VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

6-DOF LOKALIZACE OBJEKTŮ V PRŮMYSLOVÝCH APLIKACÍCH

6-DOF OBJECT LOCALIZATION IN INDUSTRIAL APPLICATIONS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **NELA MACUROVÁ**
AUTHOR

VEDOUCÍ PRÁCE **Ing. MICHAL HRADIŠ, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání diplomové práce

Studentka: **Macurová Nela, Bc.**
Program: Informační technologie
Obor: Počítačová grafika a multimédia
Název: **6-DOF lokalizace objektů v průmyslových aplikacích**
6-DOF Object Localization in Industrial Applications
Kategorie: Zpracování obrazu
Zadání:

1. Nastudujte současné metody používané pro odhad pózy objektů obraze z kamery a v bodových mračnech se zaměřením na lesklé objekty bez textury.
2. Vytvořte si sadu 3D modelů objektů a připravte si sadu scén s těmito objekty a simulátor zobrazení pomocí stereo kamery.
3. Vyberte metodu použitelnou pro přesnou lokalizaci lesklých netexturovaných objektů z hloubkové mapy.
4. Vybranou metodu implementujte.
5. V experimentech na umělých datech vyhodnoťte vlastnosti lokalizační metody a ověřte její funkčnost na reálných scénách.
6. Zhodnoťte dosažené výsledky a diskutujte možnosti budoucího vývoje.
7. Vytvořte stručný plakát prezentující vaši práci, její cíle a výsledky.

Literatura:

- Xiang et al.: PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. 2018.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Hradiš Michal, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 30. října 2020

Abstrakt

Cílem práce je navrhnout metodu, která lokalizuje objekt v bodovém mračně a co nejpřesněji odhadne 6D pózu předem známých objektů v průmyslové scéně pro bin picking. Návrh řešení je inspirován sítí PoseCNN. Součástí řešení je i simulátor scén, který generuje umělá data. Simulátor je použit k vygenerování trénovací datové sady obsahující 2 objekty pro trénování konvoluční neuronové sítě. Síť je otestována na anotovaných reálných scénách a dosahuje nízké úspěšnosti, pouze 23.8 % a 31.6 % úspěšnosti pro odhad translace a rotace pro jeden typ a pro druhý objekt 12.4 % a 21.6 %, přičemž tolerance pro správný odhad je 5 mm a 15°. Avšak použitím algoritmu ICP na odhadnuté výsledky je dosažena úspěšnost odhadu translace 81.5 % a rotace 51.8 % a pro druhý objekt 51.9 % a 48.7 %. Přínosem této práce je vytvoření generátoru a otestování funkčnosti sítě na malé objekty.

Abstract

The aim of this work is to design a method for the object localization in the point could and as accurately as possible estimates the 6D pose of known objects in the industrial scene for bin picking. The design of the solution is inspired by the PoseCNN network. The solution also includes a scene simulator that generates artificial data. The simulator is used to generate a training data set containing 2 objects for training a convolutional neural network. The network is tested on annotated real scenes and achieves low success, only 23.8 % and 31.6 % success for estimating translation and rotation for one type of obejct and for another 12.4 % and 21.6 %, while the tolerance for correct estimation is 5 mm and 15°. However, by using the ICP algorithm on the estimated results, the success of the translation estimate is 81.5 % and the rotation is 51.8 % and for the second object 51.9 % and 48.7 %. The benefit of this work is the creation of a generator and testing the functionality of the network on small objects.

Klíčová slova

6-DOF, odhad pózy, bodové mračno, hluboké učení, bin picking, PoseCNN, simulátor scén, ICP

Keywords

6-DOF, pose estimation, point cloud, deep learning, bin picking, PoseCNN, scene simulator, ICP

Citace

MACUROVÁ, Nela. *6-DOF lokalizace objektů v průmyslových aplikacích.* Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Hradiš, Ph.D.

6-DOF lokalizace objektů v průmyslových aplikacích

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně pod vedením pana Ing. Michala Hradiše, Ph.D. Uvedla jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpala.

.....
Nela Macurová
31. května 2021

Poděkování

Ráda bych poděkovala Ing. Michalu Hradišovi, Ph.D. za odborné vedení, ochotu, trpělivost, podporu a cenné rady. Dále děkuji společnosti Kinali s.r.o. za možnost využití potřebných prostředků a všem kolegům za rady a podporu.

Obsah

Kapitola 1

Úvod

Lokalizace a odhad pózy objektu je aktuálně hodně zkoumaný problém z důvodu zavádění robotizace, inteligentních zařízení a kolaborativních robotů do různých průmyslových oblastí. Robotizace má široké uplatnění v rámci zvyšování rychlosti a produktivity výroby. Využívá se hlavně pro vykonávání rutinních úkonů jako je manipulace s materiálem - zakládání materiálu do stroje, přesouvání a skládání do beden. Poslední dobou jsou hojně využíváni právě kolaborativní roboti, kteří mají vyšší kritéria pro bezpečnost kvůli práci v těsné blízkosti s člověkem. Pro ty může mít odhad pozice objektu velké uplatnění. Hodí se například pro definici scény, ve které se může robot pohybovat. Dále se odhad pozice objektu hodí pro detekci kolizních objektů, kterým je třeba se vyhnout, aby nedošlo ke srážce a robot si mohl naplánovat trasu, ve které se tomuto objektu vyhne. Znalost pozice lze také využít pro manipulaci robota s objekty a pro bin picking, kdy díky znalosti pozice a rotace lze objekt uchopit a přemístit. Problém je však náročný kvůli rozmanitosti objektů v reálném světě. Objekty mají různé 3D tvary a povrchy a jejich vzhled na snímcích je ovlivňován různými světelnými podmínkami, nepořádkem ve scéně a vzájemným překrytím objektů. V rámci počítačového vidění je odhad pozice objektu spojen s mnoha dalšími problémy, jako je chybějící textura objektu, vysoký šum v datech, symetrie předmětu a lesklý materiál.

Cílem této práce je vytvořit nástroj pro lokalizaci a odhad 6DOF pozice objektů v bodovém mračně nasnímaném pomocí 3D kamery. Kamera snímá scénu, kde je umístěna krabice se známými objekty, určenými pro manipulaci s robotem. Objekty jsou menší velikosti a mohou se vzájemně překrývat, což tvoří velmi náročnou a nepřehlednou scénu. Cílem je tedy správně nalézt objekty vhodné pro uchopení, tedy z vrchu hromady, a odhadnout co nejpřesněji jejich pozici a rotaci tak, aby mohl být tento objekt správně uchopen robotem.

Nejprve je v kapitole 2 definován daný problém, popsána scéna, hledané objekty a 3D kamera, která snímá danou scénu. V další kapitole 3 je shrnutí současných metod a algoritmů, využívaných pro odhad pozice objektů. Zaměřuji se na využití konvolučních neuronových sítí při odhadu pozice a rotace. Následující kapitola 4 obsahuje zvolené metody, které jsou v rámci práce využity. Kapitola 5 se věnuje simulátoru určenému ke generování umělých scén pro trénování sítě. V rámci práce byla vytvořena umělá datová sada a také anotovány snímky reálných scén.

Kapitola 2

Definice řešeného problému

Cílem této práce je nalézt přesnou 6DOF pozici předem známých objektů v datech, které lze získat z 3D kamery, tedy v point cloudu či hloubkové mapě. Objekty leží na hromadě v krabici a tvoří tedy množství stejných objektů, které se mohou navzájem překrývat. Cílem je odhadnout pozici s co největší přesností pro lehce uchopitelné objekty, tedy předměty z vrchu a okraje hromady. Tento podproblém je součástí většího a komplexnějšího problému zvaného bin picking [42].

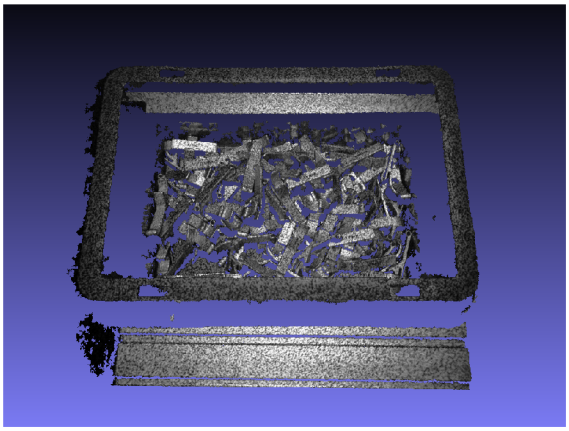
6DOF (Degrees of freedom) , neboli 6 stupňů volnosti, jsou parametry, jimiž definujeme stav systému, v našem případě tedy scény. Šest stupňů volnosti je označení pro možné směry posunu a otočení v souřadném systému scény, kterými se objekt může pohybovat. Jedná se o posun podél os x, y a z a o otočení kolem těchto os. Počátek souřadného systému může být dán buď pozicí kamery, nebo pozicí robota.

Bodové mračno (Point cloud) Bodové mračno je množina bodů v prostoru definovaných pomocí hodnot x, y a z souřadnic. Toto mračno je formátem, který lze získat snímáním scény pomocí stereokamer či 3D skenerů. Modely objektů mohou být definovány také pomocí bodových mračen. Množina bodů může uchovávat i další informace. Nejen hodnoty souřadnic, ale i informace o barvě či normále.

Bin picking Bin picking je jeden z problémových úkolů v počítačovém vidění a robotice. Cílem je, aby robot úspěšně vytahoval pomocí sacího chapadla, uchopovacího chapadla nebo jiného koncového efektoru, z krabice známé předměty tak, aby nedocházelo ke kolizím. Součástí robota jsou senzory a kamery, které slouží k vyhledávání, navigaci a vyhodnocování, případně i k zabezpečení.

2.1 Definice scény

Scéna, ve které bude docházet k odhadu pozice objektů, je plocha, na které jsou buď přímo umístěny objekty pro manipulaci nebo jsou tyto objekty umístěny v krabici, ze které jsou postupně vytahovány. Na ploše je umístěn i kolaborativní robot. Zkalibrovaná kamera je připevněna nad manipulační plochou. Na obrázku lze vidět příklad reálné scény, nasnímané pomocí 3D kamery 2.1. S objekty robot manipuluje, proto je potřeba odhadnout přesnou pozici a rotaci daného objektu. Pro odhad pozice objektů, pokud jsou na hromadě, je důležité nalézt a vybrat jednoduše uchopitelné objekty, které jsou na vrchu hromady.



(a) Vizualizace point cloudu scény.

(b) Ukázka objektů.

Obrázek 2.1: Reálná data jsou snímána pomocí Ensenso 3D kamery. Vizualizace dat reálné scény zachycující krabici obsahující otvíráky. Na druhém obrázku je ilustrace hledaných objektů.

Problémovými částmi je tedy přesnost na milimetry pro manipulaci drobných předmětů, a proto je velmi důležitý přesný odhad pozice v prostoru. Dalším problémem je lesklý materiál, který odráží světlo a může zrcadlit okolní předměty, což odhad také značně komplikuje.

Objekty Hledané objekty mohou být například nástroje, kovové komponenty, součástky nebo jiné malé, kovové předměty bez textury. Objekty jsou předem známé a definované jejich modelem a rozměry. Jedná se o rigidní objekty, které během manipulace nemění svůj tvar a nedochází k jejich deformaci. Převážně to jsou malé předměty, a proto je třeba znát co nejpřesněji jejich polohu. Některé předměty je potřeba uchopit tak, aby nedošlo k poškození určitých částí nebo výstupků, proto je nutná přesná pozice kvůli manipulaci s nimi. Místa pro uchopení některých předmětů mohou být tedy značně omezená. Jako demonstrační objekt je použit otvírák, velikost otvíráku je 60 mm na délku, 10 mm na šířku a 2 mm na tloušťku. Materiál otvíráku je kovový, může tedy docházet k reflexím. Dalším problémem může být také odraz v krabici s lesklým povrchem, kdy dochází k zrcadlení objektů na stěně krabice a může mylně vznikat neexistující objekt. Pokud dojde k vysbírání krabice, tak další problémovou částí může být i neschopnost detekovat a nalézt objekt ležící při okraji krabice. Nebo naopak detekovat, že scéna žádné objekty neobsahuje.

2.2 3D kamera

Stereoskopická kamera Ensenso N35 [15] má 2 monochromatické snímače CMOS (global shutter) s rozlišením 1280x1024 pixelů a projektor. Snímek kamery lze vidět na obrázku 2.2a. Projektor je k dispozici buď s modrým světlem ve viditelné oblasti (465 nm), nebo s infračerveným světlem (850 nm). Maximální snímková frekvence je 30 fps a 64 úrovní disparity. Pracovní vzdálenost je až 3 000 mm a ohniskové délky jsou od 6 do 16 mm. Komunikace a přenos dat probíhá prostřednictvím ethernetu. Výstupní data mají formát hloubkové mapy, dané souřadnicemi x, y (pozice) a z (hloubka) nebo formát mračna bodů.

Použitá kamera funguje pomocí stereovidění, které napodobuje lidské vidění. Dva fotoaparáty získávají snímky ze stejné scény ze dvou různých pozic. Přestože kamery vidí stejný

(a) Kamera Ensenso N35.

(b) Projektor a promítaný vzor.

Obrázek 2.2: Stereo kamera s projektorem, obrázky jsou převzaty z [15].

obsah scény, existují různé pozice objektů podle projekčních paprsků kamer, kde se bod P promítne na levém snímku jako bod P_L a na pravém P_R. Poté dochází k porovnávání dvou obrázků, ve kterých se hledají odpovídající body neboli korespondence a vypočítají se všechna posunutí bodů do mapy disparity. Pomocí znalosti vzdálenosti, úhlu pohledu kamery a ohniskové vzdálenosti f objektivu se převádí tyto mapy disparit do jednotek délky pomocí principu triangulace, takže je možné určit 3D souřadnice každého obrazového pixelu. Výsledkem je hloubková mapa či mračno 3D bodů, které je základem pro další aplikace, založené na informacích o 3D objektech. Proces porovnávání během porovnávání obrazu je založen na odstupňování kontrastu a jasu pixelů senzoru. Kvalita stereovidění tedy přímo závisí na světelném stavu scény a texturách povrchu objektu. Nalezení a výpočet souřadnic odpovídajících bodů na méně strukturovaných nebo reflexních plochách je velmi obtížné. Rozdíl nelze jednoznačně určit. Výsledkem je neúplná hloubková informace o scéně.

