VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2024 Bc. Matej Kajan

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

ROZŠÍŘENÁ REALITA V PRŮMYSLOVÉ VÝROBĚ A ÚDRŽBĚ

AUGMENTED REALITY IN INDUSTRIAL PRODUCTION AND MAINTENANCE

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Matej Kajan

AUTHOR

VEDOUCÍ PRÁCE — Ing. Karel Horák, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Matej Kajan ***ID:*** 211151

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Rozšířená realita v průmyslové výrobě a údržbě

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je navrhnout experimentální úlohu rozšířené reality v průmyslových podmínkách, tedy implementovat vizuální rozpoznávání objektu a vizualizovat v real-time jeho model v zobrazovacím zařízení uživateli.

1. Proveďte zevrubnou rešerši zařízení využívaných pro úlohy rozšířené příp. mixované reality.
2. Proveďte rešerši metod použitelných pro rozpoznání geometricky a opticky jednoduchých objektů v definované průmyslové scéně.
3. Po domluvě s vedoucím vyberte vhodnou úlohu, objekt a snímací a zobrazovací zařízení pro demonstraci průmyslové (de)montáže. Objekty mohou být např. sada montážních nástrojů, výrobek rozložitelný na části, sestava geometrických tvarů apod.
4. Do frameworku vybraného zařízení implementujte zvolenou metodu rozpoznávání objektů.
5. Proveďte experimenty s detekcí stacionárně položeného i operovaného objektu při volně se pohybujícím záznamovým zařízením (např. brýlí na hlavě operátora).
6. Experimenty vyhodnoťte z hlediska přesnosti lokalizace objektů, latence (operabilita) a robustnosti, tj. okluzních podmínek (zákryt objektu uchopením ruky atp.).

DOPORUČENÁ LITERATURA:

1. E. Pangilinan, S. Lukas, V. Mohan: Creating Augmented and Virtual Realities. OReilly Media, 2019. ISBN 9781492044192.
2. Theissler A., et al.: Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. Reliability Engineering & System Safety, 2021. ISSN 0951-8320.

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 15.5.2024

Vedoucí práce: Ing. Karel Horák, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

Abstrakt

Práca sa zaoberá témou použitia XR (extended reality) v priemyselnej montáži. Snahou je implementovať systém schopný vizuálne navádzať operátora pri konštruovaní výrobku pomocou rozpoznania objektu a augmentáciou výsledného obrazu. Na úvod sa uvedie využitie rozšírenej reality v priemysle. Ďalšú časť tvorí rešerš na tému zariadení rozšírenej alebo virtuálnej reality a ponúka stručné porovnanie súčasných možností. Potom sa prezentuje metodika rozpoznávania ľubovoľného objektu v scéne a jeho modelová reprezentácia. Následná implementácia je schopná detegovať objekt v reálnom čase na CPU, je odolná voči oklúzii a disponuje informáciou o jeho orientácii.

Kľúčové slova

Rozšírená realita, priemyselná montáž, rozpoznávanie objektu, OpenCV, C++, počítačové videnie, multithreading, OpenMP, významné body, oklúzia, RANSAC, Homografia,

Abstract

This paper seeks to explore the possibility to utilize XR (extended reality) in industrial assembly. The aim is to implement a system, which is able to visually navigate the operator during the product assembly process by the means of object recognition and image augmentation. The first chapter presents the use-case of augmented reality in the industry. The next part consists of research on the topic of augmented and virtual reality devices and provides a brief comparison of the current state of the art. Afterwards, a methodology is presented for object recognition of an arbitrary object. The implementation is able to detect the object in real-time, is resilient to occlusion and contains the information about the object's orientation.

Keywords

Augmented reality, industrial assembly, object recognition, OpenCV, C++, computer vision, multithreading, OpenMP, keypoints, occlusion, RANSAC, Homography

Bibliografická citace

KAJAN, Matej. *Rozšířená realita v průmyslové výrobě a údržbě* . Brno, 2024. Dostupné také z: https://www.vut.cz/studenti/zav-prace/detail/159975. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Karel Horák.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	Bc. Matej Kajan
VUT ID studenta:	211151
Typ práce:	Diplomová práca
Akademický rok:	2023/24
Téma závěrečné práce:	Rozšířená realita v průmyslové výrobě a údržbě

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

.

V Brne dňa: 15. mája 2024 ----------

podpis autora

Poďakovanie

V prvom rade chcem poďakovať svojej rodine za ich neustálu podporu a dôveru. Pánovi Ing. Karel Horák, Ph.D. za jeho odborné vedenie, ochotu a rady počas tvorby tejto práce. V neposlednom rade mojím aiki a ne-aiki priateľom, bez ktorých by bol svet menej farebný.

V Brne dňa: 15.5 2024 ----------

podpis autora

OBSAH

Zoznam obrázkov

Zoznam tabuliek

ÚVOD

Snaha využiť technológie XR (extended reality) má stále čoraz vyššiu tendenciu vo výrobných podnikoch a závodoch. Celková cena práce závisí na efektivite a presnosti výrobného procesu, avšak so stupňujúcimi sa požiadavkami nielen na samotný výrobok, ale aj na operátora sa situácia komplikuje. Pracovník je nútený podstúpiť radu školení a tréningov, len aby bol schopný zostaviť tieto komplexnejšie výrobky [1]. Z tohto dôvodu je možné usúdiť, že je čoraz relevantnejšie ich podporiť s vhodným typom montážnych inštrukcií. V tomto bode by sa XR mohla spojiť práve vo výrobných halách v rukách operátorov.

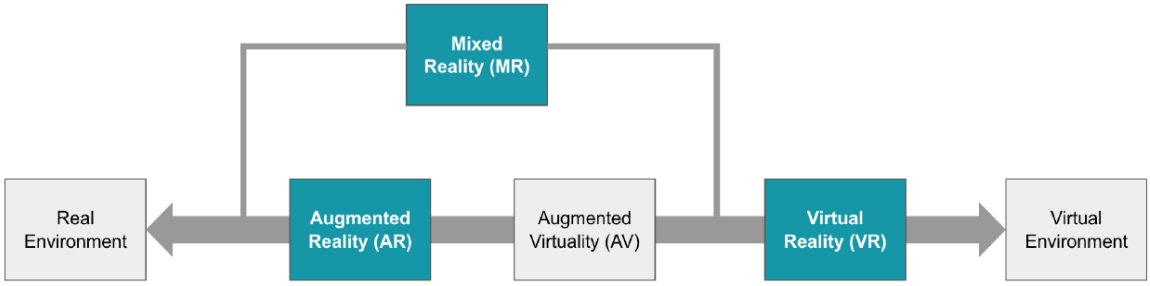
Využitím tzv. WGS (Worker guidance systems) by bolo možné operátorov zásobovať relevantnými informáciami s ohľadom na úkon a na spôsob/dôvod jeho vykonania [2]. Táto práca sa skúma možnosť vytvoriť systém, ktorý by bol schopný navádzať operátora, ktorý nemá apriori znalosť o produkte a spôsobe jeho montáže. Preto je nutné definovať spôsob a médium, na ktoré sa bude informácia prenášať. Prirodzeným kandidátom na túto úlohu sú HMD (head-mounted displays), ktoré jednak informujú operátora o postupe priamo v jeho zornom poli a zároveň sú schopní voľne používať ruky, tzv. „handsfree“ riešenie.

Cieľom práce je vytvoriť aplikáciu pre priemyselnú montáž. Program by mal byť schopný rozpoznať jednotlivé komponenty výrobku a pomocou inštrukcií navádzať operátora k zostaveniu produktu. Najprv sa formuluje využitie systémov rozšírenej reality v priemysle. Druhá kapitola sa sústreďuje na výber vhodných okuliarov pre rozšírenú, respektíve virtuálnu realitu. Kapitola je vedená formou rešerše a porovnania aktuálne dostupných riešení. Ďalšia časť sa zaoberá metódami rozpoznania ľubovoľného objektu v definovanej scéne. Následne je popisované zvolené riešenie a jeho implementácia. Na záver sú vyhodnotené výsledky použitej metódy s ohľadom na latenciu, presnosť a vplyv oklúzie.

1. FORMULÁCIA PROBLÉMU

Ako bolo spomenuté v úvode, cieľom je vytvoriť AR („Augmented Reality“ – Rozšírená realita) systém, ktorý by bol schopný jednoducho a efektívne navádzať operátora pri montáži výrobku. Motiváciou je fakt, že celková cena výrobku závisí na efektivite a presnosti výrobného procesu, ktorý nemusí byť triviálny. Následne je tak operátor nútený podstúpiť školenia a štúdium obsiahlych manuálov. Zároveň v priebehu tzv. „on-ramp“ fázy vznikajú prestoje a operátori musia absolvovať ďalšie zaučenia, čím sa proces predlžuje a naberá na cene.

Pre ilustráciu, na obrázku dole je zobrazený vzťah rozšírenej/virtuálnej reality k reálnemu, respektíve virtuálnemu svetu. AR systémy majú najbližšie k realite spomedzi ostatných typov XR. Užívateľ má priamu väzbu na reálny svet, napríklad cez okuliare, alebo cez prenos kamery, a iba obohacuje vnímanie reality. VR („Virtual reality – virtuálna realita) je kompletne virtualizované prostredie, v ktorom sa človek pohybuje, takže v ňom nemá takmer žiadny vzťah k fyzickému svetu. MR je niekde na rozmedzí, v ktorej sa zmieša reálny a virtuálny svet s možnosťou manipulácie v oboch doménach. Využite AR je preto vhodné pre implementáciu v úlohách, ktoré vyžadujú napríklad iba anotáciu a vizualizáciu stavov riešenia.



Obrázok 1.1-1 - Kontinuum reality a virtuality [3]

Použitie AR systémov pri montáži výrobku, v prípade jednoduchších úloh, operátorovi umožňuje rýchlejšie lokalizovať komponenty výrobku a urobiť menej chýb, avšak pomocou papierového manuálu užívateľ dokáže zložiť výrobok v kratšom čase [4]. Pre úlohy s vyššou komplexnosťou sa zvyšuje presnosť priestorového ukladania komponentov a užívatelia sú viac priazniví voči AR systému v porovnaní k papierovým inštrukciám. Avšak celkový čas dokončenia úlohy sa značne nezmení [5].

V ďalšej časti je snahou priblížiť použitie AR systémov v priemysle. Potom sa definuje miesto na prácu s AR systémami, v ktorej sú popisované spôsoby usporiadania pracoviska, pomocou čoho sa môžu určiť hardvérové prostriedky. V druhej podkapitole sa znázorňuje priebeh AR systému, ktorým sme schopní vymedziť potrebné softvérové prostriedky na realizáciu systému.

1.1 Rozšírená realita v priemysle

AR sa v priemysle vyskytuje v štyroch podobách - v procese montáže (1), v rámci tréningu (2), pri údržbe (3) alebo diagnostike (4). V rámci diagnostiky sa jedná o systémy, ktoré vizualizujú informácie a reportujú operátorom stav zariadení alebo výroby. Požiadavkou je kvalita výrobku, takže prepojením s priemyselnými systémami, ako napríklad CAQ (Computer-Aided Quality), je možné identifikovať a prediktívne zakročiť voči potenciálnym nedostatkom vo výrobnom procese [6].

Údržba zariadení, či už prediktívna alebo post-poruchová, riadi operátora inštrukciami pre správnu demontáž/montáž prístroja. V prípade prítomnej poruchy je známe jej miesto a operátorovi sa tak počas úkonu zobrazujú virtuálne elementy, ktoré majú za cieľ urýchliť a zefektívniť proces výmeny komponentu [7].

Počas tréningu je cieľom naučiť operátora daný postup a metodiku skladania výrobku. V tomto smere sa využíva jednak VR, kde je proces kompletne virtualizovaný, alebo AR, ktorého snahou je čo najviac sa priblížiť reálnemu svetu. Práve to je výhodou použitia AR, že zaškolovaný operátor je si schopný osvojiť proces montáže jednoduchšie a rýchlejšie pomocou AR oproti iným metódam [8][9][10].

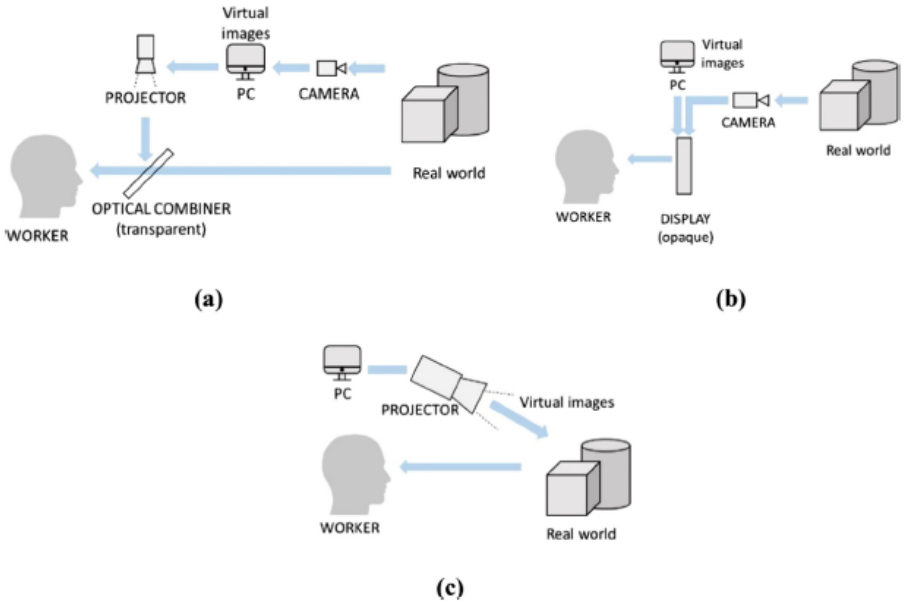
V prípade možnosti realizácie AR systému priamo na pracovisku, je predpokladom, že vhodným navádzaním operátora pomocou inštrukcií sa zníži chybovosť a čas procesu montáže výrobku. Principiálne sa jedná o podobné systémy ako pri tréningu, avšak obsahujú vyššiu komplexnosť. Zároveň je produkovaný väčší tlak na operátora, mentálny a fyzický, keďže by sa mali používať počas značnej časti pracovnej doby. Tu treba dohliadať, napríklad, na váhu HMD zariadenia alebo na samotné inštrukcie, aby boli jednoduché a zmysluplné.

1.2 Pracovisko rozšírenej reality

Samotné AR pracovisko je tvorené priestorom, kde sa práca vykonáva, napríklad pracovným stolom (1), nástrojmi (2) a jednotlivými komponentami (3). Ďalším predpokladom je prítomnosť zobrazovacieho zariadenia (4) – informačného panelu, na ktorom operátor môže sledovať inštrukcie. Nakoniec snímací element (5), pomocou ktorého sa zachytia a rozpoznajú objekty použité pri práci (komponenty), alebo akcie od užívateľa (interakcia).

Existujú 3 základné usporiadania AR pracoviska:

a) Opticky priehľadné
b) Video priehľadné
c) Obrazová projekcia



Obrázok 1.2-1 - Spôsoby zobrazovania v AR scéne (a) opticky priehľadné, (b) video priehľadné, (c) Obrazová projekcia [2]

Opticky priehľadné riešenie vkladá virtuálnu zložku scény na priehľadný displej, ktorý je medzi užívateľom a scénou. Hlavným zástupcom sú HUD (head-up-display), poprípade HMD, ako napríklad AR okuliare umiestnené na hlave operátora. Výhodou je poskytnutie možnosti vnímať priamo reálny svet cez transparentný displej. Ďalej nie je limitované rozlíšením (keďže sa jedná o reálnu scénu), skreslením alebo latenciou spôsobenou prenosom obrazu [11] (čo samozrejme neplatí pre virtuálnu zložku). Užívateľ tak vníma svet prirodzene a priamo. Nevýhodou je vkladanie virtuálnych objektov do scény, pri ktorom dochádza k miere latencie a k problému zarovnania virtuálnej zložky s reálny svetom.

Video transparentné zostavy snímajú reálny svet pomocou kamery, čím sa digitalizuje scéna. Následne v PC alebo na inom hardvéri sa spojí virtuálna zložka s obrazom kamery. Zvyčajne sa kamera pripevní na displej, aby sa vytvoril pocit pre užívateľa, že sa pozerá priamo cez displej. V zásade je cieľom dosiahnuť podobný pocit ako pri opticky priehľadnom variante. Jedná sa o časté riešenie, z dôvodu dostupnosti hardvéru – kamera a PC [11]. Výhodou je väčšia kontrola nad procesom kombinovania reálnej a virtuálnej zložky pomocou metód počítačového videnia. Naopak problémom je limitovanie zorného poľa užívateľa, ktoré je definované parametrom FOV (field of view) kamery. Taktiež je nutná vyššia výpočtová sila na spracovanie a augmentáciu obrazu.

Pri montáži výrobku sa tieto varianty používajú najčastejšie [12]. Jedná sa o neinvazívne riešenie, pri ktorom nie je potrebný fyzický zásah do priestoru montáže a zároveň má operátor ruky voľné na vykonávanie práce. Avšak je nutný hardvér, ktorým musí byť operátor vybavený (HMD alebo prítomnosť kamery s displejom).

Obrazová projekcia pozostáva z premietania virtuálnej zložky priamo na reálne objekty. V tomto prípade operátor nemusí disponovať žiadnym hardvérom a môže sa voľne pohybovať na pracovisku. Naopak je nutné inštalovať dodatočný hardvér (projektory) na pracovisko. Jedná sa o jednoduché navádzanie pomocou inštrukcií, keďže projekcia je v scéne statická. V prípade komplexnejšieho úkonu je potrebné väčšie množstvo projektorov a zároveň komponenty sa musia nachádzať na vopred definovaných miestach. Výhodou je teda jednoduchosť a intuitívne pochopenie inštrukcií operátorom. Na druhú stranu scéna a komponenty sú statické alebo sa môžu nachádzať iba na konkrétnych miestach a rovnako môže byť problémom obštrukcia projekcie operátorom, čím sa zakryje samotná inštrukcia [12].

1.3 Priebeh typického AR programu

Typická slučka AR systémov používaných v priemyselnej montáži pozostáva zo šiestich štádií. Prvým je akvizícia snímku (1) pomocou snímacieho zariadenia, zvyčajne kamerou. Následne sa snímok spracuje pomocou metód počítačového videnia (2), ktoré obsahujú etapy ako predspracovanie obrazu, segmentácie, popisu a klasifikácie. V prípade úspešnej klasifikácie sa jednotlivé objekty sledujú (3), aby sme boli schopní určiť ich polohu v obraze, na ktoré je následne možné pridať virtuálnu zložku. Ďalším krokom je rozpoznanie interakcie operátora so systémom (ak to systém podporuje), napríklad gestami, pomocou hlasu a podobne (4). Potom sa na základe predošlých krokov a vnútorného stavu systému určia budúce montážne inštrukcie (5). Posledným krokom je vizualizácia týchto inštrukcií operátorovi (6). Tento proces sa opakuje až do dokončenia procesu montáže.

Obrázok 1.3-1 - Typická slučka AR systémov

Snímanie pracoviska je možné realizovať staticky alebo dynamicky. V prvom prípade sa snímok získava z vopred pevne daného miesta v priestore. Napríklad priamo nad pracovným stolom a podobne. Pri druhom spôsobe sa kamera pohybuje v priestore. Najčastejšie je kamera vstavaná alebo pevne upevnená na okuliare typu AR/VR.

Výsledný snímok sa spracováva vo výpočtovej jednotke, ktorá v závislosti od realizácie celého systému, môže byť PC, dedikovaný hardvér alebo samotné AR okuliare. V úlohách tohto typu sa najmä rozpoznávajú predmety alebo lokácie pomocou techník počítačového videnia. Inými slovami komponenty alebo pri tzv. „order picking“, konkrétne miesta (výber objektov z konkrétnych miest).

Po rozpoznaní objektu je dôležité sledovať daný objekt, napríklad pre jeho anotáciu alebo registráciu inej virtuálnej zložky. Po pojmom registrácia sa chápe umiestnenie virtuálneho objektu (3D/2D) do priestoru reálneho sveta. Bežnou požiadavkou je presnosť umiestnenia, v opačnom prípade by sa rušila pozornosť užívateľa.

Interakcia od užívateľa môže byť haptická pomocou prídavných zariadení (ovládače), gestami (potrebná ďalšia forma vizuálneho rozpoznávania) alebo hlasom. V ideálnom prípade má operátor ruky voľné na prácu.

Po vyhodnotení predošlých informácií, tzv. manažér montážnych inštrukcií, čo je forma databázy a programovej logiky, vyberie ďalšie inštrukcie.

Posledný krok je vizualizácia potrebných informácii, čo spočíva v pridávaní virtuálnej zložky do obrazu. Dôležitá je včasná a dostatočne presná vizualizácia.

1.3.1 Montážne inštrukcie

AR inštrukcie sa definujú ako séria vizuálnych informácií, ktorá odráža zámer pri montáži výrobku [13]. Zároveň majú za úlohu popísať vzťah medzi krokmi a komponentami. Definujú sa preto dôležité charakteristiky pre AR inštrukcie:

1. Zámernosť
2. Včasnosť
3. Nízka kognitívna záťaž

Pre operátora je žiadúce, aby inštrukcie neboli dvojzmyselné, ale jednoznačné. Dôležitý je prenos informácie a zámeru (1). Vizualizácia inštrukcií v reálnom čase dovoľuje operátorovi včas reagovať, zároveň podaná informácia je vždy aktuálna a tak sa môže správne rozhodovať (2). Zaplavenie zobrazovacej plochy prvkami AR zvyšuje mentálnu záťaž, zneprehľadňuje situáciu, zvyšuje trvanie procesu a unavuje operátora(3).

Vhodné vizuálne inštrukcie pozostávajú z textu, 2D obrázkov/animácií a 3D grafických prvkov. Už krátky text prenáša značnú informáciu a zároveň sa jednoducho interpretuje. Animácie sú vhodné na popis pohybu, ktorý má operátor vykonať pri montáži a 2D obrázky napríklad pre verifikáciu. 3D grafické prvky vkladajú informáciu priamo do fyzického sveta a presnejšie navádzajú operátora ku konkrétnemu úkonu.

2. ZARIADENIA PRE XR

Všeobecne tieto zariadenia vyžadujú výpočtový systém a displej na zobrazovanie XR. Z tohto hľadiska do toho spadajú už napríklad smartfóny či tablety, avšak fenoménom posledných rokov sa stali práve okuliare na XR.

Okuliare samotné pozostávajú z plôch, na ktoré sa zobrazujú alebo premietajú (na základe použitej technológie) prvky XR. V závislosti od realizácie majú buď vlastnú, alebo externú výpočetnú platformu. Momentálne z dôvodu obrovského rastu výpočtovej sily a paralelizácii je možné analyzovať scénu a následne renderovať potrebné objekty už priamo na čipe okuliaroch. Práve v minulých rokoch sa bežnými prostriedkami nedalo dostať na rozumné latencie pri real-time aplikáciách, teraz už napríklad známe Microsoft Hololens 2 ponúkajú dynamické navádzanie pracovníkov v reálnom čase. Dnes už sa svet XR dostal do štádia, aby bolo možné tieto zariadenia používať aj v profesnej sfére a nielen na komerčné účely.

V rámci rešerše sa okuliare vyberali s ohľadom na konkrétne parametre tak, aby sa prekrývali, aspoň čiastočne, s problematikou práce. Parametre, ako napríklad, dostupnosť, cena, váha, podpora a kompatibilita. V dnešnej dobe je skutočne bohatý výber možností, hlavne čo sa týka AR okuliarov. Výsledné porovnanie okuliarov je v prílohe A.

2.1 Rozšírená realita

Najčastejším riešením AR zariadení sú HMD. Ich charakteristickými vlastnosťami sú možnosť operátora priamo vnímať okolitý svet, avšak za cenu nízkeho FOV. V závislosti od aplikácie, latencia a nesprávne zarovnanie reálnej a virtuálnej zložky môžu spôsobovať nedostatky v jeho práci.

Obrázok 2.1-1 - Použitie HMD pri tréningu personálu vo výrobe [2]

2.1.1 RealWear

RealWear je jedným z lídrov zaoberajúcich sa riešeniami v oblasti aR (assisted reality). Cieľom je umožniť operátorom v prvej línii, napríklad AR okuliarmi , aby boli schopní vykonať úlohy bezpečne a s dostatkom informácií v reálnom čase.

Na pravej strane je namontovaný displej, ktorý premieta obraz z kamery (48MP kamera) alebo sa zobrazuje práve používaná Android aplikácia. Novší model, Navigator 520, prichádza s HD displejom s 60 FPS a navyše s 24° FOV oproti predošlému 20° FOV a 848x480 displejom.

Zariadenie je ovládané pomocou hlasu, kde vďaka vstavanému systému WearHF sú funkcionality (ich spúšťanie) danej Android aplikácie prevádzané z hlasu na príkazy**.** [14]

Obrázok 2.1-2 - RealWear Navigator 520 [15]

Chipset sa skladá z 2.0 GHz 8-core Qualcomm® Snapdragon™ 662 a Adreno 610 GPU - OpenGL® ES 3.2 & OpenCL™ 2.0. Obsahuje 64GB internej pamäte a 4GB RAM. Pripojenie je možné cez Bluetooth 5.1, Wi-Fi 2.4/5 GHz a taktiež disponuje GPS, GNSS lokalizáciami. Čo sa týka praktického používania, zariadenie váži 274g a batéria má stanovenú výdrž na 6-8 hodín, čiže operátor je schopný zariadenie používať počas jedenej smeny [16]**.**

Keďže sa jedná o Android aplikácie, je možné použiť Android Studio alebo Unity C#, Unreal Engine C++. Kľúčovými prvkami tohto typu zariadenia, že disponuje malým externým displejom, sú práve možnosť si zvoliť kde a kedy si vizualizáciu zobraziť. Teda možnosť displej odvrátiť od svojho zorného poľa a pokračovať v práci bez rušivého vplyvu [17]**.**

2.1.2 XREAL

Pôvodne známy pod menom *NReal* je firma na vývoj okuliarov pre rozšírenú realitu. Ich hlavným špecifikom je jednoduchosť, komfort a autentický zážitok. Zároveň sú ich výrobky zamerané skôr pre širokú spotrebiteľskú verejnosť. Vlajkovou loďou sú okuliare *Light*, ktoré už podľa názvu odhalia, že ide o výrobok s nízkou váhou – 106g. Na prvý pohľad vyzerajú ako obyčajné okuliare, ale skrývajú v sebe OLED displeje 1920x1080 s obnovovacou frekvenciou 60Hz, ktoré sa premietajú na šošovky okuliarov**.** [18]

Obrázok 2.1-3 - NReal Light Dev kit [19]

V prednej časti obsahujú dva SLAM (Simultaneous localization and mapping) senzory, na sledovanie objektov a prostredia, 5MP RGB kameru a disponujú 6DOF priestorovým sledovaním. Ponúkajú FOV 52°, čo však môže byť nedostačujúce pre dynamické scény.

Hardwarovo sú vybavené kontrolérom, ktorý obsahuje SoC (system-on-a-chip) Qualcomm Snapdragon TM 845, 6 GB RAM a 64 GB hlavnú pamäť. Operačným systémom je Android 8 [20].

Okuliare sa dajú použiť v dvoch módoch, v ktorých musia byť neustále pripojené k smartfónu pomocou USB-C. Prvý je „AR Space“, pri ktorom užívateľ má pred sebou virtuálny priestor, kde môže interagovať s rôznymi aplikáciami a oknami. Druhý, „AirCasting“, čo je len zrkadlenie obrazovky telefónu.

Tabuľka 2-1 - Porovnanie XReal Light & Air [21]

Feature	XREAL Light	XREAL Air
Size(folded)	156mm * 52mm * 44mm	151mm * 41mm * 51mm
Weight (Excluding cable)	106g	~77g
Resolution Per Eye	1920 * 1080	1920 * 1080
FOV	52 Degrees	46 Degrees
Frame Rate - MR Space	60Hz	60Hz; 72Hz in future release
Supporting System	Android	Android
RGB Camera	Yes	No
Grayscale Camera	Yes	No
IMU	Yes	Yes
Connection	USB-C Cable	USB-C Cable
Audio	Dual Speakers and Microphones	Dual Speakers and Microphones

Existuje aj starší model, XReal Air, ktorý však neponúka rovnaké funkcionality. Vývoj aplikácií je sprostredkovaný pomocou Android Studia alebo Unity C# / Unreal Engine C++. Príkladom je výskum na univerzite v Catánii, v ktorom sa bádalo nad tzv. HOI (Human-Object Interaction). Výber XReal okuliarov bol opodstatnený práve kvôli ich nízkej hmotnosti a komfortu, a zároveň možnosti využiť niektoré natívne funkcionality NRSDK [20][22].

Avšak najväčším úskalím pri vývoji je obmedzená podpora zo strany NRSDK a XReal pre verejnosť. XReal sa momentálne sústreďuje hlavne na B2B zákazníkov a pre vývojárov ako takých, sú niektoré funkcionality nedostupné, ako napríklad sledovanie viacerých objektov alebo niektoré hrubé dáta, dáta zo stereo kamery, dáta zo snímačov, podpora OpenXR a iné[1] [2].

[1] **https://community.xreal.com/t/openxr-support-for-nreal/3889/2 [cit. 11.10.2023]**
[2] **https://community.xreal.com/t/control-stereo-camera/4483 [cit. 11.10.2023]**

2.1.3 Vuzix Blade 2

Vuzix, podobne ako predošlé spoločnosti, sa zameriava na AR technológie do priemyslu alebo aj konzumnej sféry. Používajú tzv. technológiu „Waveguide“, pri ktorej sa jedná a tenké optické elementy, na ktoré sa zobrazí projekcia, a tá sa následne odráža do oka[23].

Vuzix okuliare majú dve hlavné využitia. Prvým sú telekonferenčné hovory, užívateľ volá „hands-free“ a je schopný zdieľať okolie alebo obrazovku. Ide o vzdialenú kolaboráciu. Druhé využitie je pre pracovníkov v prvej línii, ktorí majú možnosť sa spojiť s expertom na diaľku tak, aby rýchlo vyriešili problém. Tým sa urýchli chod podniku alebo dokonca školenie pracovníka.

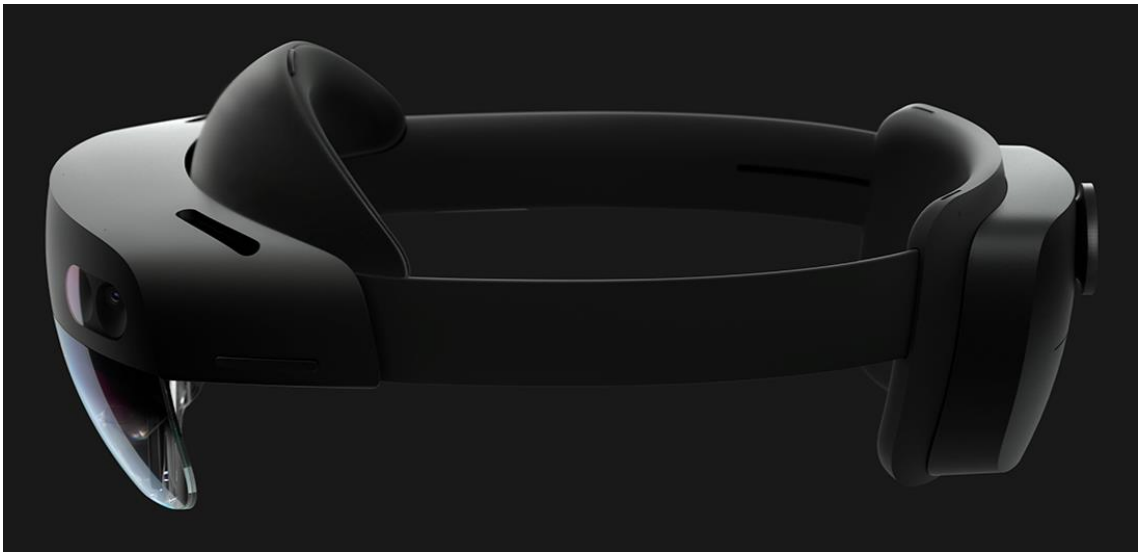
Obrázok 2.1-4 - Vuzix Blade 2 [24]

Projekcia sa zobrazí s rozlíšením 480x480 a FOV 20°. Okolie sleduje jediná 8MP kamera. Ovládanie je možné rovnako pomocou hlasu alebo touchpadom, ktorý je uložený na ráme okuliarov. Pripojiť sa je možné pomocou Bluetooth 5.1, Wi-Fi 2.4/5 GHz. Okuliare disponujú 3DOF a „head-trackerom“. V rámci HW sú vybavené 4-jadrovým ARM procesorom a 40GB ROM. Všetko je riadené operačným systémom Android 11, a teda vývoj aplikácii je rovnaký ako pre každé iné Android zariadenie. [24]

2.1.4 Microsoft Hololens 2

S Hololens 2 sa Microsoft snažil zamerať skôr na podniky ako na široké komerčné využitie. Cieľom bolo zdokonaliť už predošlý produkt, čo sa týka precíznosti, inovácie a kolaboratívnej práce. Momentálne ponúkajú moduly do priemyslu ako napríklad *Microsoft Dynamics 365 Guides, Dynamics 365 Remote Assist* a iné.