Pro zpřesnění informací o scéně je použit vzorový projektor, znázorněný na obrázku 2.2b. Světelný projektor vytváří texturu s vysokým kontrastem na povrchu objektu pomocí masky vzoru. Promítnutá textura doplňuje slabou nebo neexistující strukturu povrchu objektu. Výsledkem je podrobnější mapa disparit a úplnější hloubková informace o scéně. K dalšímu vylepšení výsledků pro statické scény může docházet při posunu tohoto vzoru a párováním výsledných disparitních map, a taktéž dojde k zhuštění bodových informací.

Kapitola 3

Metody pro odhad pózy

Existuje mnoho metod a prací, zabývajících se odhadem 2D a 3D pozic objektů. Problém odhadu pozice objektu lze rozdělit do více částí jako jsou detekce a segmentace objektu a odhad pozice a rotace daného objektu.

Metody mohou být také závislé na reprezentaci dat. Nejrozšířenější a hojně zkoumané jsou RGB a RGBD data. Další obvyklou reprezentací jsou hloubková data reprezentovaná bodovými mračny.

Metody odhadu pozice objektů 6D lze zhruba rozdělit na metody založené na šablonách (template) a metody založené na příznacích (features).

V metodách založených na šabloně je zkonstruována pevná šablona, která je použita k pasování v různých umístěních ve vstupním obrazu. V každém místě se vypočítá skóre podobnosti a nejlepší shoda se získá porovnáním těchto skóre podobnosti [18]. V odhadu 6D pozice je šablona obvykle získána vykreslením odpovídajícího 3D modelu. V poslední době se metody detekce 2D objektů používají jako porovnávání šablon (template matching) a jsou rozšířeny pro odhad 6D pozice, zejména u detektorů objektů založených na hlubokém učení [36, 41]. Metody založené na šablonách jsou vhodné při detekci objektů bez textury, ale mají však problém s okluzemi (překryvy) kvůli nízkému skóre podobnosti.

V metodách založených na příznacích jsou lokální příznaky extrahovány buď z bodů zájmu nebo z každého pixelu v obrazu a přiřazeny k prvkům na 3D modelech, aby se vytvořily korespondence 2D na 3D, z nichž lze obnovit 6D pozici [29, 33]. Tyto metody jsou schopny zpracovat překryvy mezi objekty, k výpočtu lokálních příznaků však vyžadují dostatečné textury na objektech. Mezi tyto metody patří i metoda RANSAC.

Další metody předpovídají 3D umístění pixelů nebo lokálních tvarů v prostoru objektů [5, 6, 28, 32]. Mezi tyto metody patří i síť Pix2Pose [32] a PoseCNN [45]. Brachmann a kol. [6] používají regresi 3D souřadnice a předpovídají třídu pro každý pixel. K dosažení nejlepšího výsledku mezi hypotézami pozice je však zapotřebí další výpočet, což tyto metody zpomaluje. 3D souřadnicová regrese naráží na problémy při zpracování symetrických objektů, protože různé orientace objektů mohou tvořit identické pohledy. Není možné tedy jedinečně odhadnout správnou orientaci a může docházet k vysoké chybě pro odhad orientace, i když je odhad správný. V takových případech lze využít upravenou chybovou funkci. Toto je řešeno v PoseCNN.

3.1 Algoritmické metody

Algoritmické metody se hodí pro jednodušší scény, kde lze lehce nalézt korespondence a klíčové body, a není tedy vhodná na složité a nepřehledné scény. Metody jsou založeny na iterativním procházení vstupu a hledání nejlepší podobnosti. Mezi tyto metody patří algoritmus Random Sample Consensus, Iterative closest point a Perspective-n-Point. Tyto metody jsou často kombinovány.

Random Sample Consensus (RANSAC)

Random Sample Consensus neboli Shoda náhodných vzorků je metoda, která byla poprvé publikována Martinem A. Fischlerem a Robertem C. Bollesem [14] v roce 1981. Algoritmus se hodí pro hledání korespondencí v obraze nebo hledání geometrických objektů, které lze popsat modelem. RANSAC je iterační metoda, která provádí testování shody hledaného modelu s náhodně vybranými vzorky dat. Počet vybraných vzorků dat odpovídá minimu pro výpočet parametrů hledaného modelu, například pro přímku dva, pro kružnici tři. Data sestávají z „inliners“, data, která odpovídají danému modelu i když mohou být do jisté míry zašumělá, a „outliers“, což jsou data neodpovídající danému modelu. Výstupem algoritmu jsou parametry hledaného modelu. Opakování se provádí, dokud míra shody nepřekročí požadované kritérium nebo maximální počet iterací. Maximální počet iterací I lze vypočítat pomocí rovnice

$$I = \frac{\log(1-p)}{\log(1-q^n)}, \tag{3.1}$$

kde p je pravděpodobnost úspěšného řešení, q je pravděpodobnost výběru inlineru a n je počet parametrů modelu.
Tato metoda se nehodí pro velmi složitou scénu, která obsahuje velké množství bodů. Pro hledání korespondencí by bylo nutné vyfiltrovat klíčové body, případně použít metodu pouze na vybraném segmentu scény. Tato metoda je využita například v pracech [6, 32].

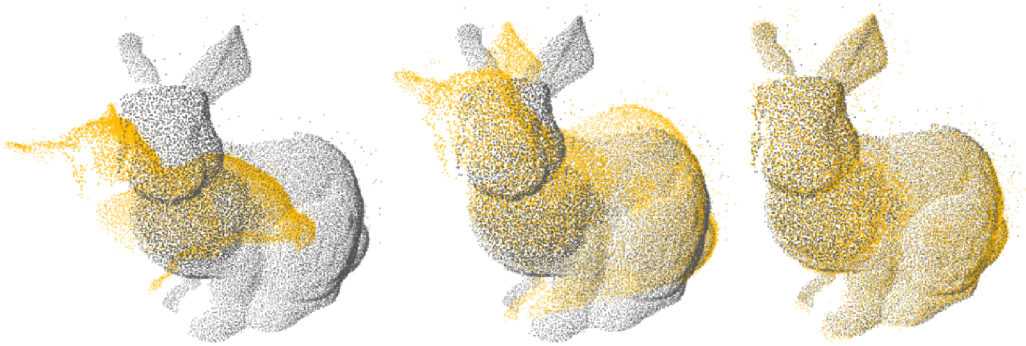
Iterative closest point (ICP)

Algoritmus Iterative closest point byl představen P. J. Beslem a N. D. McKayem [3] a také Y. Chenem a G. Medionim [9] v roce 1992. Je to algoritmus, který minimalizuje vzdálenost mezi dvěma bodovými mračny, což je vidět na obrázku 3.1. Algoritmus se hodí pro jemné zarovnání modelu s bodovým mračnem. Pomocí translace a rotace se snaží nalézt co nejlepší korespondenci s referenčním bodovým mračnem. Vstupem je referenční X a zdrojové P bodové mračno, případně kritéria pro zastavení iterací. Výstupem je cílová transformace R a t. Dochází k hledání posunu t a rotace R takové, která minimalizuje střední kvadratickou chybu v euklidovském prostoru

$$E(R,t) = \frac{1}{N_p}\sum_{i=1}^{N_p}\|x_i - Rp_i - t\|^2, \tag{3.2}$$

kde x_i a p_i jsou korespondující body. Existuje mnoho variant tohoto algoritmu, z nichž nejoblíbenější jsou point-to-point a point-to-plane, který využívá informace o normálách a je využívaný spíše ve strukturovaných prostředích.

Tato metoda se hodí spíše na závěrečné dorovnání bodů, protože může spadnout do lokálního minima. Proto se opět na složitou scénu hodí pouze na vybraný segment scény. Tato metoda je často využívána, například v PoseCNN [45] právě pro dorovnání extrahovaných segmentů.



Obrázek 3.1: Ukázka algoritmu Iterative closest point. Na obrázku je znázorněn posunutý model králíčka reprezentovaný bodovým mračnem oproti cílové pozici modelu, postupnou iterací dochází ke snižování vzdálenosti bodů modelu a tím se model dorovná na cílovou transformaci. Převzato z [2].

Perspective-n-Point (PnP)

Metoda Perspektive-n-Point řeší vztah mezi 3D souřadnicovými systémy v závislosti na projekci do 2D roviny. Vztahy jednotlivých souřadnicových systémů jsou znázorněny na obrázku 3.2. Metoda se používá k nalezení transformace, tedy rotace a translace, která transformuje 3D bod ze souřadného systému objektu do souřadného systému kamery. Tento vztah lze vyjádřit rovnicí

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{21} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{3.3}$$

Pro perspektivní promítání platí, že pokud známe sadu bodů X, Y, Z v souřadném systému objektu korespondující k sadě projekcí x, y v obraze a známe vnitřní parametry kamery, lze vztah vyjádřit rovnicí

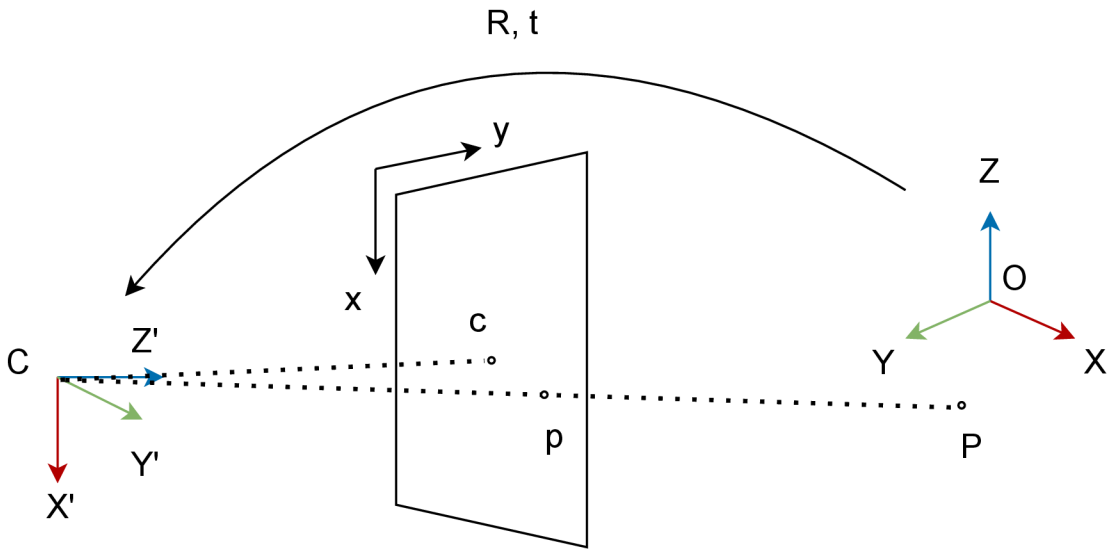
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{21} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{3.4}$$

kde f_x a f_y jsou ohniskové vzdálenosti a c_x, c_y je optický střed v souřadnicích obrazu.

Narozdíl od algoritmu RANSAC nepočítá PnP se šumem, tedy data představující „outliers“. Tato metoda je využita je při odhadu pózy v Pix2Pose [32] a Real-Time Seamless Single Shot 6D Object Pose Prediction [41]. Knihovna OpenCV obsahuje různé varianty implemetací tohoto algoritmu a mimo jiné i kombinaci PnP a RANSAC.

3.2 Metody využívající hluboké učení

Existuje mnoho prací zkoumajících odhad pózy objektů pomocí hlubokého učení. Nejčastěji se využívají konvoluční sítě, zpracovávající RGB a RGBD obrázky. Například pro detekci objektů se často vyžívá síť YOLOv3 [37], tento rychlý detektor je využit dále v síti [41],



Obrázek 3.2: Ukázka vztahů mezi kamerou, objektem a projekční rovinou. Souřadnicový systém kamery je definován středem C a osami X', Y',Z', projekční rovina osami x a y a souřadnicový systém objektu středem O a osami X, Y,Z. Pomocí rotace R a translace t bodu vyjádřeného v systému O lze bod transformovat do systému C. Bod c na obrazové rovině je optický střed obrazu a bod P je promítán do roviny na pozici p.

která slouží pro real-time predikci 6D pozy objektů. Dále se také pro detekci a segmentaci objektů využívá síť MaskRCNN [17], která efektivně detekuje objekty v obraze a generuje masku segmentace pro každou instanci. To je využito v sítích jako LieNet [12] a Pix2Pose [32], které dále zpracovávají výstup sítě pro odhad pózy objektu.

Využívají se ale i sítě, které zpracovávají přímo bodová mračna. Mezi tyto sítě patří síť PointNet[35], která je zaměřená na sémantickou segmentaci a klasifikaci objektů, a PointRCNN[40] zabývající se 3D detekcí objektů. Existují ale i sítě, které pracují s jinými datovými reprezentacemi a převádí bodové mračno do pravidelných struktur. Touto reprezentací mohou být třeba voxely, které zpracovává například síť VoxNet[30] pro rozpoznávání objektů. Nebo jsou 3D bodová mračna promítnuta do 2D obrazů, zachycujících více pohledů, a jsou zpracovávána pomocí Více-pohledové konvoluční neuronové sítě [31]. Tyto reprezentace však data mohou zkreslit.

PoseCNN

PoseCNN[1] je konvoluční neuronová síť, navržená pro odhad 6D pozice objektu v nepřehledné scéně s překrývajícími se objekty [45]. Vstupem sítě je buď RGB nebo RGBD obrázek.

PoseCNN odhad rozděluje na 3 části, jak je vidět na obrázku 3.3, nejprve pro každý pixel předpoví označení kategorie objektu, pomocí konvolučních a dekonvolučích vrstev vznikne tedy sémantické značení. Výstupem této vrstvy je n kanálů s n počtem sémantických tříd. Při trénování sémantického značení se aplikuje softmax cross-entropy loss. Během testování se používá funkce softmax pro výpočet pravděpodobnosti třídy pixelů.

[1] https://github.com/yuxng/PoseCNN

Ve druhé části dochází k odhadu posunu T, kde $T = (T_x, T_y, T_z)^T$ jsou souřadnice počátku objektu v souřadnicovém systému kamery. Odhaduje souřadnice 2D obrazových bodů středu objektu předpovídáním jednotkového vektoru z každého pixelu směrem do středu. Pomocí sémantických označení obrazové pixely spojené s objektem hlasují o umístění středu objektu v obraze. Kromě toho síť také odhaduje vzdálenost středu objektu. Projekce T je $c = (c_x, c_y)^T$. Pokud síť dokáže lokalizovat c v obraze a odhadnout hloubku T_z, můžeme nalézt T_x a T_y podle následující projekční rovnice:

$$\begin{bmatrix} c_x \\ c_y \end{bmatrix} = \begin{bmatrix} f_x \frac{T_x}{T_z} + p_x \\ f_y \frac{T_y}{T_z} + p_y \end{bmatrix} \tag{3.5}$$

kde f_x a f_y jsou ohniskové vzdálenosti kamery a $(p_x, p_y)^T$ je hlavní bod.

Výstupní kanál druhé části sítě má tedy velikost 3 x n, kde n je počet tříd objektu. Při trénování je využitá smoothed L1 loss aplikovaná na regresi. Pro odhad středu objektu je využitá Houghova hlasovací vrstva [24, 34], která bere na vstup sémantické značení a výsledek regrese středu. Hlasování probíhá tak, že každý pixel vrhá paprsky daným směrem a tím volí hlasy pro umístění obrazu podél paprsku odhadnutého ze sítě. Po zpracování všech pixelů ve třídě objektů získá skóre hlasování pro všechna umístění obrázku. Poté je jako centrum objektu vybráno místo s maximálním skóre. Pro případy, kdy se na obrázku může objevit více instancí stejné třídy objektů, je aplikováno potlačení maximálního skóre, a poté vybráno umístění se skóre větším než je určitý práh. Po vygenerování sady středů objektů jsou považovány pixely, které hlasují pro střed objektu, za inliery středu. Pak je predikce hloubky středu, T_z, jednoduše spočítána jako průměr hloubek předpovídaných inliners. Nakonec pomocí rovnice 3.5 lze odhadnout 3D posun T. Kromě toho síť generuje bounding box objektu jako 2D obdélník, který ohraničuje všechny inliery, a ten se použije pro regresi 3D rotace.

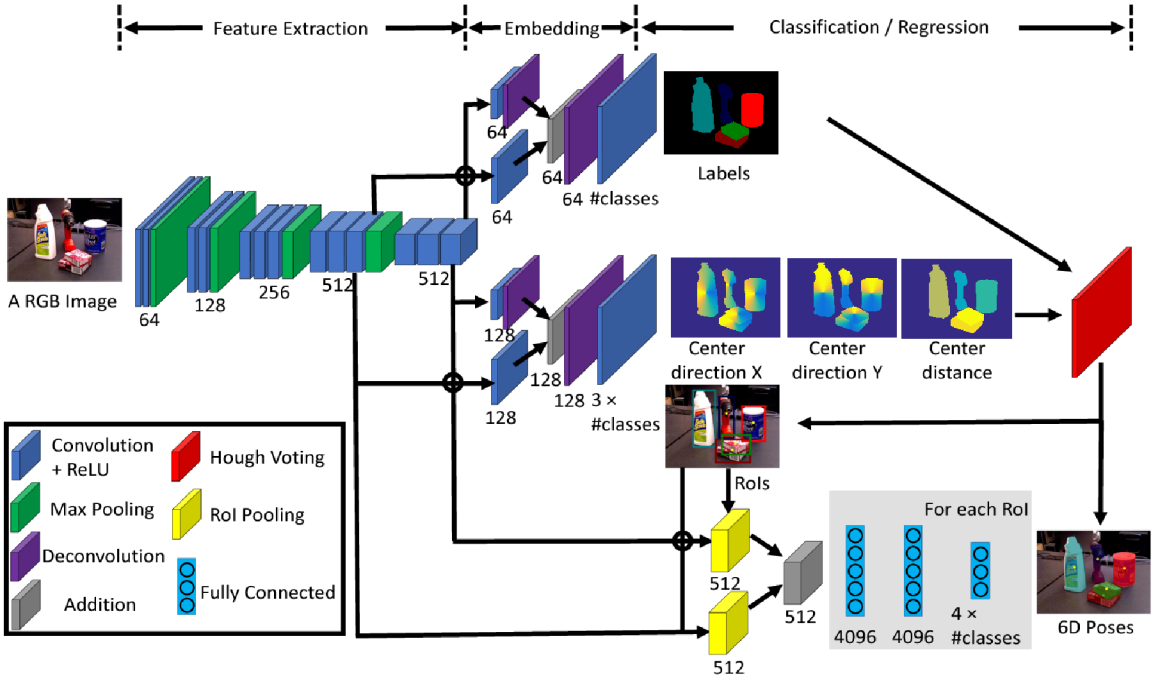
Ve třetí části sítě dochází k odhadu 3D rotace pomocí regrese. Pomocí bounding boxu objektů, předpovídaných z Houghovy hlasovací vrstvy, jsou využívány dvě vrstvy RoI pooling, pro oříznutí a seskupení vizuálních příznaků, generovaných první fází sítě pro 3D rotaci regrese. Seskupené příznakové mapy se sčítají a vloží do tří plně propojených vrstev (FC). Poslední vrstva FC má rozměr 4 x n, kde n je s počet tříd objektů. Pro každou třídu je tedy výstupem poslední vrstvy 3D rotace R reprezentovaná kvaternionem. Pro trénink kvaternionové regrese jsou dvě ztrátové funkce PLoss a SLoss. PLoss udává průměrnou kvadratickou vzdálenost mezi body cílové orientace modelu a korespondujících bodů odhadované orientace modelu. Funkce je dána rovnicí

$$PLoss(q', q) = \frac{1}{2m} \sum_{x \in \mathcal{M}} \|R(q')x - R(q)x\|^2, \tag{3.6}$$

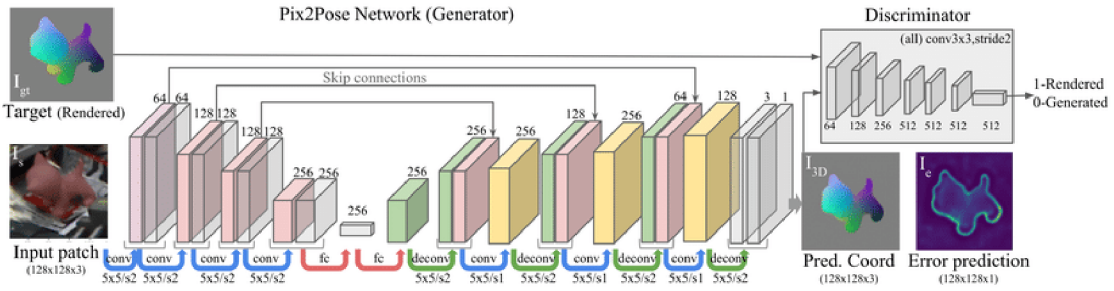
kde $\mathcal{M}$ je množina bodů 3D modelu a m je počet bodů. $R(q')$ a $R(q)$ značí rotační matice vypočítané z odhadnutého a cílového kvaternionu. Síť umožňuje zpracovávat i symetrická data, díky upravené ztrátové funkci

$$SLoss(q', q) = \frac{1}{2m} \sum_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \|R(q')x_1 - R(q)x_2\|^2, \tag{3.7}$$

udávající vzdálenost odhadované orientace a nejbližšího cílové orientace modelu.



Obrázek 3.3: Architektura sítě PoseCNN. V první části jsou konvoluční vrstvy, které slouží pro extrakci příznaků, pomocí nichž a pomocí dekonvoluce dochází k sémantické segmentaci objektů a pro každý pixel obrazu je odhadnuty kategorie objektu. Dále jsou odhadovány středy objektů a vzdálenost. Pro každý RoI (Region of interest) se odhaduje 3D rotace. Převzato z [45]



Obrázek 3.4: Architektura sítě Pix2Pose. Pomocí konvoluce a dekonvoluce dochází k odhadu pozice každého pixelu. Síť diskriminátoru se pokouší určit, zda je 3D souřadnicový obraz vykreslen pomocí 3D modelu nebo je odhadnut. Převzato z [32]

Pix2Pose

Pix2Pose [32] předpovídá 3D souřadnice každého pixelu objektu netexturovaných modelů[32]. Architektura auto-enkodéru je navržena pro odhad 3D souřadnic a očekávaných chyb na pixel. Tyto pixelové předpovědi jsou pak použity v několika fázích k vytvoření 2D-3D korespondence pro přímé výpočet pozic pomocí algoritmu PnP s iteracemi RANSAC. Tato metoda je robustní vůči překryvům díky GAN (Generative Adversarial Network), které umožní regeneraci překrytých částí. Využívá se i upravená chybová funkce, která umožňuje manipulaci se symetrickými objekty pomocí předpovědí k nejbližší symetrické póze. Pix2Pose předpovídá 3D souřadnice jednotlivých pixelů pomocí oříznuté oblasti obsahující objekt. Robustní odhad se stanoví obnovením 3D souřadnic překrytých částí a použitím všech pixelů objektu pro predikci pozice.

Pro každou třídu objektů se používá jediná síť. Vstupem do sítě je oříznutý obrázek, který používá bounding box detekované třídy objektů. Výstupy sítě jsou normalizované 3D souřadnice každého pixelu I_{3D} v souřadnici objektu a odhadované chyby I_e každé predikce. Predikce chyby I_e je považována za skóre spolehlivosti každého pixelu, který se přímo používá ke stanovení outlier a inlier pixelů před výpočtem pozice. Batch normalizace a aktivace LeakyReLU se aplikuje na každý výstup mezivrstev s výjimkou poslední vrstvy. V poslední vrstvě vytváří výstup se třemi kanály a aktivační funkcí tanh vytváří 3D souřadnicový obraz I_{3D} a další výstup s jedním kanálem pomocí aktivační funkce sigmoid odhaduje očekávané chyby I_e. Chyba rekonstrukce je definována

$$L_r = \frac{1}{n}[\beta \sum_{i \in M} \| I^i_{3D} - I^i_{gt} \|_1 + \sum_{i \notin M} \| I^i_{3D} - I^i_{gt} \|_1], \tag{3.8}$$

kde n je počet pixelů, $\beta (\geq 1)$ je faktor zvýhodňující pixely ležící pod maskou objektu před pozadím a I_{gt} je i-tý pixel cílového obrázku a M značí masku objektu cílového obrázku obsahující i pixely, které patří objektu, když je plně viditelný. Upravená funkce počítající se symeterií je dána

$$L_{3D} = \min_{p \in sym} L_r(I_{3D}, R_p I_{gt}), \tag{3.9}$$

kde $R_p \in \mathbb{R}^{3\times3}$ je transformace pozice, sym obsahuje matice udávající identitu pro dané pózy. Chybová funkce pro predikci chyby

$$L_e = \frac{1}{n} \sum_i \| I^i_e - \min[L^i_r, 1] \|^2_2, \beta = 1, \tag{3.10}$$

udává rozdíl mezi předpovězeným obrázkem I_{3D} a cílovým obrázkem I_{gt}.

Trénink s GAN generuje přesnější a realističtější obrazy v cílové doméně pomocí obrázků z jiné domény. K trénování sítě se používá diskriminátor a funkce ztrát GAN, L_{GAN}. Jak je znázorněno na 3.4, síť diskriminátoru se pokouší rozlišit, zda je 3D souřadnicový obraz vykreslen pomocí 3D modelu nebo je odhadnut. Ztráta je definována jako

$$L_{GAN} = \log D(I_{gt}) + \log(1 - D(G(I_{src}))), \tag{3.11}$$

kde D značí diskriminační síť (discriminator network). Trénování s GAN je formulováno jako funkce

$$G^* = arg \min_G \max_D L_{GAN}(G, D) + \lambda_1 L_{3D}(G) + \lambda_2 L_e(G), \tag{3.12}$$

kde λ_1 a λ_2 označují váhy pro vyvážování různých úkolů.

Souřadnice 2D obrazu a predikované 3D souřadnice přímo tvoří korespondenci. Konečná pozice je vypočítána pomocí algoritmu PnP s RANSAC, kdy dochází k maximalizací počtu inlierů, které mají menší reprojekční chyby, než je prahová hodnota.

PointNet

PointNet je síť, kterou vytvořili autoři Charles R. Qi, Hao Su, Kaichun Mo Leonidas a J. Guibas ze Stanfordské University v roce 2016 [35]. PointNet je neuronová síť zpracovávající přímo bodová mračna. Tato síť je navržená pro klasifikaci objektů, segmentaci částí objektů a sémantickou segmentaci scény. Tato síť je robustní vůči chybějícím bodům a překryvům, je efektivnější oproti sítím jiných reprezentací a zároveň dosahuje standardních výsledků.

Bodový mrak je jen množina bodů, a proto je invariantní k permutacím svých členů, což vyžaduje určité symetrizace při výpočtu sítě. Rovněž je třeba zvážit další invarianty k rigidním pohybům. Pro základní architekturu sítě v počátečních fázích je každý bod zpracováván identicky a nezávisle. Každý bod je reprezentován pouze svými třemi souřadnicemi (x, y, z). Další dimenze sítě mohou přidat vypočítané normály a další lokální nebo globální rysy.

Klíčem k jejich přístupu je použití jediné symetrické funkce, maximální sdružování neboli max pooling. Síť se efektivně učí sadě optimalizačních funkcí / kritérií, které vybírají zajímavé nebo informativní body z mračna bodů a zakódují důvod jejich výběru. Finální plně připojené vrstvy sítě agregují tyto získané optimální hodnoty do globálního deskriptoru pro klasifikace tvaru, nebo se použijí k predikci labelů podle bodu pro segmentaci tvaru. Síť se učí shrnout mračno vstupních bodů pomocí řídké sady klíčových bodů, což přibližně odpovídá kostře objektů.

3.3 Datové sady

Existují datové sady, na kterých lze trénovat sítě a vyhodnocovat úspěšnost a přesnost odhadu 6D póz objektů, tedy translaci a rotaci. Datové sady obsahují snímky scén s objekty a informace o jejich skutečné póze. Sady sestávají jak z trénovacích tak testovacích snímků, jak z reálných tak z umělých obrázků. Vytvořila jsem přehled těchto datových sad, ukázky snímků jsou zobrazeny na obrázku 6.3.

T-LESS T-LESS [19] je datová sada obsahující 6D pózu 30 průmyslových objektů bez textur. Objekty jsou symetrické a některé jsou navzájem velmi podobné, například když se jedná o různý typ jednoho produktu nebo když je objekt částí jiného. Mezi objekty jsou

například objímky na žárovky, zásuvky a vypínače. Datová sada obsahuje 38000 snímků jednotlivých objektů na černém pozadí, 77000 trénovacích a 10000 testovacích snímků z 20 různých scén. Scény mají různou složitost, která se stupňuje, od jednoduchých scén, kde se objekty nepřekrývají a obsahují jednu instanci objektu daného typu až po složitější scény obsahující překryvy, více instancí objektu a také nepořádek (neznámé předměty). Snímky jsou nasnímány RGB-D senzory.