V prednej časti sa nachádzajú priehľadné holografické šošovky, s finálnym rozlíšením 2048x1080, a 8MP kamera. V ráme sú vstavané dve IR kamery na snímanie pohybu očí a ďalšie 4 kamery na snímanie a vyhodnotenie pozície hlavy. Na čipe sa nachádza SOC Qualcomm Snapdragon 850, špeciálne vyvinutá HPU – holographic processing unit, 64GB UFS a 4 GB RAM. Pripojenie je sprostredkované cez Bluetooth 5 alebo Wi-Fi 5GHz. Celková váha zariadenia je 566g a batéria má výdrž okolo 2-3 hodín, čo môže byť nevyhovujúce pre operátora, ohľadom komfortu. [25]



Obrázok 2.1-5 - Microsoft Hololens 2 [25]

Vynikajúcimi vlastnosťami Hololens sú 6DOF (six degrees of freedom) priestorové sledovanie, mesh mapovanie prostredia, vďaka ktorému je možné si vytvoriť model okolitého prostredia a vkladať virtuálne objekty s vysokou presnosťou. Ďalším parametrom je rozpoznávanie rúk a gest, čo umožňuje jednoducho manipulovať s virtuálnymi objektami. Operačným systémom je Windows 10, takže vývoj je pomerne silno podporovaný, od Unity, Unreal Engine až po OpenXR [26].

Obdobné aplikácie rádovo vznikajú práve na tejto platforme, už len vďaka pokročilým funkcionalitám od samotného Microsoft (Mixed Reality Toolkit) a kompatibilitou Hololens s rôznymi softvérovými komponentami. Jednou z nich je MA2RA (Manual Assembly Augmented Reality Assistant), ktorej cieľom bolo vytvoriť asistenčný montážny systém pre nezaučeného operátora [27]. Výskumy v blízkej tematike hľadajú ako využiť AR práve pri montáži výrobkov [4] [28] [29] [30].

2.1.5 Moverio BT-45CS

Produkt od spoločnosti Epson pozostáva z binokulárnych AR okuliarov a inteligentného kontroléra. Okuliare využívajú Si-OLED technológiu, ktorá vytvára obraz s väčším kontrastom. Obraz je následne premietnutý pomocou série zrkadiel do obzoru užívateľa, kde čierne pixely nie sú vôbec premietnuté, práve vďaka Si-OLED. [31]

Pre každé oko je navyše generovaný 1920x1080 obraz, ktorý je navyše kontrastný a priehľadný, v závislosti na jasových podmienkach. Výsledkom je zorné pole s FOV 34° a vytvára dojem, akoby sa človek pozeral na displej s 300cm uhlopriečkou vo vzdialenosti 5m. V prednej časti sa nachádza 8MP kamera, senzor svetla a senzory pre 6DOF priestorové sledovanie. Hardvérovo sú vybavené CPU Snapdragon XR1 2.52GHz Octa Core, 4GB RAM a 64 GB úložiska s možnosťou až 2TB externej Micro SD karty.[32]

Obrázok 2.1-6 - Moverio BT-45CS [32]

Pripojenie je možné pomocou USB-C (s DisplayPort Alternate módom) do ľubovoľného zariadenia s Windows 10 alebo Android 8.0 a vyššie. Poprípade využiť kontrolér, ktorý je zahrnutý v balíčku. Ten sa v princípe správa ako Android (smartfón) zariadenie s dotykovým displejom, ktoré ma navyše ešte Wi-Fi/Bluetooth pripojenie a vlastnú kameru. Celková váha je 550g, čo je porovnateľné s ostatnými AR headset riešeniami. Použité aplikácie sú vyvíjané pod Androidom (Unity C# s vlastným pluginom alebo Unreal Engine C++) alebo pod Windows.

2.2 Virtuálna realita

Virtuálna realita sa odlišuje od tej rozšírenej v absencii priameho pohľadu užívateľa na reálny svet. V zásade sa človek pohybuje v úplne graficky vytvorenej realite, alebo sa užívateľovi zobrazuje prenos z kamery, do ktorej sa následne vykresľuje virtuálny obsah. Tu sa jedná o hardvérovo vyspelejšie systémy, keďže je potrebné, napríklad, renderovať grafické prvky v reálnom čase tak, aby bol zážitok pre jednotlivca čo najviac autentický.

2.2.1 Meta Quest 2

Predtým známe pod menom Oculus Quest 2, nadväzujú na predchodcu, ale s nižšou hmotnosťou, 503g, s vylepšeným hardvérom a displejom. Všeobecne boli okuliare pozitívne prijaté verejnosťou, najmä vďaka vylepšeniam, väčšiemu komfortu, širšiemu výberu dostupných aplikácií a cene. Hlavným využitím headsetu je zábava a relax, napríklad VR hry, aplikácie na - fitness, vzdelávanie, creativitu - alebo Meta virtuálny svet (Metaverse).

Headset uvádza LCD displej s rozlíšením 1832x1920px (pre každé oko) s obnovovaciou frekvenciou 90Hz. FOV je 97° pre horizontálnu a 93° pre vertikálnu rovinu. Pomocou 4 interných kamier sú schopné 6DOF trekovanie. O výkon sa stará SoC

Qualcomm Snapdragon XR2 so 6 GB RAM a úložisko je o veľkosti 128-256GB. Okuliare prichádzajú aj s dvoma kontrolérmi, ktoré zabezpečujú odozvu na zápästia a hmat. [33]

Obrázok 2.2-1 - Meta Quest 2 [33]

Ide o zariadenie, ktoré je schopné fungovať samostatne, avšak je možné ho pripojiť k PC pomocou USB-C alebo pomocou „Air Link“ cez WiFi. Týmto spôsobom sme schopní vyvíjať a testovať softvér, mať potenciálne prístup k lepšiemu hardvéru alebo používať SteamVR a iné PC VR hracie platformy. Na zariadení beží operačný systém založený na Androide, takže aplikácie sa dajú vyvíjať pomocou Unreal Enginu alebo Unity a Oculus SDK. Následne sa aplikácie podajú do Oculus Store. Výdrž batérie je okolo 2-3 hodín.

2.2.2 HTC Vive Pro 2

Vive, ako korporátna značka HTC, sa snaží vytvoriť prostredie na kolaboráciu, komunikáciu alebo zábavu. Jedným z produktov je HTC Vive Pro 2, ktorý je nasledovníkom predošlého úspešného HTC Vive Pro. Tento model sa najviac pýši svojím displejom o 2488x2488px, 120 Hz obnovovacou frekvenciou a 120° FOV. Okuliare majú integrovanú stereo kameru 640x480 [34], ktorá sa používa, napríklad, pre vymedzenie priestoru na pohyb pri používaní VR aplikácií. Na sledovanie polohy samotného užívateľa v priestore, slúžia tzv. „basestations“ s 150° horizonálnym FOV (SteamVR Tracking 2.0 [35]), ktoré sa nainštalujú v miestnosti a dve dokážu pokryť priestor 5x5m. Ďalšími doplnkami sú dva ručné ovládače, pomocou ktorých sa interaguje s jednotlivými aplikáciami**.** [36]

Obrázok 2.2-2 - HTC Vive Pro 2 full kit [37]

Headset musí byť pripojený k PC via „link box“ cez USB 3.0, display port kábel a napájací kábel. Samotné PC má uvedené minimálne nároky, CPU Intel® Core™ i5-4590 alebo AMD Ryzen 1500, GPU NVIDIA® GeForce® GTX 1060 alebo AMD Radeon RX 480, 8GB RAM a Windows 10/11. Približná doba výdrže batérie na jedno nabitie je 6 hodín. [36]

3. ROZPOZNÁVANIE OBJEKTU

Úlohou rozpoznávania v počítačovom videní je identifikácia a klasifikácia objektov v obraze. Program je tak schopný interpretovať snímaný priestor a na základe tejto informácie má schopnosť ďalej sa rozhodovať. Konkrétna metóda by mala dokázať zdolať prekážky ako oklúzia, zmena veľkosti alebo orientácie objektu. Najprv je popisovaný koncept modelu objektu a následne spôsoby jeho reprezentácie a detekcie.

3.1 Model objektu

Pre správne navádzanie operátora je dôležité rozoznať daný komponent. Preto je potrebné apriori vytvoriť model objektu, ktorý sa následne v online fáze vyhľadáva v snímanom obraze. Bližší popis metód na vytvorenie/reprezentáciu modelu je v ďalších kapitolách. Nižšie je uvedená koncepcia použitia modelu pri rozpoznávaní.

Pri návrhu systémov využívajúcich model objektu je nutné brať do úvahy:

1. Typ senzoru
2. Spôsob vytvorenia a reprezentácie modelu objektu
3. Vhodnú metódu pre hľadanie korešpondencie (rozpoznanie)

Bežným typom senzoru v optických sústavách je digitálna kamera. Výsledkom je digitalizovaný snímok, matica o jasovej intenzite. Pri rozpoznávaní objektov môže byť vhodné rozšíriť poskytnutú 2D znalosť o 3D aspekt. Pri monokulárnych systémoch by išlo o zaostrenie, tieňovanie, pohybovú paralaxu [38]. Iným prístupom je použitie binokulárneho systému, stereo kameru, RGB-D kameru [39], ktoré dodajú hĺbku obrazu ako ďalší parameter. Podobne použitie vysokofrekvenčných optických sensorov (laser, LiDar, radar...), ktoré vytvoria 3D mapu priestoru, tiež známe ako „point-cloud“. V konečnom dôsledku ide o spôsob a typ získanej informácie zo senzoru. (1)

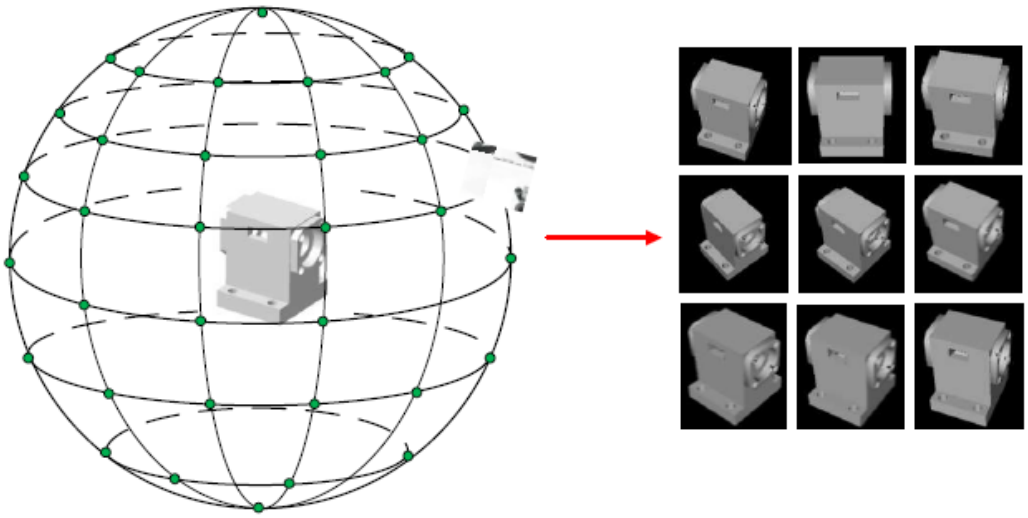
Principiálne existujú dva prístupy na vytváranie modelu. Prvý spôsob vychádza zo snímkov reálneho objektu z rôznych pohľadov, čím sa vytvorí súbor príznakov, ktoré popisujú celý objekt (360°). Druhý spôsob spočíva v použití virtualizovaného modelu, napríklad CAD (computer aided design) modelu, skonštruovaného pomocou vopred definovaných geometrických primitív [40]. Obidvomi spôsobmi získavame popis objektu.

Tvorenie modelu zo snímkov má istú nevýhodu. Jedná sa totiž o formu merania, takže vchádza do toho ľudský faktor, ktorý môže ovplyvniť výsledný model. Chyby ako nedodržanie ekvidištantného uhlu medzi snímkami, nedokonalý model – identické alebo chýbajúce snímky. Naopak vytvorenie modelu je možné z ľubovoľného objektu a zároveň proces je veľmi intuitívny.

CAD modely sú štandardizované, ľahko opísateľné a výrobcovia už majú tieto systémy integrované, čo značí nemalú výhodu. Avšak pri rozpoznávaní nasleduje opačná úloha, v ktorej z daného obrazu je potrebné vyvodiť popis objektu, ktorý sa bude následne porovnávať [40]. (2)

Rozpoznanie objektu pozostáva z hľadania korešpondencie medzi objektom a jeho modelom. Zvolená metóda by mala proces zvládnuť aj s neúplným popisom objektu, keďže dochádza napríklad k oklúzii alebo snímania objektu z iného pohľadu a podobne. Po nájdení korešpondencie môže byť potrebné stanoviť geometrickú transformáciu medzi modelom a nájdeným objektom[40], ak treba mať znalosť o orientácii objektu v priestore. (3)

Podobný koncept sa použil napríklad v práci na Pekinskej univerzite [41] , kde vytvorili systém lokalizácie a sledovania objektu v scéne. Model objektu je v tomto prípade súbor virtuálnych snímkov z rôznych uhlov pohľadu (tak aby sa vytvoril kompletný model) v CAD prostredí. Tento proces je možné aj automatizovať [42].



Obrázok 3.1-1 - Tvorba modelu objektu [41]

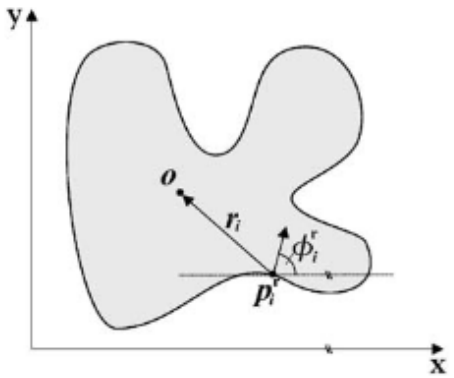
Následne sa z virtuálnych snímkov extrahovali hrany pomocou Laplacového operátora a vytvoril sa kontúrny model objektu. Pokračuje sa estimáciou pózy objektu v priestore a registráciou 3D modelu z CAD systému na správne miesto.

3.2 Metódy popisu založené na detekcii hrán

Prirodzenou voľbou reprezentácie objektu sú jeho hranice (hrany), ktoré presne vyznačujú jeho tvar. Tie sú definované ako miesta s veľkou zmenou gradientu jasovej intenzity a tak v prvej fáze sa detegujú hrany pomocou napríklad Canny detektoru alebo kombináciou operátorov na detekciu hrán. Vo fáze druhej sa definuje vzťah jednotlivých hrán k referenčnému bodu alebo ich vzájomný vzťah, čím sa vytvorí model útvaru. V tretej fáze sa jedná o spôsob „template matching“, kde sa hľadá podobnosť referenčného tvaru k inštancii v obraze, napríklad pomocou Hausdorffovej, Chamferovej vzdialenosti alebo iných metrík. Podrobnejšie sú metodiky popisované nižšie.

3.2.1 Všeobecná Houghova transformácia

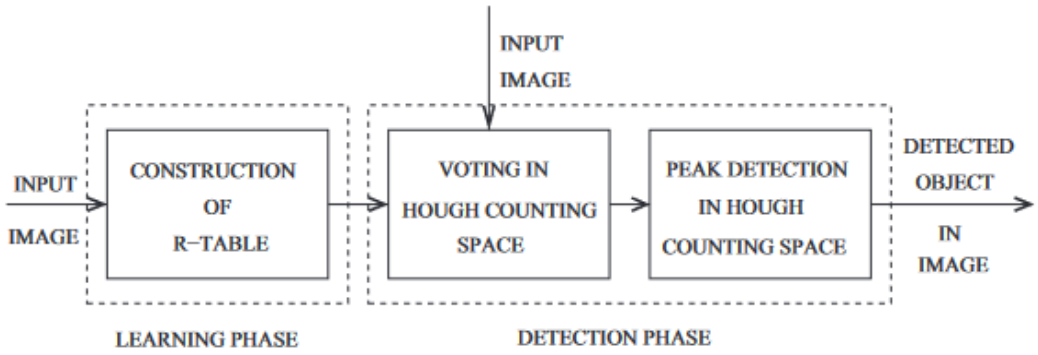
Všeobecný tvar Houghovej transformácie, tzv. GHT, dokáže popísať ľubovoľný útvar, ktorý sa nemusí skladať z analyticky opísateľných tvarov. V off-line fáze sa vytvorí tzv. R-tabuľka, ktorá popisuje pozíciu a orientáciu hrán v referenčnom obraze [43]. Najprv sa definuje referenčný bod objektu, zvyčajne ťažisko (centroida množiny všetkých hrán). Následne sa pre každý bod hrany určí vzdialenosť **r** a orientácia **Φ** od centroidy, kde gradient **Φ** predstavuje index tabuľky.



j	ϕ_j	r_i
0	0	$\{r_i \mid \phi_i = 0\}$
1	$\Delta\phi$	$\{r_i \mid \phi_i = \Delta\phi\}$
2	$2\Delta\phi$	$\{r_i \mid \phi_i = 2\Delta\phi\}$
⋮	⋮	⋮

Obrázok 3.2-1 - Model objektu (vľavo) a vygenerovaná R-tabuľka (vpravo) [43]

V online fáze každý bod hrany hlasuje za hypotetický referenčný bod, v tzv. „Hough counting space“, podľa gradientu a R-tabuľky. Najprv sa vypočíta gradient a vytvoria sa všetky páry (uhol a vzdialenosť) z riadku R-tabuľky (1), v ďalšom kroku sa pre každý pár vypočítajú referenčné body – hlasovanie (2) a v poslednej fáze sa hľadá maximum z množiny referenčných bodov, teda sa nájde ten správny (3). [44]



Obrázok 3.2-2 - Princíp Všeobecnej Houghovej transformácie [44]

V prípade, ak je na daný objekt aplikovaná transformácia – translácia, rotácia, zmena merítka – teda sa objekt môže nachádzať pod iným uhlom, inou vzdialenosťou a inej pozícii, tak pridaním ďalšej dimenzie do viacrozmerného poľa sa vytvorí ďalší vektor kombinácií parametrov [45] .Vo výsledku teda vzniká n-dimenzionálne pole, ktorého každý prvok je kombináciou parametrov (r, Φ, rotácia, translácia, zmena merítka, …).

Problémom je však obrovský nárok na pamäť n-dimenzionálneho priestoru parametrov, ktoré majú zachytiť všetky možné kombinácie, s čím súvisí následná náročnosť prehľadávania a eventuálne hlasovania.

Samozrejme nedostatky, ktoré sa týkajú rýchlosti a nárokov na pamäť je možné zlepšiť, napríklad použitím modifikovanej verzie GHT, ktorá limituje prehľadávaný priestor parametrov a používa hierarchickú stratégiu vyhľadávania. Táto metóda umožňuje relatívne rýchlu detekciu a je zároveň použiteľná v prípade oklúzie objektu a prítomnosti šumu. [43]

3.2.2 Kontúry objektu

Predmety v scéne je možné definovať pomocou ich obrysu, ktorý je výsledkom spájania súvislých bodov, pozdĺž hranice, s podobnými vlastnosťami, napríklad intenzita jasu. Opäť sa vychádza z obrazu hrán (poprípade z prahovaného obrazu), z ktorého sa následne extrahujú kontúry. Tie sú uložené buď ako (x,y) súradnice každého pixelu hrany, alebo primitívne geometrické tvary sú aproximované, čím sa skomprimuje výsledná kontúra. Napríklad v prípade obdĺžnika sú potrebné iba jeho rohy a nie celé hrany.

Obrázok 3.2-3 - Kontúry obrazu [46]

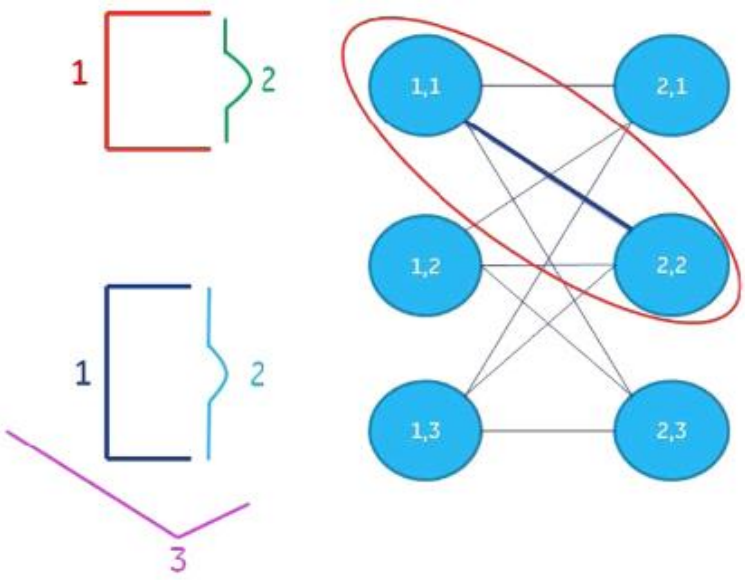
Akonáhle sú z obrazu extrahované kontúry, tak vykazujú rôzne vlastnosti, ktorými sme ich schopní popísať. Medzi základné metriky napríklad patria : momenty – až do 3.rádu - (1), obsah plochy uzavretej kontúrami (2), obvod (3), konvexnosť (4), orientácia (5), hlavná/vedľajšia osa (6) a iné.

Avšak metódy, ktoré vychádzajú z tvaru objektov, musia prekonať viaceré prekážky, ako napríklad zmena pózy/orientácie objektu, zmeny pohľadu na objekt alebo chýbajúce hrany – napríklad kvôli oklúzii. [46]

Keďže kontúry nie je vždy možné uzavrieť, tak v [46] sa detekcia objektu rieši ako hľadanie zhody medzi kontúrami a časťami modelu objektu. Oklúzia alebo chýbajúce kontúry tak nemajú vplyv na detekciu objektu. Algoritmus pozostáva z off-line fázy, v ktorej sa definujú časti modelu (1) a následne segmenty v obraze (2). Potom sa hľadá najbližšia zhoda medzi časťami modelu a segmentami v obraze pomocou vzdialenosti od dotyčnice („tangent distance“). Z toho sa vytvorí graf, kde každý uzol je pár (model, segment) a hrany grafu sú váhované podľa metriky, ktorá určuje podobnosť medzi modelom a segmentom (3). Posledným krokom je hľadanie najlepšej hypotézy pre detegovanie objektu (4).

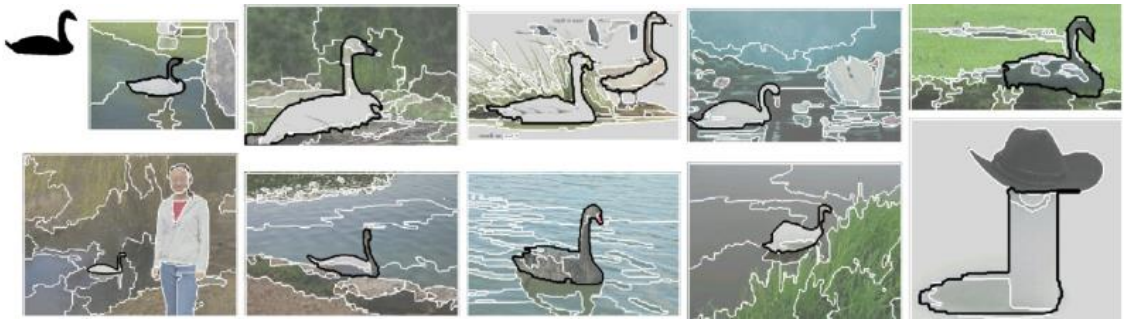
Na obrázku 3.2-3 je viditeľný model objektu, vľavo hore, a pod ním detegované segmenty kontúr, z ktorých sa vytvoril graf na pravej strane. Vidíme, že najsilnejšia hypotéza bola cesta z uzlu (1,1) do uzlu (2,2), čo odpovedá modelu.

Táto metóda teda rieši chýbajúce časti objektu pri rozpoznávaní, avšak pri zmene pózy modelu by bolo nutné generovať toľko modelov, koľko by bolo možných transformácií i keď je metóda invariantná na zmenu merítka, afínna transformácia by sa značila byť ako problematická.



Obrázok 3.2-4 - Ukážka reprezentácie grafu algoritmu [46]

Robustnejšou metódou je detekcia na základe globálneho tvaru kontúr [47]. Pri tomto spôsobe sa v prvom kroku nepoužíva obraz hrán alebo segmentácia, čím sa napríklad mohli vytvoriť neuzavreté kontúry alebo v prípade príliš silného prahovania by mohla nastať strata informácie. Segmentácia pomocou štatistického spájania oblastí rozdelí snímok na oblasti s podobnou charakteristikou, čím sa určia hranice, na základe ktorých je následne možné vytvoriť a hľadať model objektu. Ide konkrétne o *elastický model*, ktorý je definovaný ako pevne daný počet bodov pozdĺž kontúr oblasti. Každý bod je potom charakterizovaný pomocou uhlu dotyčnice, tým pádom je model invariantný voči translácii a zmene merítka. Pri hľadaní podobnosti medzi obrysmi sa vychádza z predpokladu, že začiatočný a koncový bod vektoru bodov modelu sú zhodné a hľadá sa iba najefektívnejšia cesta pozdĺž kontúr, ktorý by tento predpoklad potvrdili. Pričom môžu nastať prípady, keď sa body modelu môžu natiahnuť (v ose X alebo Y) – odtiaľ je názov elastický model. Pre dosiahnutie invariantnosti voči rotácii sa použijú rôzne začiatočné body vektoru, čím sa nájde obrys objektu v rôznej orientácii. Na obrázku 3.2-5 3.2-53.2-5 môžeme vidieť snímok rozdelený pomocou štatistického spájania oblastí a nájdení obrazec modelu.



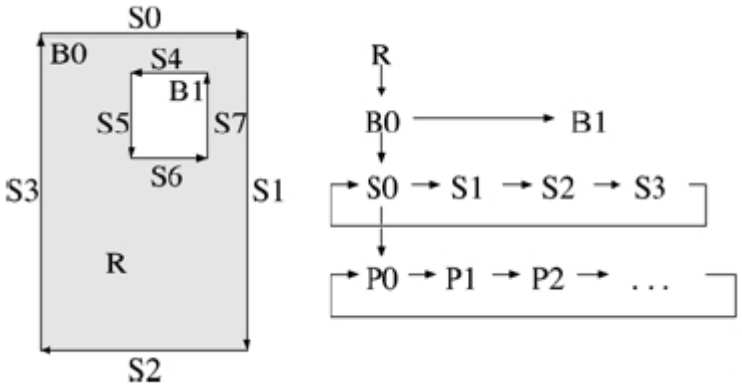
Obrázok 3.2-5 - Výsledky hľadania zhody pomocou elastického modelu [47]

3.2.3 Reprezentácia pomocou tvarov hrán

Rozpoznávanie modelu objektu je možné realizovať aj pomocou 3D senzorov. Tie ponúkajú ďalší rozmer (hĺbku), ktorým je možné objekt definovať. Typy senzorov sú rôzne, napríklad vysokofrekvenčné optické senzory, RGB-D alebo stereo kamery.

Typicky sa však zostava stereo kamier nevyužíva na rozpoznávanie objektov, pretože neposkytuje husté a presné 3D dáta , avšak je možné extrahovať niekoľko vlastností z obrazu, ktoré sme následne schopní využiť [48].

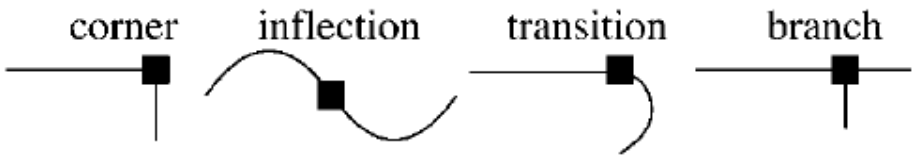
Metóda je založená na extrahovaní 3D hraníc zo stereo obrazu, ktorým musia byť pevne dané hrany. Teda hrany ako rovné priamky, kruhové oblúky alebo krivky. Týmto spôsobom sa obraz reprezentuje ako súbor tzv. B-rep štruktúr – oblasť, hranica, segment a bod. Obdobne je možné tento popis extrahovať z CAD modelu objektu, keďže B-rep je štandard podporovaný v CAD softvéroch (napríklad SolidWorks) [49].



Obrázok 3.2-6 - Datová štruktúra B-rep [48]

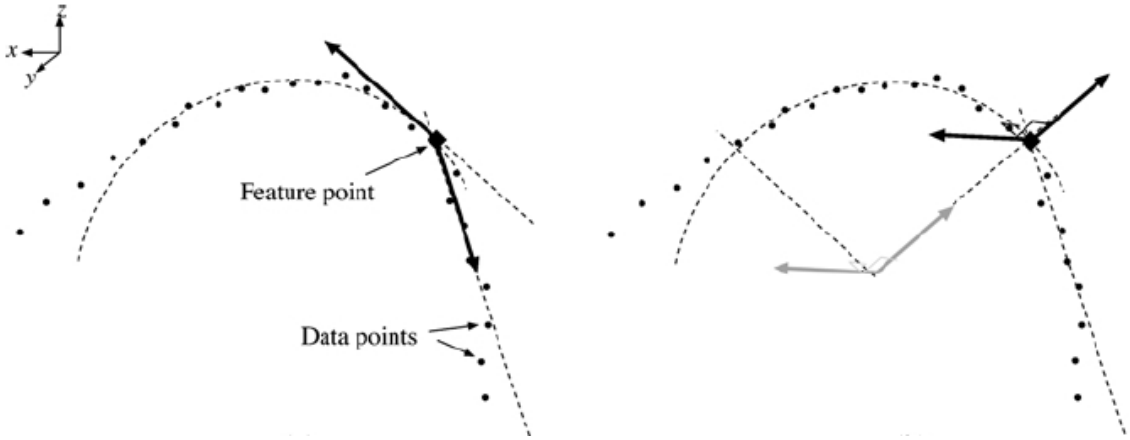
R – kontinuálna plocha segmentovaného obrazu
B – hranica, ktorá obklopuje oblasť (R)
S – priamka alebo krivka, ktorá vznikne segmentáciou hraníc (B) pomocou významných bodov (Obrázok 3.2-7Obrázok 3.2-7)
P – pixel na segmente

Samotná reprezentácia vznikne segmentáciou šedo tónového obrazu do oblastí (R) a segmentáciou hraníc (B) na segmenty (S), ktoré môžu byť rovné, konvexné alebo konkávne. [50]



Obrázok 3.2-7 - Významné body pre segmentíciu hraníc (B) na segmenty (S) [48]

Významné body na hraniciach (BF – boundary features) slúžia na vytvorenie korešpondencie medzi obrazom a modelom objektu, z ktorej je možné stanoviť pozíciu a orientáciu objektu . Ďalej sú definované ako vektorový pár z významného bodu (z ktorého vznikli) - BF-vertex, BF-arc, BF-point.

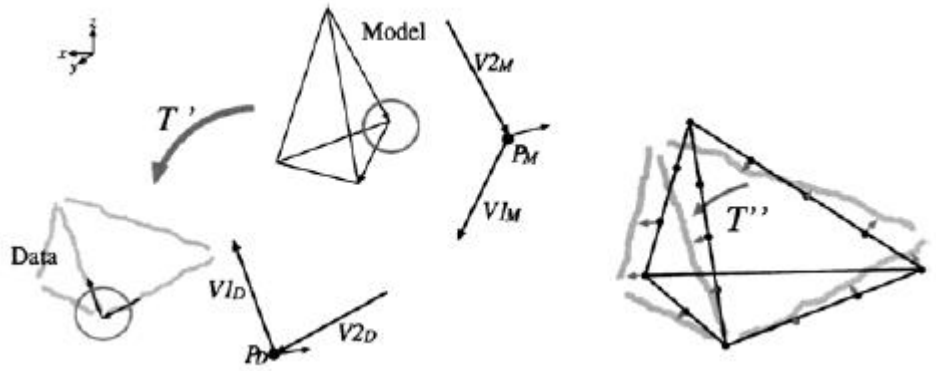


Obrázok 3.2-8 - BF-Vertex (vľavo), BF-Arc (vpravo) [48]

Týmto spôsobom z obrazu vznikne sada geometrických významných bodov, pod termínom dátové (data-vertex, data-arc, data-point). Na popis modelu sa použije rovnaká datová štruktúra BF a vzniknú tak model-vertex, model-arc, model-point. [51]

Vo fáze rozpoznávania objektu, ktorá je zložená z prvotného porovnávania („initial matching“) a úpravy („fine adjustement“), sa hľadá zhoda medzi objektom a modelom pomocou matice rotácie (R) a translačného vektoru (t).

$$\mathbf{T} = \begin{bmatrix} \boldsymbol{R} & & & \boldsymbol{t} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \tag{4.1}$$



Obrázok 3.2-9 - Inital matching (vľavo), Fine adjustement (vpravo) [48]

Metóda je rozšíriteľná na detekciu viacerých objektov v scéne, kde sa vypočítajú transformačné matice (T) nezávisle pre každý objekt.

3.3 Popis pomocou lokálnych príznakov

Pri reprezentácii objektu pomocou lokálnych príznakov sa oproti napríklad reprezentácii kontúrami, objekt popisuje globálne po celej jeho ploche. Získavame tak jeho podrobný popis. Dôležitou vlastnosťou niektorých z nich je invariantnosť na jas, rotáciu, zmenu merítka, čo z nich robí aj robustnú metódu.

Všeobecne je prvým krokom detekcia príznakov, tzv. významných bodov a následne sa hľadá ich korešpondencia s apriori vytvoreným modelom – jeho deskriptormi.

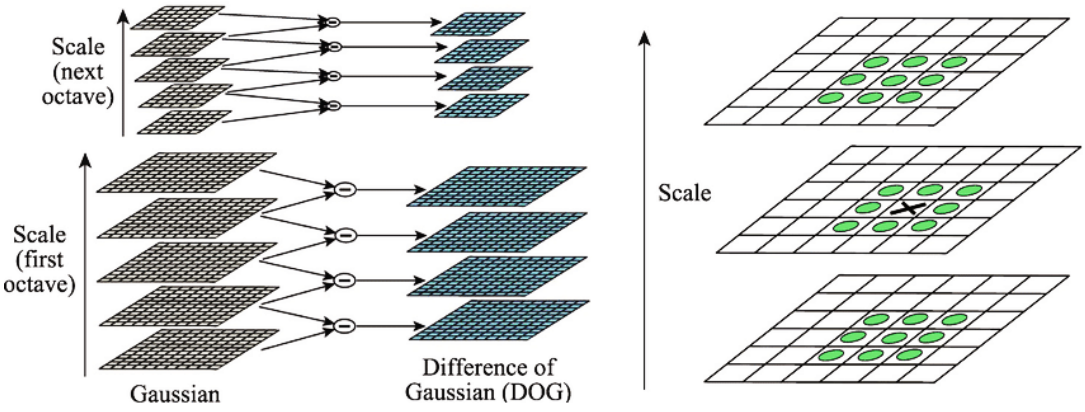
3.3.1 SIFT (Scale-invariant feature transform)

Jeden z prvých algoritmov detekcie, popisu a hľadania korešpondencie významných bodov v obraze. Zároveň je principiálne základným kameňom pre tieto typy algoritmov. Obraz, respektíve objekt, sa popíše pomocou lokálnych vlastností, ktoré sú aspoň čiastočne invariantné zmene jasu, transformáciám (rotation, scaling) a variácii objektov, čo umožňuje jeho detekciu aj v komplexnej scéne [52].

Metóda pozostáva zo štyroch krokov:

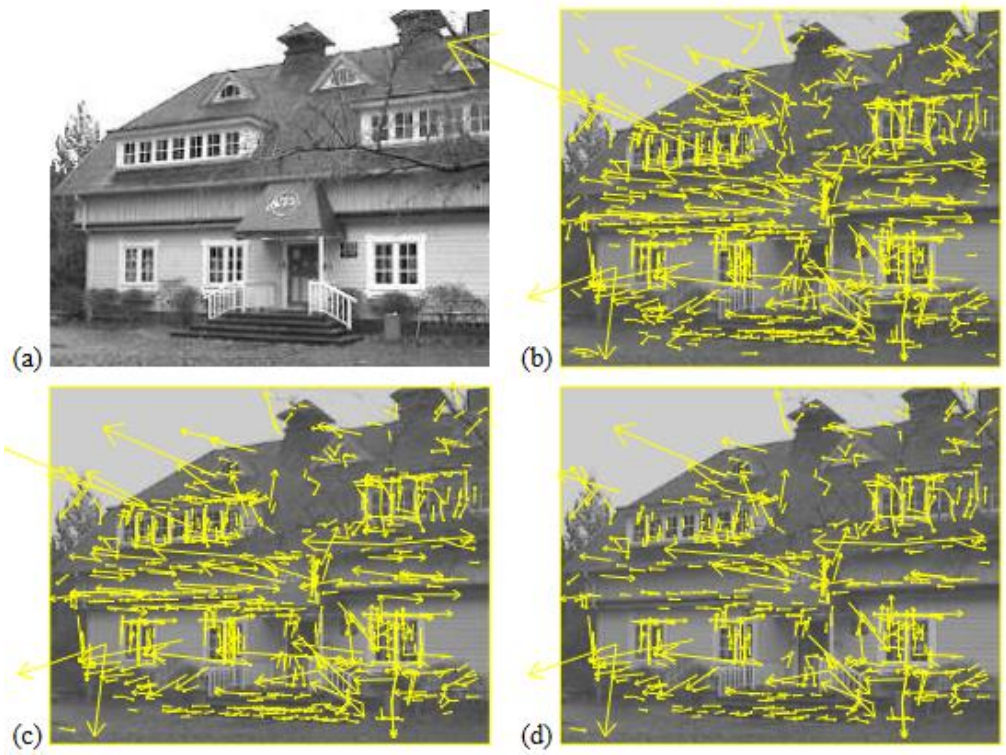
1. Detekcia extrémov v mierke
2. Lokalizácia významných bodov
3. Priradenie orientácie
4. Tvorba deskriptoru významných bodov

V prvom rade sa identifikujú potencionálne body, ktoré sú invariantné voči zmenšeniu/zväčšeniu („scaling“) pomocou rozdielov obrazov po aplikovaní Gaussovej funkcie. Tento proces sa vykoná v rámci každej oktávy, ktorá vzniká podvzorkovaním obrazu predošlej oktávy. Tým sa vytvorí tzv. „scale-space“. Následne sa vypočíta DoG (difference of Gaussian), čo je rozdiel σ_{k_i} a $\sigma_{k_{i+1}}$, kde *k* značí oktávu a *i* výstup po filtrácii Gaussovým kernelom. Potom sa aplikuje nemaximálna supresia a vyberie sa lokálny extrém, teda kandidát. Bod sa označí ako významný v oktáve *k* v DoG obrazu σ_{k_j}, kde *j* značí rozdielový obraz dvoch obrazov zo scale-space, v prípade odlišnosti hodnoty pixelu od ostatných v DoG $\sigma_{k_{j+1}}$ a $\sigma_{k_{j-1}}$.(1).



Obrázok 3.3-1 - Konštrukcia "scale space" (vľavo), hľadanie extrému (vpravo) [53]

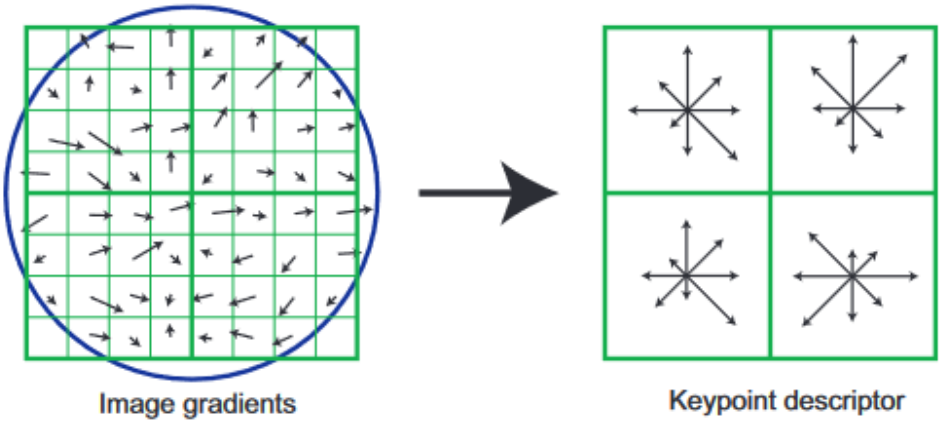
Pri každom kandidátovi sa následne overí jeho stabilita, čiže body ktoré majú nízky kontrast (sú citlivé na šum), alebo sú nesprávne nájdené na hranách, sa zamietnu ako potenciálny kandidát. Na obrázku 3.3-2 vidíme (a) Originálny obraz, (b) Prvotnú lokalizácia významných bodov (832) v extrémoch rozdielov Gaussovej funkcie, (c) Po aplikovaní prahu pre minimálny kontrast (729 bodov), (d) Aplikovanie ďalšieho prahu na eliminovanie odozvy na hranách (536 bodov). (2)



Obrázok 3.3-2 - Selekcia významných bodov [54]

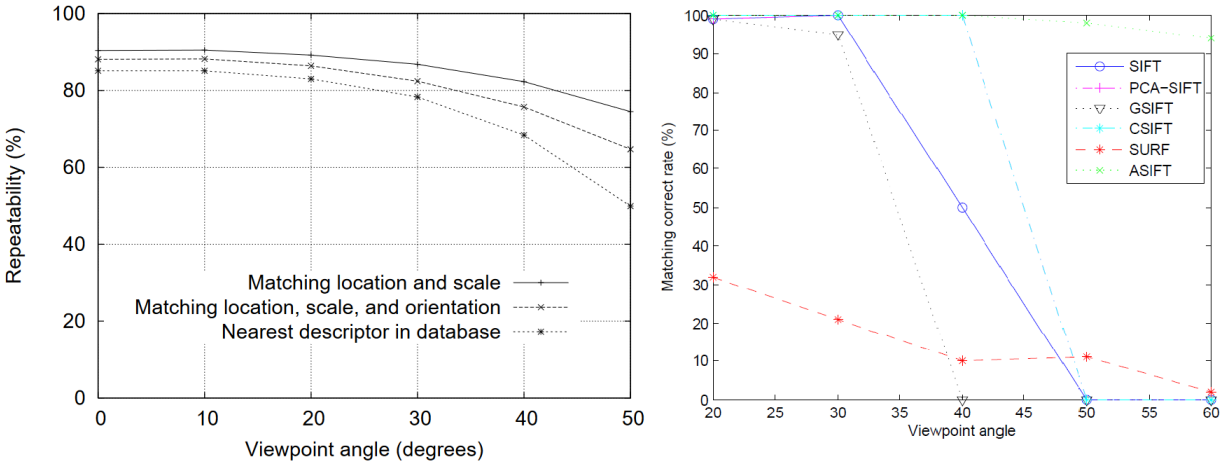
Ďalej sa každému bodu priradí orientácia oproti okolitým vlastnostiam obrazu a tým sa zabezpečí invariancia voči rotácii. (3)

Posledný krok pozostáva z vytvorenia deskriptoru. Zoberie sa okolie významného bodu a určí sa jeho gradient – veľkosť a smer. Následne je gradient vážený pomocou Gaussového okna a výsledok je naakumulovaný do histogramov pre pod-regióny. Napríklad pre vzorku obrazu o veľkosti 8x8 vznikne deskriptor o veľkosti 2x2. Veľkosť deskriptoru sa určí podľa dvoch parametrov, r – počet orientácií v histograme, a n – $n \times n$ matica histogramov orientácií. V predošlom príklade by išlo – n = 2 a r = 8, čím vznikne vektor o veľkosti 2*2*8 = 32 elementov popisujúcich významný bod. Prirodzene čím bude vektor väčší tým, bude jednoznačnejšie ho nájsť vo veľkej skupine iných vektorov a naopak pre menšie vektory rýchlejšia prehľadáva v databázy. V práci [54], ktorá priamo nadväzuje na [52] – kde metóda bola prvý krát popísaná – je odporúčané tvoriť deskriptor o veľkosti 4*4*8 = 128 elementov (kapitola 6.2). Zväčšovaním vektoru sa stane náchylnejším na skreslenie [54]. (4)



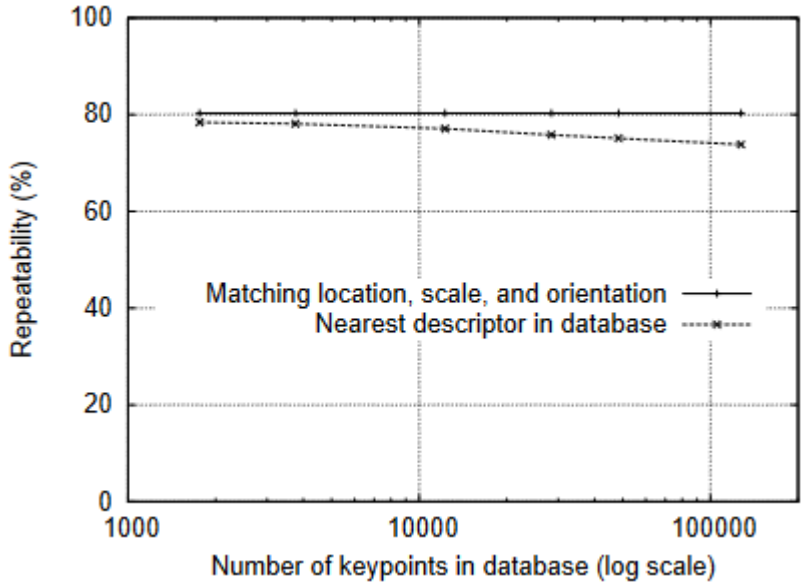
Obrázok 3.3-3 - Vznik deskriptoru zo vzorky obrazu [54]

Ďalším faktorom je do akej miery sú deskriptory invariantné voči afínnej transformácii. Boli prevedené dva rôzne experimenty [54][55], ktoré skúšali vplyv afínnej transformácie na SIFT deskriptory. Na obrázku 3.3-4Obrázok 3.3-4 môžeme vidieť, že zhoda začína výraznejšie klesať so zmenou uhlu pohľadu niekde okolo 30° (rozdiel v zhode vyjadrenej na osiach *y*, sú spôsobené len iným spôsobom, akým sa nachádza najlepšia zhoda medzi deskriptormi a veľkosťou ich databáze).



Obrázok 3.3-4 - Stabilita zhody deskriptorov pri afínnej transformácii.(vľavo [54], vpravo [55])

Vplyv veľkosti databázy je znázornený na obrázku 3.3-5Obrázok 3.3-5. Na osi *x* v logaritmickej mierke je počet významných bodov (vektorov o 128 elementov). Je viditeľné, že zvyšovaním počtu významných bodov sa znižuje presnosť alebo opakovateľnosť nájdenia zhody. Môžeme usúdiť, že toto je ďalší faktor nutný brať do úvahy, s čím je úzko spojená aj rýchlosť nájdenia zhody (pri prehľadávaní databázy).



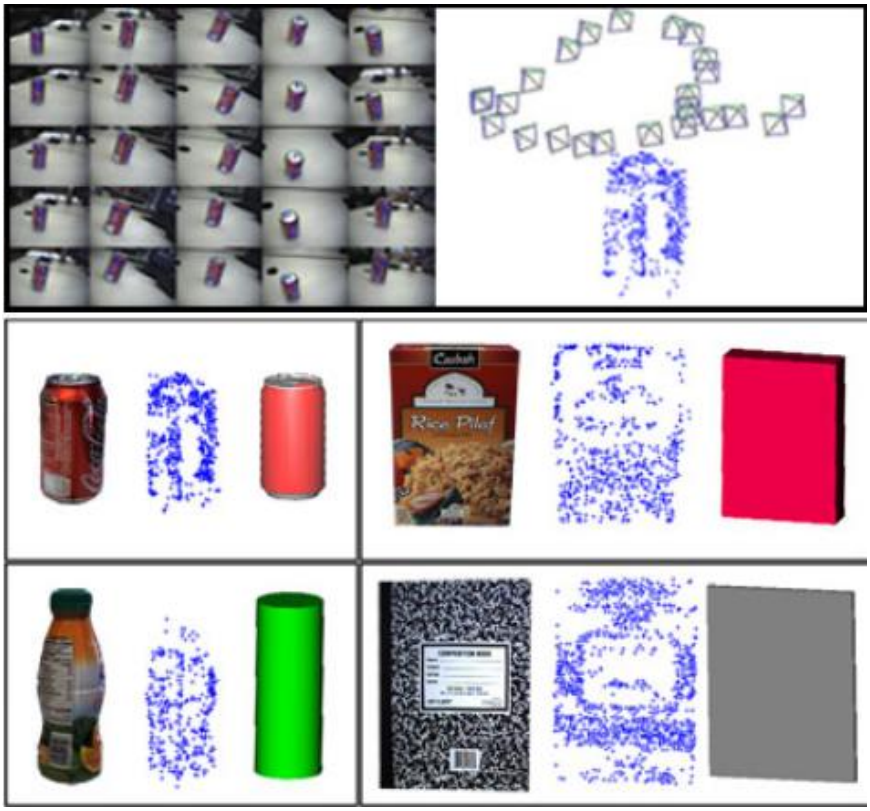
Obrázok 3.3-5 - Vplyv veľkosti databáze na úspešnosť zhody [54]

Zhoda medzi nájdeným deskriptorom a deskriptorom z tréningovej množiny sa nájde pomocou najbližšieho suseda („nearest neighbor“). Ten je definovaný ako významný bod s najmenšou Euklidovskou vzdialenosťou. Na efektívne vyhľadávanie najbližšieho suseda sa používa BBF (Best-Bin-First) algoritmus. [54]

Pri implementácii, napríklad pomocou OpenCV, je možné využiť dve stratégie hľadania zhody. Prvá je tzv. „Brute force“ algoritmus, ktorý je vhodný iba pre menšie datasety, keďže sa deskriptor porovnáva so všetkými ostatnými. Naopak FLANN algoritmus obsahuje kolekciu optimalizovaných algoritmov pre hľadanie najbližšieho suseda vo veľkých datasetoch s vysokou dimenzionalitou. [56] [57]

Všeobecne SIFT je robustná metóda na rozpoznávanie objektov. Pokiaľ je scéna príliš komplexná alebo obraz je priveľký, tak na zrýchlenie algoritmu sa v obraze nájdu ROIs – „regions of interest“. Napríklad podľa farby objektu sme schopní sa zamerať iba na časť obrazu a hľadať najlepšiu zhodu tam. [58]

Čo sa týka 3D rozpoznávania, tak na vytvorenie modelu je postup podonbý ako na obrázku 3.1-1, kde sa vytvorí model pomocou niekoľkých snímkov z rôznych pohľadov, čím vzniká tzv. „sparse model“. Následne sa hľadá optimálna transformácia (R, t) modelu do snímku.



Obrázok 3.3-6 - Tvorba "sparse" modelu [59]

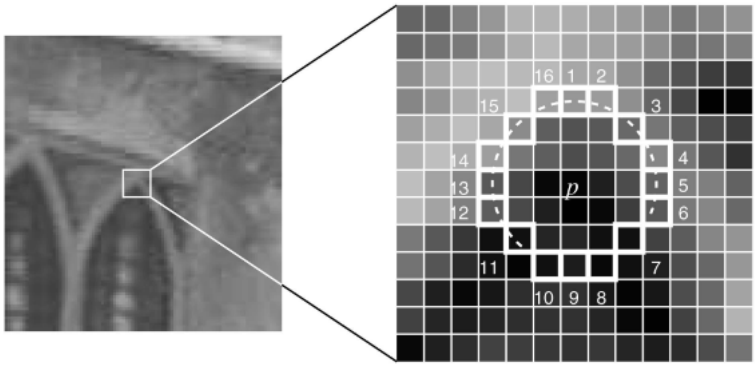
Obdobne je možné skombinovať SIFT so stereo kamerou na vytvorenie robustnejšej metódy detekcie objektu a jeho pózy [60]. Iný spôsob vymedzuje použitie algoritmov strojového učenia, napríklad SVM na vyriešenie rozpoznania orientácie objektu bez apriori vytvorenia modelu [61].

3.3.2 ORB (Oriented FAST and Rotated BRIEF)

Jedna z alternatív k SIFT je metóda ORB [62], ktorá vznikla kombináciou FAST [63], detektoru významných bodov, a BRIEF [64] deskriptoru. Cieľom bolo umožniť nasadenie algoritmov tohto charakteru na embedded zariadenia alebo pre real-time aplikácie, v prípade ak nie je možné použiť akcelerátor (GPU, FPGA, ...).

FAST detektor sa zameriava na detekciu rohov v obraze. Motivácia bola rovnaká ako pre celú metódu ORB, teda schopnosť detekcie a ďalšieho spracovania na úrovni rýchlosti získavania obrazu („*at frame rate*“). Dôležitým parametrom bola taktiež opakovateľnosť detekcie z rôznych pohľadov na objekt.

FAST - Features from Accelerated Segment Test – je test, ktorý berie do úvahy 16 pixelov z okolia arbitrárneho bodu *p*. Kandidát sa začne klasifikovať pokiaľ v rámci tohto okolia existuje kontinuálny blok *n* pixelov, ktoré majú nižšiu alebo vyššiu jasovú (+ threshold) intenzitu ako bod *p*. [63]



Obrázok 3.3-7 - Ukážka segment testu. 16 pixelov z okolia bodu *p* a následné hľadanie kontinuálneho bodu [63]

Kandidát sa považuje za významný bod (roh), ak má nižšiu alebo vyššiu jasovú intenzitu ako *n* (zvyčajne 9 alebo 12) pixelov okolia. Poradie testovania jednotlivých pixelov bolo zistené empiricky pomocou rozhodovacieho stromu ID3. Ďalším krokom je priradenie orientácie detegovanému významnému bodu zistením dominantného gradientu v jeho okolí.

Druhá časť detektoru pozostáva z **BRIEF** - Binary Robust Independent Elementary Features. Ide o bit string deskriptor, ktorý uvažuje jasovú intenzitu pixelových párov v okolí významného bodu. Každý významný bod je teda popísaný bit string deskriptorom o dĺžke 128-512 bitov [64]. Jeho výhodou je práve rýchlosť hľadania zhody. Pre metódu ORB bol BRIEF modifikovaný na tzv. rBRIEF, ktorý je už invariantný voči rotácii, tým pádom ORB spĺňa rovnaké parametre invariantnosti (zmena merítka, rotácia,...) ako SIFT.

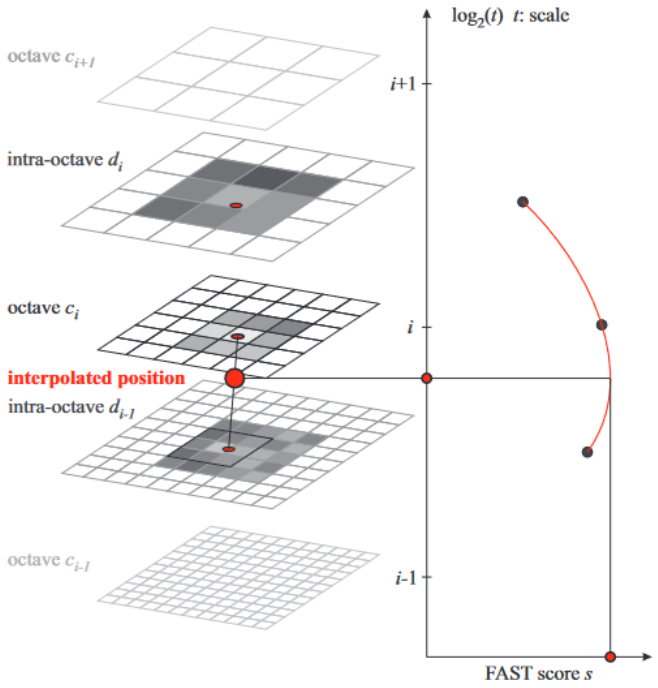
Nasadenie ORB prezentujú práce [65][66], v ktorých sa účelovo porovnáva práve so SIFT, a inými metódami. Cieľové zariadenie, alebo aplikácia, je vždy limitované výpočtovou silou a zároveň je snaha dosiahnuť rozumný počet snímkov za sekundu.

3.3.3 BRISK (Binary Robust invariant scalable keypoints)

Myšlienkou metódy BRISK je nájsť kompromis medzi vysokou kvalitou deskriptoru a nízkou výpočetnou náročnosťou algoritmu. Jedná sa o binárny deskriptor, takže umožňuje rýchle porovnávanie deskriptorov a taktiež využíva SIMD (single instruction, multiple data) model pre urýchlenie výpočtov.

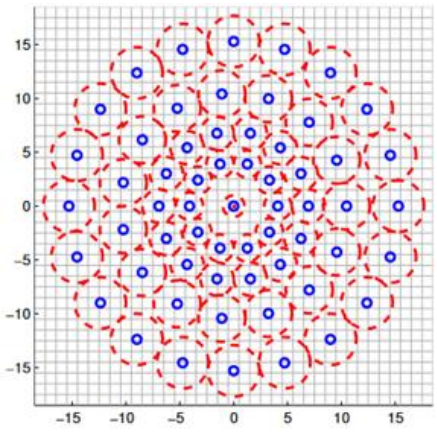
Prvým krokom je detekcia významný bodov v tzv. „scale-space“. Ten predstavuje pyramídu s *n* vrstvami a *n* medzivrstvami. Vrstvy sú tvorené podvzorkovaním obrazu, v tomto prípade na polovicu a medzivrstvy vznikajú rovnakým spôsobom. Zápis pre

veľkosť by bol nasledovný - ak *t* indukuje veľkosť tak t(c_i) = 2^i and t(d_i) = $2^i \cdot 1.5$, kde *c* reprezentuje vrstvy, *d* medzivrstvy a *i* je krok. [67]



Obrázok 3.3-8 - Detekcia významný bodov v "scale-space" [67]

Pre detekciu sa používa **FAST**, takže rovnako ako v prípade ORB sa uvažuje okolie 16 pixelov a hľadá sa aspoň 9 za sebou idúcich pixelov s nižším alebo vyšším jasom ako pixel *p*. Tento proces sa aplikuje na každú vrstvu a medzivrstvu pyramídy, tak aby sa identifikovali oblasti záujmu. Následne uvažovaný bod musí splniť požiadavku FAST v rámci okolia 8 pixelov, rovnako to platí pre hornú a dolnú (medzi-)vrstvu. Ďalej sa pre každý významný bod definuje kruhový vzor, pomocou ktorého sa vypočíta jeho orientácia, na základe distribúcie jasovej intenzity. [67]



Obrázok 3.3-9 - Príklad použitého kruhového vzoru [67]

Aplikáciou kruhového vzoru na významný bod v rôznych veľkostiach a rotáciách sa zabezpečí jeho invariantnosť. Výsledný deskriptor, o dĺžke 512 bitov, sa získa pomocou porovnávania jasových hodnôt v kruhovom vzore. Výhodou bitového deskriptoru je pri hľadaní zhody v databáze. Na porovnávanie sa používa Hammingová vzdialenosť (namiesto Euklidovej v prípade napríklad SIFT), ktorá sa vypočíta cez bitový XOR a rozdielu počtu bitov, čo na moderných procesoroch býva jedna inštrukcia.

Využitie BRISK je napríklad aj v oblasti AR [68], kde je potrebné nájsť daný objekt v obraze, v tomto prípade prekryť plochu objektu vlastným 2D obrazom.

3.3.4 KAZE (Kintinuous-Accelerated-Zero)

Prístup KAZE je podobný ako v prípade SIFT, kde sa v prvom kroku vypočíta tzv. „scale space“. Oproti SIFT sa však pracuje s rovnakým rozlíšením obrázku bez pod vzorkovania v každej oktáve a zároveň sa nepoužíva Gaussov filter, ale nelineárne difúzne filtrovanie („nonlinear diffusion filtering“). V porovnaní s Gaussovým filtrom sa zachovajú dominantné hrany v obraze [69].

Obrázok 3.3-10 - Porovnanie použitia Gaussového filtra (hore) a nelineárneho difúzneho filtrovania (dole) [69]

V prvom rade sa (paradoxne) použije Gaussov kernel na zníženie šumu a určí sa histogram gradientu spolu s parametrom *k*, ktorý sa používa na vytvorenie „scale space“. Na lokalizáciu významného bodu sa postupuje obdobne ako pri SIFT, určuje sa lokálne maximum naprieč úrovňami „scale space“. [69]

Popis významného bodu, deskriptor, vychádza z metódy SURF, v ktorej sa určí dominantná orientácia významného bodu v kruhovom okolí. Pre každý bod sa priradí jeho prvá derivácia v oboch osiach, ktoré sa následne váhujú Gaussovou funkciou. Vytvorí sa tak vektor s orientáciou a veľkosťou odozvy pre každý bod v okolí. Na stanovenie dominantnej orientácie sa použije kruhová časť o veľkosti $\frac{\pi}{3}$ v okolí kandidáta, v ktorej sa sčítajú odozvy predošlej derivácie. Najväčšia suma určí, ktorý vektor je dominantný. Pre každý bod sa tak priradí gradient orientácii, čím sa zaistí invariancia na rotáciu. [69]

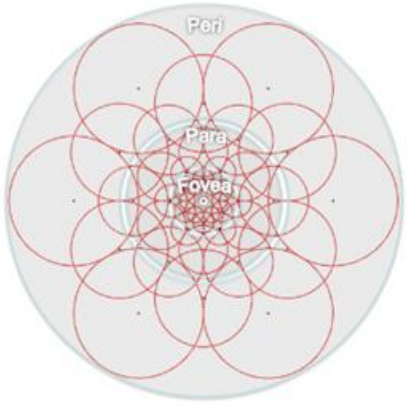
Samotný deskriptor sa tvorí z okolia $24\sigma_i \times 24\sigma_i$ pixelov, kde σ_i je úroveň v „scale space“, na ktorej sa bod detegoval. To sa ďalej rozdelí na subregióny, v ktorých sa pomocou odozvy derivácie, kde odozvy bližšie k bodu majú vyššiu váhu, definuje deskriptor pre daný subregión. Získajú sa tak 4 hodnoty pre subregión. Kombináciou hodnôt z každého subregiónu sa vytvorí deskriptor s dĺžkou 64 floatov. [63]

Existuje aj rozšírenie tejto metódy pod názvom **AKAZE** (‘A‘ symbolizuje „accelerated“). Jedná sa hlavne o optimalizáciu algoritmov pri tvorení „scale space“ (použitie Fast Explicit Diffusion (FED)), hľadaní významných bodov a kalkulácii deskriptoru. [70]

3.3.5 FREAK (Fast Retina Keypoint)

Fast Retina Keypoint (FREAK) sa, podobne ako BRISK alebo ORB, zameriava na možnosť využiť korešpondenčné algoritmy v real-time aplikáciách, najmä v mobilných zariadeniach. Práve tu je dôležité implementovať algoritmy, ktoré sú pamäťovo a výpočetne menej náročné. Cieľom je teda získať deskriptor, a jeho korešpondenciu za minimálny čas, ktorý zároveň disponuje robustnosťou voči zmene veľkosti, rotácii a šumu.

Metóda je inšpirovaná ľudskou sietnicou, kde pre určenie významného bodu sa používa kruhová vzorkovacia sieť s vyššou hustotou bodov v okolí arbitrárneho bodu. Napríklad BRISK používa podobný princíp, pri ktorom body sú stredmi kružníc, ktoré sú od seba rovnako vzdialené. V tomto prípade, sú body hustejšie v okolí bodu a polomery kružníc sa zároveň exponenciálne menia a prekrývajú.

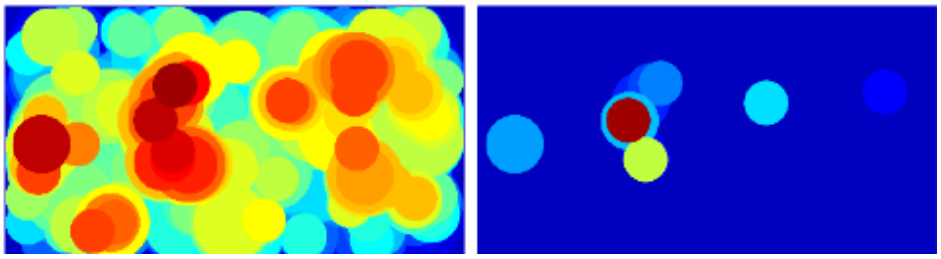


Obrázok 3.3-11 - Kruhový vzor pre vzorkovanie FREAK metódy [71]

Je diskutované, že zmenou veľkosti kružníc sa zlepšujú výsledky významných bodov a ich vzájomné prekrývanie (aj keď sa môžu získať redundantné informácie) dovoľuje použiť menšie množstvo bodov/kružníc. [71]

Deskriptor sa zostaví rozdielom výsledku prahovania pomocou Gaussovho kernelu medzi párom kružníc. Týchto párov môže vzniknúť obrovské kvantum, takže na filtráciu párov sa používa spôsob podobný ako u ORB. Pomocou priemeru sa extrahujú páry s najväčšou variáciou a nakoniec sa pridajú ešte ďalšie páry, ktoré majú najnižšiu koreláciu s ostatnými. Týmto vznikne deskriptor o veľkosti 512 prvkov.

Hľadanie zhody sa skladá z dvoch krokov. Najprv sa uvažuje nad prvými 16 bajtami, ktoré eliminujú 90% kandidátov, čím sa estimuje potenciálna oblasť kde sa objekt nachádza (1). Potom sa kaskádovo porovnáva zvyšok deskriptoru medzi jednotlivými kaskádami (2). [71]



Obrázok 3.3-12 - Vzdialenostná mapa pre prvú kaskádu (vľavo) a poslednú kaskádu (vpravo) hľadania zhody [71]

Použitie v rozšírenej realite sa uplatnilo pri úlohe registrácie virtuálnych objektov [72], kde namiesto značiek („marker“) sa využili obrazce popísané pomocou FREAK a riešením homografie sa odhadla póza kamery voči obrazcu.

3.4 Strojové učenie – metódy založené na CNN

V prípade, ak je k dispozícii hrubá výpočtová sila, je možné nasadiť algoritmy hlbokého učenia na dosiahnutie výsledkov v reálnom čase. Najmä možnosťou paralelizácie výpočtov na GPU, CPU, TPU, a na iných dedikovaných zariadeniach, a taktiež do veľkej miery zdokonalením algoritmov hlbokého učenia. Práve modely hlbokého učenia tvoria kostru algoritmov na rozpoznávanie objektov, hlavne konvolučné neurónové siete (CNN). V tomto smere sa vyslovene nejedná o popis objektu pomocou jeho vlastností, ale skôr o tvorbu modelu, ktorý svojou architektúrou a parametrami popisuje daný objekt.