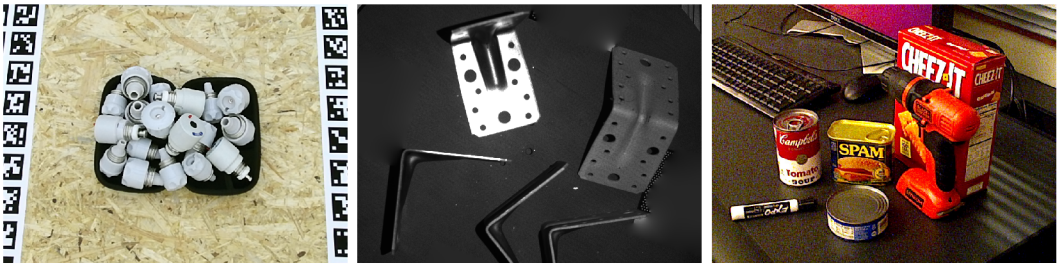
MVTec Industrial 3D Object Detection Dataset Datová sada MVTec ITODD [13] obsahuje 28 průmyslových objektů v 800 scénách. Velikost objektů je od 2,4 cm do 27 cm. Objekty mají různé charakteristiky jak v povrchu při odrážení světla, tak jednoduché i komplexní tvary, symetrické nebo ploché. Celkem je anotováno 3500 instancí objektu. Scény jsou nasnímány různými Gray-D senzory. Anotace 6D póz jsou dostupné pouze pro validační snímky ne pro testovací [23].

Linemod a Linemod-Occluded Datová sada Linemod [18] obsahuje 15 barevných objektů. Objekty představují různé domácí předměty jako hrnek, konvička, kačenka, žehlička, vrtačka a jiné. Každý objekt je spojen se sadou testovacích obrazů, která zobrazuje jednu instanci objektu s mírným překryvem, ale s významným nepořádkem. Datová sada obsahuje 20000 trénovacích a 18000 testovacích RGBD snímků. Linemod-Occluded [5] obsahuje pro danou scénu anotace všech objektů, to vytváří náročné případy s překryvy.

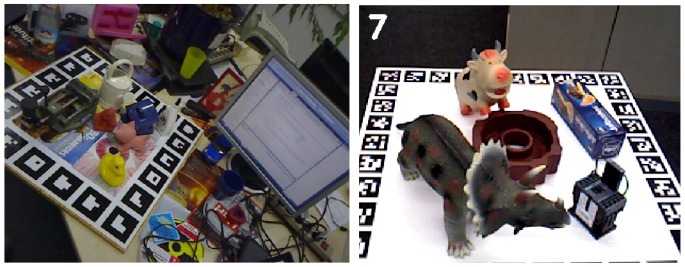
YCB-Video Datová sada YCB-Video [45] obsahuje 21 barevných objektů, opět se jedná o domácí předměty mezi které patří například různé plechovky, krabice, hrnek, nůžky a vrtačka. Některé objekty jsou symetrické. Sada obsahuje 92 videí s 133827 snímky nasnímané RGB-D kamerou. Dataset obsahuje také 80000 umělých trénovacích obrázků. Každá scéna obsahuje různý počet objektů, ale pouze jednu instanci typu objektu.

HomebrewedDB Datová sada HomebrewedDB [27] obsahuje 33 objektů (17 hraček, 8 domácích a 8 průmyslových předmětů) ve 13 scénách s různou složitostí. Snímky jsou zachyceny pomocí RGBD senzorů. Pro každou scénu je nasnímáno 340 validačních snímků a 1000 testovacích snímků. Ve scéně je pokaždé jedna instance daného typu objektu, ale obsahuje více druhů objektů.

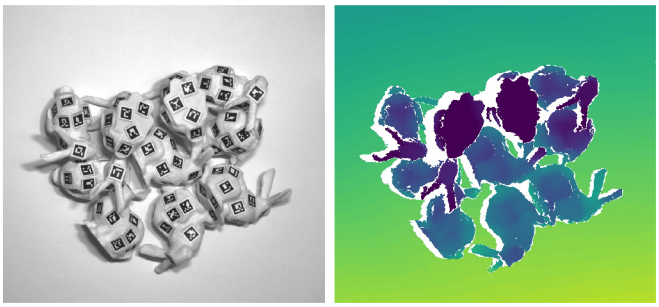
Siléane Dataset for Object Detection and Pose Estimation Siléane [8] datová sada obsahuje 8 objektů zachycených ve 2601 scénách. Každá scéna obsahuje různý počet instancí jednoho typu objektu. Datová sada obsahuje obrázky syntetických i reálných dat. Syntetická data jsou tvořena scénami představujícími vnitřek krabice, ve kterých byl vygenerován náhodný počet instancí objektů. Reálná data obsahují 5 různých objektů a syntetická 9 různých objektů. Datová sada je zaměřena na vyhodnocení symetrických předmětů.



(a) T-LESS [19] (b) ITODD [13] (c) YCB-V [45]



(d) LINEMOD [18] (e) HomebrewedDB [27]



(f) Siléane [8], RGB+D

Obrázek 3.5: Ukázka datových sad.

Kapitola 4

Návrh řešení

Vstupem mého řešení je snímek scény, obsahující známé objekty, pořízený 3D kamerou. Snímek sestává ze tří kanálů, tedy hodnoty xyz, kde z je hloubka a x,y je posun v daných osách. Objekty a jejich modely jsou předem známé. Objekty jsou rigidní, malé a lesklé. Scény obsahují větší počet instancí objektů jednoho typu a často dochází k vzájemným překryvům.

Převážná většina řešení pro odhad pózy, zpracovávaných pomocí neuronových sítí, jsou založeny na RGB obraze, například Pix2Pose [32], ale některé částečně využívají i hloubková data PoseCNN[45], dokonce PointNet[35] využívá pouze point cloud.

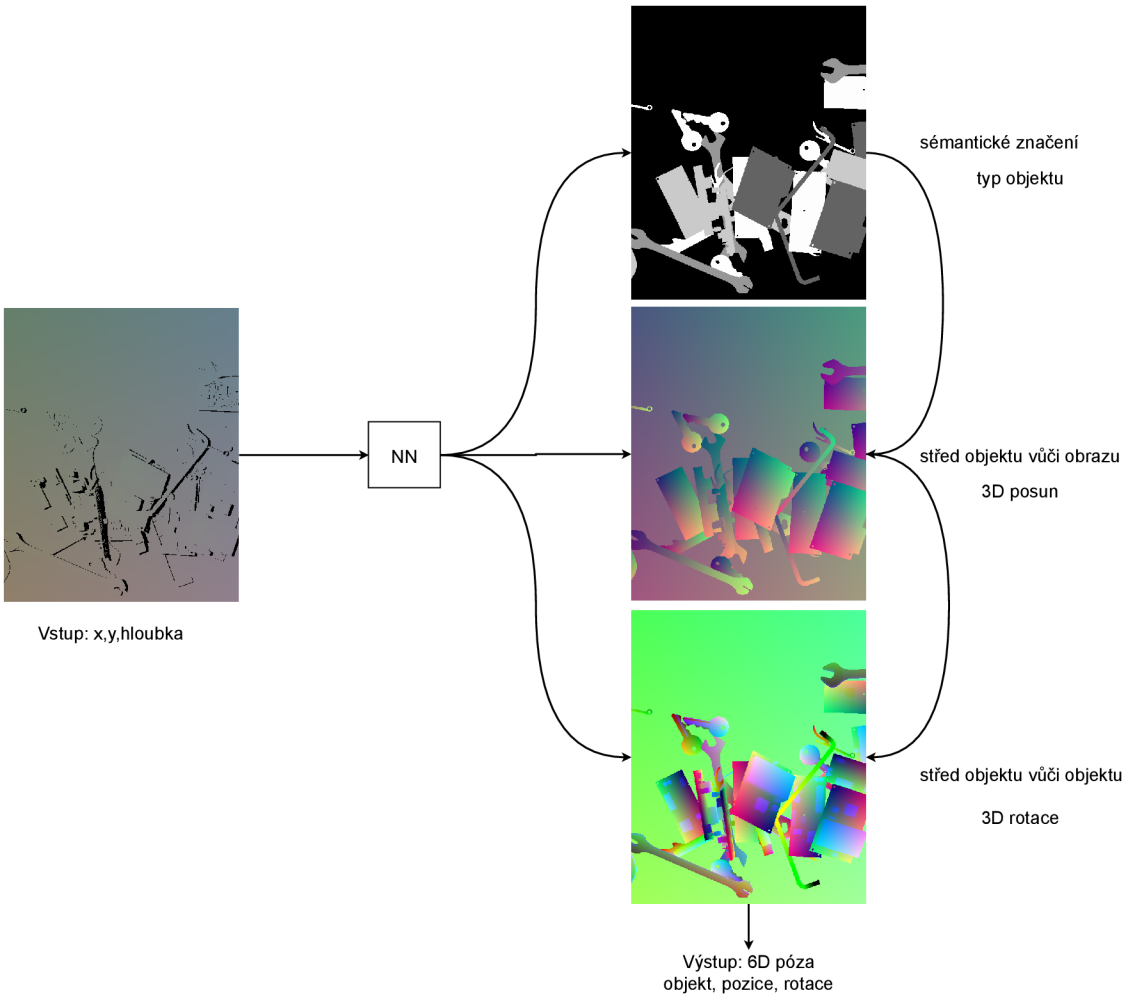
Konvoluční neuronové sítě lze použít jak pro segmentaci objektů, tak pro odhad pozice a rotace. Nejvíc jsem se inspirovala sítí PoseCNN popsané v kapitole 3, která tyto úlohy spojuje do jedné sítě. Jako vstup místo hodnot RGB jsou pouze hloubková data xyz. Zjednodušený návrh řešení je na obrázku 4.1.

Protože jsem při hledání vhodné datové sady nenalezla žádný volně dostupný, který by obsahoval podobné předměty a typ scén, rozhodla jsem se v rámci práce implementovat simulátor scén pro generování dat. Simulátor je popsán v kapitole 5. Díky simulátoru si mohu sama definovat data pro uložení a trénování a je možné vygenerovat velké množství různých scén. Navíc si můžu určit parametry tak, aby se co nejvíce podobaly reálným scénám.

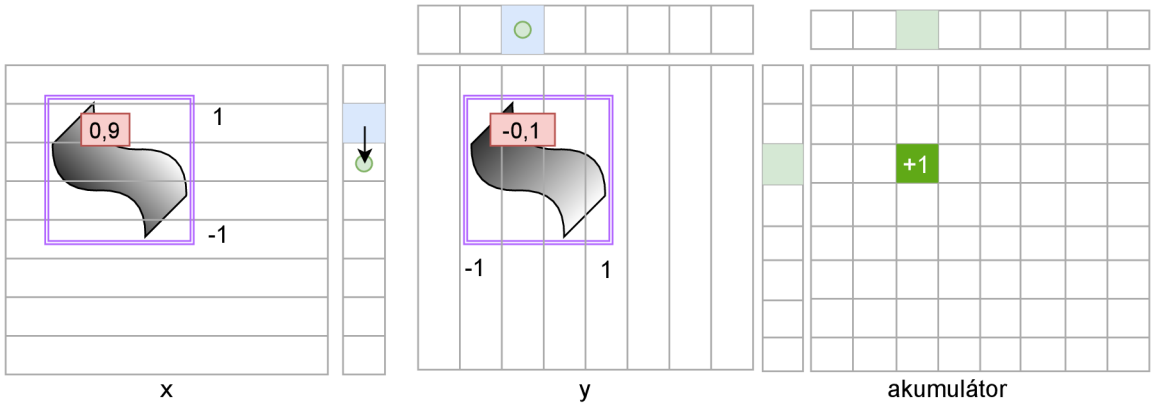
Navrhované řešení se skládá z částí jako je segmentace objektů, odhad středu objektu vzhledem k obrazu, odhad 3D posunu a odhad rotace.

4.1 Segmentace

Segmentace v obraze propojí konkrétní část obrazu s typem objektu. Pro každý pixel obrazu je odhadnut určitý typ objektu, který se na daném pixelu nachází. To umožní výběr segmentu obrazu pro konkrétní objekt. Pro získání segmentace byla využita konvoluční neuronová klasifikační síť, jejímž výstupem je vektor pravděpodobností objektu pro daný pixel. Síť je inspirována segmentační sítí U-net [38] a sítí PoseCNN [45]. Vstup je zpracován kaskádou konvolučních vrstev a následovaný kaskádou dekonvolučních vrstev, architektura sítě je zachycena na obrázku 4.3. Výstupem sítě je odhad typu objektu pro každý pixel, máme tedy sémantické značení pro každý pixel obrazu, těmito pixely pak lze filtrovat další výstupy ze sítě. Díky znalosti typu objektu pak lze použít parametry, které máme k dispozici o daném modelu, například rozměry objektu a model.



Obrázek 4.1: Vstupem sítě jsou xyz souřadnice (point cloud), vstup je zpracován pomocí neuronové sítě, nejprve dochází k ohodnocení každého pixelu do tříd objektů, následně dochází k odhadu 3D posunu objektu lokalizací jeho středu a následné využití segmentu pro odhad rotace daného objektu.