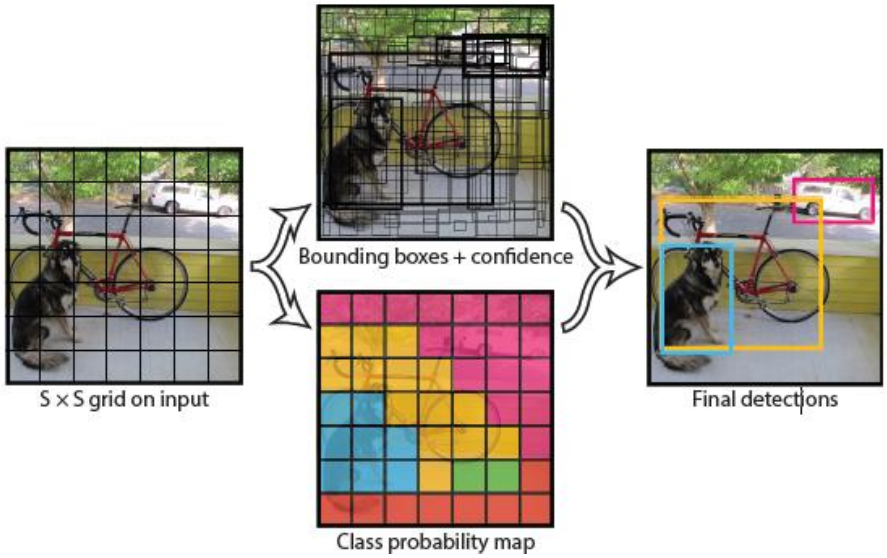
Existujúce detektory môžu byť rozdelené do dvoch kategórií a to jednostupňové a dvojstupňové. Prvé spomenuté majú inherentne rýchlejšiu schopnosť detekcie a naopak druhé majú vyššiu presnosť a lokalizáciu detegovaných objektov. [73]

3.4.1 Jednostupňové detektory

Pri tomto type detektorov sa pracuje priamo nad celým vstupným obrazom. Miesta kde sa objekt môže nachádzať sú vopred pevne dané. Tento spôsob urýchli proces detekcie, keďže sa nepredikujú potencionálne oblasti s objektom, avšak za cenu presnosti.

YOLO algoritmus („You-only-look-once“) rieši problematiku detekcie objektu ako jeden regresný problém, v ktorom sa, zároveň, predikujú tzv. „bounding boxes“ – BBs - (oblasť kde sa objekt nachádza) a pravdepodobnosti klasifikácie do jednotlivých tried. Vstupný obraz sa tak spracuje iba jedenkrát. Rovnako sa uvažuje nad celým obrazom, takže pri trénovaní sa zakomponujú aj kontextové informácie. Oproti R-CNN, ktorý patrí do kategórie dvojstupňových detektorov, má polovičnú chybovosť kvôli pozadiu [74].

Najprv sa obraz rozdelí na S*S mriežku (obrázok 3.4-1 vľavo), kde každá bunka má za úlohu detegovať objekt, ak sa jeho centrum nachádza v bunke. Ďalej každá bunka predikuje BBs a ich istotu. Rovnako bunky predikujú podmienenú pravdepodobnosť tried, teda pravdepodobnosť, že sa v bunke nachádza objekt obrázok 3.4-1Obrázok 3.4-1 (obrázky v strede). Nakoniec sa estimujú finálne klasifikácie – pravdepodobnosť triedy, a predikcia či oblasť obsahuje objekt.

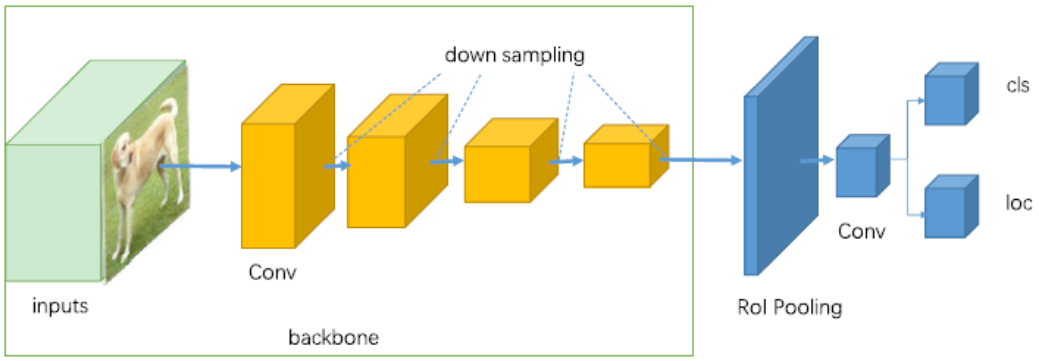


Obrázok 3.4-1 – Postupnosť YOLO [74]

Výhodou YOLO je dosiahnutie real-time rýchlosti pri rozpoznávaní objektov, za cenu presnosti a správnej lokalizácie v obraze. Obecne YOLO deteguje objekty v 2D rovine, nemá teda znalosť o hĺbke obrazu, ale napríklad kombináciou s detekciou roviny [20] je možné estimovať pozíciu objektu v priestore.

SSD (single shot detector) je detektor, ktorý je schopný rozpoznať objekty už pri prvom priechode architektúrou. V prvotnej fáze sa predikujú BBs, tentokrát však, o rôznych veľkostiach a v rôznych príznakových mapách. Týmto spôsobom je detektor schopný zachytiť objekty rôznej veľkosti. Následne sa súbežne predikuje ofset pre BBs v rámci lokalizácie objektu a pravdepodobnosť klasifikácie do danej triedy. V konečnej fáze sa odstraňujú nadbytočné BBs, čím sa zvýši presnosť predikcie.

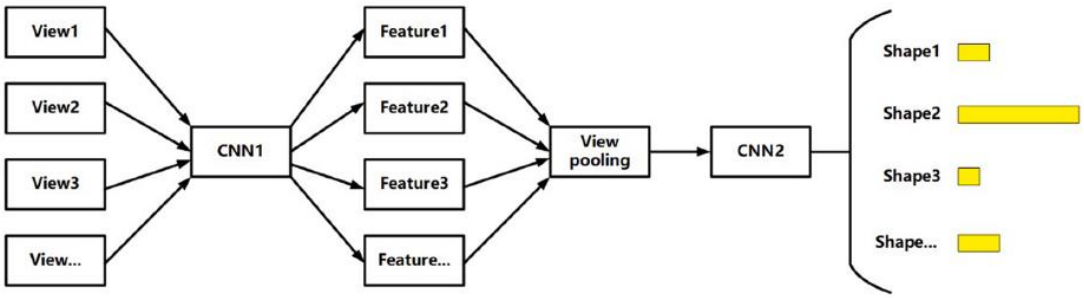
V prípade, ak scéna je komplexná a je nutné detegovať viac objektov o rôznej veľkosti, tak pre učenie CNN je SSD vhodný kandidát [42].



Obrázok 3.4-2 - Generický jednostupňový detektor [73]

Pri rozpoznávaní objektu býva dôležitá aj póza, teda natočenie objektu oproti arbitrárnemu bodu (napríklad kamera). Keďže sa jedná o hlboké učenie, dôležité je získať dostatočný dataset ako tréningovú množinu. Ak ide o priemyselnú aplikáciu, tak sa môže použiť CAD model objektu a spomínaný dataset vytvoriť synteticky alebo pomocou skriptu [42][75].

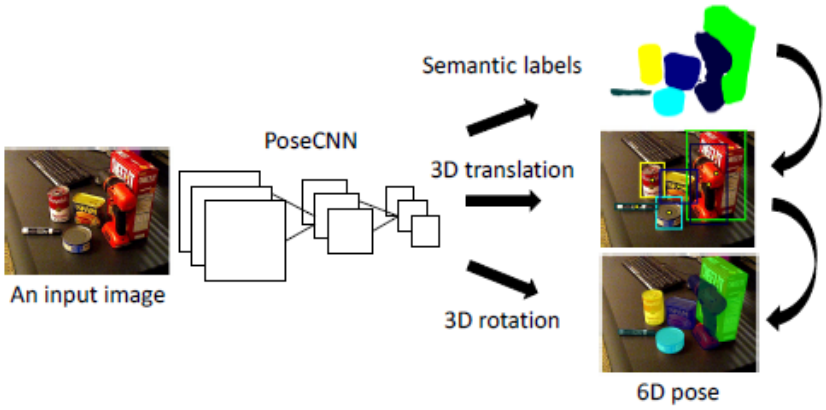
Ďalej na detekciu pózy sa použije už konkrétna architektúra CNN siete, uspôsobená na túto úlohu. Napríklad MVCNN používa ako jadro CNN, ktorá exceluje v klasifikácii, takže sa použije na začiatku pre extrahovanie vlastností a na klasifikáciu na konci.



Obrázok 3.4-3 - Architektúra MVCNN [76]

Vstupom sú teda pohľady na konkrétny objekt, kde každý vygeneruje svoj vlastný vektor príznakov. Nasleduje fúzia týchto vektorov príznakov do jedného globálneho [76], na základe ktorého sa klasifikuje objekt – trieda a póza.

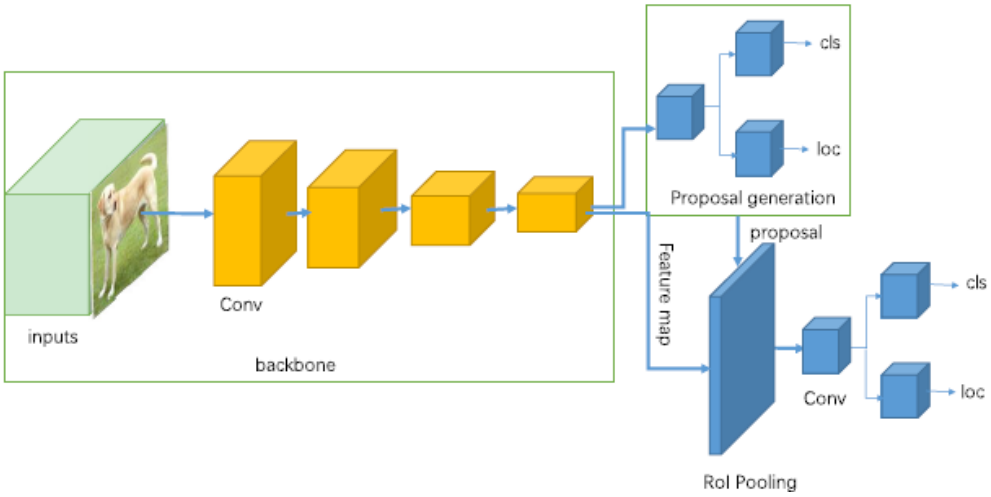
Iný rámcový model, PoseCNN, sa sústredil na rozdelenie estimácie pózy na pod-úlohy. Prvou je predikcia objektu pre každý pixel (1), ďalej estimácia centroidy objektu na základe predikcie vektoru z každého pixelu (2) a rovnako rotáciu a transláciu objektu voči kamere (3). Výsledkom je metóda, ktorá je schopná detegovať orientáciu objektu bez znalosti hĺbky obrazu. [77]



Obrázok 3.4-4 – Charakteristika Pose CNN [77]

3.4.2 Dvojstupňové detektory

Dvojstupňové detektory sú založené na tzv. „region based“ CNN detektoroch (R-CNN). Namiesto toho aby sa architektúra zaoberala celým obrazom, ktorý už prešiel cez konvolučné vrstvy, tak RPN (region proposal network) navrhne oblasti, kde by sa objekt mohol nachádzať.

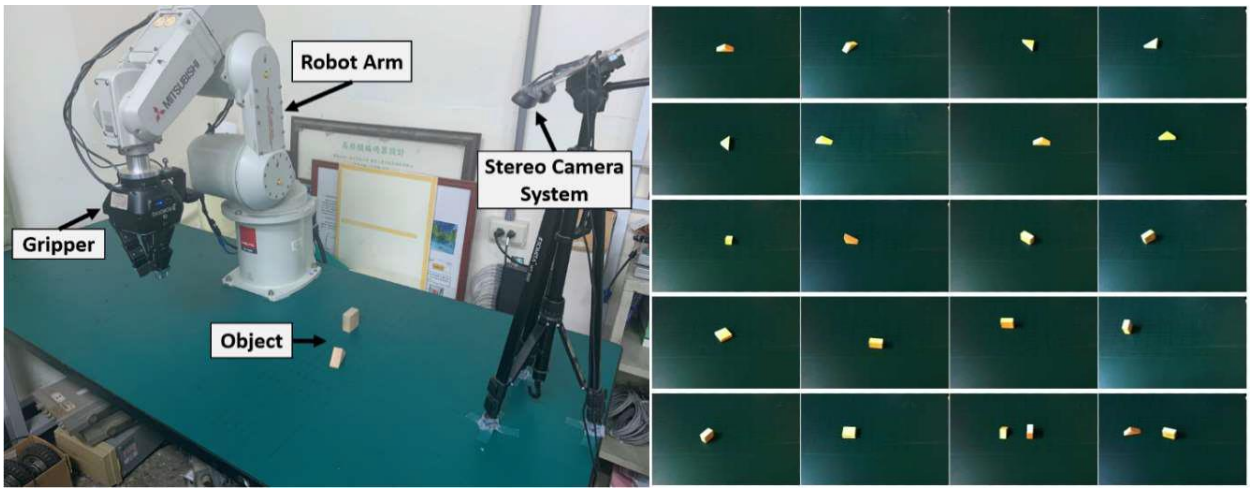


Obrázok 3.4-5 - Generický dvojstupňový detektor [73]

Konvolučné vrstvy najprv vyfiltrujú pomocou konvolúcie z obrazu vlastnosti, napríklad geometrické primitívy, teda z komplexného obrazu vznikajú jeho jednoduché reprezentácie. Po každej operácii konvolúcie sa aplikuje aktivačná funkcia, zvyčajne ReLU, aby sa napríklad zachovala nelinearita. Ďalšia vrstva, „pooling layer“, zabezpečuje zmenšenie obrazu, čím sa zníži výpočetná náročnosť. Pôvodný obraz sa teda zmenší, avšak dôležité vlastnosti sa zachovajú. Vo finále sa obraz transformuje na 1D vektor, na ktorom plne-prepojená vrstva („fully-connected layer“) tvorí predikcie. Pri R-CNN sa tento postup aplikuje pre každý navrhnutý región.

V praktickom príklade [78] sa R-CNN použila v systéme manipulácie s objektami pomocou robotickej ruky. Scéna sa skladala zo stereo kamery, objektov a robotickej ruky.

Model objektu sa vytvoril vyhotovením snímkov z rôznych pohľadov, čo vlastne bolo použité ako tréningová množina. Pre zjednodušenie detekcie sa z obrazu extrahovali/určili regióny s objektami pomocou farby (HSV). Prostredníctvom stereo kamery a triangulácie sa určila poloha objektu voči kamere v scéne. Robotická ruka tak bola schopná presne uchopiť predmet a uložiť ho na potrebné miesto.



Obrázok 3.4-6 - Použitá zostava (vľavo), tvorba modelu pre R-CNN (vpravo) [78]

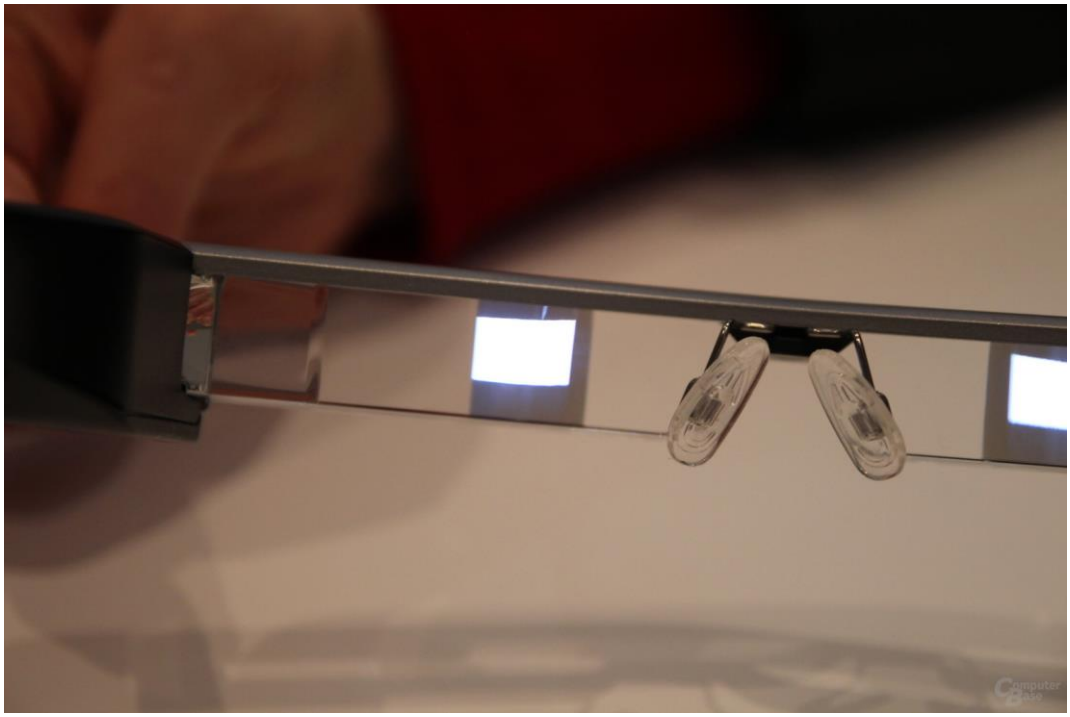
4. ZVOLENÉ RIEŠENIE

V druhej kapitole boli opísané aktuálne zariadenia pre XR, ktoré by mohli byť potencionálne súčasťou riešenia. Avšak kvôli relatívne zložitej dostupnosti zariadení, čo sa týka napríklad ceny, nie je možné vyskúšať a porovnať ich všetky. Podstatou bolo zvoliť riešenie, ktorým by sme boli schopní zrealizovať navrhovaný koncept (1) za predpokladu zachovať pre operátora prirodzený vnem – stereo videnie (2) a snažiť sa rešpektovať komfort pri práci operátora s XR systémami (3). Je preto potrebné zabezpečiť:

1. Kompatibilné časti systému
2. Autenticitu
3. Schopnosť dlhodobo používať zariadenie

4.1 Porovnanie a voľba XR zariadení

Okuliare Moverio BT-45CS sa používajú už v dnešnej dobe na výpomoc pracovníkov v priemyselných zriadeniach. Príkladom je vzdialená podpora (remote-assistance), pomocou ktorej sa pracovník spojí s expertom na diaľku a dostáva inštrukcie. Video-hovor sa zobrazuje priamo do okuliarov, takže operátor je schopný sledovať a plniť jeho pokyny[3].



Obrázok 4.1-1 - Pohľad na zobrazenie na okuliaroch Moverio BT-300

[3] https://www.youtube.com/watch?v=w9hwhBHbOCU

Avšak problémom ktorý som vnímal, bol nedostatočne veľký displej na zobrazovanie prenosu kamery s rozšíreným obsahom (obrázok 4.1-1), relatívne vysoká váha, čo spolu s faktom, že samotný displej nie je opticky komfortný, sa prikláňajú k výsledku, že použitie tohoto zariadenia nie je vhodné pre navrhovanú aplikáciu.

Keďže okuliare Meta Quest 2 sú VR okuliare, bolo nutné nájsť spôsob, ktorým by sa zobrazoval reálny svet priamo do nich. Prakticky by išlo o snímací systém, ktorý by prenášal obraz, ktorý by sa následne zobrazil do zorného poľa užívateľa. Statická konštalácia nedáva zmysel, keďže sa pracovník môže voľne pohybovať. Zaujímavým riešením je stereo kamera ZED-mini, ktorá je rozmerovo dostatočne malá, aby sa pripevnila na VR headset [79] [80], má solídnu podporu od výrobcu a je kompatibilná s rôznymi zariadeniami (Meta Quest 1/2, HTC Vive, Vive Pro) a nástrojmi (Unity, Unreal Engine, OpenCV).

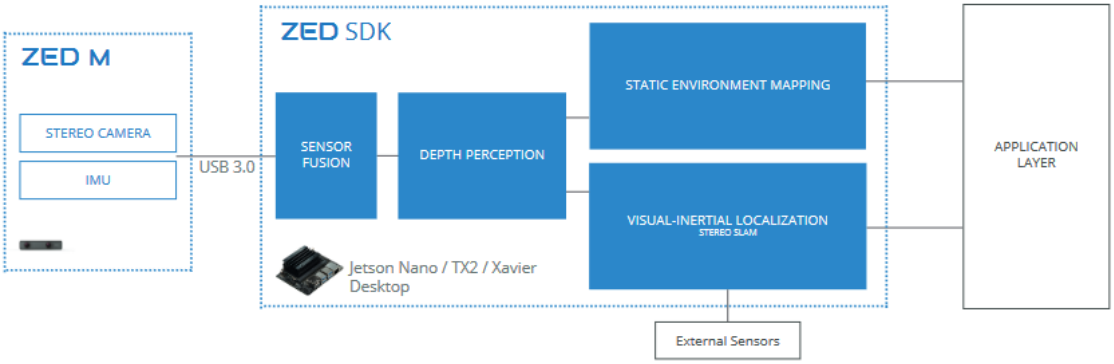
Obrázok 4.1-2 - Kamera ZED-mini na VR headsete [81]

Okuliare typu RealWear a Vuzix nespĺňajú parametre ako FOV alebo rozlíšenie, ktoré by boli nedostačujúce pri montáži výrobku. Pre Microsoft Hololens 2 je zas faktorom dostupnosť, teda cena, oproti iným okuliarom. HTC Vive Pro 2 sú rovnako kompatibilné so stereo kamerou ZED-min, avšak mínusom je nutnosť použitia „basestations“, ktoré limitujú rozsah prostredia a komfort. XReal okuliare by boli vhodným kandidátom, najmä vďaka ich nízkej váhe, ale keďže je podpora pre vývojárov v NRSDK značne obmedzená, nebolo by pravdepodobne možné vytvoriť konečnú aplikáciu so všetkými prvkami.

Zvoleným riešením je teda kombinácia Meta Quest 2 a ZED-mini. Výhodou je možnosť vývoja bez použitia okuliarov, zároveň sa jedná o autentické riešenie z pohľadu zobrazovania. Nevýhodou sa môže zdať nezanedbateľná váha okuliarov pre dlhodobé používanie. Avšak v porovnaní s ostatnými alternatívami, v rámci dostupnosti a potrebných parametrov, sa aktuálne jedná o vhodné riešenie.

4.2 ZED-mini

Ako bolo spomenuté v predošlej kapitole, ZED-mini je stereo kamera , ktorá je určená na kombináciu s headsetmi XR. Najmä vďaka jej rozmerom a veľkosťou báze šošoviek, ktoré kopírujú IPD **(**Interpupillary Distance) človeka. Keďže sa jedná o stereo kameru (RGB 4 MP), sme schopní získať aj hĺbku obrazu. FOV značí 90° v horizontálnej, 60° vo vertikálnej a 100° v diagonálnej rovine. Rozlíšenie je závislé na FPS, napríklad pre 60 FPS je rozlíšenie pre obe kamery 1280x720. Rovnako disponuje akcelerometrom a gyroskopom, takže umožňuje 6DOF trekovanie. Kameru je možné pripojiť cez USB-C do externého zariadenia.



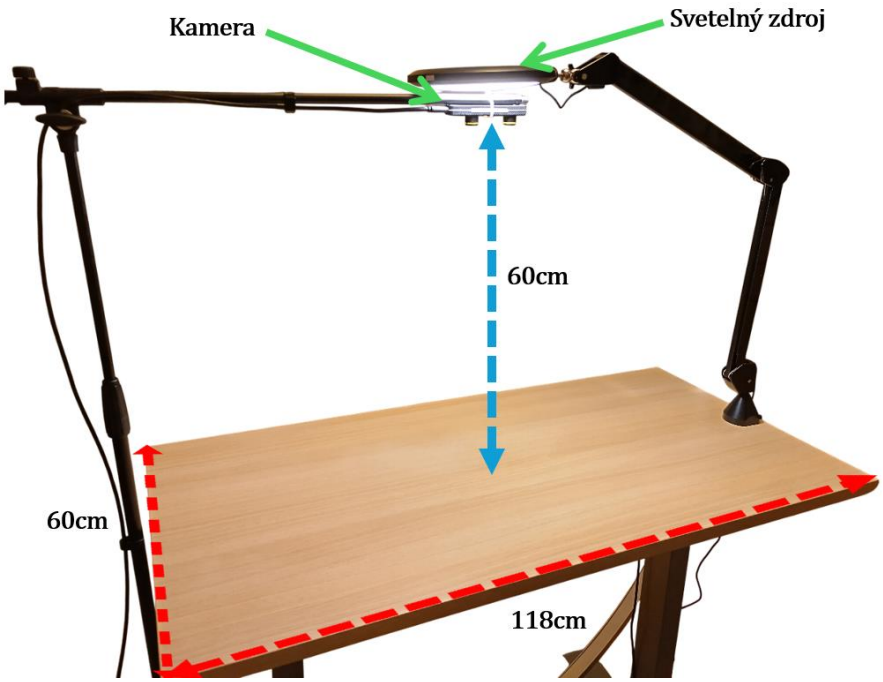
Obrázok 4.2-1 - Integrácia ZED-mini do reťaze spracovania [82]

Výrobca ponúka vlastné SDK, ktorá pridáva hĺbkové snímanie, rozpoznávanie objektov, trekovanie, detekciu roviny a API pre jednotlivé kamery. Na vývoj sú minimálne požiadavky pre externú jednotku Dual-core 2.3GHz alebo viac, minimum 4GB RAM pamäte a Nvidia GPU výpočetná schopnosť ≥ 3.0. (CUDA kompatibilita).

4.3 Riešená úloha

Cieľom je rozpoznať a lokalizovať objekt rozložiteľný na časti. Či už by sa jednalo o komponenty, ktoré prirodzene do seba zapadajú alebo je potrebná ďalšia akcia od operátora (použitie náradia) je v zásade nepodstatné. V prípade použitia náradia by sa napríklad rozpoznali potrebné nástroje.

Ďalej sa vychádza z faktu, že kamera je upevnená na hlave operátora nad pracovným stolom. To jest vo výške prirodzenej na prácu a pohľadom zvrchu na pracovisko. Vzdialenosť by sa mala udržovať relatívne konštantná. Pracovisko pozostáva z pracovného stola, na ktorom bude prebiehať proces montáže. Pre statickú scénu je v strede nad stolom pripevnená kamera so svetlom vo výške 60cm, čo by malo simulovať približnú vzdialenosť ruky.



Obrázok 4.3-1 - Pohľad na pracovný stôl pre statickú scénu

Použitým objektom je Kärcher vysávač na okná WV 75, ktorý sa skladá z troch častí. Jednotlivé komponenty do seba zapadajú bez nutnosti použitia ďalších nástrojov.

Obrázok 4.3-2 - Použitý objekt v implementácii

4.4 Výber metódy

Azda najdôležitejšou požiadavkou pre detekciu v tomto type úlohy je rýchlosť. Z pohľadu užívateľa je dôležitá real-time, alebo čo najbližšia real-time, informácia o objektoch či už sa jedná o pozíciu objektu v obraze, alebo včasné inštruovanie. Druhá metrika je relatívna presnosť detekcie a robustnosť.

Metódy založené na detekcii hrán čelia viacerým problémom, ako napríklad zmena pózy objektu, zmena pohľadu na objekt, avšak najväčším sú chýbajúce hrany spôsobené oklúziou. To môže značiť nedostatok pri manipulácii s objektami a ich následným skladaním. Spomínaná rýchlosť je ďalšou dôležitou oblasťou, v ktorej žiaľ nedosahujú dobré výsledky. Jedná sa o rýchlosti v rádoch sekúnd, najmä, ak by sa jednalo o obrazy s vysokým rozlíšením. Najlepšia v oblasti rýchlosti dosahuje metóda GHT, konkrétne MGHT podľa [43], ktorá dosahuje rýchlosť 0,07s pre obraz 600x600px a v desiatkach milisekúnd. Avšak v prípade implementácie pre rôzne objekty, v rôznych pózach sa potrebný čas niekoľkonásobne zvýši. V rámci precíznosti, keďže sa jedná o detekciu hrán, tak naopak vykazujú subpixelovú presnosť detekcie. Výhodou je aj schopnosť reprezentovať hladké objekty s minimálnou textúrou, na druhej stranu však nemusia byť dostatočne deskriptívne v prípade triviálneho tvaru objektu (kváder, kocka, ihlan) – mohla by nastať dvojzmyselnosť pri rozpoznávaní na základe iba tvaru objektu. Preto by bolo potrebné implementovať aj komplementárnu metódu, ktorá by brala do úvahy obsah predmetu alebo jeho textúru.

Strojové učenie na druhú stranu ponúka, najmä CNN, state-of-the-art detektory na všeobecné rozpoznávanie objektov. Dosahujú najvyššej presnosti, avšak rýchlosť detekcie je silno závislá na dostupnom hardvéri. Jedná sa o počet operácií v rádoch miliárd za sekundu, čo na bežnom superskalárnom procesore nie je možné v reálnom čase. Preto je nutný dedikovaný hardvér ako TPU (tenzorové procesory), ktoré disponujú špecializovanými maticovými operáciami, alebo GPU, ktoré rádovo tieto operácie sú schopné vykonať, pomocou SIMD modelu vykonávania inštrukcií. Ďalším dôležitým faktorom pre presnosť je vstupný dataset, na ktorom sa model učí. Pre proces učenia je potrebné dodať stovky vstupných dát, pričom treba dohliadať na diverzifikáciu jednotlivých snímkov (alebo je možné použiť nejaký generatívny model na umelú augmentáciu dát). Naopak výhodou sú už bežne dostupné architektúry, ktoré sa špecializujú na konkrétnu úlohu.

Lokálne príznaky produkujú relatívne jedinečné body v danej množine a nesú charakteristickú informáciu. Následne súborom týchto významných bodov je možné reprezentovať objekt, respektíve jeho časti. Jedná sa o relatívne jednoduchý spôsob detekcie objektov. V online fázy ide o hľadanie korešpondencií referenčného súboru, deskriptoru, s detegovanými význačnými bodmi. Potom sa hľadá transformácia pôvodných bodov do tých detegovaných. Vieme teda zistiť kde sa presne nachádza hľadaný súbor významných bodov. Jednotlivé fázy je možné vykonávať paralelne s oneskorením. Detektor pracuje v kroku $k+1$, zatiaľčo hľadanie korešpondencií a transformácie v kroku k. Detektor teda spracováva nový snímok, zatiaľčo hľadanie korešpondencií ten predošlý. Výhodou je schopnosť paralelizovať úlohu, čím sa prekryjú jednotlivé časti. Realizácia je tak možná aj na bežnom hardvéri s rozumnou latenciou. Oproti metódam vychádzajúcich z popisu hrán objektu, funguje aj pri oklúzii a pre objekty s podobnou štruktúrou kontúr. Nevýhodou je, že objekty musia byť

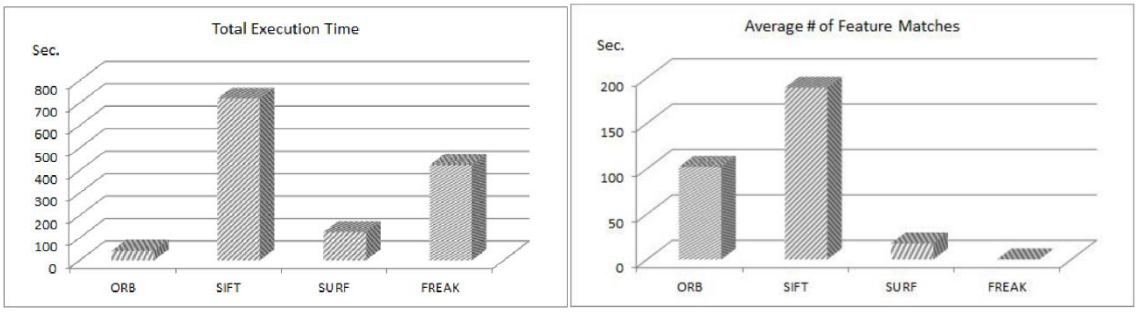
štruktúrované, obsahovať textúru, z ktorej sa detegujú významné body. Nesmú byť hladké bez textúry.

4.4.1 Zvolená metóda

Po zvážení sa vybrala metóda na základe lokálnych príznakov, kvôli rýchlosti a účinnosti aj pri oklúzii predmetu. V rámci hardvéru sa používa kamera, ktorej obraz je potrebné následne spracovať, napríklad na PC, preto si prihliadlo aj na schopnosť implementovať metódu na dostupnom hardvéri.

Ako bolo spomenuté v kapitole 3.1, tvorba modelu pomocou významných bodov spočíva v akvizícii snímku z rôznych pohľadov. Tak aby nebolo nutné disponovať snímkami z každého možného pohľadu, je dostatočný, obmedzený počet snímkov v prípade, ak daná metóda garantuje aspoň čiastočnú invarianciu voči rotácii a zmene merítka. Je potrebné preto definovať granularitu akvizície a tvorby datasetu. Príliš hrubá granularita by znamenala neschopnosť detegovať objekt v nejakej konkrétnej orientácii. Príliš jemná granularita by znamenala v prvom rade opakovať proces akvizície, detekcie a tvorby deskriptoru niekoľkokrát a zároveň by sa databáza deskriptorov toľkokrát zväčšila. Problém s veľkosťou databáze je ilustrovaný v kapitole o SIFT metóde. Vo finále sa detekcia 3D objektu dekomponovala na 2D problém, čím je možné proces paralelizovať, ale stále sa zachováva informácia o jeho orientácii (cez popísané strany).

Pri voľbe metódy na tvorbu modelu pomocou lokálnych príznakov som teda prihliadal na predošlé spomenuté požiadavky a porovnania týchto metód, ako napríklad [83][84]. Prvá porovnáva výkonnosť algoritmov SIFT, SURF, FREAK a ORB. Najprv sa z referenčného snímku určili významné body a deskriptor a vo video sekvencii referenčného obrázku, v rôznej výške a rotácii, sa hľadala zhoda. Na vymedzenie počtu nájdených korešpondencií je použitá imy definovaná metrika prahu.



Obrázok 4.4-1 - Celkový čas detekcia hľadania zhody (vľavo), Počet zhôd (vpravo), pre threshold = 120 [83]

Na obrázku je možné vidieť, že najrýchlejší spomedzi algoritmov je ORB, kde celkový čas je 17x lepší ako v prípade SIFT a priemerný počet zhôd je 101/253 oproti 189/251 pre SIFT. SURF a FREAK vykazujú nedostatočné výsledky, kde v prípade FREAK sa nenašla zhoda žiadna a celkový čas je 10x pomalší ako pre ORB. Pre SURF

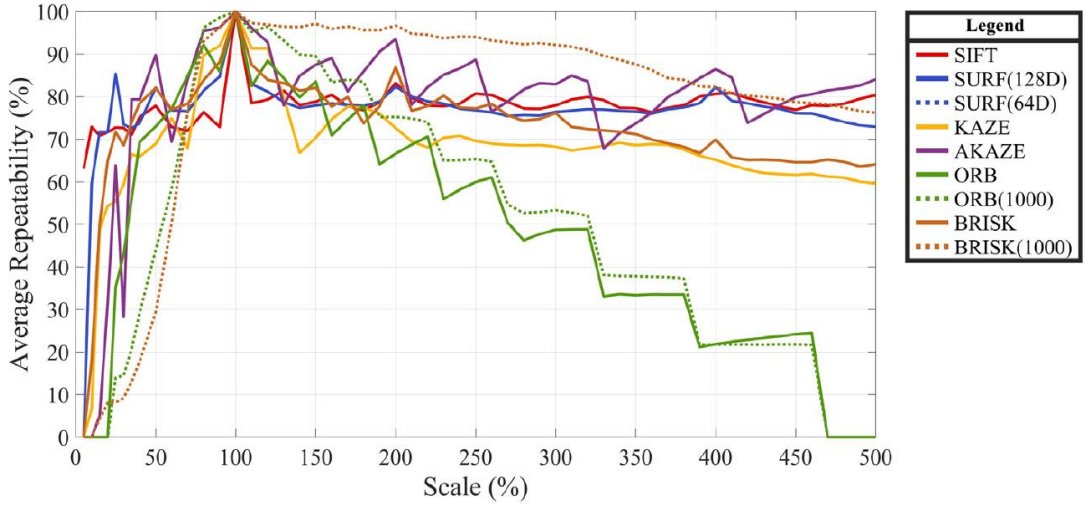
je celkový čas porovnateľný s ORB, ale pre hľadanie správnej zhody je len 17/268. Samozrejme nie sú popísané parametre jednotlivých metód ani popis prahu, avšak udáva to predstavu o jednotlivých algoritmoch.

V druhej spomenutej práci sa porovnávajú SIFT, SURF, (A)KAZE, ORB a BRISK. Pre SURF, ORB a BRISK boli ešte vytvorené varianty rôznych dĺžok deskriptoru alebo počtu významných bodov. Metodika spočíva v šiestich pároch rôznorodých obrázkov a hľadanie korešpondencie s nimi. Použila sa zmena mierky od 5% do 500% a rotácia v 360°. Stratégia hľadania korešpondencie vychádza z metriky NNDR podľa [54], kde sa prah nastavil na úroveň 0,7. Nastavenie RANSAC metódy pre zamietnutie tzv. „outliers“ je 2000 iterácií a úroveň istoty 0,995.

Tabuľka 4-1 - Porovnanie metód z hľadiska času na významný bod [84]

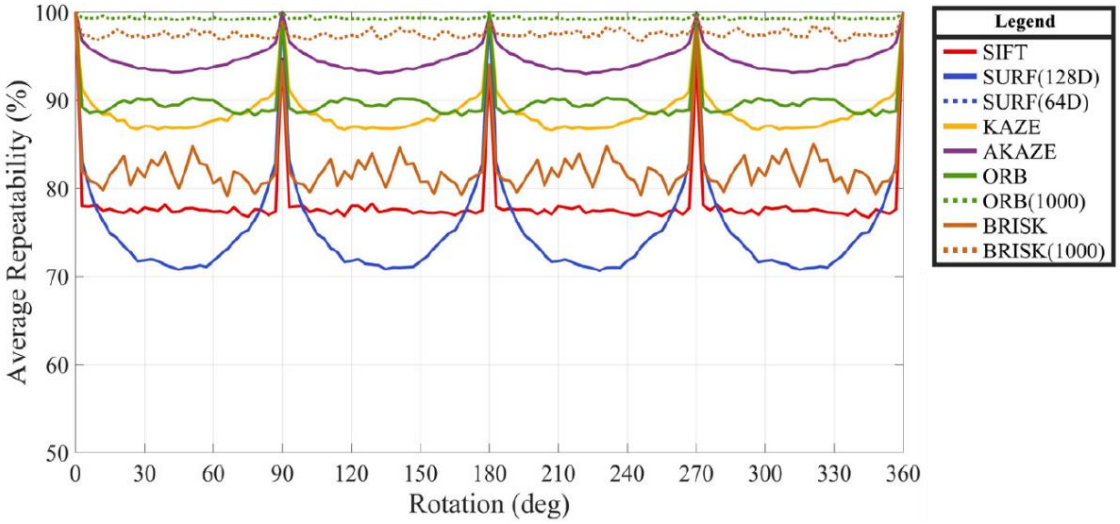
Algorithm	Mean Feature-Detection-Description Time per Point (µs)		Mean Feature Matching Time per Point (µs)
	1st Images	2nd Images	
SIFT	90.44	85.15	**142.02**
SURF(128D)	42.78	42.22	**168.55**
SURF(64D)	41.83	41.18	89.66
KAZE	**191.24**	**177.09**	60.58
AKAZE	60.93	57.04	24.61
ORB	**3.94**	**3.94**	97.25
ORB(1000)	13.51	13.92	**11.82**
BRISK	16.59	16.76	**124.64**
BRISK(1000)	20.70	21.49	**15.42**

Najrýchlejšie časy dosiahli BRISK a ORB v prípade obmedzenia počtu významných bodov na 1000. Na druhú stranu SIFT je najpomalší spomedzi všetkých algoritmov.



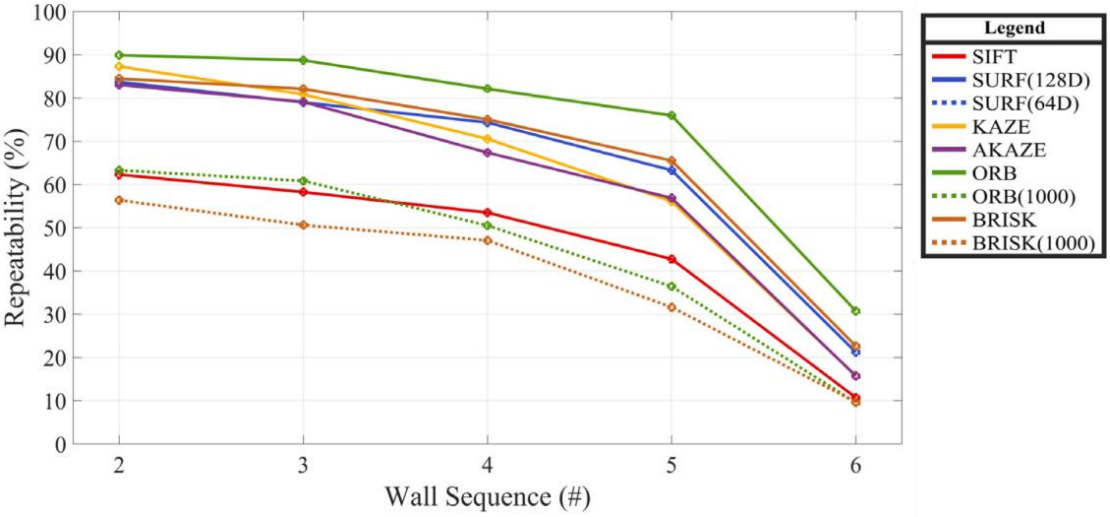
Obrázok 4.4-2 - Porovnanie metód pre zmenu veľkosti oproti referenčnému obrazu [84]

Čo sa týka invariancie na zmenu veľkosti objektu tak, pre ORB sú výsledky dostatočné v intervale 60% až 150%, podobne ako pre BRISK, avšak pre mimo tento interval sa úspešnosť zníži dramaticky, najmä vo veľkostiach menších ako referenčný obraz. Najstabilnejšie sú SIFT a SURF.



Obrázok 4.4-3 - Porovnanie metód pre rotáciu oproti referenčnému obrazu [84]

Ohľadom rotácie, najlepší výsledok dosiahne ORB, BRISK a AKAZE. Naopak najhoršie výsledky dosiahne SURF a SIFT.



Obrázok 4.4-4 - Porovnanie metód pre zmenu pohľadu voči referenčnému obrazu [84]

Pre zmenu pohľadu (afínnu invarianciu) sa najlepšie choval ORB, BRISK a SURF, na opačnom konci je SIFT.

Autori nakoniec vyhodnotili metódy nasledovne :

- **ORB>BRISK>SURF>SIFT>AKAZE>KAZE**

V ďalšom kroku sa prihliadalo na už implementované algoritmy v OpenCV. V základnom balíku sú implementované metódy SIFT, ORB, BRISK, (A)KAZE, SURF a iné. Napríklad metóda FREAK nie je dostupná v základnej kompilácii, ale až v extra moduloch, ktoré sú špecializované alebo experimentálne.

SURF bol zamietnutý z dôvodu najnižšej invariancie voči rotácii spomedzi všetkých zvyšných metód [83][55]. Na základe rýchlosti detekcie, tvorby deskriptoru a hľadaní korešpondencie v generickej scéne úlohy sa rozhodovalo medzi SIFT, ORB, BRISK.

SIFT sa ďalej neuvažoval z dôvodu rýchlosti detekcie a hľadania zhody. Jedná sa rádovo o desiatky až stovky milisekúnd, čo nie je vhodené pre real-time aplikáciu. Naopak ORB a BRISK sú rýchlosťou porovnateľné, i keď ORB je najrýchlejší. Avšak prihliadnutím na zvolenú metodiku, pri ktorej vyššia robustnosť znamená menšie množstvo potrebných snímkov na tvorbu modelu a vyššia kvalita lokalizácie predmetu, zvolil sa BRISK.

5. IMPLEMENTÁCIA

Nasledujúce kapitoly opisujú spôsob a postup implementácie detekcie objektu pomocou lokálnych príznakov BRISK. Najprv sa charakterizuje metóda tvorby modelu pomocou 2D rovín. Potom jeho reprezentácia a samotný proces rozpoznávania. Nakoniec sa predostrie štruktúra kódu a použité softvérové prostriedky.

5.1 Tvorba modelu

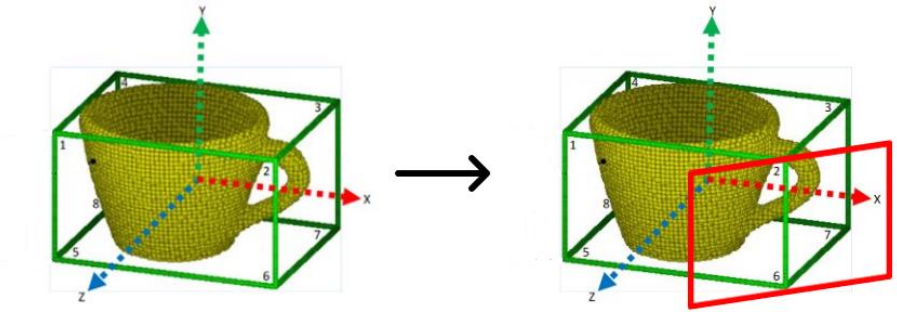
Tvorba modelu spočíva v akvizícii snímkov objektu z rôznych pohľadov, pričom vzdialenosť snímacieho zariadenia je približne rovnaká ako vzdialenosť kamery od pracovného stola. Keďže v demonštračnej úlohe by mala byť kamera hypoteticky umiestnená na hlave operátora. Aj keď detektory a deskriptory významných bodov sú do istej miery invariantné voči zmene merítka (scale), tak pre dosiahnutie najlepšieho výsledku sa použila práve táto vzdialenosť. Zároveň by sa objekty mali zväčša nachádzať približne v konštantnej vzdialenosti, na dĺžku ruky. Nepredpokladá sa extrémna zmena vzdialenosti, priblíženie objektu priamo k snímaciemu elementu a naopak snímanie pracoviska z extrémnej vzdialenosti.

Každý objekt môže byť všeobecne definovaný ako objekt so 6 stranami. Každá strana je definovaná 4 bodmi (rohmi) a časťou samotného objektu. V prípade tvorby modelu sa využila akvizícia snímkov z rôznych pohľadov, čím sa definovali 4 rohy roviny objektu a zároveň významné body danej ohraničenej roviny (ako na obrázku 5.1-1).

V prvej fáze boli získané snímky každej strany objektu v stabilnej polohe. Vychádza sa z faktu, že pokiaľ bude predmet položený na pracovnom stole, nebude sa nachádzať vo vratkej polohe. V prípade nepravidelných útvarov to znamená, že sú uvažované stabilné strany objektu a nie reálne strany objektu.

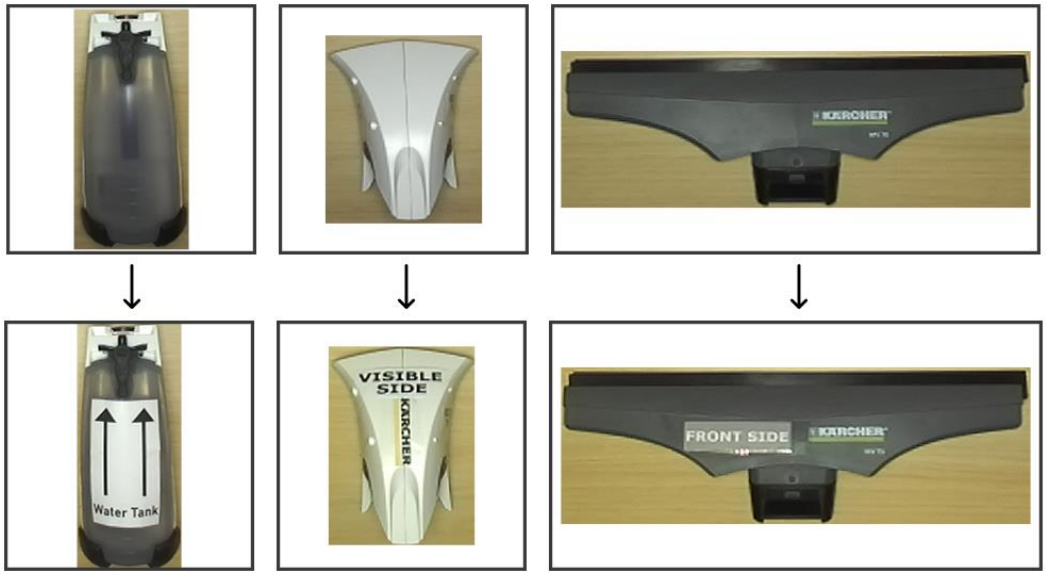
Je dôležité poznamenať, že každý pohľad by mal byť jednoznačný, aby nevznikli dvojzmyselné pohľady. Napríklad v prípade symetrického objektu je zbytočné tvoriť model z každého pohľadu, keďže sú totožné.

Pre zabezpečenie vyššej robustnosti a presnosti, je možné vytvoriť komplementárne snímky z pohľadov 30°- 45° na normály strán.



Obrázok 5.1-1 - Príklad modelu objektu definovaného 6 stranami [85] (vľavo), ilustrácia pridania komplementárnych pohľadov (vpravo)

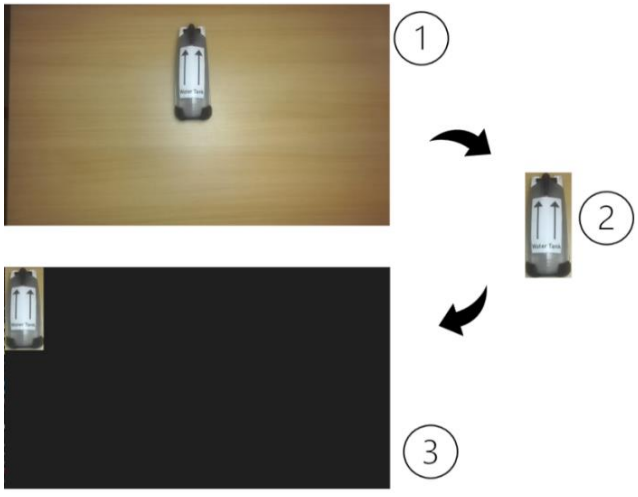
Samotné strany komponentov nevykazujú vysoký počet významných bodov, preto na každú stranu objektu sa nalepili dodatočné značky, čím sa zvýši počet detegovaných významných bodov a detekcia strany objektu bude viac pravdepodobná.



Obrázok 5.1-2 - Augmentácia komponentov

5.1.1 Postup tvorby modelu

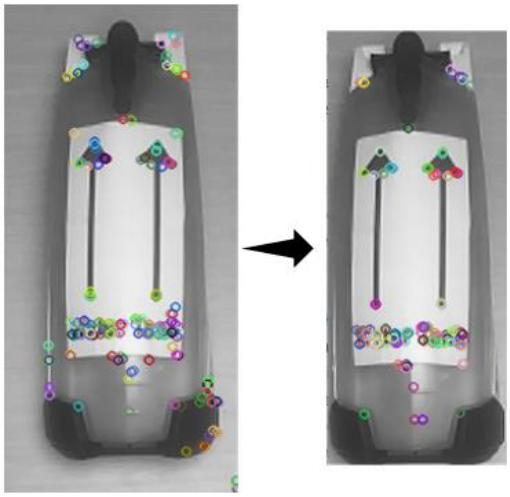
Pre získanie datasetu sa použil nasledujúca postupnosť krokov. Prvá časť je akvizícia snímku pracoviska s danou pózou daného objektu (1). Samotné pracovisko je v stave v akom sa bude vykonávať detekcia (hľadanie korešpondencie) pri behu aplikácie. Ide najmä o svetelné podmienky, keďže metódy lokálnych príznakov nie sú kompletne invariantné na zmenu jasu. Druhý krok spočíva vo vyrezaní daného objektu z obrazu (2). Ten sa následne umiestnil na čierne pozadie obrázku o rozmeroch snímku kamery (3). Až z tohto obrázku sa detegujú významné body a vytvára deskriptor.



Obrázok 5.1-3 - Tvorba datasetu, kroky 1-3

Pri hľadaní korešpondencie (a homografie) sa používa obrázok z kroku (3), respektíve jeho významné body a deskriptor. Pri riešení perspektívnej transformácie sa používa obrázok z kroku (2), keďže sú potrebné rozmery objektu v obraze. Na akvizíciu snímky sa použil projekt *DatasetAcquisition*, ktorý vytvorí snímky z kroku (1). Po manuálnej extrakcii objektu, projekt *KeypointsFromSmaller* deteguje významné body a vytvorí deskriptor pre daný obrázok. Tie sa presunú do dezignovanej zložky v hlavnom projekte, spolu s obrázkom z kroku (2). Náhľad organizácie datasetu vo filesystéme je v nasledujúcej kapitole.

Keďže v metóde BRISK sa pri určovaní významného bodu uvažuje jeho okolie, tak v prípade použitia celého obrazu, alebo jeho časti vrátane objektu, by boli významné body ovplyvnené aj pozadím. Síce sa eliminuje časť významných bodov, ale určia sa iba tie najstabilnejšie. Z toho vyplýva, že predmety, respektíve strany predmetov, musia obsahovať bohatú textúru mimo okolia hrán.



Obrázok 5.1-4 – Detegovaných 176 významných bodov s výplňou od hranice (vľavo), Detegovaných 108 významných bodov bez výplne (vpravo)

Na obrázku vľavo je možné vidieť, že body blízko hranice obrazu, ktoré sú detegované ako významné, sa na obrázku vpravo už takmer nenachádzajú.

Výsledným datasetom je sada snímkov (+ deskriptorov a významných bodov) pokrývajúca minimálne stabilné polohy objektu. Podľa počtu komplementárnych snímkov sa počet teoreticky navýši na 26. Tie popisujú objekt z každého pohľadu po otočení o 45° v osách x,y,z.

Obrázok 5.1-5 - Základný dataset pre objekt, stabilné polohy z každej strany objektu

5.2 Kompozícia datasetu

Tvorba datasetu pozostáva z akvizícii snímku pomocou ZED-mini kamery z každého definovaného pohľadu. Výstupom je snímok (1), význačné body (2) a deskriptor (3). V zložke *resources*/*objectX* (kde '*X*' značí poradie objektu) sa nachádzajú dva .json súbory – *descriptor_object_X.json* a *keypoints_object_X.json* (1,2) – a snímky pohľadov (3), kde každý je označený v poradí ako *object_X_Y.png* (kde '*X*' značí poradie objektu a '*Y*' značí poradie pohľadu).

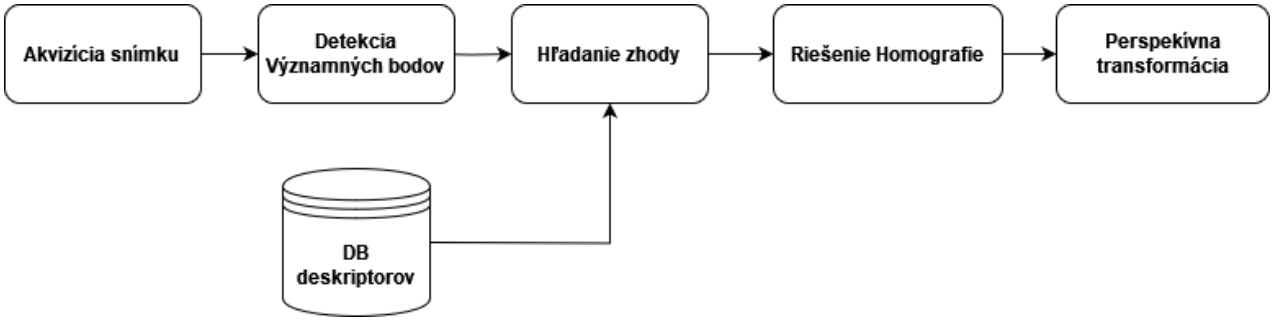
```
AugmentedAssembly
|
└──resources
      |
      └──objects
            |
            ├──object_0
            |    ├──descriptor_object_0.json
            |    ├──keypoints_object_0.json
            |    ├──object_0_0.png
            |    ├──object_0_1.png
            |    ├──...
            |    └──object_0_Y.png
            |
            ├──object_1
            |    ├──descriptor_object_1.json
            |    ├──keypoints_object_1.json
            |    ├──object_1_0.png
            |    ├──object_1_1.png
            |    ├──...
            |    └──object_1_Y.png
            |
            ├──...
            |
            └──object_X
                 ├──descriptor_object_X.json
                 ├──keypoints_object_X.json
                 ├──object_X_0.png
                 ├──object_X_1.png
                 ├──...
                 └──object_X_Y.png
```

Potom každý *.json* (descriptor/keypoints) obsahuje korešpondujúce vektory pre každý snímok objektu. Teda v *.json* deskriptoru je zaznamenaný *descriptor_0*, ktorý korešponduje v *.json* význačných bodov s *keypoints_0* , čo korešponduje so snímkom *object_X_0.png*.

Nastavenie kamery je totožné s nastavením použitím vo finálnej aplikácii, čo sa týka rozlíšenia, ostrosti a podobne.

5.3 Detekcia objektu

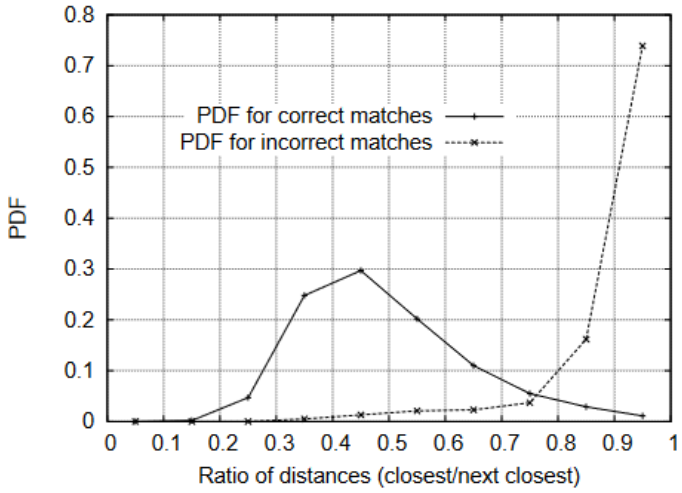
Priebeh detekcii pozostáva z akvizícii snímku, v ktorom sa detegujú významné body pomocou metódy BRISK. Tie sa následne filtrujú a hľadá sa korešpondencia v rámci databázy deskriptorov objektu, teda najlepšia zhoda medzi deskriptorom obrazu a jednotlivých deskriptorov pohľadov daného objektu. Použitý bol *bruteforce matcher*, *flann based matcher* neprodukoval dostatočné korešpondencie. V prípade dostatočného množstva kvalitných korešpondencií, vyžaduje sa aspoň 10 viz. kapitolu 6.2, sa rieši matica homografie **H**. Tá ma za úlohu nájsť vzťah medzi významnými bodmi strany modelu objektu a jeho reprezentácii v obraze. Ide teda o mapovanie 2D-2D roviny (plocha objektu – plocha v obraze), ktorých vzťah je definovaný pomocou korešpondencie významných bodov Na vymedzenie správnosti matice H, sa vypočíta jej determinant. Pokiaľ sa nenachádza v danom intervale, napríklad podľa [86], tak sa matica zamieta. Hodnoty boli nájdené empiricky. V poslednej fáze sa počíta perspektívna transformácia pôvodných štyroch rohov strany do obrazu. Následne sa spoja, aby tvorili štvoruholník. V princípe ide o „bounding box“ pre danú stranu objektu. Dodatočný a posledný test správnosti je určenie uhlov, ktorý zvierajú strany vzniknutého štvoruholníka. Hodnotami mimo interval sa riešenie opäť zamieta.



Obrázok 5.3-1 - Základná logika detekcie

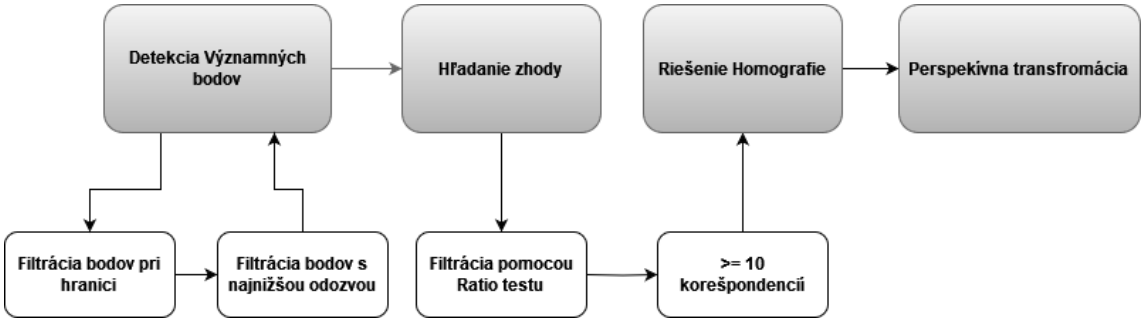
5.3.1 Filtrácia a overovanie správnosti

V [54] je popísaný tzv. „ratio test“, ktorý vymedzuje pravdepodobnosť nájdenia správnej zhody. Vychádza sa z faktu, že pri určovaní zhody sa používa metrika vzdialenosti (Euklidova, Hamming) medzi referenčným významným bodom a významným bodom v scéne. Na filtráciu sa použijú dve zhody s najnižšou vzdialenosťou k referenčnému významnému bodu, označme ich r_1 a r_2. Ak je pomer $r = \frac{r_1}{r_2} > r_{max}$, čo znamená, že zhoda nie je jednoznačná, tak sa kandidát zamietne. Predpokladom je, že pokiaľ bod so vzdialenosťou r_1 je nositeľom značnej informácie (je skutočne významný), nemal by nastať stav, kedy by sa našla ďalšia podobná/blízka zhoda. Autor udáva, že pre $r = 0.8$ sa eliminuje 90% falošných korešpondencií.



Obrázok 5.3-2 - Ratio test na filtráciu falošných zhôd. Ukazuje PDF (probabilty density function) „ratio test-u“ pre pravdivé a falošné zhody

Prvotná filtrácia bodov nastáva už po detekcii bodov. Uchováva sa najsilnejších 90% a z nich sa odstránia významné body v blízkosti 20px od hraníc obrazu (celkový maximálny počet významných bodov je limitovaný na KP_MAX). Pre filtráciu korešpondencií sa použila teda NNDR (Nearest Neighbor Distance Ratio) podľa [54]. Ďalšou z metrík je prakicky aj samotné riešenie homografie, ktoré rozdelí vstupné dáta na „outliers“ a „inliers“, čím sa filtrujú zhody ležiace, pravdepodobne, mimo objektu.



Obrázok 5.3-3 - Priblíženie filtrácie významných bodov a korešpondencií

Poslednými metrikami sú determinant matice **H** a tolerancia uhlov štvoruholníka, tvorený bodmi výslednej perspektívnej transformácie

Zhrnutie filtrácií a podmienok:

1. Filtrácia významných bodov
2. Filtrácia korešpondencií pomocou ratio testu
    a. kvalitných korešpondencií ≥ 10, inak preskoč iteráciu
3. Riešenie matice homografie
    a. H ϵ intervalu, inak preskoč iteráciu
4. Výpočet perspektívnej transformácie
    a. $uhol\ \alpha_i \geq$ threshold $(i = 1,2,3,4)$, inak preskoč iteráciu

5.3.2 Homografia

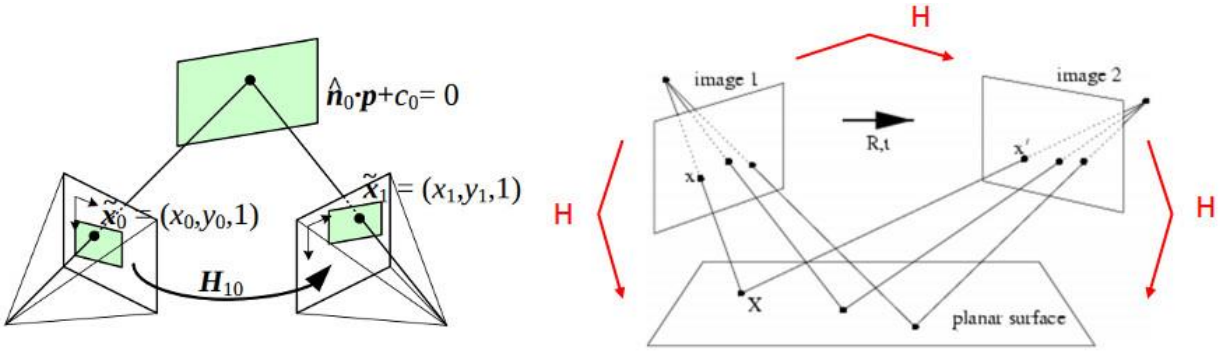
Plošná homografia je metóda, ktorá hľadá transformáciu medzi dvoma plochami, ktorá je definovaná ako 3x3 matica **H.**

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{h_{11}} & \boldsymbol{h_{12}} & \boldsymbol{h_{13}} \\ \boldsymbol{h_{21}} & \boldsymbol{h_{22}} & \boldsymbol{h_{23}} \\ \boldsymbol{h_{31}} & \boldsymbol{h_{32}} & \boldsymbol{h_{33}} \end{bmatrix} \tag{6.1}$$

Prvky $\boldsymbol{h_{11}, h_{12}, h_{21}, h_{22}}$ predstavujú afínnu transformáciu, $\boldsymbol{h_{13}, h_{23}}$ transláciu a $\boldsymbol{h_{31}, h_{32}}$ perspektívnu transformáciu. Pre aplikáciu matice H na arbitrárny bod sa používajú **homogénne súradnice**, ktoré pre každý bod (x,y) v 2D priestore pridajú ďalšiu súradnicu ω. Nastane tak zjednotenie reprezentácie bodu a jeho transformácie pomocou matice a zároveň sa výpočet mapovania bodu zjednoduší na operáciu násobenia matíc. Ďalej pre transformáciu súradníc pomocou matice **H** platí:

$$\begin{bmatrix} \boldsymbol{x'} \\ \boldsymbol{y'} \\ \boldsymbol{1} \end{bmatrix} = \boldsymbol{H} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{h_{11}} & \boldsymbol{h_{12}} & \boldsymbol{h_{13}} \\ \boldsymbol{h_{21}} & \boldsymbol{h_{22}} & \boldsymbol{h_{23}} \\ \boldsymbol{h_{31}} & \boldsymbol{h_{32}} & \boldsymbol{h_{33}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{1} \end{bmatrix} \tag{6.2}$$

Kde bod $\boldsymbol{P'}$ (hľadaný bod), je definovaný podľa homogénnych súradníc $(\boldsymbol{x'y', 1})$ a referenčný bod $\boldsymbol{P}$ ako $(\boldsymbol{x, y, 1})$. [87]



Obrázok 5.3-4 - Ilustračné znázornenie homografie[4]

Aby sme boli schopní nájsť maticu **H**, je potrebné nájsť aspoň štyri navzájom korešpondujúce body [87]. Tradičnou metódou je použitie významných bodov definovaných pomocou napríklad SIFT, ORB, BRISK a podobne. Postup spočíva v detekcii významných bodov, v hľadaní zhody a nakoniec v riešení homografie. V prípade prítomnosti väčšieho množstva korešpondujúcich párov je výhodné na jej riešenie použiť metódu RANSAC.