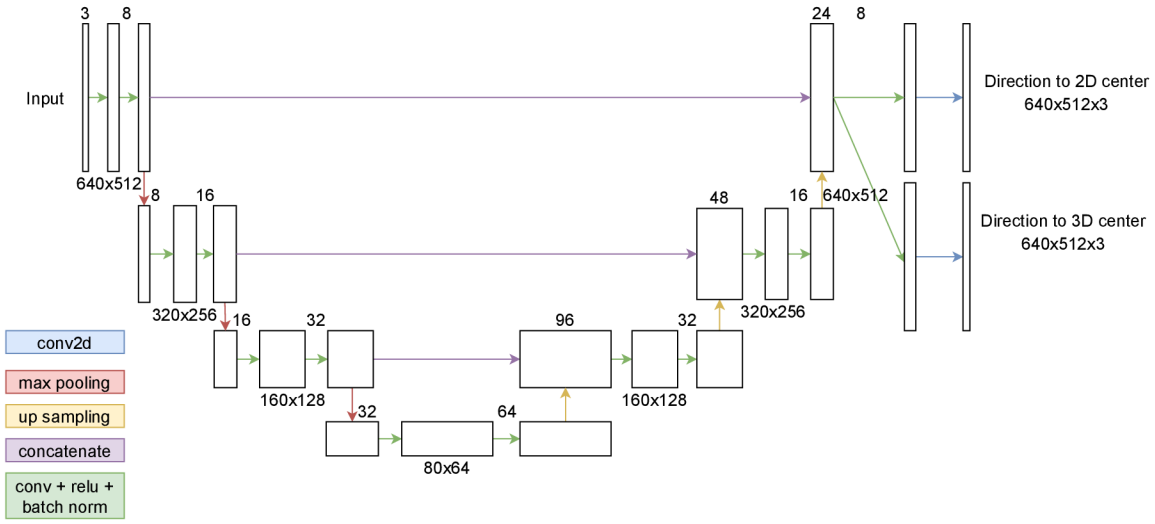


Obrázek 4.2: Akumulace hlasů středů. Aktuální hlasující pixel je značen modrou barvou a hodnota odhadu je znázorněna červenou barvou, zvolený pixel je znázorněn zelenou barvou. Pozice hlasujícího pixelu je v daném řádku x a sloupci y. Pro konkrétní objekt známe maximální rozměr, znázorněn fialově, a ten nám společně s hodnotou odhadu udává vzdálenost od aktuálního pixelu.

4.2 Lokalizace a odhad pozice

Při detekci a lokalizaci objektu se využívá volení středů v obraze neboli Houghovo volení [24, 34]. To se využívá při detekci jednoduchých vzorů v bodových vzorcích pro detekci maxim v parametrickém prostoru. Tato metoda se využívá i v PoseCNN pro lokalizaci středu objektu, kdy každý pixel vrhá paprsky daným směrem a tím volí hlasy pro umístění obrazu podél paprsku odhadnutého ze sítě. Já tuto metodu využívám tak, že je objekt lokalizován na základě výstupu ze sítě obsahující hlasování pro 2D střed, kdy každý pixel obrazu volí směr a vzdálenost středu v dané ose, tyto hlasy se akumulují, poté se vytvoří shluky reprezentující množinu hlasů pro daný střed. Akumulace hlasů probíhá v rámci obrazu, máme tedy počítadlo s rozměry výstupního obrazu, počítající hlasy pro každý pixel. Hlasy jsou sečteny tak, že hodnoty z první vrstvy, tedy hlasování v ose x udávají směr posunu od daného pixelu, který hlasuje a také vzdálenost. Hodnota hlasování je v rozsahu od 1 do -1, znaménko udává směr od středu a hodnota vzdálenost, centrum je v 0, ta je dána typem objektu, známe o jaký objekt se jedná a pro daný objekt víme maximální velikost. Z první vrstvy vyjde hlas pro řádek a z druhé vrstvy hlas pro sloupec, akumulujeme tedy počítadlo na daném pixelu, jako je zobrazeno na obrázku 4.2. Dále jsou z těchto hlasování vytvořeny shluky a pomocí metody Non-maxima suppression je vybrán nejlepší výsledek z množiny detekcí [39]. Shluky shlukují hlasy v rámci daného rozsahu. Zde dochází k problému u překrývajících se objektů, protože odhad středu jednoho objektu se může překrývat s odhadem středu jiného objektu stejného typu. U náročnějších scén je kvůli tomu problém určit správně, které hlasy patří konkrétnímu objektu.

Pro daný shluk máme daný střed a seznam hlasujících. Třetí odhadovaná hodnota je vzdálenost objektu od kamery. Z těchto hodnot lze vypočítat posun objektu vzhledem ke kameře. V mém případě je to ke středu obrazu. Zde je potřeba dodat, že střed obrazu se bere jako výchozí pozice kamery. A rozměry scény a orientace kamery se vzhledem k obrazu nemění.



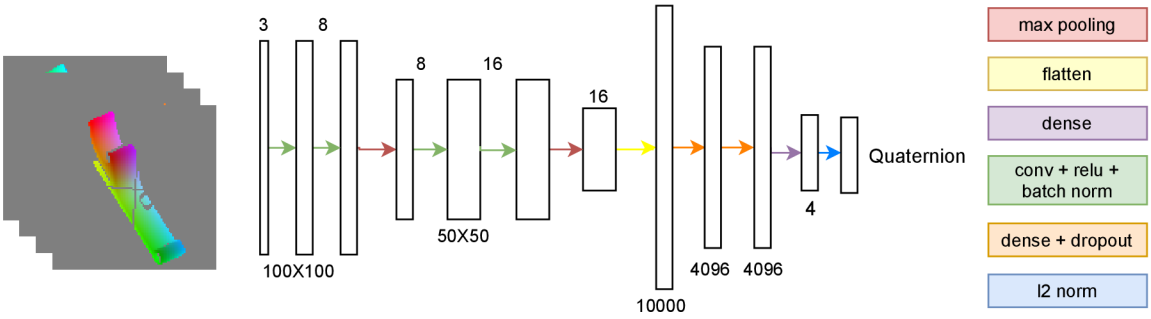
Obrázek 4.3: Architektura sítě. Vstupní vrstva sítě je dána velikostí vstupních dat 640x512x3, poté pokaždé následuje kaskáda vrstev - konvoluční, relu a batch normalizace dvakrát po sobě a poté následuje vrstva max pooling. Toto je dvakrát zopakováno a dále je vrstva max pooling nahrazena vrstvou up sampling, která je propojená z výstupem z vrstev odpovídající úrovně. Opět se dvakrát opakuje. Poslední vrstva je vrstva konvoluční a dle výstupu je aktivátorem buď funkce softmax pro odhad kategorie objektu (segmentace), lineární pro odhad směru ke středu objektu relativně k obrazu a nebo hyperbolický tangens pro odhad směru k centru objektu relativně k souřadnému systému objektu. Architektura je navržena dle vzoru sítě U-net [38] pro segmentaci obrazu.

4.3 Architektura sítě

Síť pro segmentaci i pro odhady středů je inspirována, jak už jsem zmínila, segmentační sítí U-net [38], jejímiž autory jsou Olaf Ronneberger, Philipp Fischer a Thomas Brox, a sítí PoseCNN [45] zmíněné v části 3.2. Vstupní vrstva je o velikosti 640x512x3, kde první dimenze představuje počet řádků, druhá dimenze počet sloupců a třetí dimenze počet kanálů (x,y,z). Výstupní vrstva je dvojice o velikosti 640x512x3 pro síť pro odhad 2D a 3D středu. Pro odhad 2D středu a hloubky je použita lineární aktivační funkce a pro výstup s odhadem 3D směru ke středu je aktivační funkcí hyperbolický tangens. Segmentační síť je shodná s touto sítí jen výstupní vrstva má rozměry 640x512x3(počet kategorií) a pro odhad kategorie je použita aktivační funkce softmax. Architektura sítě je zobrazena a podrobněji popsána na obrázku 4.3.

4.4 Odhad rotace objektu

V navrženém řešení jsem rozhodla síť natrénovat podobně jak na 2D směr ke středu, inspirovaný z PoseCNN, tak i na 3D směr ke středu v rámci objektu. Rozhodla jsem se využít přímo odhad 2D středu k výběru segmentu z odhadu 3D směru ke středu a tento výběr využít jako vstupní data do dalších metod pro odhad rotace. Tyto informace lze využít buď jako vstup do iteračních algoritmů ICP nebo pro PNP nebo jako vstup do sítě pro odhad rotace. Vstupem sítě pro odhad rotace je tedy segment obrazu extrahovaný pomocí hlasů z odhadu relativní pozice povrchu daného objektu. Díky segmentaci daného typu objektu



Obrázek 4.4: Architektura sítě pro regresi 3D rotace. Vstupem sítě je výřez ze segmentace 3D odhadu pixely hlasujícími pro daný střed. Jsou využity konvoluční vrstvy s aktivační funkcí relu a batch normalizací následované max poolingem. Dále jsou využity plně propojené vrstvy a nakonec je regresní funkce L2. Výstupem sítě je kvaternion.

víme o jaký typ se jedná a známe i jeho rozměry. Použila jsem tedy odlišný postup oproti PoseCNN, kde používají pro odhad rotace také větev pro 3D regresi rotace. Ta je ale napojená na výstupy z předchozích vrstev sítě s houghovým výběrem (hough voting) a vrstvy ROI pooling pro výběr segmentu obrázku. Moje síť pro regresi 3D rotace obsahuje kaskádu konvolučních a plně propojených vrstev, poslední vrstva pro regresi využívá L2 normalizaci, výstupem je kvaternion, architektura sítě je na obrázku 4.4.

Kapitola 5

Simulátor scény a generování dat

Pro trénování sítě je nutné mít velké množství dat. A protože získání reálných dat je zdlouhavý a náročný proces, vytvořila jsem místo toho generátor, který slouží k simulování reálné scény a 3D kamery pro generování trénovacích dat. Scéna představuje prostor s různým počtem objektů a 3D kamerou simulující skenování této scény. Kamera zachycuje pohled na hromadu objektů, určených pro manipulaci robota. Objekty jsou známé a definované modelem, který lze využít pro generování. V simulované scéně lze jednoduše měnit a nastavovat různé parametry jako je pozice kamery, model objektu, počet objektů pro vygenerování a další. V generátoru je také funkce pro opakované generování a snímání, které se hodí pro ukládání dat v cyklu.

5.1 Simulátor

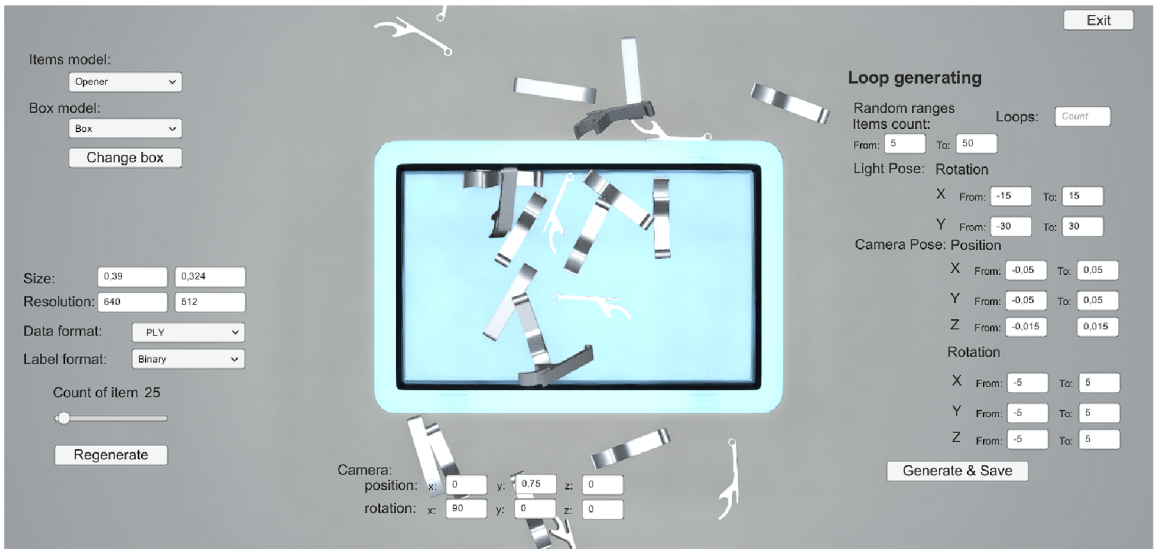
Scéna simulátoru je reprezentována podstavcem, kamerou, směrovým světlem a generátorem. Vygenerované objekty padají z výšky na podložku či do krabice, takže tvoří pokaždé jinak uspořádanou hromadu. V simulátoru dochází ke snímání scény a ukládání informací o scéně. Simulátor zohledňuje stíny, které při snímání scény reálnou kamerou vznikají při vržení stínu snímaných objektů. Ale nezohledňuje materiál objektu, proto kameru nesimuluje dostatečně přesně.

V generátoru je také funkce pro opakované generování, které se hodí pro ukládání dat v cyklu. Pro každý cyklus lze generovat, různý počet objektů, měnit pozici kamery relativně k počáteční pozici a náhodně měnit rotaci směrového světla, které způsobuje stín.

Snímek je ukládán jako ortografický snímek, proto je vytvořen objekt v pozici kamery, který generuje paprsky z daných bodů kolmo ke scéně a snímá danou vzdálenost kolmo od kamery. Zjištění hloubky je tedy provedeno pomocí vržení paprsku vycházejícího z prostoru kamery kolmo na scénu a pozicí průniku tohoto paprsku s objektem. Pro tento zasažený objekt máme ve scéně danou pozici a rotaci a také lze zjistit pozice zasažení relativně k souřadnému systému daného objektu. Generátor opakovaně generuje definované objekty v definovaném množství, dle vybraného modelu objektu.

Volitelné parametry

V generátoru lze jednoduše měnit a nastavovat různé parametry jako je počáteční pozice a rotace kamery ve scéně, model generovaného objektu z výběru již nahraných modelů a typ krabice také z daného výběru. Dále lze nastavit počet objektů pro vygenerování. Generátor umožňuje změnit počet pixelů, udávající rozlišení ukládaného snímku a velikost zachycované



Obrázek 5.1: Ukázka UI generátoru objektů s volitelnými parametry pro generování a ukládání.

scény v metrech. V generátoru je možné vybrat formát pro uložení snímku buď PLY, nebo YAML a formát po anotace buď binární, nebo textový.