[4] **https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html#tutorial_homography_Demo4**

5.3.3 RANSAC

RANSAC (Random sample consensus) je všeobecne metóda na vyhľadávanie modelu/parametrov v dátach, ktoré podliehajú šumu (prítomnosť tzv. „outliers“). V prvom kroku sa náhodne vyberie minimálny počet bodov z dát, potrebných na estimáciu modelu (1). V ďalšom kroku sa hľadá samotné riešenie modelu (2). Ak sa v odhadnutom modeli nenašlo dostatočné množstvo tzv. „inliers“, algoritmus sa opakuje od kroku 1. V opačnom prípade sa model dezignuje ako správny (3). V poslednej fáze sa odhaduje model iba z „inliers“ bodov, čím sa vylepší finálny model (4). Algoritmus končí, pokiaľ sa nenájde najlepšie riešenie, alebo sa nedosiahne maximálny počet iterácií. RANSAC týmto spôsobom rozdelí dáta na tzv. „outliers“ a „inliers“, čo je možné využiť na ďalšie spracovanie. Problémom sa môže javiť, že nie sme schopní zistiť, že sa našiel optimálny model. Algoritmus môže skončiť na maximálnom počte iterácií, čo však neznamená, že sa našiel najoptimálnejší model.

5.4 Popis softvéru

Softvérová časť aplikácie pozostáva zo šiestich tried, kde každá implementuje konkrétnu funkcionalitu. Podrobnejšie sú jednotlivé triedy rozobraté v nasledujúcich kapitolách. Celý program je písaný v C++, najmä kvôli potencionálnej rýchlosti oproti iným jazykom a kvôli kompatibilite, jednak s kamerou a OpenCV knižnicou. Minimálnym štandardom je **C++ 17**, z dôvodu použitých knižníc STL, napríklad *filesystem* alebo *synchronizačné mechanizmy* pre vlákna.

Na tvorbu dokumentácie sa použil nástroj **Visual Paradigm** 17.1. Jedná sa o nástroj na vývoj softvéru, ktorý napríklad umožňuje modelovanie UML diagramov a následné generovanie kódu z týchto modelov. Je k dispozícii ako verzia pre komunity zdarma.

Na dokumentáciu samotného kódu bol použitý **Doxygen**, čo je nástroj na generovanie dokumentácie zdrojového kódu. Podporuje viacero jazykov, vrátane C++. Po parsovaní návestí sa vytvorí krížový model dokumentácie programu, čo sprehľadňuje účel jednotlivých metód a tried aplikácie.

OpenCV je open source knižnica pre počítačové videnie a strojové učenie. Ponúka tak spoločnú bázu pre aplikácie počítačového videnia a rovnako tak zvyšuje tendenciu použitia strojového videnia v komerčných produktoch. Knižnicu je možné použiť na rôznych platformách, Windows, Unixové distribúcie a v rôznych jazykoch, C++, Python, C#, Java alebo Matlab. Rovnako tak disponuje širokou škálou optimalizovaných algoritmov a kvalitnou dokumentáciou. Práve z tohto dôvodu bola zvolená ako východisková vo vývoji práce. V implementácii je používaná verzia 4.8., ktorá bola skompilovaná pomocou CMake verzie 3.27.7.

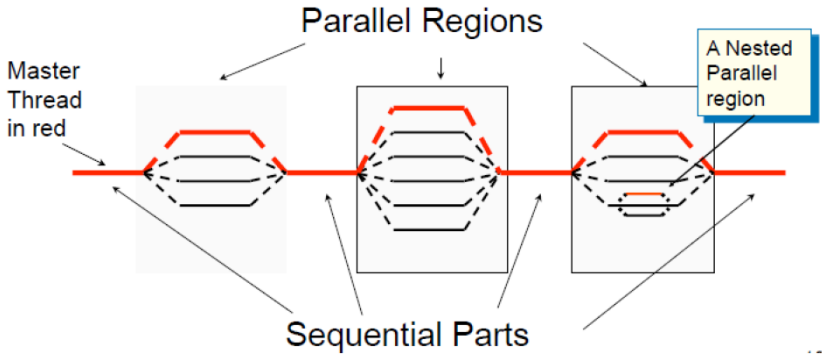
Kamera má k dispozícia svoje vlastné SDK (**ZEDSDK**), ktoré ponúka API na manipuláciu, ako prevzatie snímku alebo pokročilejšie funkcionality.

Windows API je použitá pre jej implementáciu časovača, ktorý zavolá callback funkciu po vypršaní času. To je využívané napríklad pri prechode stavového automatu. Dependencie aplikácie sú nasledovné:

- C++ 17
- OpenCV 4.8
- ZED SDK 4.0.8
- Windows API
- OpenMP ≥ 2.0

5.4.1 OpenMP

Jedná sa o jednu z najpoužívanejších knižníc na tvorbu paralelných aplikácií v C/C++ a Fortrane. Jedná sa de facto o štandard v oblasti paralelného programovania. Samotné OpenMP obsahuje súbor direktív pre kompilátor a knižničné rutiny, ktoré značne uľahčujú písanie viacvláknových aplikácií. Ide o vyššiu kontrolu nad aplikáciou, kde programátor je schopný explicitne určiť, ktorú časť kódu paralelizovať alebo vektorizovať, pokiaľ to kompilátor nezvládne sám.



Obrázok 5.4-1 - Programovací model OpenMP

Základná logika OpenMP modelu spočíva z hlavného vlákna, v ktorom sa dynamicky vytvára súbor vlákien. Oblasti sa vytvárajú pomocou direktívy *#pragma omp parallel* a tie paralelne riešia danú časť programu. Model by sa dal popísať ako princíp *fork-join* pre procesy v jazyku C. Paralelizácia je možná aj rekurzívne, teda vytvárať paralelné oblasti v paralelných oblastiach.

Počet vlákien v jednotlivých oblastiach sa určuje buď explicitne, alebo implicitne. Implicitne sa odvodí od počtu logických jadier systému alebo explicitne dovetkami:

- *num_threads(thread_count)* – pre danú paralelnú oblasť
- *omp_set_num_threads(int)* – pre všetky nasledujúce paralelné oblasti
- OMP_NUM_THREADS = num – pre celý program

Samotná direktíva *parallel* predstavuje SPMD (single program multiple data) model, teda každé vlákno vykonáva rovnaký kód. Na diverzifikáciu práce vlákien sa používajú ďalšie direktívy. Najbežnejšia je pre slučky *for*. Spojením direktív teda vzniká *#pragma*

omp parallel for, čím sa paralelizuje slučka priamo pod direktívou. Ďalšie vnorené slučky paralelizované nie sú.

Touto direktívou sa vytvorí počet vlákien na iteráciu (do maximálneho počtu vlákien), čo nemusí byť vždy ideálne. Preto je dôležité buď určiť počet vlákien, ktoré si prácu rozdelia spravodlivo (explicitné dovetky), alebo pomocou *#pragma omp for schedule(dynamic[, chunk_size])*. *Schedule* umožňuje dynamicky priradiť počet iterácií pre dané vlákno. Druhá možnosť je použiť direktívu *guide*, ktorá sama dynamicky vyvažuje záťaž, čo je vhodné v prípade, ak sa čas jednotlivých iterácií líši

Pre Microsoft Visual Studio kompilátor (MSVC) je možné zapnúť podporu OpenMP, síce je podporovaná verzia 2.0, ale je to postačujúce pre základné direktívy (bohužiaľ tam chýba, napríklad, direktíva *schedule*).

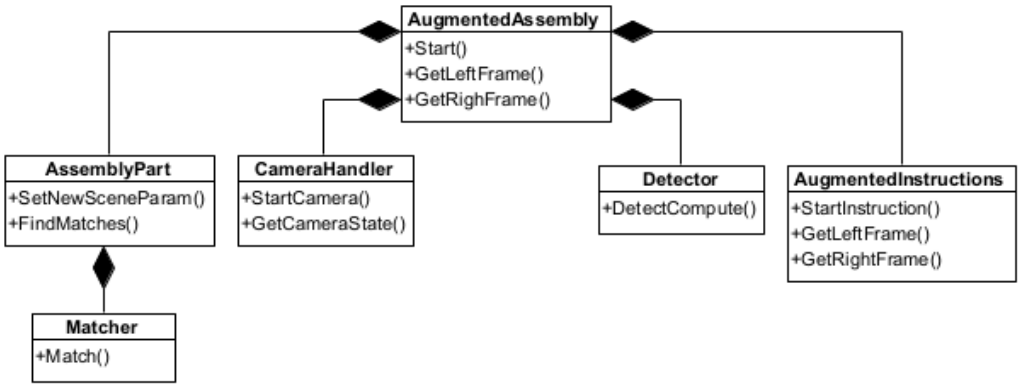
5.4.2 Synchronizácia

Na synchronizáciu medzi vláknami sa používajú z STL knižnice, triedy *std::mutex, std::atomic* a *std::conditional_variable_any*. Atomické premenné slúžia ako signalizačné premenné, pomocou ktorých sa spúšťa, respektíve pokračuje vo vykonávaní. Napríklad inštancia triedy *Detector* po spustení metódy *DetectCompute(...)* čaká na zmenu *std::atomic<bool>*, čím sa signalizuje, že dostala nové dáta do zdieľanej pamäte a môže pokračovať vo vykonávaní.

Na druhú stranu *std::mutex* sa používa pri zápise do zdieľanej pamäte v prípade, ak sa nepoužijú synchronizačné premenné (*std::atomic<bool>*). Tento spôsob je použitý napríklad pre inštanciu triedy *CamerHandler*, ktorá nepretržite predáva snímky do zdieľanej pamäte hlavnému vláknu.

5.5 Štruktúra kódu

Architektúra je zložená zo šiestich tried. Hlavnou triedou, ktorá inštancuje ostatné, je trieda *AugmentedAssembly*. Formy komunikácie medzi jednotlivými inštanciami sú v prílohe B.



Obrázok 5.5-1 - Diagram tried implementácie

5.5.1 Trieda CameraHandler

Táto trieda slúži ako wrapper pre triedu *sl::Camera* ZEDSDK, kde okrem inicializačnej metódy *InitCamera(sl::Camera& zed)* je metóda *StartCamera(...)*. Tá je použitá v samostatnom vlákne, v ktorom sa prevezmú snímky z kamery (pravá + ľavá). Tie sa následne uložia do zdieľanej pamäte. Predávanie snímkov je nepretržité bez žiadnej bariéry. Kamera je nastavená na rozlíšenie 1280x720, kvôli zrýchleniu detekcie objektu, ostrosť obrazu v ZEDSDK je na hodnote 5 a FPS na úrovni 60.

5.5.2 Trieda Detector

Realizácia detekcie významných bodov je pomocou inštancie triedy Detector. V konštruktore sa zvolí použitý detektor (BRISK) a metóda *DetectCompute(...)* uskutočňuje detekciu významných bodov a tvorbu deskriptoru. Významné body sa po detekcii filtrujú priestorovo a podľa odozvy. Neuvažujú sa významné body detegované 20px od hranice obrazu a z tejto podmnožiny sa vyberie 90% s najvyššou odozvou (najsilnejšie). Vo finále sa tak zmenší aj samotný deskriptor, čím sa eventuálne urýchli proces hľadania korešpondencií. Maximálny počet významných bodov je zhora obmedzených na 1000, aby jednak bolo možné hľadať korešpondencie v rozumnom čase, a zároveň aby sa vyhlo realokáciám vektorov pri zmene veľkosti. Táto metóda beží v samostatnom vlákne a s hlavným vláknom si predávajú dáta cez zdieľanú pamäť. S hlavným vláknom zdieľa:

- Matice snímkov z kamery (write)
- Vektory významných bodov a deskriptora scény (read)

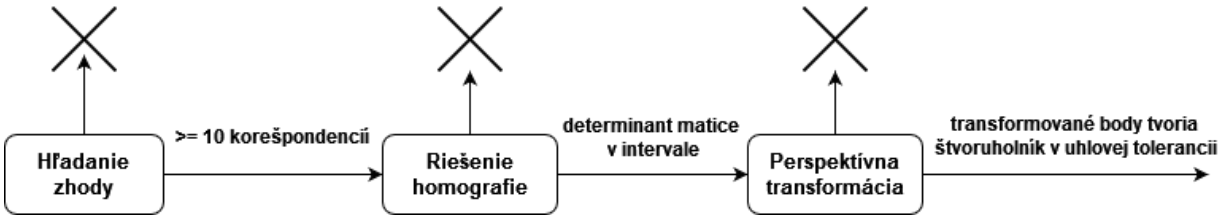
5.5.3 Trieda AssemblyPart

Trieda AssemblyPart obsahuje všetky informácie o danom komponente. Obrázky jednotlivých častí objektu (na perspektívnu transformáciu), ich významné body a deskriptory. Tie sa načítajú zo súborov popisovaných v kapitole 5.2.

Každá inštancia triedy je spustená v samostatnom vlákne cez metódu *FindMatches(...)*, ktorá hľadá korešpondencie medzi deskriptorom scény a objektu pomocou triedy *Matcher*. Počet objektov triedy *Matcher* je rovnaký ako počet hľadaných deskriptorov objektu, tj. pre objekt definovaný 6 stranami je vytvorených 6 objektov triedy *Matcher*. Objekt *AssemblyPart* najprv vystaví flag, že je pripravený na novú sadu významných bodov a deskriptor. To je synchronizované pomocou mutexu a *std::conditional_variable_any*. Vo fáze čakania na nové body je vlákno neaktívne a očakáva notifikáciu od hlavného vlákna, že do zdieľanej pamäte mu boli posunuté nové dáta. Týmto spôsobom je možné selektovať aktívnosť vlákna. S hlavným vláknom zdieľa:

- Vektor významných bodov scény (read)
- Vektor deskriptoru scény (read)
- Vektor polôh rohov každej strany daného komponentu (write)

Následne v slučke *for* sa spúšťa hľadanie korešpondencií pre každý deskriptor objektu. Pri nájdení aspoň 10 korešpondencií sa rieši matica homografie. Ak je determinant v danom intervale, tak sa počíta perspektívna transformácia rohov strany objektu. Keďže sa jedná o *for* cyklus, na zrýchlenie je možné použiť OpenMP, ktoré dodatočne paralelizuje výpočet. V ideálnom prípade bude počet vlákien na počet deskriptorov objektu. V prípade odmietnutia niektorého z výpočtov sa iterácia ukončí, strana objektu sa nedeteguje a vynuluje sa posledná detekcia na indexe iterácie.



Obrázok 5.5-2 - Priebeh slučky pre AssemblyPart

5.5.4 Trieda Matcher

Hľadanie zhody je obalené v triede *Matcher*. Trieda používa metódu *knnMatch(...)* na hľadanie korešpondencie medzi deskriptormi scény a danej strany objektu. Po nájdení korešpondencií sa filtrujú pomocou ratio testu. Pôvodne boli implementované aj iné metódy filtrácie nájdených korešpondencií (priestorová), ale nakoniec neboli nutné. Implementácia triedy ale zostala.

5.5.5 Trieda AugmentedInstructions

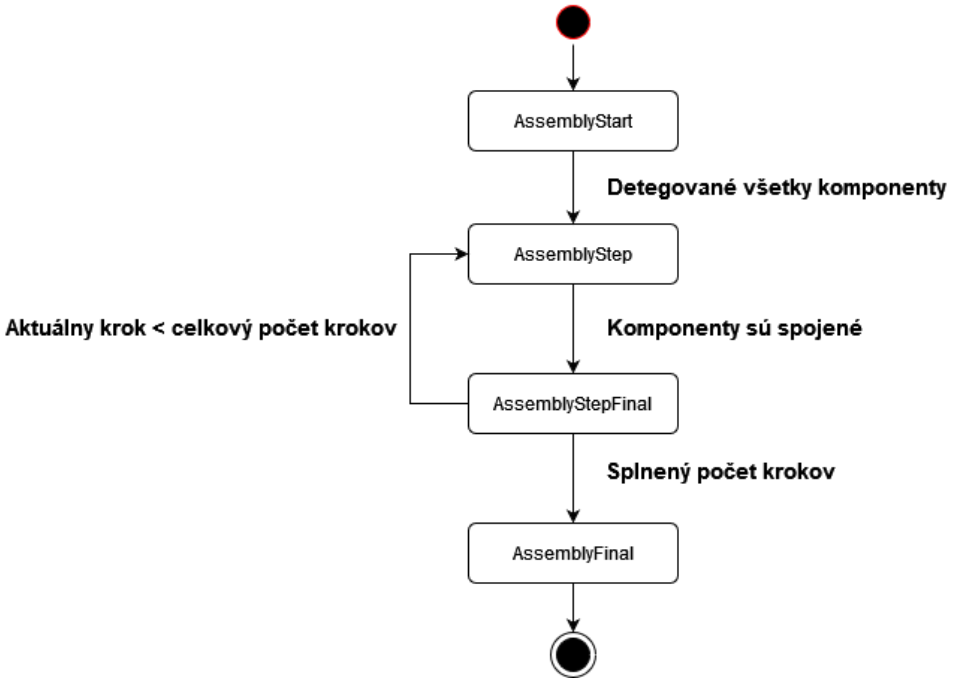
Úlohami inštancie sú reprezentácia manažéra inštrukcií a vizualizácia obsahu. Metóda *StartInstructions()* beží v samostatnom vlákne. S hlavným vláknom zdieľa:

- Vektor polôh rohov každého komponentu (read)
- Obraz z kamery (read)
- Vektor udávajúci aktuálne detekovateľné objekty (write)

Pomocou posledného vektoru sa signalizuje hlavnému vláknu, ktoré objekty detegovať, a ktoré nie (ktoré vlákna objektov *AssemblyPart* má hlavné vlákno notifikovať).

Vo vlákne sa vykonáva vkladanie animácie, vizualizácia inštrukcií (písomných, grafických), ohraničenie detegovaných objektov a ich anotácia. Druhou úlohou je kontinuálne overovanie prechodu stavového automatu. Začína sa v stave *AssemblyStart*, kde po detekcii všetkých komponentov sa prechádza do *AssemblyStep*. V nej sa detegujú iba komponenty potrebné na splnenie kroku a hľadá sa ich vzájomné spojenie (vznik medzikomponentu). Konzekventne sa zredukuje počet hľadaných komponentov a zníži sa tak záťaž na CPU. Po ich vzájomnom spojení sa prechádza do stavu

AssemblyStepFinal, v ktorom sa hľadá výsledný medzikomponent. Po splnení sa opakuje predošlý postup pre *Step* až do dosiahnutia počtu potrebných krokov. Posledným stavom je *AssemblyFinal*, v ktorom sa hľadá výsledný komponent, a tým sa aplikácia môže ukončiť.



Obrázok 5.5-3 – Zjednodušený diagram stavového automatu montáže výrobku

5.5.6 Trieda AugmentedAssembly

Jedná sa o triedu, ktorá koordinuje prácu ostatných inštancií. Hlavnými úlohami sú prevzatie a konverzia snímku z kamery, a kopírovanie dát do zdieľanej pamäte pre *Detector* a inštancie *AssemblyPart*. Pomocou metódy *Start()* sa zaháji sled udalostí celej aplikácie.

5.6 Montážne inštrukcie a vizualizácia

Pre navigovanie užívateľa sa vizualizujú jednotlivé inštrukcie v snímanom obraze. Primárnym vizualizačným prvkom je anotácia objektov. V prípade úspešnej detekcie sa objektu priradí ID(číslo značiace triedu komponentu) a objekt, jeho strana, sa ohraničí.(1)

Prvým typom inštrukcie je textová. Užívateľovi podáva základnú informáciu o ďalšom kroku, napríklad pre jednotlivé stavy.

Tabuľka 5-1 - Inštrukcie v jednotlivých stavoch

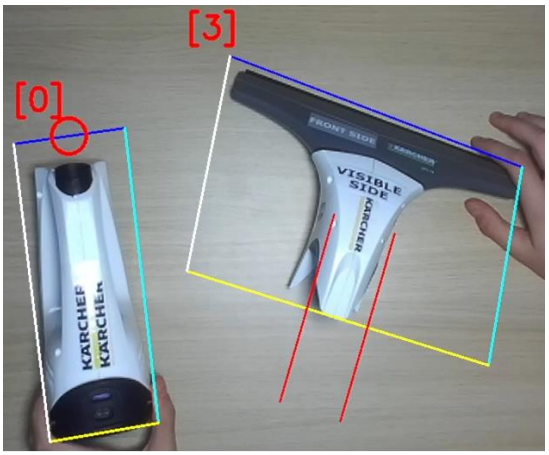
AssemblyStart	`Make sure all the objects are visible`
AssemblyStep	`Connect the object [X] with the object [Y]`
AssemblyFinal	`Assembly has finished sucessfully !`

V prvom stave (Obrázok 5.5-3) sa vyžaduje viditeľnosť všetkých komponentov, teda po ich detekcii je možné začať proces montáže. V stave spájania komponentov sa uvedie, ktoré komponenty, podľa ID, sa majú spojiť. V konečnej fáze sa zobrazí, že proces montáže sa úspešne dokončil. (2)

Ďalší typ inštrukcie je animácia. Jej úlohou je naznačiť užívateľovi, akým spôsobom sa objekty majú spájať, teda akým pohybom. (3)

Počas stavu spájania komponentov sú prvou pridanou inštrukciou naznačené konkrétne miesta, v ktorých sa objekty majú spojiť. Pomocou dvoch paralelných čiar sa indikuje, z ktorého smeru danej strany sa druhý objekt má vložiť. Obdobne na vkladanom objekte sa značí hrana strany, ktorá má byť vkladaná. (4)

Pre uvedenie terminológie, *strana* sa myslí strana objektu, tj. plocha, ktorá sa hľadá v obraze (uzavretá hranami). *Hrana* je uvažovaná ako hrana danej strany, takže ak máme stranu reprezentovanú štvoruholníkom, tak sa jedná o jeho hranu (na obrázku dole znázornené pomocou farebných čiar).



Obrázok 5.6-1 - Vizualizácia vkladanej hrany (pre objekt 0 vľavo) a indikácia miest vloženia (objekt 3 vpravo)

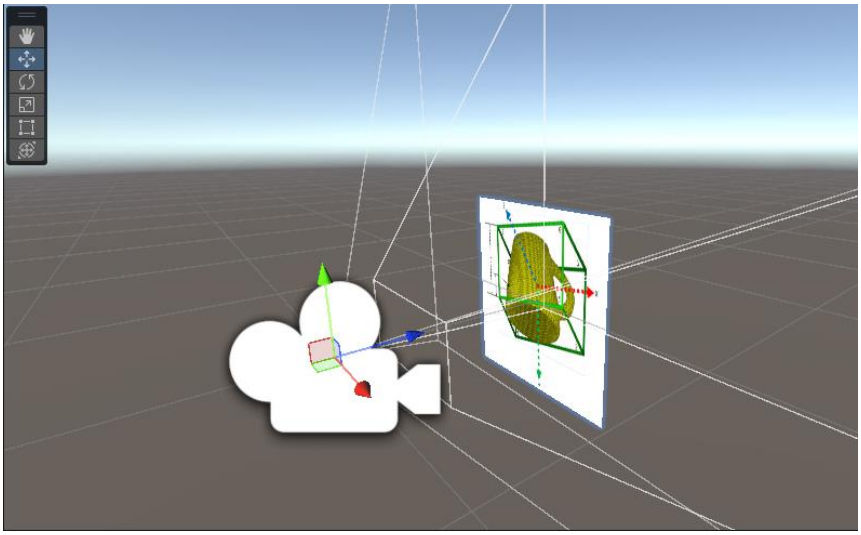
Posledným typom inštrukcie je indikácia správnej strany. V procese montáže sa objekt deteguje pomocou strán, ktorými je definovaný jeho model. Pri spájaní komponentov sa jeden z nich (v tomto prípade ide o druhý v poradí) dezignuje ako vodiaci. Teda podľa vodiaceho komponentu sa určuje súhlasnosť strán komponentov. Inými slovami, ak pre vodiaci komponent je viditeľná strana A, tak pre druhý komponent musí byť rovnako viditeľná strana A. V závislosti od konkrétnej implementácie sa dané strany definujú podľa dvojíc, napríklad A-D, D-A, a podobne. Overujú sa iba štyri základné strany komponentov. Myšlienkou je teda navádzanie užívateľa na správne dokončenie kroku, a zároveň je predpokladom, že strana komponentu je dostatočne viditeľná, aby sa vizualizácia zobrazila správne. Konkrétna inštrukcia je implementovaná ako trojitý záblesk danej strany. Zelená znamená, že strana je v poriadku, naopak červená, že objekt je potrebné otočiť. (5)

5.7 Konfigurácia s Unity a Meta Quest 2

V zásade existujú tri spôsoby vytvárania aplikácií pre Meta Quest, buď pomocou Unity Unreal Engine 5, alebo natívne. V tomto prípade sa jedná iba o jednoduchý prenos obrazu do okuliarov, takže nie je potrebné používať komplexnejšie prostredie ako Unreal Engine. Pre nastavenie samotného prostredia je vhodné použiť návod[5] priamo od Meta. Prvotné je konfigurácia a agregácia potrebných softvérových balíčkov:

- Unity (v. 2022.3)
- Meta Quest Link
- Unity Asset Meta XR All-in-One SDK

Koncepcia scény je priamočiara. Pred zorným poľom kamery sa umiestnia zobrazovacie plochy, na ktoré sa premieta obraz z kamery. Postupnou aktualizáciou textúry plôch sa na nich zobrazí prenos kamery. Do Unity scény sa vloží kamera typu *OVRCameraRig*, ktorá zobrazí pohľad kamery na scénu do okuliarov. Priamo pred ňu sa umiestnia objekty *Quad* (alebo *RawImage*), na ktoré sa zobrazí obraz (pre objekt *Quad* je nutné ešte vložiť *Material*, na ktorý sa obraz renderuje). Zobrazovacie plochy sa navzájom prekrývajú, preto sú maskované pomocou vlastnej vrstvy (Layer). Každá kamera potom pomocou Culling Mask, sníma iba im určené objekty (zobrazovacie plochy) danej vrstvy. Obrázky v Oculusoch sa zobrazujú vertikálne otočené, preto riešením je zvoliť *shader Oculus/OVRMRCameraFrame*, ktorý otáča výsledný obraz. V opačnom prípade by bolo potrebné otáčať obraz v CPU, čo vkladá dodatočnú latenciu.



Obrázok 5.7-1 Unity scéna

[5] https://developer.oculus.com/documentation/unity/unity-tutorial-hello-vr

Unity používa C#, takže implementovaná aplikácia *AugmentedAssembly (*ďalej ako *AA)* bola importovaná ako *.dll* knižnica. V AA boli napísané funkcie (a označené pre export) na vytvorenie/zničenie inštancie AA, jej spustenie a funkcie pre získanie obrazu z oboch kamier (už augmentovaného). Obraz z kamier sa prenáša ako *char[]* a na strane C# sa konvertuje späť na obrázok. Jednotlivé funkcie sa následne v skripte, použitom v Unity, importujú. Ten sa potom priloží aktívnemu *GameObject* (ako napríklad náš *quad*) a v nastaveniach projektu sa pridá do zoznamu vykonávaných skriptov. Samotný skript pozostáva z dvoch základných Unity metód. Metódy *Start()*, ktorá sa spustí po zahájení skriptu. Tu sa vytvoria a nájdu (v scéne) potrebné inštancie. A metódy *Update()*, ktorá mení textúry zobrazovacích plôch (je periodicky volaná enginom na každom frame). Tým by sa mal zabezpečiť chod aplikácie v Unity editore. Pre jej prenos do okuliarov je potrebný dodatočný súbor krokov.

- V Meta Quest Link aplikácii:
  - *Settings* > *General*, povoliť *Unknown sources*, a nastaviť *Meta Quest Link* ako aktívny OpenXR Runtime
- V Unity:
  - *Edit* > *Project Settings* > *XR Plugin management* a zaznačiť *Initialize XR on startup* a *OpenXR* v karte *Plug-in providers* pre Android a Windows/Mac/Linux

Aplikácia je potom viditeľná a spustiteľná v okuliaroch, a zároveň sa renderuje obraz.

Čo sa týka šírky zbernice, tak okuliare sú pripojené k PC cez USB.3.1 gen 1, ktorá pre jeden port dosahuje rýchlosti 2.0 Gbps. Potrebný bandwidth sa vypočíta ako $\frac{Rozlíšenie*počet\ kanálov*počet\ obrázkov*FPS}{1024*1024*8}$ [Gbps]. Pre 30 FPS, dvoch snímkov z kamery, a obrázku 1280*720*4 ≈ 2.11 Gbps, čo je hranične dostačujúce, avšak vznikne dodatočná latencia. Finálna latencia je ovplyvnená aj faktom, že samotná aplikácia *AugmentedAssembly* je spustená na rovnakom systéme a súperi o zdroje s Unity enginom. V ideálnom prípade je beh oboch aplikácií na odlišných hardvérových zostavách alebo na zariadení, ktoré dokáže obstarať obe naraz.

6. EXPERIMENTY A VYHODNOTENIE

Táto kapitola sa venuje vyhodnoteniu výsledkov implementovanej metódy rozpoznávania . Primárnym účelom je evaluovať samotný algoritmus, preto experimenty prebiehali na izolovanej aplikácii *AugmentedAssembly,* tj. bez okuliarov. Použitým hardvérom boli procesor *Intel® Core™ i7-8750H 2.20-4.10 GHz* a 16 GB DDR4 RAM. Prvou metrikou bola latencia detekcie, potom jej presnosť a tolerancia voči oklúzii.

6.1 Latencia

Jeden z najdôležitejších parametrov v úlohách tohto charakteru je rýchlosť detekcie. Najmä v dynamickej scéne, keď sa pohybuje objekt alebo kamera a v obraze sa tak mení poloha rozpoznávaného objektu. Motiváciou je včasne inštruovať operátora.

Štruktúra implementovaného algoritmu pozostáva z 3 základných krokov:

1. Detekcia významných bodov
2. Hľadanie korešpondencie
3. Vizualizácia

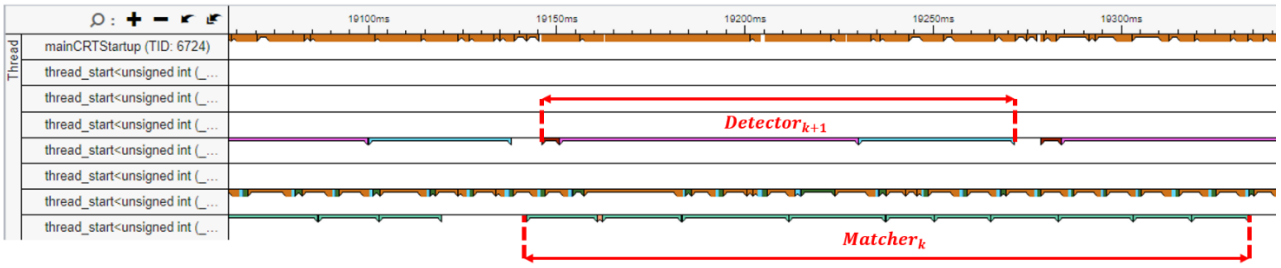
Každý krok vkladá formu latencie, preto je vhodné úlohy paralelizovať. Síce paralelizáciou tiež vznikajú formy latencie, kedy vlákna na seba čakajú alebo nemajú ešte aktuálne dáta, avšak celkový čas detekcie sa radikálne zníži..

Na analýzu jednak rýchlosti, paralelizácie a použitia OpenMP sa zvolil nástroj **VTune**. Ide o pokročilý profiler, ktorý analyzuje výkonnosť a efektivitu sériového alebo paralelného kódu na x86 Linuxe alebo Windows operačných systémoch. Dokáže analyzovať aplikáciu na CPU, GPU alebo FPGA a v rôznych kombináciách programovacích jazykov, čo z neho vytvára mocný nástroj pri vývoji aplikácií.

Používa sa napríklad na optimalizáciu kódu, hľadanie tzv. „hotspots“ alebo „bottlenecks“, analýzu efektívneho využitia pamäti (využitie cache pamätí, prístup ku pamäti), paralelizmus (identifikácia problémov vo viacvláknových aplikáciách, efektivita využitia CPU, vektorizácia,...), optimalizácia GPU/FPGA kernelov, analýza I/O intenzívnych aplikácií alebo viacuzlové aplikácie s použitím MPI a OpenMP.

VTune je možné stiahnuť ako samostatnú aplikáciu a zároveň je možné VTune integrovať ako rozšírenie priamo do Microsoft Visual Studia. Samotné použitie spočíva v priložení procesu alebo binárneho súboru programu na sledovanie. Následne si je možné zvoliť, ktorú konfiguráciu, respektíve, ktorú analýzu chceme obdržať. Výsledkom je niekoľko súborov pozorovaných dát, ktoré je možné otvoriť v profileri a získať tak ich grafickú reprezentáciu.

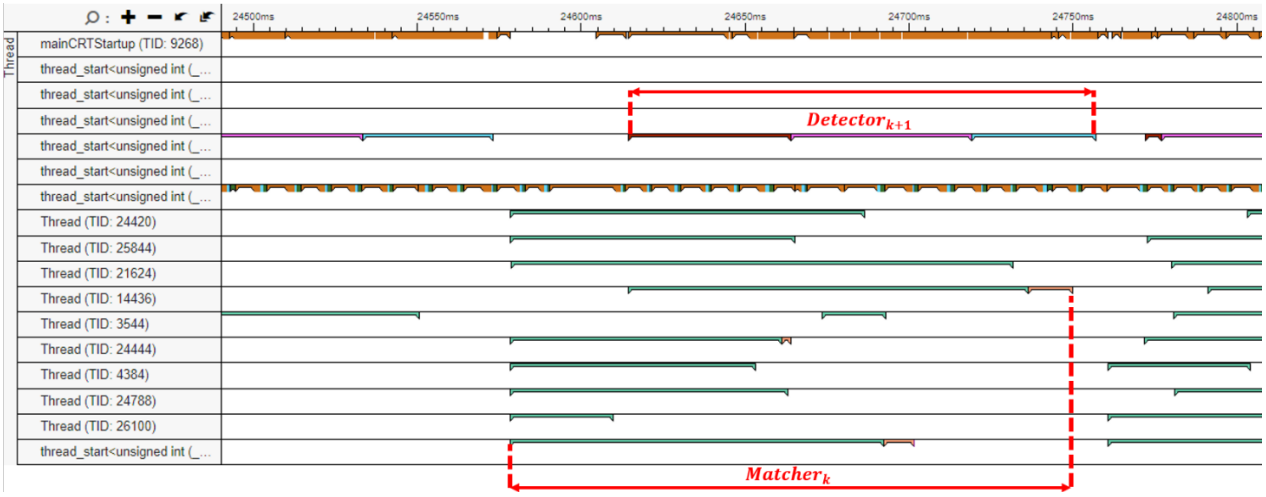
Algoritmus je teda rozdelený na tri kroky. Ideálny stav nastáva, keď hľadanie korešpondencie sa pre jednotlivé objekty prekryje s detekciou významných bodov. Tým sa zabezpečí, že pre výstup detektoru v kroku ***k***, sa hľadá korešpondencia a samotný objekt počas detekcie významných bodov v kroku ***k+1*** (Obrázok 6.1-2).



Obrázok 6.1-1 - Výstup profileru pre verziu bez OpenMP (2000 významných bodov)

Vidíme, že v prípade ak sa nepoužije OpenMP (Obrázok 6.1-1), môže trvanie procesu hľadania zhody trvať dlhšie ako detekcia významných bodov, čím sa prakticky zahodí jeden snímok. Naopak pri použití OpenMP (Obrázok 6.1-2) je opačný stav viac konzistentný, kedy je detekcia včasná. Počet vytvorených vlákien sa rovná počtu strán modelu objektu.

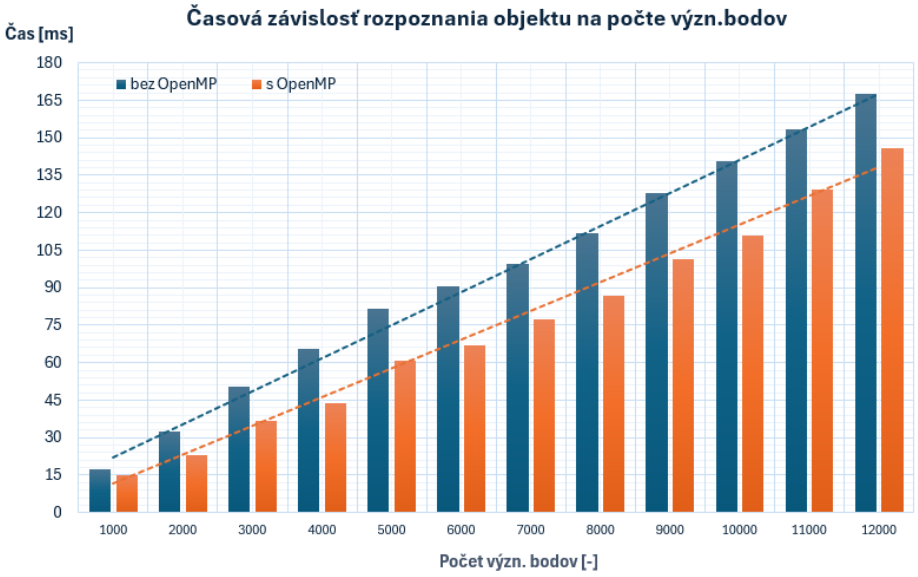
Na obrázku dole je zároveň vidieť oneskorené spustenie jednotlivých vlákien pre *Matcher* (tyrkysová farba), kvôli obmedzenému počtu paralelne vykonávateľných vlákien (konkrétne 12 pre použitý systém). Predĺži sa tak celkový čas rozpoznávania a môže dochádzať k dodatočnému oneskoreniu voči detektoru.



Obrázok 6.1-2 - Výstup profileru pre verziu s OpenMP (2000 významných bodov)

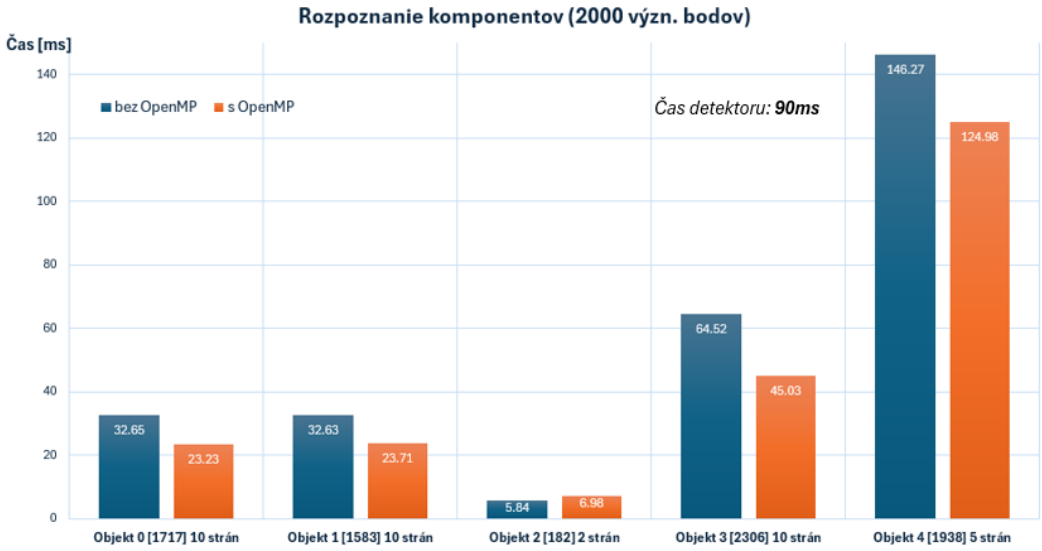
Podstatné je ideálne disponovať počtom vlákien na počet hľadaných strán, čím by sa prekryla detekcia významných bodov a hľadania korešpondencií. Vyvážením záťaže pomocou dodatočných direktív OpenMP, v závislosti na veľkostiach porovnávaných deskriptorov, je možné navyše redukovať počet potrebných vlákien.

Použitím OpenMP sa rovnako urýchli celkový proces detekcie objektu. Na obrázku 6.1-3Obrázok 6.1-3 je vidieť, že sa jedná o lineárnu závislosť a zrýchlenie je na úrovni 20-30% oproti variante bez OpenMP. Realizovalo sa 100 meraní pre daný počet významných bodov a zaznamenával sa čas trvania detekcie objektu (hľadanie zhody + riešenie homografie + perspektívna transformácia).



Obrázok 6.1-3 - Graf časovej závislosti rozpoznania objektu od počtu významných bodov (pre objekt 0)

Na rýchlosť rozpoznania objektu vplýva viacero faktorov. Jedným je počet rozpoznávaných strán, čo súvisí s potrebným počtom vlákien. Ďalším faktorom je počet významných bodov v scéne a počet významných bodov objektu, respektíve jeho strán. V tomto smere môžu nastať tri prípady. Počet významných bodov pre každú stranu je porovnateľný alebo je možné pri výpočte záťaž vyvážiť (1) – ideálna granularita. Ako môžeme sledovať na obrázku 6.1-4. pre objekt č. 2, počet významných bodov je nízky, kedy už úlohu paralelizovať nedáva zmysel. Tam už réžia ohľadom vlákien navyšuje čas oproti sekvenčnej verzii - granularita úlohy je príliš jemná (2). Tretí prípad je opačný, kedy sa rozpoznáva menší počet strán, ktoré však obsahujú značné množstvo významných bodov a tak úlohu nie je možné na úrovni vlákien viac paralelizovať – príliš hrubá granularita (objekt č. 4) (3).



Obrázok 6.1-4 - Graf trvania rozpoznania pre jednotlivé komponenty (2000 význ. bodov)

Ďalší zdroj latencie je LLC (last-level cache) miss, ktoré nútia procesor pristupovať až k pamäti DRAM, čím sa zastaví exekúcia na niekoľko cyklov. To nastáva najmä pri práci nad maticami pomocou OpenCV, pravdepodobne pri kopírovaní matíc pre jednotlivé vlákna.

V reáli sú merané časy vyššie kvôli prepínaniu kontextu vlákien iných procesov, ktoré sú spustené na pozadí. Nastávajú tak napríklad aj výplachy pamätí cache, ktoré spätne súvisia s predošlým bodom. Ideálne by na dedikovanom systéme boli spustené iba vlákna aplikácie.

Celkový čas rozpoznania objektu je teda závislý na:

1. Čase detekcie významných bodov
2. Počte strán objektu ($\approx$ Počet potencionálne súbežne vykonávajúcich vlákien)
3. Počte významných bodov v scéne
4. Granularite/rozložení významných bodov objektu

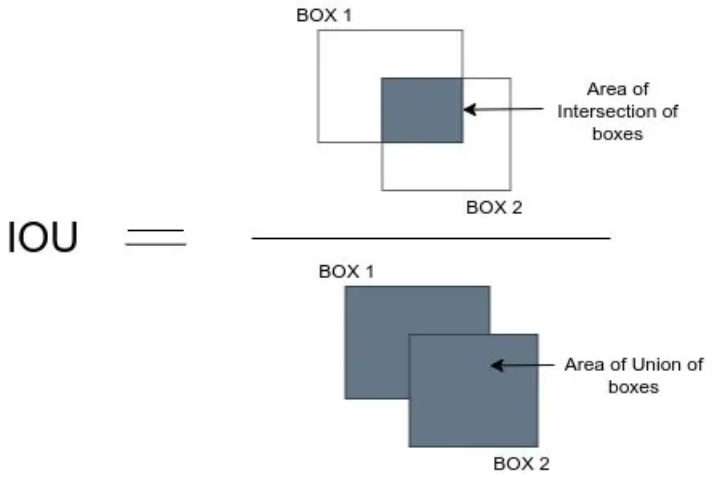
Na zhrnutie, žiadúcim predpokladom je prekrytie výpočtu detekcie významných bodov a hľadania korešpondencie pre jednotlivé objekty. V princípe počas skladania výrobku sa jedná o maximálne 2 súbežne hľadané komponenty, a tak pre priblíženie sa real-time času musí platiť $\sum strán\ komponentov(2) = počet\ súbežných\ vlákien,$ alebo pri vyvážení záťaže môže byť počet vlákien nižší. Samotný počet významných bodov v scéne ovplyvňuje rýchlosť detekcie a rovnako aj rýchlosť hľadania korešpondencie. V neposlednom rade granularita a rovnomerné rozloženie významných bodov na objekte, i.e. počet strán a významných bodov.

6.1.1 Úzke hrdlo

Analýzou je možné usúdiť, že potencionálne najväčšie úzke hrdlo je použitý detektor. V prípade hľadania korešpondencie jednotlivých strán sa úloha paralelizáciou rozloží na viac vlákien/jadier, avšak detektor je knižničná funkcia. Pravdepodobne sa tiež interne paralelizuje, ale z implementačného pohľadu ju nie je možné ďalej upravovať. Jedná sa teda o kompromis použitého detektoru (BRISK) medzi rýchlosťou a invariantnosťou na afínne transformácie/zmeny pohľadu. Nižšia invariantnosť znamená potrebu väčšieho množstva snímkov objektu, čo naopak predlžuje proces hľadania korešpondencie v databáze. Potom by sa karta obrátila a úzkym hrdlom by sa stal proces rozpoznávania a nie detekcia významných bodov.

6.2 Oklúzia a presnosť

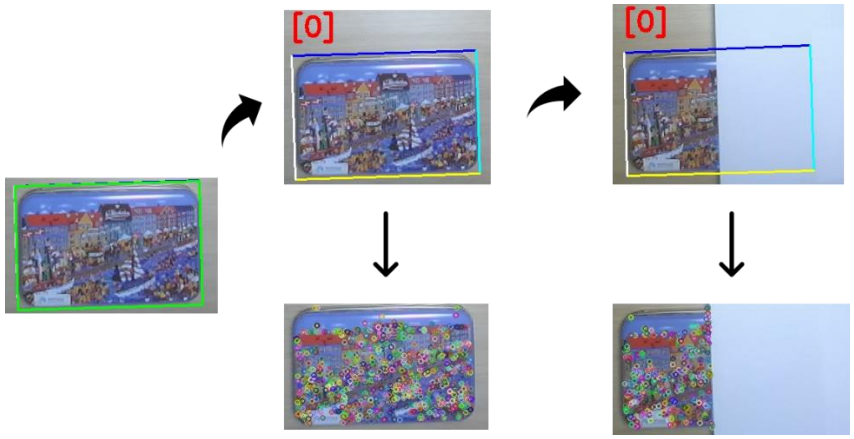
Vyhodnotenie oklúzie prebiehalo nasledujúco. Vybral sa objekt s bohatou a približne rovnomerne rozloženou textúrou. V princípe to znamená, že hustota významných bodov je približne rovnaká na celom objekte. Experiment prebiehal na jednej strane objektu. Objekt sa postupne s 10% krokom zakrýval pomocou jednoliateho papiera, na ktorom sa nenachádzajú významné body. Komponent bol snímaný z konštantnej vzdialenosti (60cm) a nachádzal sa na rovnakom mieste. Spoločne s oklúziou sa použila metrika IoU („Intersection over union“), ktorá určuje presnosť detekcie objektu (lokalizáciu), tým pádom zároveň vymedzuje kvalitu detekcie, jej správnosť. Druhá použitá metrika bola tzv. „reprojection error“ – premietacia chyba. Pre výpočet matice H pomocou homografie platí: $H \epsilon < 0.2 ; 2.0 >$, z dôvodu obmedzenia nesprávnosti výsledkov. Pre hodnoty mimo interval mohol nastať stav, kedy perspektívna transformácia premietla body do záporných hodnôt (mimo obraz).



Obrázok 6.2-1 - Ilustrácia výpočtu IoU[6]

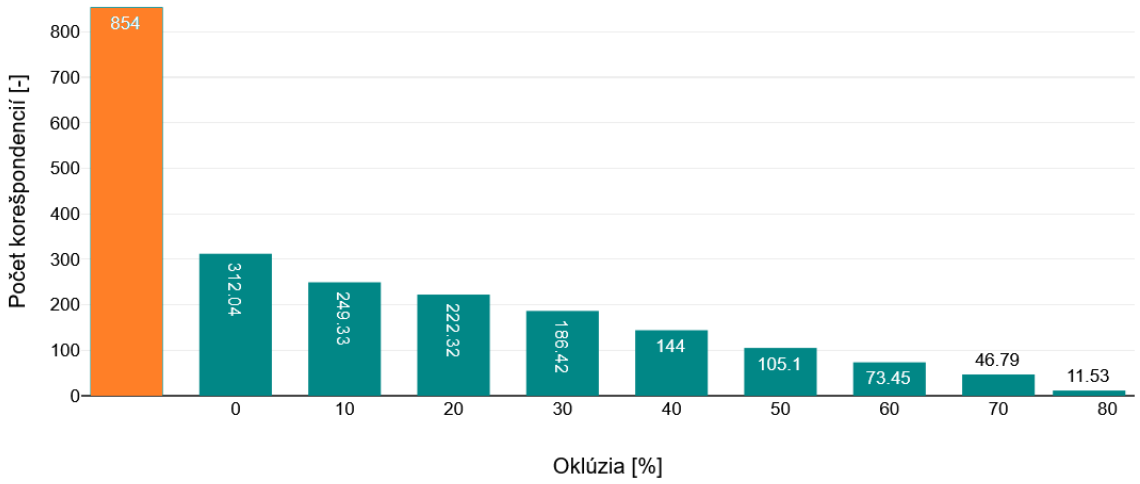
IoU je metrika bežne používaná v oblasti rozpoznávania. Pri detekcii sa zvyčajne objekt vyhradí tzv. „bounding box“ (ďalej ako BB), ktorý ohraničuje detegovaný objekt. Vzniká tak predikovaný BB pre daný objekt a IoU teda uvádza pomer správnosti predikovaného BB k reálnemu. Vzťah pre IoU sa definuje ako $\frac{Plocha\ prieniku}{Plochu\ zjednotenia}$ oboch BB. Pri perfektnom prekrytí oboch BB (alebo ak je predikovaný BB menší a vo vnútri reálneho) sa IoU = 1, inak hodnota klesá k 0.

[6] https://medium.com/analytics-vidhya/iou-intersection-over-union-705a39e7acef



Obrázok 6.2-2 - Spôsob vyhodnocovania oklúzie a presnosti detekcie

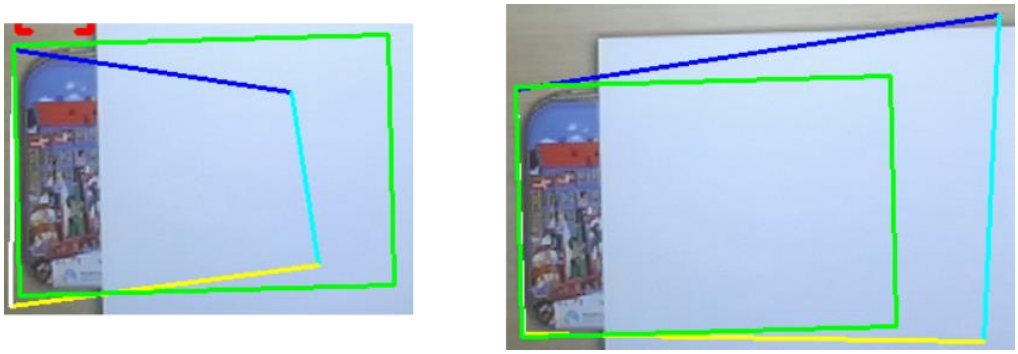
Na začiatku sa vymedzil správny BB pre objekt. V ďalšom kroku sa vyhodnocoval počet nájdených korešpondencií, po filtrácii „ratio testom“, a IoU. Vľavo na obrázku 6.2-2 je správny (ground truth) BB. V hornej rade sú ilustrované predikované BB vypočítané pomocou matice homografie a perspektívnej transformácie. V dolnej rade sú zobrazené všetky detegované významné body. Experiment prebiehal v intervale od <0, MAX> s krokom 10%, kde MAX sa približne určilo ako hodnota 78,32%. Pri zvyšovaní tejto hodnoty oklúzie, objekt už nebolo možné lokalizovať. Jednotlivé merania prebiehali po 100 prvkoch (detekciách).



Obrázok 6.2-3 - Počet detegovaných korešpondencií v závislosti od oklúzie

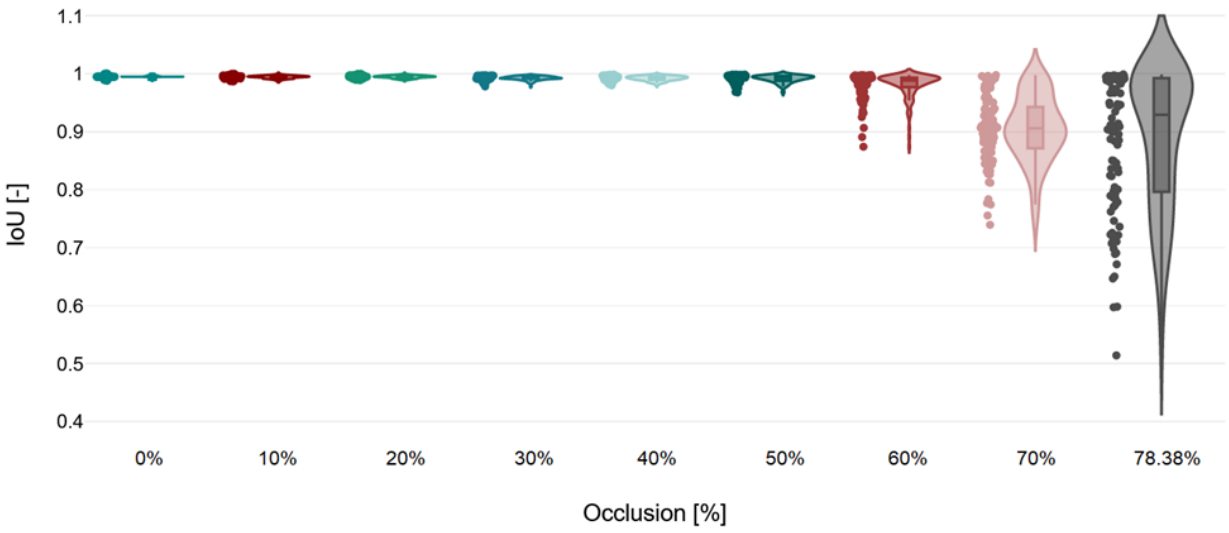
Z pôvodných 854 významných bodov objektu sa deteguje pri nulovej oklúzii v priemere 312,04 korešpondencií, čo je zhruba tretina z celkového počtu. Tu vidíme, že síce 854 bodov bolo určených ako významných, avšak reálne má dostatočnú odozvu len časť z nich. Absencia detekcie objektu nastala niekde v priemere 11,53 korešpondencií. V prípade menšieho počtu korešpondencií už matica homografie nesplňovala metriku

alebo perspektívna transformácia nebola schopná transformovať rohy strany do obrazu. Na základe tejto hodnoty sa v implementácii určuje, či vôbec započať proces hľadania matice homografie a následnej perspektívnej transformácie.



Obrázok 6.2-4 - Ilustrácia nepresnej lokalizácie predmetu s oklúziou. Vľavo (**IoU = 0.983; $\mathbf{mean_{dist}}$ = 34.57**), vpravo (**IoU = 0.697; $\mathbf{mean_{dist}}$ = 36.36**)

Pri zvyšovaní oklúzie predmetu sa môžu správne určiť niektoré rohy predmetu, avšak na základe matice homografie, perspektívna transformácia nesprávne určí tie zvyšné. To má za výsledok stav, kedy lokalizácia predmetu je nestabilná voči reálnemu BB. Nastávajú detekcie ako na obrázku hore.



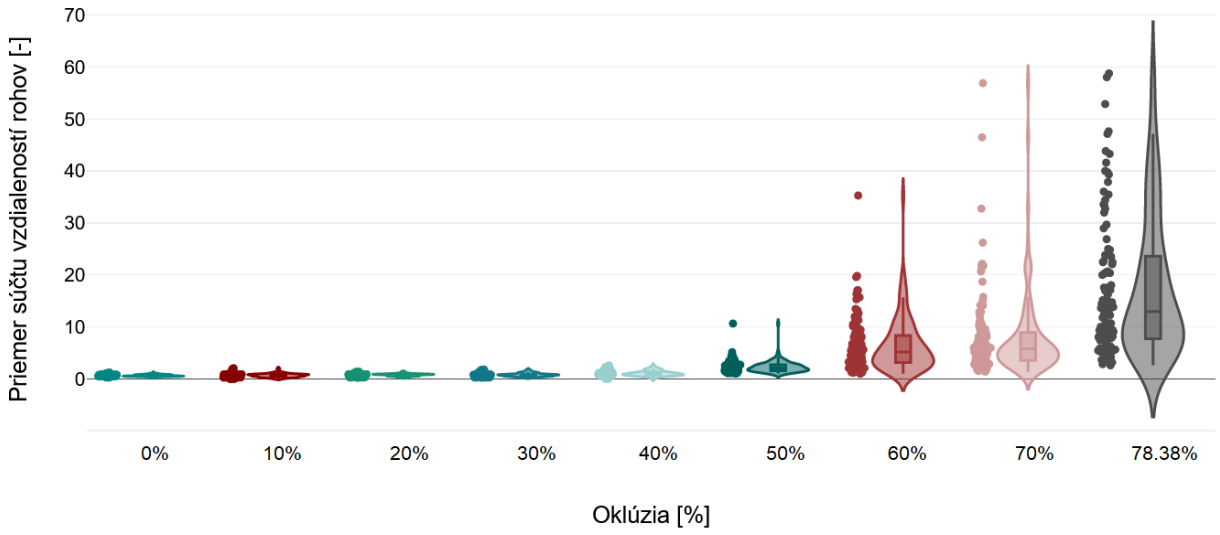
Obrázok 6.2-5 - Distribúcia IoU pre hodnoty oklúzie

Pre ilustráciu distribúcie tejto variácie sa zmeralo 100 hodnôt IoU pre dané hodnoty oklúzie. Môžeme sledovať, že relatívne stabilné hodnoty IoU sa dosahujú v okolí 50%. Avšak ďalším zvyšovaním oklúzie sa destabilizuje presnosť lokalizácie predmetu. Pre hodnotu 70% je priemer IoU 0.90 a pre hraničnú hodnotu (78.38%) okolo 0.89. Dôležitá

je však distribúcia týchto hodnôt a minimálna hodnota. Pre hraničnú hodnotu sa značný počet detekcií nachádza niekde pod hranicou 0,8. Keďže operátor vníma každú detekciu, je zaujímavá otázka, či uprednostniť absenciu detekcie pred nedostatočnou.

Navyše pre hodnoty oklúzie blízke maximálnej hodnoty nastáva aj stav, kedy predikovaný BB je trochu zdeformovaný a kompletne vo vnútri reálneho BB. Tu metrika IoU vyhodnocuje, že ide o správnu detekciu, avšak nie úplne korektnú (čo sa týka tvaru a rozmeru, Obrázok 6.2-4). Preto napríklad na obrázku 6.2-6 vidíme nárast, napríklad mediánu, IoU pre hraničnú hodnotu oklúzie oproti nižším.

Premietacia chyba môže slúžiť ako ďalšia metrika. V tomto prípade sa porovnávajú vzdialenosti premietaných bodov a bodov výsledných. Pri perspektívnej transformácii sa jedná o 4 body, takže vo výsledku ide o priemer súčtu štyroch vzdialeností. Znova sa uskutočnilo 100 meraní.



Obrázok 6.2-6 - Ilustrácia distribúcie premietacej chyby

Použitím premietacej chyby je jasnejšie viditeľný vplyv oklúzie. Z obrázku 6.2-6 si môžeme všimnúť, že premietacia chyba okolo 30 je už značná pri objekte tohto rozmeru a snímania zo vzdialenosti 60cm. Chyba rovnako ako v prípade IoU začne rásť v okolí 50% hodnoty oklúzie.

6.3 Detekcia objektu

Všeobecne detekcia objektu pomocou zvolenej metódy závisí na jeho použitom modeli. Ako bolo diskutované, model pozostáva zo snímkov objektu, pohľadov. Počet týchto snímkov úzko súvisí s veľkosťou databáze deskriptorov a jej zväčšovaním sa predlžuje čas rozpoznania objektu. Preto je otázka kompromisu medzi rozpoznaním objektu v každom snímku, v každom pohľade, alebo uprednostniť včasnú detekciu.

Ďalším úskalím môže byť snímanie objektu z úzkeho profilu. Vtedy sa počet významných bodov objektu a následné korešpondencie s vysokou odozvou (silné korešpondencie) blížia k nule a nie je možné objekt detegovať. Zároveň zvoleným postupom tvorby datasetu sa eliminujú body na kontúrach objektu, tj. v okolí hraníc obrazu (Obrázok 5.1-4), takže celkový počet bodov je nižší.

Rozpoznanie objektu je teda konzistentné v prípade dostatočného množstva významných bodov strany (pohľadu) telesa, a že daný pohľad je obsiahnutý v databáze. V prípade neprítomnosti sa objekt jednoducho nedeteguje. Ďalším faktorom je použitá metóda detekcie a popisu významných bodov – ich invariantnosť na zmenu pohľadu a afínne transformácie. Napríklad nepravidelnosť, konkávnosť alebo konvexnosť telesa zmenou pohľadu značne mení rozloženie významných bodov a finálny popis telesa. Vtedy je potrebné väčšie množstvo snímkov, avšak pohľady sú jemne dvojzmyselné, a môžu sa detegovať obe strany naraz. V tomto smere by sa už ideálne použil 3D model objektu na jednoznačnú detekciu.

6.4 Zhrnutie

Prvým faktorom je latencia. Celkovo o meraní latencie by sa malo uvažovať v relatívnych číslach, keďže najväčším faktorom je použitý hardvér (počet jadier, inštrukčná sada, frekvencia a iné) a izolácia testov. Pri už konkrétnej implementácii sa paralelné procesy optimalizujú na daný hardvér. Je však viditeľné, že algoritmus pri správnej paralelizácii jednotlivých úloh dokáže rozpoznať objekty v reálnom čase.

Presnosť detekcie vychádza z vlastností a riešení homografie pomocou metódy RANSAC. Nájdením dostatočného množstva silných korešpondencií je RANSAC schopný transformovať model do scény s vysokou presnosťou. Závisí však na rozložení významných bodov telesa, respektíve hľadanej strany.

Vplyv oklúzie rieši tiež v zásade metóda RANSAC, ktorá je schopná transformovať model do scény aj v prípade značného množstva chýbajúcich dát. Pre upresnenie by sa však o oklúzii objektu malo uvažovať ako o oklúzii významných bodov. Usporiadanie významných bodov na povrchu objektu tak udáva odolnosť voči oklúzii, čo znamená, že ak napríklad objekt disponuje významnými bodmi iba v jednej jeho časti, tak zakrytím zvyšnej plochy sa detekcia objektu neovplyvní a naopak ich zakrytím sa objekt nedeteguje.

7. ZÁVER

Podstatou práce bolo vytvoriť algoritmus detekcie (častí) priemyselného výrobku, ktorý by bolo možné eventuálne implementovať v zariadeniach AR/VR. V prvom kroku bola vykonaná rešerš aktuálne dostupných zariadení rozšírenej, respektíve virtuálnej reality. V kapitole druhej sa uviedli metódy rozpoznávania objektu v scéne s apriori vytvoreným modelom. Ako experimentálnou úlohou sa zvolila priemyselná montáž výrobku rozložiteľného na časti, v ktorej by finálny program bol schopný navádzať užívateľa k jeho zostaveniu pomocou virtualizovaných montážnych inštrukcií. V rámci hardvéru sa vybrala kombinácia stereo kamery ZED-mini a VR okuliarov Meta Quest 2, ktoré ponúkajú široké možnosti a podporu zo strany výrobcov. Zvolenou metódou detekcie objektu bolo použitie lokálnych príznakov, pomocou ktorých sa popísal model objektu. V tomto prípade sa problematika rozpoznávania 3D objektu dekomponovala na viacnásobný 2D problém, v ktorom sa objekt definoval jeho stranami. Konkrétne sa zvolil detektor/deskriptor významných bodov BRISK, ktorý je schopný dosiahnuť real-time rýchlosti, a zároveň je robustný z hľadiska invariancie na geometrické vplyvy. Z pohľadu implementácie sa na vyriešenie vplyvu oklúzie použila metóda RANSAC pri hľadaní transformácie modelov strán do scény. Výsledná viacvláknová aplikácia bola napísaná v C++ s použitím OpenCV, ZEDSDK a OpenMP. V experimentálnej časti bola diskutovaná latencia detekcie, kde je možné dosiahnuť real-time rýchlosti. Závisí to však na použitom hardvéri a na rozložení a počte významných bodov v scéne a na objektoch. V rámci presnosti detekcie sa dosahujú dobré výsledky aj v prípade značnej oklúzie objektu, kde v ideálnych podmienkach môže dochádzať okolo 50% oklúzii s vysokou presnosťou detekcie a až takmer k 80% pri stálej schopnosti detekcie. Finálna aplikácia bola prenesená aj do samotných okuliarov, avšak výška latencia už bola značná.

Práca do budúcna

Pokračovaním práce by mohla byť implementácia detekcie arbitrárneho objektu popísaného 3D modelom. Proces tvorby modelu by bol rovnaký, od akvizície snímkov z rôznych pohľadov, detekcie významných bodov, ale v nasledujúcej fáze by sa pomocou napríklad SfM (structure from motion) metódy vytvoril 3D model objektu. Ten by bolo možné vyhľadávať v obraze pomocou PnP (Perspective-n-Point) algoritmu.

Ďalší smer by sa mohol zaoberať implementáciou algoritmu na GPU, kde by bolo možné dosiahnuť real-time rýchlosti pre objekty s ľubovoľným počtom významných bodov.

V neposlednom rade plnohodnotná implementácia v okuliaroch. Pravdepodobne by bolo vhodné vytvoriť celú pipeline renderovania obrazu priamo do okuliarov, napríklad v OpenGL alebo v niektorých herných enginoch.

LITERATÚRA

[1] WERRLICH, S.; EICHSTETTER, E.; NITSCHE, K. a NOTNI, G. An Overview of Evaluations Using Augmented Reality for Assembly Training Tasks. *International Journal of Computer and Information Engineering*. 2017, roč. 11, č. 10, s. 1080-1085.

[2] RATCHEV, Svetan (ed.). *Smart Technologies for Precision Assembly: 9th IFIP WG 5.5 International Precision Assembly Seminar, IPAS 2020 Virtual Event, December 14–15, 2020 Revised Selected Papers*. Springer Cham, 2020. ISBN 978-3-030-72632-4.

[3] DOOLANI, Sanika; WESSELS, Callen; KANAL, Varun; SEVASTOPOULOS, Christos; JAISWAL, Ashish et al. A Review of Extended Reality (XR) Technologies for Manufacturing Training. In: *Technologies 2020, 8(4), 77*. MDPI, 2020. Dostupné z: https://doi.org/10.3390/technologies8040077.

[4] BLATTGERSTE, Jonas; STRENGE, Benjamin; RENNER, Patrick; PFEIFFER, Thies a ESSIG, Kai. Comparing Conventional and Augmented Reality Instructions for Manual Assembly Tasks. In: *PETRA '17: Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*. New York, NY, United States: Association for Computing Machinery, 2017, s. 75-82. ISBN 9781450352277. Dostupné z: https://doi.org/10.1145/3056540.3056547.

[5] DESHPANDE, Abhiraj a KIM, Inki. The effects of augmented reality on improving spatial problem solving for object assembly. *Advanced Engineering Informatics*. 2018, č. 38, s. 760–775.