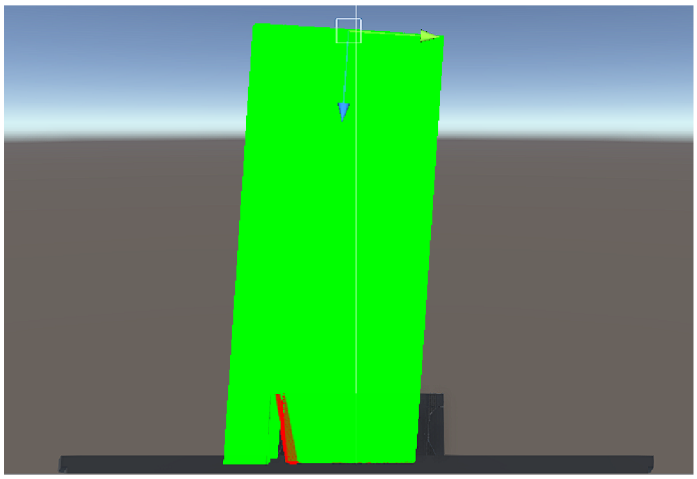
Pro generování v cyklu lze nastavit parametry jako je počet cyklů pro generování a ukládání, rozsah počtu generovaných objektů, rozsah změny pozice kamery v osách x, y, z relativně od počáteční pozice a také rozsah změny rotací v osách x, y, z. Dále je možné určit rozsah rotací bodového světla kolem os x a y relativně od počáteční hodnoty. Tyto parametry lze měnit pomocí grafického uživatelského rozhraní zobrazeného na obrázku 5.1. Přímo v projektu přes prostředí Unity je možné nastavit další proměnlivé parametry, které nejsou zavedeny v UI. Další parametry, které lze měnit jsou zapnutí možnosti generování náhodných typů objektů, vypnutí možnosti náhodné změny typu krabice při generování v cyklu, umožnění generování náhodné sady objektů jednoho typu při cyklickém generování. Přímo v Unity lze nahrát i model objektu nebo krabice, který lze přidat do seznamu pro generování.

Modely

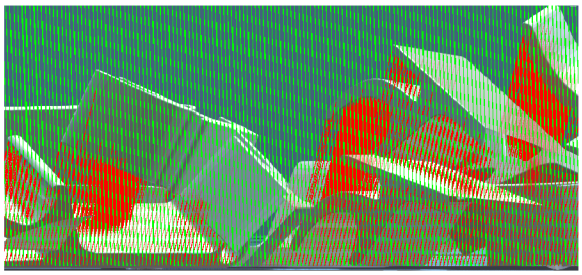
Modely pro generování jsou předem definovány. Generátor počítá s tím, že pivot modelu se nachází ve středu ohraničujícího kvádru objektu neboli bounding boxu. Pivot objektu, v Unity objekt transform, jednoznačně určuje pozici a orientaci objektu ve scéně. Některé modely objektů v generátoru jsou převzaty z firmy Kinali, některé jsou vytvořeny v Blenderu a nebo staženy z volně dostupných zdrojů. Simulátor obsahuje 3 typy krabic a 15 druhů objektů, které lze generovat, mezi ně patří například otvírák, inbus, klíč nebo vršek.

Proces snímání

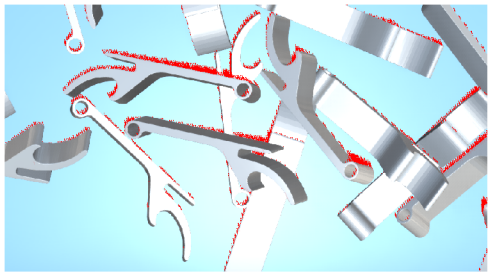
Snímání simuluje proces skenování hloubky pomocí 3D kamery. Při tomto procesu se pro každý pixel vrhne paprsek od kamery kolmo směrem dolů, jak je zobrazeno v obrázku 5.2a, a při protnutí nejbližšího objektu respektive jeho kolizní obálky je zjištěna délka vrženého paprsku. Při skenování scény 3D kamerou může docházet k vrhání stínů objektů. Toto



(a) Vrhání paprsků při skenování scény kolmo k rovině kamery.



(b) Detail paprsků dopadajících na objekty.



(c) Ukázka simulace stínů objektů.

Obrázek 5.2: Proces snímání scény pomocí paprsků v prostředí Unity. Zelené paprsky znázorňují snímání hloubky kolmo k rovině kamery a červené paprsky znázorňují stín objektu kolmý ke světlu. Snímání simuluje skenování hloubky pomocí 3D kamery.

je simulováno tak, že pokud paprsek zasahuje do stínu objektu, tak se místo délky uloží prázdná hodnota. Stín je zjištěn pomocí vržení paprsku z daného bodu směrem ke světlu a pokud daný paprsek zasáhne nějaký objekt je tento bod ve stínu, což je zobrazeno na obrázku 5.2c. Ukládání hloubkových dat probíhá dle nastavených parametrů rozlišení v ose x a y a v daném rozsahu velikosti v souřadnicích x a y. Rozlišení definuje počet generovaných paprsků a velikost definuje v jakém rozsahu budou generovány vzhledem od pozice kamery, z těchto parametrů se tedy vypočte velikost kroku tedy posun v ose x a y pro vržení paprsku respektive velikost jednoho pixelu. Vztah je popsán rovnicí $x = s/r$, kde x je velikost kroku, s je velikost snímku v dané ose a r je rozlišení snímku v dané ose.

5.2 Ukládání a reprezentace dat

Data jsou generována pro účely trénování sítě, jsou tedy třeba vstupy a výstupy. Jako vstupní data jsou snímky scény vygenerované pomocí snímání, jak je popsáno výše. Dále pro segmentaci a lokalizaci je vhodné vědět typ objektu a pozici v obraze a pro odhad 3D pózy je třeba znát translaci a orientaci jednotlivých objektů od daného počátku. Z toho důvodu jsou ukládány směry ke středu ve 2D a 3D. Celkově je tedy pro jeden snímek scény vygenerováno 5 souborů, jak je popsáno níže, a jeden globální pro všechny snímky.

Při skenování scény jsou hloubková data ukládána jako souřadnice x, y, z, kde x a y je posun paprsku od kamery a z je délka paprsku. Jedná se o ortogonální snímek. Všechny pozice se odvíjí od pozice kamery. Jak již bylo zmíněno, prázdná hodnota Nan značí stín.

Další požadovaná data jsou zjištěna z informací o průsečíku, získaném při skenování, ze kterých lze zjistit nejen pozice, ale také informace o zasaženém objektu. Další ukládanou hodnotou je směr ke středu objektu ve 2D relativní ke snímku. Tedy rozdíl průsečíku s pozicí objektu v osách x a y respektive z. A ke směru je získána i vzdálenost objektu od kamery. Vychází se z pózy, kterou jsou definovány objekty, tedy z pivotu. Při ukládání se počítá s tím, že je střed bounding boxu objektu totožný s jeho pivotem. Směr ke středu objektu vzhledem k osám x a y snímku je uložen jako hodnoty v rozsahu -1 až 1 pro každý objekt relativně k maximální hodnotě z rozměrů bounding boxu a z je vzdálenost objektu od kamery v metrech.

Také jsou ukládány pozice na povrchu objektu v souřadném systému objektu. Tato hodnota je vypočítána pomocí inverzní transformace získaného průsečíku se středem (pivotem) objektu. Opět jsou ukládány hodnoty x, y a z v rozsahu -1 až 1, pro každý objekt je to relativně ke středu jeho bounding boxu.

Pro každý snímaný pixel je také ukládáno id typu objektu respektive id modelu objektu do souboru, kde je každý pixel na jednom řádku. Tato data se hodí pro segmentaci.

Pro každý snímek jsou uloženy informace o objektech ve scéně, tedy transformační matice kamery a informace o zasažených objektech jako je id objektu, id modelu, pozice a rotace v souřadném systému kamery a pozice středu objektu ve snímku. Všechny rozměry jsou udávány v metrech.

Pro scénu nebo sadu scén je generován jeden globální soubor, ukládající obecné informace o scéně a snímku. První řádek obsahuje parametry kamery, rozlišení je reprezentováno jako počet pixelů vertikálně v ose x a horizontálně v ose y. Dále se ukládá také velikost snímané scény v metrech reprezentovaný jako výška x v a šířka y snímané scény a výčet všech modelů. Každý záznam o modelu objektu je na jednom řádku a ten udává id typu objektu, jméno objektu a rozměry bounding boxu v metrech v osách x, y a z.

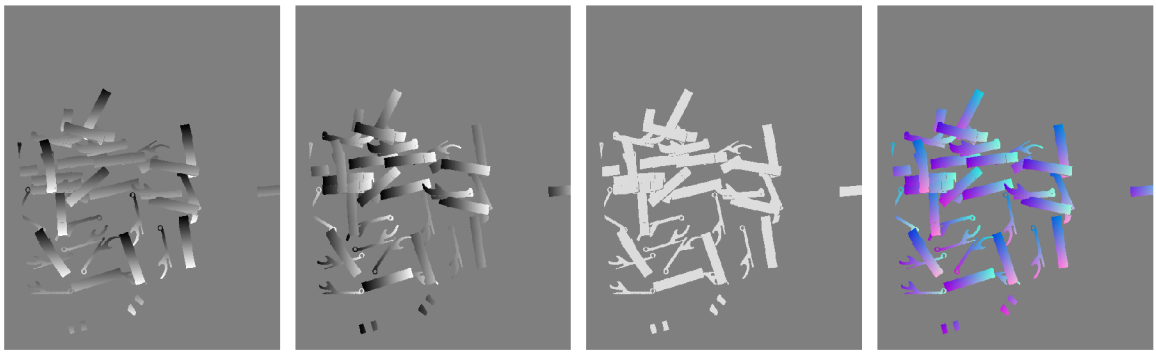
Formát ukládaných dat

Snímky se vstupními daty obsahující informace o hloubce jsou uloženy buď do souboru .ply jako vrcholy reprezentující point cloud, nebo do textového formátu .yaml. Soubory ukládající relativní pozice povrchu objektů a směry ke středu vzhledem ke snímku je možné uložit buď binárně do souboru .bin nebo v textově do souboru .txt. Soubory s informacemi o scéně a s globálními informacemi jsou uloženy jako textové soubory formátu .txt.

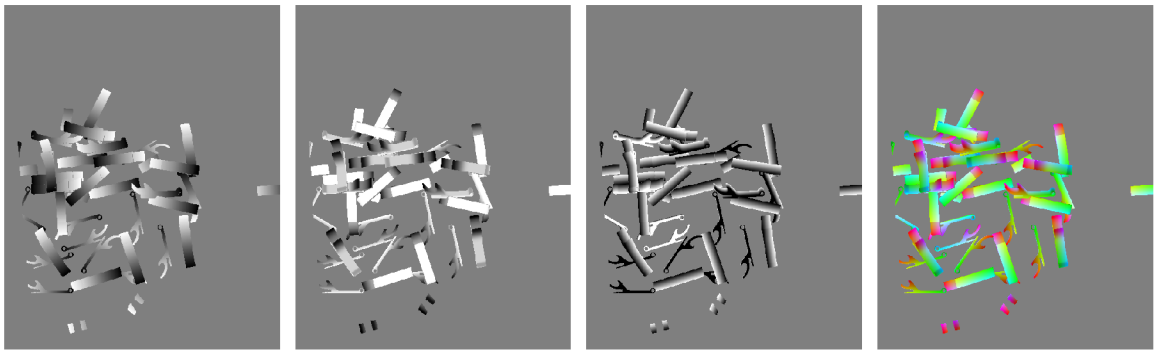
5.3 Implementace

Simulátor scény je vytvořen v prostředí Unity. Toto prostředí bylo zvoleno hlavně z důvodu intuitivního a jednoduchého vytvoření a manipulace scény, světla, objektů a kamery. Simulátor využívá fyzikální simulaci gravitace, rigidních objektů, paprsků a také objektů typu canvas a prefab.

Unity Unity [26] je herní engine, který umožňuje vývoj 2D, 3D her, simulací a virtuální reality pro různé platformy. Jádrem softwarové architektury je kolekce znovupoužitelných komponent a nástrojů používaných při vývoji, jako je editor scén a uživatelské rozhraní s funkcemi drag and drop, ale podporuje i tvorbu skriptů v jazyce C#. Prostředí Unity také



Obrázek 5.3: Vizualizace vygenerované anotace pro 2D směr ke středu objektu vzhledem k obrazu. Každý obrázek reprezentuje jeden kanál. První směr ke středu objektu vzhledem k obrazu v ose x, druhý v ose y a třetí je vzdálenost od kamery. Čtvrtý obrázek je vizualizace všech kanálů zároveň.



Obrázek 5.4: Vizualizace vygenerované anotace pro 3D směr ke středu objektu. První obrázek znázorňuje směr ke středu v ose x, druhý v ose y a třetí v ose z. Čtvrtý obrázek je vizualizace všech kanálů zároveň.

obsahuje komponenty jako jsou kamera, osvětlení, simulace fyziky jako je například pohyb, hmotnost, gravitace a kolize.

Editor Unity umožňuje prototypování a vývoj i testování v reálném čase, které mají okamžitý efekt bez nutnosti kompilace a restartování běžící simulace. Editor je jednoduše rozšiřitelný o další pluginy a nástroje. Projekt má specifickou adresářovou strukturu, obsahující konfigurační soubory a data. Projekt má logickou strukturu skládající se ze scény, herních objektů a komponent. Scéna je specifická část prostředí a je tvořena herními objekty. Herní objekt je kontejner pro komponenty, které definují vzhled, chování a vlastnosti objektů, komponenta je i skript, definující dané chování. Každý herní objekt obsahuje komponentu transform, udávající pozici, rotaci a měřítko objektu ve scéně.

Vestavěné komponenty jsou kamera, geometrický a fyzikální model objektu, světlo, render, canvas adt. Prefab je šablona vytvořená z herního objektu či jejich hierarchie a lze ji opakované vkládat do scény. Uživatelské rozhraní editoru zobrazuje přehled zdrojů, hierarchii herních objektů a pohledy na scénu s možností manipulace objektů a pohled na hru či simulaci, kterou lze spustit a testovat.

Scéna simulátoru je tvořena herními objekty pro podstavec, kameru, směrové světlo, generátor objektů, canvas a objekt, který generuje paprsky a ukládá soubory. Funkcionalita těchto objektů je implementována pomocí skriptů v jazyce C#. Skripty jsou komponenty daných objektů. Všechny generované modely jsou rigidní, tedy obsahují komponentu *RigidBody*, podléhají gravitaci a obsahují kolizní síť, tedy komponentu *MeshCollider*, díky níž jsou objekty pevné a paprsek či jiné objekty kolidují s tímto objektem. Generátor generuje nové instance objektů daného typu definované šablonou *Prefab*. Při vrhání paprsků je využit vestavěný objekt *Ray* a vrhání paprsku je pomocí funkce *Raycast*, která vrací parametr *RaycastHit* uchovávající informace o zásahu. Veškeré informace o póze daného objektu jsou ve vestavěné komponentě *transform*, díky tomu jsou v objektu pro ukládání dat tyto informace dostupné.

Kapitola 6

Implementace, datové sady a metriky

6.1 Implementace

V této části jsou popsány některé implementační detaily použité při vyhodnocování snímku. Dále jsou zmíněny nástroje, které byly využité při vývoji.