[6] SEGOVIA, Daniel; MENDOZA, Miguel; MENDOZA, Eloy a GONZÁLEZ, Eduardo. Augmented Reality as a Tool for Production and Quality Monitoring. In: *Procedia Computer Science*. 2015, s. 291-300. ISSN 1877-0509. Dostupné z: https://doi.org/https://doi.org/10.1016/j.procs.2015.12.250.

[7] NINI, G. a DURA, M. Dalle. Application of Augmented Reality Techniques in Through-life Engineering Services. In: *Procedia CIRP*. 2015, Pages 14-23. ISSN 2212-8271. Dostupné z: https://doi.org/https://doi.org/10.1016/j.procir.2015.07.044.

[8] BOUD, A.C.; HANIFF, D.J.; BABER, C. a STEINER, S.J. Virtual reality and augmented reality as a training tool for assembly tasks. In: *1999 IEEE International Conference on Information Visualization*. London, UK: IEEE, 1999. ISBN 0-7695-0210-5. ISSN 1093-9547. Dostupné z: https://doi.org/10.1109/IV.1999.781532.

[9] VALIMONT, R.B.; VINCENZI, D.A.; GANGADHARAN, S.N. a MAJOROS, A.E. The effectiveness of augmented reality as a facilitator of information acquisition. In: *The 21st Digital Avionics Systems Conference*. Irvine, CA, USA: IEEE, 2002. ISBN 0-7803-7367-7. Dostupné z: https://doi.org/10.1109/DASC.2002.1052926.

[10] PATHOMAREE, N. a CHAROENSEANG, S. Augmented reality for skill transfer in assembly task. In: *ROMAN*. Nashville, TN, USA: IEEE, 2005. ISBN 0-7803-9274-4. ISSN 1944-9437. Dostupné z: https://doi.org/10.1109/ROMAN.2005.1513829.

[11] BILLINGHURST, Mark; CLARK, Adrian a LEE, Gun. *A Survey of Augmented Reality*. Now Foundations and Trends, 2015. ISBN 9781601989215.

[12] MURA, M. Dalle a , G. Augmented Reality in Assembly Systems: State of the Art and Future Perspectives. In: *IFIP Advances in Information and Communication Technology*. 2021. Dostupné z: https://doi.org/10.1007/978-3-030-72632-4_1.

[13] WANG, Zhuo; BAI, Xiaoliang; ZHANG, Shusheng; BILLINGHURST, Mark; HE, Weiping et al. A comprehensive review of augmented reality-based instruction in manual assembly, training and repair. Online. *Robotics and Computer-Integrated Manufacturing*. 2022, č. 78. Dostupné z: https://doi.org/https://doi.org/10.1016/j.rcim.2022.102407.

[14] *RealWear Developer Documentation*. Online. 2023. Dostupné z: https://developer.realwear.com/docs/developer-guide. [cit. 2023-10-16].

[15] *RealWear Navigator 520*. Online. In: Realwear. 2023. Dostupné z: https://shop.realwear.com/products/realwear-navigator-520. [cit. 2023-10-16].

[16] *RealWear Hyperdisplay*. Online. Realwear. 2023. Dostupné z: https://www.realwear.com/hyperdisplay. [cit. 2023-10-16].

[17] GERDENITSCH, Cornelia; DEINHARD, Lisa; KERN, Bettina; HOLD, Philipp a EGGER-LAMPL, Sebastian. Cognitive Assistance to Support Maintenance and Assembly Tasks: Results on Technology Acceptance of a Head-Mounted Device. In: *Smart Technologies for Precision Assembly*. 11. Springer Cham, 2020, s. 276-284. ISBN 978-3-030-72631-7.

[18] *XREAL Light AR Glasses*. Online. Deviestore. 2023. Dostupné z: https://deviestore.com/product/nreal-light-dev-kit/. [cit. 2023-10-11].

[19] *Nreal Light Dev Kit*. Online. In: Nreal. 2023. Dostupné z: https://shop.nreal.ai/cart. [cit. 2023-10-11].

[20] MAZZAMUTO, Michele; RAGUSA, Francesco; RESTA, Alessandro; MARIA FARINELLA, Giovanni a FURNARI, Antonino. *A Wearable Device Application for Human-Object Interactions Detection*. Research paper. University of Catania, IT: University of Catania, 2022.

[21] *XREAL Glasses*. Online. Xreal. 2023. Dostupné z: https://xreal.gitbook.io/nrsdk/nrsdk-fundamentals/xreal-devices/readme. [cit. 2024-10-11].

[22] *NRSDK User Guide*. Online. NRSDK. 2023. Dostupné z: https://nrealsdkdoc.readthedocs.io/en/latest/index.html. [cit. 2023-10-11].

[23] *Vuzix Technology*. Online. Vuzix. 2023. Dostupné z: https://fr.vuzix.com/pages/technology. [cit. 2023-10-16].

[24] *Vuzix Blade 2™ Smart Glasses*. Online. Vuzix. 2023. Dostupné z: https://www.vuzix.com/products/vuzix-blade-2-smart-glasses. [cit. 2023-10-16].

[25] *HoloLens 2*. Online. Microsoft. 2023. Dostupné z: https://www.microsoft.com/en-us/hololens/hardware#document-experiences. [cit. 2023-10-17].

[26] *Introduction to mixed reality development*. Online. Microsoft. 2023. Dostupné z: https://learn.microsoft.com/en-us/windows/mixed-reality/develop/development. [cit. 2023-10-17].

[27] KÖNIG, M.; STADLMAIER, M.; RUSCH, T.; SOCHOR, R.; MERKEL, L. et al. MA2RA - Manual Assembly Augmented Reality Assistant. In: *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. Macao, China: IEEE, 2019, s. 501-505. ISBN 978-1-7281-3804-6. ISSN 2157-362X. Dostupné z: https://doi.org/10.1109/IEEM44572.2019.8978844.

[28] BLATTGERSTE, Jonas; RENNER, Patrick; STRENGE, Benjamin a PFEIFFER, Thies. In-situ instructions exceed side-by-side instructions in augmented reality assisted assembly. In: *PETRA '18: Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference*. New York, NY, United States: Association for Computing Machinery, 2018, s. 133-140. ISBN 9781450363907. Dostupné z: https://doi.org/10.1145/3197768.3197778.

[29] HANSON, Robin; FALKENSTRM, William a MIETTINEN, Mikael. Augmented reality as a means of conveying picking information in kit preparation for mixed-model assembly. In: *Computers and Industrial Engineering, Volume 113, Issue C*. United States: Pergamon Press, 2017, s. 570-575. ISSN 0360-8352. Dostupné z: https://doi.org/10.1016/j.cie.2017.09.048.

[30] GEMITO, Gennaro a SANTORO, Adolfo. A Mixed Reality Guidance System to Assist the Operator in the Assembly Process of Complex Products. In: *2023 IEEE International Conference on Industrial Technology (ICIT)*. Orlando, FL, USA: IEEE, 2023, s. 1-6. ISBN 979-8-3503-3651-1. ISSN 2641-0184. Dostupné z: https://doi.org/10.1109/ICIT58465.2023.10143166.

[31] *Si-OLED Technology for Smaller Display Modules*. Online. Epson. 2023. Dostupné z: https://corporate.epson/en/technology/search-by-products/other/smaller-display-modules.html. [cit. 2023-10-18].

[32] *Moverio BT-45CS*. Online. Epson. 2023. Dostupné z: https://www.epson.eu/en_EU/products/smart-glasses/see-through-mobile-viewer/moverio-bt-45cs/p/36980. [cit. 2023-10-18].

[33] *Meta Quest 2*. Online. Meta. 2023. Dostupné z: https://www.meta.com/quest/products/quest-2/. [cit. 2023-12].

[34] *Vive Pro AR (SRWorks) vs. ZED Mini*. Online. Stereolabs. 2023. Dostupné z: https://www.stereolabs.com/blog/vive-pro-ar-zed-mini. [cit. 2023-10-19].

[35] *Tracking SteamVR™*. Online. Steamworks. 2023. Dostupné z: https://partner.steamgames.com/vrlicensing. [cit. 2023-10-19].

[36] *Vive Pro 2*. Online. Vive. 2023. Dostupné z: https://www.vive.com/us/product/vive-pro2-full-kit/specs. [cit. 2023-10-19].

[37] *Vive Pro 2 - Product*. Online. Vive bussines. 2023. Dostupné z: https://business.vive.com/us/product/vive-pro2. [cit. 2023-10-19].

[38] REICHELT, Stephan; HÄUSSLER, Ralf a LEISTER, Norbert. Depth cues in human visual perception and their realization in 3D displays. In: *Proceedings Volume 7690, Three-Dimensional Imaging, Visualization, and Display 2010 and Display Technologies and Applications for Defense, Security, and Avionics IV*. Orlando, Florida, United States: SPIE, 2010. Dostupné z: https://doi.org/10.1117/12.850094.

[39] ALVES, João; MARQUES, Bernardo; OLIVEIRA, Miguel; ARAÚJO, Tiago; DIAS, Paulo et al. Comparing Spatial and Mobile Augmented Reality for Guiding Assembling Procedures with Task Validation. In: *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. Porto, Portugal: IEEE, 2019, s. 1-6. ISBN 978-1-7281-3559-5. Dostupné z: https://doi.org/10.1109/ICARSC.2019.8733642.

[40] ARMAN, Farshid a AGGARWAL, J.K. Model-based object recognition in dense-range images — a review. In: *ACM Computing Surveys, Volume 25, Issue 1*. New York, NY, United States: Association for Computing Machinery, 1993, s. 5-43. Dostupné z: https://doi.org/10.1145/151254.151255.

[41] MENG, Shaohua; LIU, Yuan; CHEN, Changyu a FAN, Qifan. Tracking and positioning of industrial objects using CAD models. In: *2022 8th International Conference on Mechanical Engineering and Automation Science (ICMEAS)*. Wuhan, China: IEEE, 2022, s. 219-223. ISBN 978-1-6654-6306-5. Dostupné z: https://doi.org/10.1109/ICMEAS57305.2022.00048.

[42] ŽIDEK, Kamil; LAZORÍK, Peter; PITEĽ, Ján a HOŠOVSKÝ, Alexander. An Automated Training of Deep Learning Networks by 3D Virtual Models for Object Recognition. In: *Symmetry*. MDPI, 2019. ISSN 2073-8994. Dostupné z: https://doi.org/10.3390/sym11040496.

[43] ULRICH, Markus; STEGER, Carsten a BAUMGARTNER, Albert. Real-time object recognition using a modified generalized Hough transform. In: *Pattern Recognition*. 2003, s. 2557-2570. ISSN 1873-5142. Dostupné z: https://doi.org/https://doi.org/10.1016/S0031-3203(03)00169-9.

[44] HASSANEIN, Allam Shehata; MOHAMMAD, Sherien; SAMEER, Mohamed a RAGAB, Mohammad Ehab. A Survey on Hough Transform, Theory, Techniques and Applications. In: *IJCSI*. 2015. ISSN 1694-0784. Dostupné z: https://doi.org/https://doi.org/10.48550/arXiv.1502.02160.

[45] STRZODKA, R.; IHRKE, I. a MAGNOR, M. A Graphics Hardware Implementation of the Generalized Hough Transform for fast Object Recognition, Scale, and 3D Pose Detection. In: *12th International Conference on Image Analysis and Processing*. Mantova, Italy. IEEE, 2003. ISBN 0-7695-1948-2. Dostupné z: https://doi.org/10.1109/ICIAP.2003.1234048.

[46] YANG, Xingwei; LIU, Hairong a LATECKI, Longin Jan. Contour-based object detection as dominant set computation. In: *Pattern Recognition*. 2012, s. 1927-1936. ISSN 1873-5142. Dostupné z: https://doi.org/https://doi.org/10.1016/j.patcog.2011.11.010.

[47] SCHINDLER, Konrad a SUTER, David. Object detection by global contour shape. In: *Pattern Recognition*. 2008, s. 3736-3748. ISSN 1873-5142. Dostupné z: https://doi.org/https://doi.org/10.1016/j.patcog.2008.05.025.

[48] SUMI, Yasushi; KAWAI, Yoshihiro; YOSHIMI, Takashi a TOMITA, Fumiaki. 3D Object Recognition in Cluttered Environments by Segment-Based Stereo Vision. In: *International Journal of Computer Vision*. Netherlands: Kluwer Academic, 2002, s. 5-23. ISSN 0920-5691. Dostupné z: https://doi.org/10.1023/A:1013240031067.

[49] SOMANI, Nikhil; PERZYLO, Alexander; CAI, Caixia; RICKERT, Markus a KNOLL, Alois. Object detection using boundary representations of primitive shapes. In: *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Zhuhai, China: IEEE, 2016. ISBN 978-1-4673-9675-2. Dostupné z: https://doi.org/10.1109/ROBIO.2015.7414632.

[50] SUGIMOTO, K. a TOMITA, F. Boundary segmentation by detection of corner, inflection and transition points. In: *Proceedings of Workshop on Visualization and Machine Vision*. Seattle, WA, USA: IEEE, 2002, s. 13-17. ISBN 0-8186-5875-4. Dostupné z: https://doi.org/10.1109/VMV.1994.324992.

[51] SUMI, Yasushi a TOMITA, Fumiaki. *3D Object Recognition Using Segment-Based Stereo Vision*. Electrotechnical Laboratory, 305, Tsukuba, Ibaraki, Japan, 2005.

[52] LOWE, D.G. Object recognition from local scale-invariant features. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece: IEEE, 1999, s. 1150-1157. ISBN 0-7695-0164-8. Dostupné z: https://doi.org/10.1109/ICCV.1999.790410.

[53] HSU, Gee-Sern; LIN, Chyi-Yeu a WU, Jia-Shan. Real-time 3-D object recognition using Scale Invariant Feature Transform and stereo vision. In: *2009 4th International Conference on Autonomous Robots and Agents*. Wellington, New Zealand: IEEE, 2009, s. 239-244. ISBN 978-1-4244-2712-3. Dostupné z: https://doi.org/10.1109/ICARA.2000.4803919.

[54] LOWE, D.G. Distinctive Image Features from Scale-Invariant Keypoints. In: *International Journal of Computer Vision*. Netherlands: Kluwer Academic Publishers, 2004, s. 91-110. Dostupné z: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[55] WU, Jian; CUI, Zhiming; SHENG, Victor S.; ZHAO, Pengpeng; SU, Dongliang et al. A Comparative Study of SIFT and its Variants. In: *Measurement Science Review*. 2013, s. 122 - 131. ISSN 1335-8871. Dostupné z: https://doi.org/10.2478/msr-2013-0021.

[56] *Feature Matching*. Online. Opencv. 2023. Dostupné z: https://docs.opencv.org/3.4/dc/dc3/tutorial_py_matcher.html. [cit. 2023-12-28].

[57] ASWIN, P.J.; CHANDANA, J.S.; REGHUNATH, Seethal a MENON, Maya. Stereo Vision Based System For Object Detection And Recognition. In: *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*. Tirunelveli, India: IEEE, 2019, s. 1284-1288. ISBN 978-1-5386-9439-8. Dostupné z: https://doi.org/10.1109/ICOEI.2019.8862588.

[58] LOURAKIS, Manolis a ZABULIS, Xenophon. Model-Based Pose Estimation for Rigid Objects. In: *Computer Vision Systems*. Volume 7963. Springer, 2013, s. 83-92. ISBN 978-3-642-39401-0.

[59] COLLET, Alvaro; BERENSON, Dmitry; SRINIVASA, Siddhartha S. a FERGUSON, Dave. Object recognition and full pose registration from a single image for robotic manipulation. In: *2009 IEEE International Conference on Robotics and Automation*. Kobe, Japan: IEEE, 2009, s. 48-55. ISSN 1050-4729. Dostupné z: https://doi.org/10.1109/ROBOT.2009.5152739.

[60] YOON, Kuk-Jin; SHIN, Min-Gil a LEE, Ji-Hyo. Recognizing 3D Objects with 3D Information from Stereo Vision. In: *2010 20th International Conference on Pattern Recognition*. Istanbul, Turkey: IEEE, 2010, s. 4020-4023. ISBN 978-1-4244-7542-1. ISSN 1051-4651. Dostupné z: https://doi.org/10.1109/ICPR.2010.1151.

[61] LIN, Chyi-Yeu a SETIAWAN, Edwin. Object orientation recognition based on SIFT and SVM by using stereo camera. In: *2008 IEEE International Conference on Robotics and Biomimetics*. Bangkok, Thailand: IEEE, 2009, s. 1371-1376. ISBN 978-1-4244-2678-2. Dostupné z: https://doi.org/10.1109/ROBIO.2009.4913200.

[62] RUBLEE, E.; RABAUD, V.; KONOLIGE, K. a BRADSKI, G. ORB: An efficient alternative to SIFT or SURF. In: *2011 International Conference on Computer Vision*. Barcelona, Spain: IEEE, 2012, s. 2564-2571. ISBN 978-1-4577-1102-2. ISSN 2380-7504. Dostupné z: https://doi.org/10.1109/ICCV.2011.6126544.

[63] ROSTEN, Edward; PORTER, Reid a DRUMMOND, Tom. Faster and Better: A Machine Learning Approach to Corner Detection. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*. IEEE, 2008, s. 105-119. ISSN 1939-3539. Dostupné z: https://doi.org/10.1109/TPAMI.2008.275.

[64] CALONDER, Michael; LEPETIT, Vincent; STRECHA, Christoph a FUA, Pascal. BRIEF: Binary Robust Independent Elementary Features. In: *Computer Vision -- ECCV 2010*. Heraklion, Crete, Greece, 2010, s. 778–792. ISBN 978-3-642-15560-4.

[65] DU, Yuanyuan; MIAO, Zhenjiang a CEN, Yigang. Markless augmented reality registration algorithm based on ORB. In: *2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*. Changchun, China: IEEE, 2015, s. 547-550. ISBN 978-1-6654-6253-2. Dostupné z: https://doi.org/10.1109/ICOSP.2014.7015197.

[66] LU, Pengsen; ZHANG, Jie; LIN, Xin; QI, Jingwen; CHEN, Yuxing et al. Research on Indoor Target Positioning System Based on Image Feature Extraction and Recognition. In: *2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*. Changchun, China: IEEE, 2023, s. 547-550. ISBN 978-1-6654-6253-2. Dostupné z: https://doi.org/10.1109/EEBDA56825.2023.10090701.

[67] LEUTENEGGER, Stefan; CHLI, Margarita a SIEGWART, Roland Y. BRISK: Binary Robust invariant scalable keypoints. In: *2011 International Conference on Computer Vision*. Barcelona, Spain: IEEE, 2012, s. 2548-2555. ISBN 978-1-4577-1102-2. ISSN 2380-7504. Dostupné z: https://doi.org/10.1109/ICCV.2011.6126542.

[68] NAZIR, Souha; ASSOUM, Ammar; EL HASSAN, Bachar a DORNAIKA, Fadi. Augmented Reality application: A new implementation chain. In: *2016 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)*. Beirut, Lebanon: IEEE, 2016, s. 69-74. ISBN 978-1-5090-5281-3. Dostupné z: https://doi.org/10.1109/IMCET.2016.7777429.

[69] FERNÁNDEZ ALCANTARILLA, Pablo; DAVISON, Andrew J. a BARTOLI, Adrien. KAZE Features. In: *ECCV*. Springer, 2012, s. 214–227. ISBN 978-3-642-33782-6. Dostupné z: https://doi.org/10.1007/978-3-642-33783-3_16.

[70] ALCANTARILLA, Pablo Fernández. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In: *British Machine Vision Conference (BMVC)*. Bristol, UK. 2013. Dostupné z: https://doi.org/10.5244/C.27.13.

[71] ALAHI, Alexandre; ORTIZ, Raphael a VANDERGHEYNST, Pierre. FREAK: Fast Retina Keypoint. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. Providence, RI, USA: IEEE, 2012, s. 510-517. ISBN 978-1-4673-1228-8. ISSN 1063-6919. Dostupné z: https://doi.org/10.1109/CVPR.2012.6247715.

[72] YU, Yang; GUO, Yingchun; WANG, Ruili; YIN, Susha a YU, Ming. Registration Based on ORB and FREAK Features for Augmented Reality Systems. In: *Transactions of Tianjin University*. Springer, 2016, s. 192–200. Dostupné z: https://doi.org/10.1007/s12209-017-0028-3.

[73] JIAO, Licheng; ZHANG, Fan; LIU, Fang; YANG, Shuyuan; LI, Lingling et al. A Survey of Deep Learning-Based Object Detection. In: *IEEE Access*. IEEE, 2019, s. 128837-128868. ISSN 2169-3536. Dostupné z: https://doi.org/10.1109/ACCESS.2019.2939201.

[74] REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross a FARHADI, Ali. You Only Look Once: Unified, Real-Time Object Detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, 2016, s. 779-788. Dostupné z: https://doi.org/10.1109/CVPR.2016.91.

[75] SU, Yongzhi; RAMBACH, Jason; MINASKAN, Nareg; LESUR, Paul; PAGANI, Alain et al. Deep Multi-state Object Pose Estimation for Augmented Reality Assembly. In: *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. Beijing, China: IEEE, 2019, s. 222-227. ISBN 978-1-7281-4765-9. Dostupné z: https://doi.org/10.1109/ISMAR-Adjunct.2019.00-42.

[76] QI, Shaohua; NING, Xin; YANG, Guowei; ZHANG, Liping; LONG, Peng et al. Review of multi-view 3D object recognition methods based on deep learning. In: *3D data computation and visualization*. Elsevier B.V., 2021. ISSN 0141-9382. Dostupné z: https://doi.org/10.1016/j.displa.2021.102053.

[77] XIANG, Yu; SCHMIDT, Tanner a FOX, Dieter. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In: *ArXiv*. 2017. Dostupné z: https://doi.org/10.48550/arXiv.1711.00199.

[78] DU, Yi-Chun; MUSLIKHIN, Muslikhin; HSIEH, Tsung-Han a WANG, Ming-Shyan. Stereo Vision-Based Object Recognition and Manipulation by Regions with Convolutional Neural Network. In: *Electronics*, *9*. MDPI, 2020. Dostupné z: https://doi.org/10.3390/electronics9020210.

[79] ZHAO, Lijun; LI, Xiaoyu; SUN, Zhenye; WANG, Ke a YANG, Chenguang. A Robot Navigation Method Based on Human-Robot Interaction for 3D Environment Mapping. In: *2017 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. Okinawa, Japan: IEEE, 2018, s. 409-414. ISBN 978-1-5386-2035-9. Dostupné z: https://doi.org/10.1109/RCAR.2017.8311896.

[80] CHEN, Song; LI, Zupei; DANGELO, Fabrizio; GAO, Chao a FU, Xinwen. A Case Study of Security and Privacy Threats from Augmented Reality (AR). In: *2018 International Conference on Computing, Networking and Communications (ICNC)*. Maui, HI, USA: IEEE, 2018, s. 442-446. ISBN 978-1-5386-3652-7. Dostupné z: https://doi.org/10.1109/ICCNC.2018.8390291.

[81] *ZED-mini mounted on a VR headset*. Online. Stereolabs. 2023. Dostupné z: https://www.stereolabs.com/. [cit. 2023-10-11].

[82] *ZED SDK integration*. Online. Stereolabs. 2023. Dostupné z: https://www.stereolabs.com/. [cit. 2023-10-11].

[83] MORSHIDI, Malik Arman; GUNAWAN, Teddy Surya a OLANREWAJU, Rashidah Funke. A Comparative Analysis of Feature Extraction Algorithms for Augmented Reality Applications. In: *2021 IEEE 7th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA)*. Bandung, Indonesia: IEEE, 2021, s. 59-63. ISBN 978-1-7281-7523-2. ISSN 2640-6535. Dostupné z: https://doi.org/10.1109/ICSIMA50015.2021.9526295.

[84] TAREEN, Shaharyar Ahmed Khan a SALEEM, Zahra. A Comparative Analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In: *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. Sukkur, Pakistan: IEEE, 2018, s. 1-10. ISBN 978-1-5386-1370-2. Dostupné z: https://doi.org/10.1109/ICOMET.2018.8346440.

[85] CHEN, Wei; JIA, Xi; CHANG, Hyung Jin; DUAN, Jinming; SHEN, Linlin et al. FS-Net: Fast Shape-based Network for Category-Level 6D Object Pose Estimation with Decoupled Rotation Mechanism. In: *2021 Computer Vision and Pattern Recognition*. IEEE/CVF, 2021, s. 1581-1590. Dostupné z: https://doi.org/10.48550/arXiv.2103.07054.

[86] VINCENT, E. a LAGANIERE, R. Detecting planar homographies in an image pair. In: *ISPA 2001. Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis. In conjunction with 23rd International Conference on Information Technology Interfaces (IEEE Cat.*. Pula, Croatia: IEEE, 2002, s. 182-187. ISBN 953-96769-4-0. Dostupné z: https://doi.org/10.1109/ISPA.2001.938625.

[87] LUO, Y.; WANG, X.; LIAO, Y.; FU, Q.; SHU, C. et al. A Review of Homography Estimation: Advances and Challenges. In: *Electronics*. 2023. Dostupné z: https://doi.org/10.3390/electronics12244977.

ZOZNAM PRÍLOH

Příloha A - Porovnanie XR okuliarov

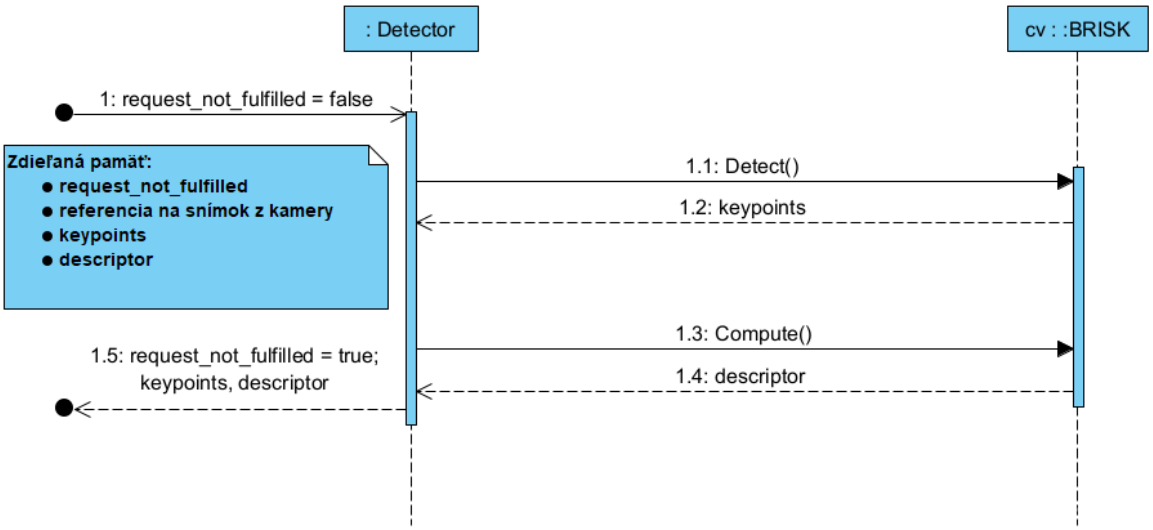
A.1 Tabuľka porovnania

Model	Rozlíšenie	Stereo kamera	FOV	Výdrž batérie	Nutnosť káblového pripojenia	Váha	Externý HW	6-DOF tracking	Cena
Real Wear Navigator 520	1280x720	Nie	24°	6-8h	Nie	274g	Nie	Nie	$ 3,300.00
XReal Light	1920x1080	Nie	52°	2,5h	Áno	106g	Nie	Áno	$ 599.00
Vizux Blade 2	480x480	Nie	20°	5-6h	Nie	90g	Nie	Nie	$ 1299.99
Microsoft Hololens 2	2048x1080	Nie	96°	2-3h	Nie	566g	Nie	Áno	$ 3500.00
Moverio BT-45CS	1920x1080	Nie	34°	Ø	Áno	550g	Áno	Nie	$ 2099.00
Oculus Quest 2	1832x1920	Nie	93°	2-3h	Nie	503g	Nie	Áno	$ 299.00
HTC VIVE Pro 2	2488x2488	Áno	120°	6h	Áno	470g	Áno	Áno	$ 799.00

Příloha B - Sekvenčné diagramy

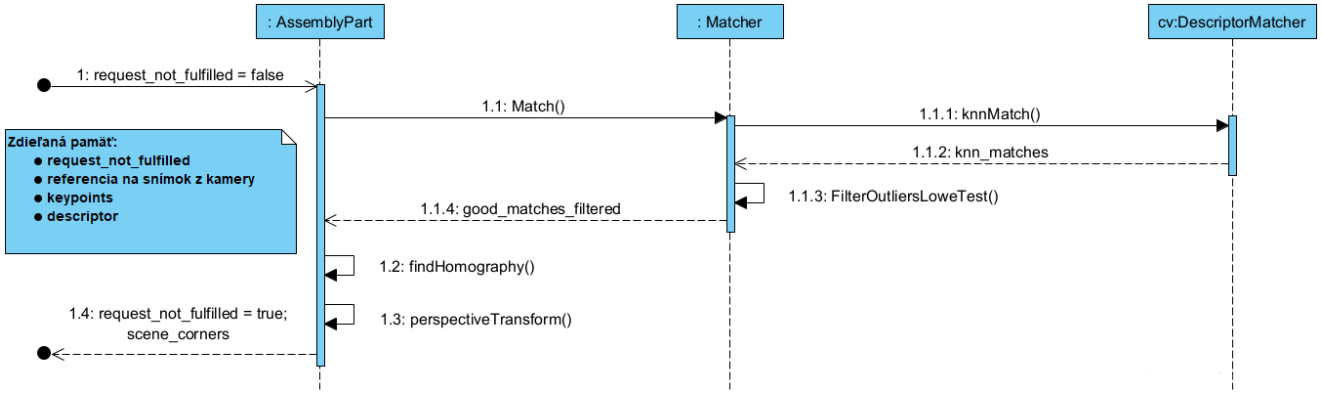
B.1

Sekvenčný diagram popisujúci priebeh inštancie triedy *Detector*.



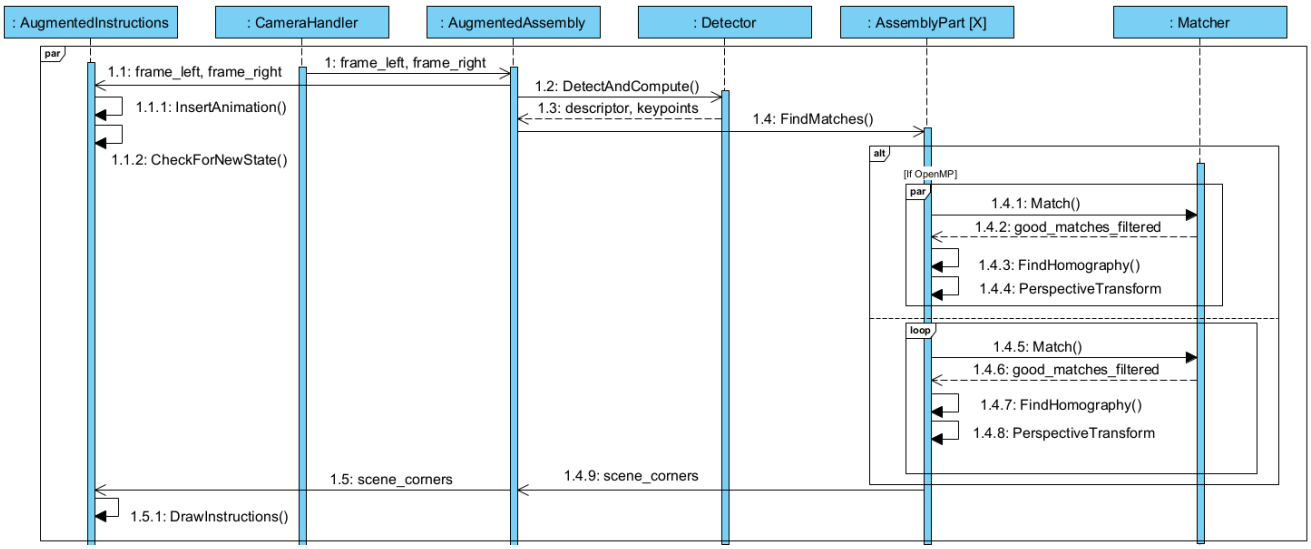
B.2

Sekvenčný diagram popisujúci priebeh inštancie triedy *AssemblyPart*.



B.3

Sekvenčný diagram vlákien aplikácie.



Příloha C - Obsah pamäťového média

/
- AugmentedAssembly
  - include……..…..Hlavičkové súbory
    - AssemblyPart.h
    - AugmentedAssembly.h
    - AugmentedInstructions.h
    - CameraHandler.h
    - Detector.h
    - Enums.hpp
    - Matcher.h
  - src……...……Zdrojové súbory
    - AssemblyPart.cpp
    - AugmentedAssembly.cpp
    - AugmentedInstructions.cpp
    - CameraHandler.cpp
    - Detector.cpp
    - main.cpp
    - Matcher.cpp
  - list_of_dlls.txt……..…….Zoznam .dll knižníc
  - OpenCV480_ZED_SDK.props…….konfigurácia MSVS DEBUG
  - OpenCV480_ZED_SDK_Release.props…….. konfigurácia MSVS RELEASE
- Doxy…….Doxygen dokumentácia
  - ...
- Measurements……...Zložka s meranými hodnotami
  - Oclusion…….Hodnoty pre meranie oklúzie
  - Performance……..Hodnoty pre meranie latencie
- Thesis
  - Rozšírená realita v priemyselnej výrobe a údržbe.pdf
- Unity
  - Assets……...Assety použité v Unity
  - main.cs…….....….Hlavný použitý skript
  - ...
  - list_of_dlls.txt……..…… Zoznam .dll knižníc
- Videos
- github_repo.txt……link pre github repozitár
- VTune profiler.....výstupy z profileru
  - ...