Použité nástroje

Implementace sítě i všech skriptů je v jazyce Python. Použitým nástrojem pro vytvoření a trénování sítě je knihovna TensorFlow [1] a Keras [10]. Pro trénování sítí jsem použila Google Colab [4]. Dále jsem použila knihovnu Open3D [46] pro práci s point couldem a jejich algoritmus pro ICP registration. Pro práci s transformacemi a rotacemi jsem využila knihovnu SciPy [44]. Metriky jsem čerpala z BOP [21] a také jsem použila funkce pro metriky definované v BOP toolkit[1] [22]. Pro práci s obrázky a pro vizualizace jsem použila knihovnu OpenCV [7] a Pillow [43]. Pro práci s polem jsem použila knihovnu NumPy [16] a pro vytváření grafů knihovnu Matplotlib [25].

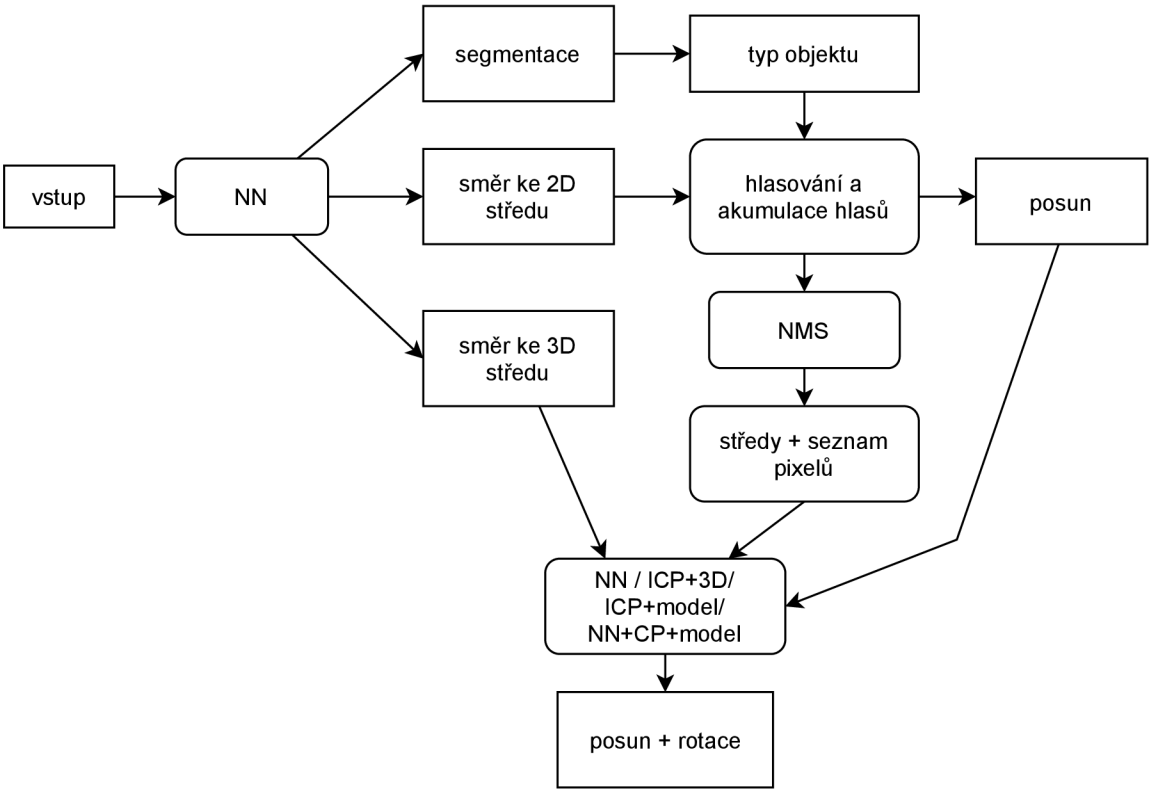
Pro simulátor a generátor scén jsem použila prostředí Unity [26] a implementace funkcionalit objektů je pomocí skriptů v jazyce C#. Simulátor jsem podrobněji rozebrala v kapitole 5.

Pro anotování scény jsem použila prostředí Blender [11] a pomocí python skriptu jsem přidala panel, který poskytuje pár funkcí pro načítání a ukládání scény a objektů do mnou používaného formátu.

6.2 Proces zpracování vstupu při odhadu pózy

Proces zpracování vstupního obrazu při odhadování pózy je naznačen na obrázku 6.1. Vstupní obraz je zpracován pomocí neuronových sítí. V mém řešení jsem použila celkem 3 sítě. Jednu pro segmentaci a druhou pro odhad 2D středu a hloubky a 3D povrchu objektu, tyto dvě sítě jsou velmi podobné až na počet výstupních vrstev, kde ze segmentační sítě je výstup o velikosti jako vstup až na počet kanálů, ten je ekvivalentní k počtu kategorií a z druhé sítě vychází dva výstupy o velikosti vstupu. Toto je podrobněji popsáno v kapitole 4 v sekci 4.3. Tedy po vyhodnocení vstupu těmito sítěmi je výstup ze segmentační

[1] https://github.com/thodan/bop_toolkit

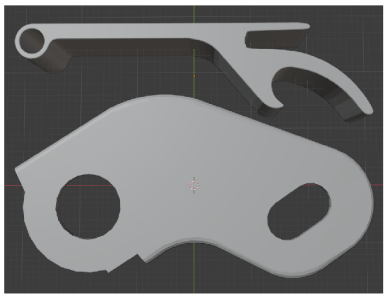


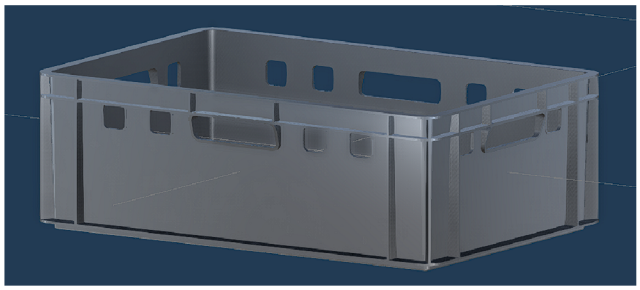
Obrázek 6.1: Proces zpracování obrazu a vyhodnocení pozice a rotace objektu.

sítě využit pro filtrování výstupu s 2D odhadem směru z druhé sítě. Poté je tento výstup zpracován akumulátorem, popsaným v sekci 4.2 a algoritmem Non-maximum suppression, který vybírá z hlasů a vytváří shluky pixelů hlasující pro daný střed v rámci stanovené vzdálenosti. Na základě středů a pixelů pro dané středy je vypočítán průměr z odhadu hloubky vzhledem ke kameře z třetího kanálu vrstvy odhadu 2D středu. Tyto pixely jsou poté dále využity pro filtrování druhého výstupu sítě s 3D odhady středu objektu a výřezu daného bounding boxu, který je dále využit jako vstup do sítě pro odhad rotace.

Takovéto použití je pouze jedno z řešení, dále je také v rámci experimentu porovnání s úspěšnosti s použitím algoritmu ICP. Využila jsem algoritmus implementovaný v knihovně Open3D [46]. Vstupem tohoto algoritmu je buď model, který známe na základě znalosti kategorie nebo místo modelu je použitý odhadu 3D směru, kdy jsou odhadnuté hodnoty vynásobeny na základě znalosti typu objektu danými rozměry v dané ose. Dále je shlukem pixelů segmentován vstupní obraz a toto je použito jako cílový point cloud do algoritmu ICP registration. Parametr *max_correspondence_distance* je nastaven na 0.015 a parametr *max_iteration* nastaven na 3000. Inicializační transformační matice je nastavena na výchozí hodnotu tedy na identitu. V rámci experimentu byl také využit odhad z rotační sítě jako vstup do inicializační transformační matice.

Tyto odhady jsou pak při vyhodnocování párovány s anotacemi, kde maximální vzdálenost pixelu od anotace je $\pm$ 5 px.





Obrázek 6.2: Modely generovaných objektů Opener(otvírák) a Metal) a ukázka krabice.

Objekt	Rozměry		
Opener	0.0593	0.0154	0.011
Metal	0.0624	0.0320	0.0032
Box	0,3	0.2	0.12
BoxResized	0.45	0.3	0.12
BigBox	0.6	0.4	0.2
Container	0.4	0.3	0.32

Tabulka 6.2: Rozměry objektů v metrech. Spodní objekty oddělené linkou jsou pro modely krabic.

6.3 Datové sady

V této kapitole je přehled datových sad využitých v mé práci. Při trénování sítí byla využita pro trénování trénovací datová sada a testovací generovaná datová sada pro validaci. Experimenty jsem pak prováděla na generované testovací sadě a na ručně anotované sadě. Stručný přehled datových sad je v tabulce 6.1.

Datová sada	počet scén	počet objektů
trénovací	200	11148
generovaná testovací	90	4946
anotovaná testovací	61	2197

Tabulka 6.1: Přehled dat v datové sadě. Trénovací datová sada je vygenerovaná. Testovací datové sady jsou dvě, jedna je vygenerovaná a druhá je ručně anotovaná.

Trénovací datová sada Síť jsem trénovala na vygenerované datové sadě ze 2 objektů, celkem tedy obsahuje 3 kategorie objektů, kde jednou z kategorií jsou jiné objekty jako například krabice, ty jsou vnímány jako kategorie None. Pro každý objekt je vygenerováno 100 scén, 50 jednodušších scén, kde při generování byl parametr počtu objektů 15-50 a 50 složitějších scén, které byly vygenerovány s parametrem pro počet objektů 50-150. Ostatní parametry simulátoru pro generování jsou uvedeny tabulce 6.3. Celkově je v trénovací sadě 200 scén obsahující dohromady 11148 objektů. V každé scéně je náhodně vybráno z různých typů krabic nebo i bez krabice. Ukázky objektů a jedné krabice jsou na obrázku 6.2 a rozměry všech použitých modelů jsou v tabulce 6.2.

Testovací datové sady Testovací datové sady mám dvě, jedna je generovaná a obsahuje anotace všech dat potřebných pro trénování, tedy data potřebná pro segmentaci, což je informace o kategorii objektu pro každý pixel, anotace 2D směru ke středu vzhledem k obrazu a hloubku objektu, anotace 3D směru ke středu v souřadném systému objektu a pak také veškeré informace o objektech ve scéně, tedy pózy všech objektů a jejich kategorie. Druhá testovací datová sada je vytvořena z reálných scén, je ručně anotovaná a obsahuje informace pouze o objektech ve scéně, tedy jejich pózu a kategorii.

Generovaná testovací datová sada obsahuje celkem 90 scén s 4946 objekty. Pro každý typ objektu je vygenerováno 30 scén a 30 scén s oběma objekty zároveň. Parametr počtu objektů při generování jednoho typu byl 15-150 a 15-50 při generování mixu. Ostatní parametry simulátoru pro generování jsou uvedeny v tabulce 6.3. V každé scéně je náhodně vybráno z různých typů krabic nebo i bez krabice, stejně jako pro trénovací sadu.

Druhá testovací datová sada je vytvořená ze snímků reálných scén nasnímaných pomocí 3D kamery. Tyto snímky jsou ručně anotovány, celkem jsem anotovala 61 scén s 2197 objekty, pro typ objektu s modelem Opener (otvírák) je 30 snímků a pro druhý typ 31 snímků. Scény jsou různě složité a obsahují různý počet objektů. Ruční anotace je vytvořená pomocí nástrojů v editoru Blender [11]. Reálné snímky z kamery byly transformovány do formátu shodného s generovanými snímky, tedy rozlišení bylo sníženo na 640x512, měřítko bylo zmenšeno z cm na m. A data byla uložena v pořadí shodném s generovanými snímky, aby bylo možné shodné načítání a zpracování těchto dat.

Parametry generátoru		od	do
Rotace světla	x	-15	15
	y	-30	30
Posun kamery	x	-0.05	0.05
	y	-0.05	0.05
	z	-0.01	0.01
Rotace kamery	x	-5	5
	y	-5	5
	z	-5	5
Parametry snímku		x	y
Rozlišení		640	512
Velikost		0.39	0.324

Tabulka 6.3: Parametry generátoru pro testovací a trénovací datové sady.

6.4 Metriky

V této sekci jsou zmíněny metriky pro vyhodnocení chyb, přesností a úspěšnosti odhadovaných hodnot. Metriky jsem použila při vyhodnocování výsledků v experimentech. Vyhodnocení natrénovaných modelů lze pomocí porovnání výsledků odhadu sítě s anotacemi na testovacích sadách. Vyhodnocení přesnosti a úspěšnosti odhadu pózy je pak pomocí porovnání výsledků metod s anotacemi.

Chyba odhadu 2D a 3D vzdálenosti od středu objektu Přesnost odhadu je vypočítána jako střední kvadratická chyba. Definice pro střední kvadratickou chybu je dána

rovnicí

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - Y_i')^2, \tag{6.1}$$

kde Y je skutečná hodnota a Y' je odhad. Odhady jsou v rozsahu hodnot -1 až 1. Odhad 2D směru je relativní k rozměrům objektu vzhledem k obrazu a zároveň v závislosti na velikosti pixelu, chyba je tedy různá pro jednotlivé objekty v závislosti na maximálním rozměru objektu a zároveň v závislosti na velikosti pixelu. Odhad 3D směru je závislý na rozměrech objektu.

Chyby odhadu 6D pózy

Póza objektu je reprezentována maticí 4×4 $P = [R, T; 0, 1]$, kde R je rotační matice 3×3 a T je translační vektor 3×1. Matice P transformuje 3D bod x_m v souřadném systému modelu na 3D bod x_c v souřadnicovém systému kamery $x_c = Px_m$.

Chyba translace Chyba pro odhad posunu objektu je vyjádřena rovnicí

$$E_t = \|T - T'\|, \tag{6.2}$$

kde T je anotovaný posun a T' je odhadovaný posun. Celková chyba translace je průměr chyby přes všechny výsledky.

Chyba rotace Chyba pro rotaci objektu je vypočítána pomocí rovnice

$$E_r = \arccos \frac{\mathrm{Tr}(R'R^{-1}) - 1}{2}, \tag{6.3}$$

kde R je anotovaná rotační matice a R' je odhadovaná rotační matice a funkce Tr je součet diagonálních prvků.

Průměrná vzdálenost odpovídajících bodů modelu [18] 3D model objektu je definován jako sada vrcholů v $\mathbb{R}^3$, které popisují povrch objektu. Póza objektu je reprezentována maticí P a odhadována póza maticí P'. Výpočet průměrné vzdálenosti dvojic mezi 3D modelovými body transformovanými podle cílové pózy a odhadované pózy lze vyjádřit rovnicí

$$E_{ADD} = \mathrm{avg}_{x \in \mathcal{M}} \|Px - P'x\|, \tag{6.4}$$

kde $\mathcal{M}$ značí množinu bodů 3D modelu. Pozice je považována za správnou, pokud je vzdálenost menší než předem určený práh. Tato metrika je použita například v PoseCNN pro vyhodnocení úspěšnosti odhadu s prahem 10 % průměru modelu.

Tyto metriky jsou vyhodnoceny na výsledcích pro objekty, jejichž odhadovaný střed je v toleranci 5 pixelů od anotace.

Definované metriky jsem čerpala z práce On Evaluation of 6D Object Pose Estimation [20] a také jsem využila volně dostupné zdrojové kódy[2]. Další metriky pro srovnání datasetů jsou uvedeny v navazující práci Benchmark for 6D Object Pose Estimation [21], kde je zpracováno srovnání 15 metod pro odhad 6D póz objektů.

[2] https://github.com/thodan/obj_pose_eval

Kapitola 7

Experimenty

V této kapitole jsou uvedeny výsledky provedených experimentů a zhodnocení vybraných metod. Je zde zhodnocena úspěšnost sítě a porovnání úspěšnosti výsledků vybraných metod pro odhad póz.

7.1 Přesnost odhadu sítě

V této části jsou experimenty prováděné na generované testovací sadě. Je zde vyhodnocena úspěšnost segmentace a chyba odhadu výstupu ze sítě vyhodnocující 2D směr ke středu a vzdálenost objektu od kamery a také 3D směr ke středu v souřadném systému objektu.

Úspěšnost segmentace pixelů

Vyhodnocení segmentace probíhalo na všech testovacích datech, tedy na 90 scénách pro všechny kategorie typu objektů, zároveň přes všechny pixely. Celkem byly 3 kategorie objektů: Opener, Metal a None. Na této sadě byla dosažena úspěšnost odhadu $96,3\%$.
To je v přepočtu na obrázek 640x512 zhruba výřez o velikosti 110x110 špatně vyhodnocených pixelů.

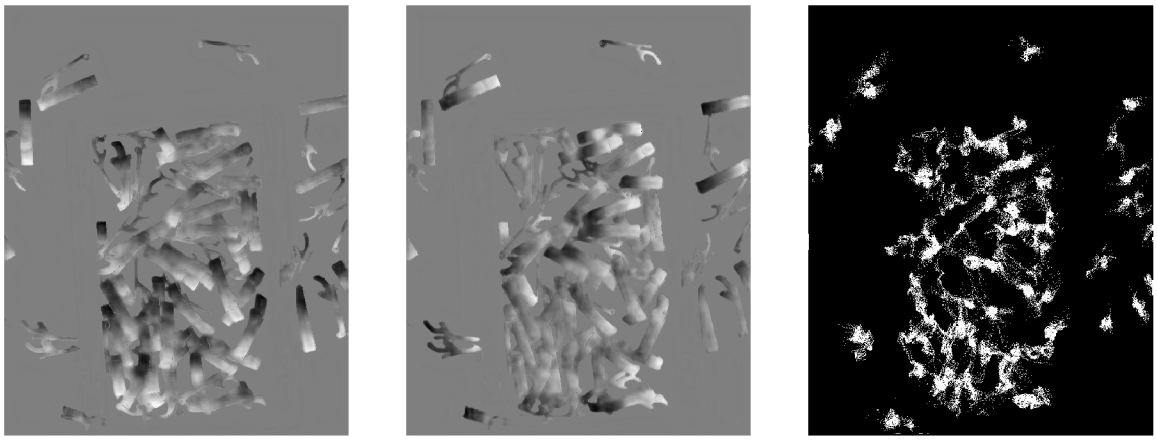
Chyba odhadu 2D středu objektu vzhledem k obrazu a chyba odhadu vzdálenosti od kamery

Výpočet chyby odhadu 2D středu objektu vzhledem k obrazu je pro konkrétní objekt vypočítán přes všechny pixely, které byly segmentovány pomocí výstupu sítě odhadující segmentaci. Pro výpočet průměrné chyby odhadu pro konkrétní objekt v pixelech lze vynásobením chyby a maximálního rozměru objektu děleno velikostí pixelu v dané ose. Protože je odhad relativní ke každému objektu v rozsahu -1 až 1. Odhad vzdálenosti je v metrech a není relativní k objektu, není tedy potřeba jej přepočítat. Chyba odhadu sítě pro směr k 2D středu objektu a vzdálenosti objektu je uveden v tabulce 7.1.

Zhodnocení Z tabulky a obrázku 7.1 lze vyčíst, že chyba odhadu 2D středu vytváří značný šum při hlasování pro daný střed objektu. To lze částečně eliminovat pomocí využitého shlukování, ale pro velký počet překrývajících se objektů jednoho typu je problémové určit, které hlasy patří jakému objektu.

		chyba 2D odhadu a hloubky	chyba v px
Opener	x	0.0223	2.2
	y	0.0225	2.1
	z	0.0081	
Metal	x	0.0251	2.6
	y	0.0237	2.3
	z	0.0149	

Tabulka 7.1: Tabulka zachycuje chybu odhadu výstupu 2D vzdálenosti ke středu ze sítě na testovacích datech pro každý typ objektu, chyba je vypočítána ze segmentované části odhadu pro daný typ objektu a každou souřadnici a je relativní k danému objektu.



Obrázek 7.1: Ukázka výstupu ze sítě pro odhad 2D středu a výstupu akumulace hlasování pro střed.

Chyba odhadu 3D relativní pozice objektu

Výpočet chyby odhadu 3D středu objektu vzhledem k objektu je pro konkrétní objekt vypočítán přes všechny pixely, které byly segmentovány pomocí výstupu sítě odhadující segmentaci. Chyba odhadu je v tabulce 7.2. Pro výpočet průměrné chyby v rámci objektu lze vypočítat chybu v metrech vynásobením chyby a rozměru daného objektu v dané ose.

		chyba 3D odhadu	chyba v mm
Opener	x	0.0603	3.5
	y	0.0554	0.8
	z	0.1360	1.5
Metal	x	0.0751	4.7
	y	0.0607	1.9
	z	0.0964	0.3

Tabulka 7.2: Tabulka zachycuje chybu odhadu 3D středu vzhledem k objektu ze sítě na testovacích datech pro každý typ objektu, chyba je vypočítána ze segmentované části odhadu pro daný typ objektu a každou souřadnici.

7.2 Přesnost odhadu póz objektů

V tomto experimentu je měřena chybovost jednotlivých metod na sady, rozdělené dle objektů. Vyhodnocení je jak na generované, tak i na reálné testovací datové sadě. V experimentu jsem srovnala výsledky použití jednotlivých metod pro odhad pozice a rotace objektu. Mezi srovnanými výsledky metod je

- použití metody pro odhad pozice na základě odhadu středu ze shluků a odhad rotace pomocí sítě, jejímž vstupem je segmentace 3D odhadu pomocí pixelů hlasujících pro daný střed
- použití algoritmu ICP na segmentovanou scénu hlasujícími pro daný střed s použitím modelu objektu
- algoritmu ICP na segmentovanou scénu se segmentovaným odhadem 3D povrchu
- použití ICP s modelem a inicializační transformací modelu pomocí odhadu rotace ze sítě

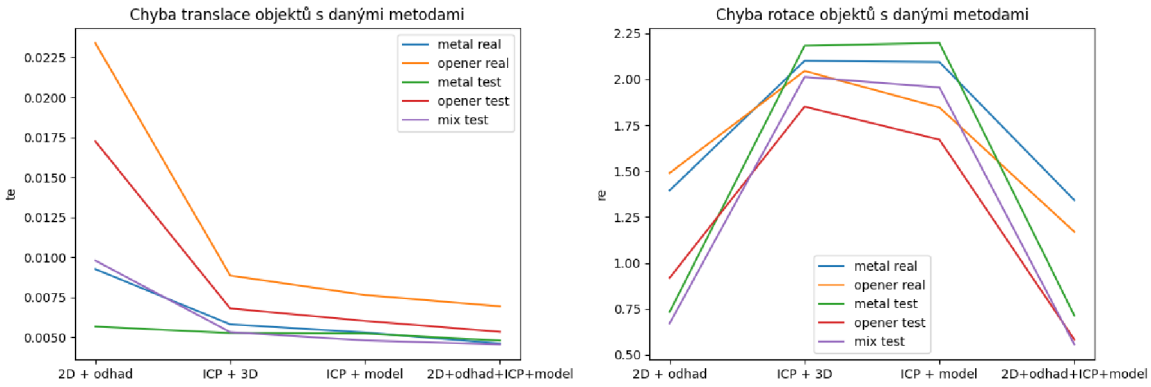
Srovnání výsledků a anotací probíhá na detekovaných a spárovaných objektech, párování probíhalo na základě 2D středu s rozsahem ± 5 px. Nutno dodat, že toto párování může také produkovat další chyby v odhadu.

Přesnost odhadu pozice Chyba přesnosti odhadu translace je vypočítána pomocí rovnice 6.4. Srovnány jsou metody zmíněné výše na jednotlivé sady s danými objekty. Přesnost výsledků jednotlivých metod pro odhad translace je vyjádřena v grafu 7.2.

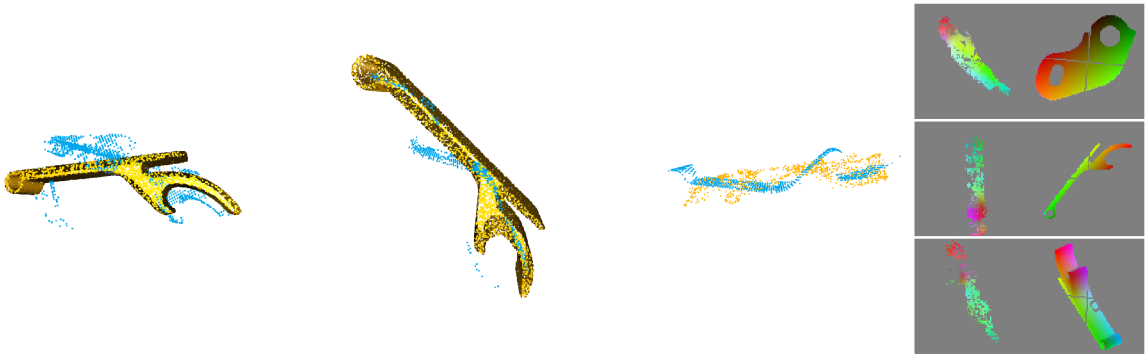
Z grafu lze vyhodnotit, že odhad translace jen na základě mojí metody je velice chybový. Pomocí dorovnání algoritmem ICP je odhad pozice o hodně lepší. Pro objekt opener na reálné scéně je odhad mojí metodou víc jak 22 mm mimo a na generované scéně víc jak 17 mm, pro objekt metal je to zhruba 1 cm. Pomocí algoritmu ICP je chyba snížena pro objekt opener zhruba o polovinu na 1 cm a nejpřesnějšího odhadu dosahuje metoda s použitím ICP a inicializací matice odhadnutou rotací, ale i tak je chyba odhadu pro objekt otvíráku průměrně víc jak 7 mm. Pro větší objekt metal je odhad pozice obecně přesnější jak pro scény reálné, tak pro generované.

Přesnost odhadu rotace Chyba přesnosti odhadu rotace je vypočítána pomocí rovnice 6.3. Srovnány jsou metody zmíněné výše na jednotlivé sady s danými objekty. Přesnost výsledků jednotlivých metod pro odhad rotace je vyjádřena v grafu 7.2. V grafu je vidět, že využití metody ICP na taková data není moc vhodné, protože v hodně případech se zarovnání zaseklo na lokálním minimu. Odhady pomocí neuronové sítě je tedy vhodnějším použitím pro takováto data. Ale i tak je chybovost poměrně vysoká. Jak lze vidět v grafu, tak na reálných scénách je chybovost odhadu sítě značně vyšší. Pohybuje se kolem 1,25-1,5 rad. Při použití metody pouze s 2D odhadem a odhadem rotace je chybovost na generované sadě s objektem opener necelý radián a na objektu metal je to zhruba 0,75 rad. Přesnost je tedy bohužel velmi nízká. Nejlepších výsledků dosahuje opět metoda s inicializací algoritmu ICP na rotaci z odhadu sítě. Zde je chybovost na generovaných sadách mezi 0,5-0,75 rad.

Pro zadané objekty je dorovnání modelu pomocí algoritmu ICP na vstupní výřez scény problémový z důvodu zarovnání bodů v lokálním minimu, jak je zobrazeno na obrázku 7.3 . Velmi problémovým může být i velké množství šumu, které vzniká při hlasování pro daný střed z pixelu jiného objektu.



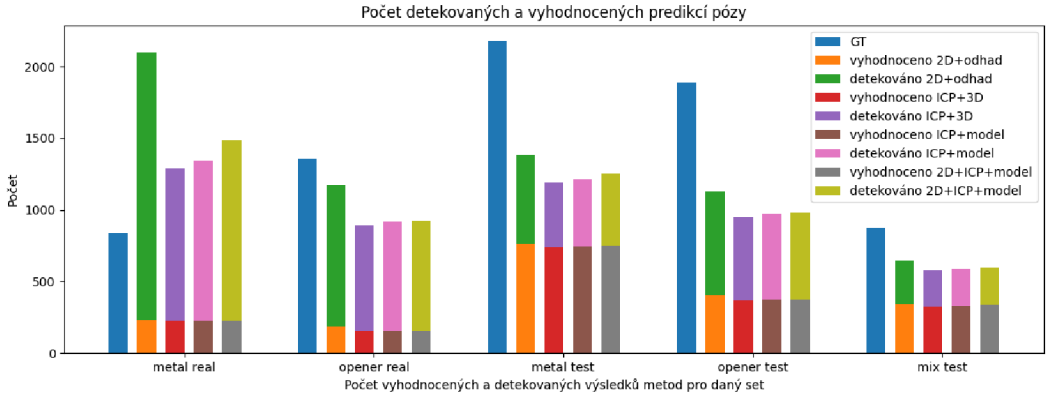
Obrázek 7.2: Grafy zobrazují chyby odhadu pro jednotlivé objekty použitím jednotlivých metod. Jsou zde zahrnuty hodnoty vyhodnocené na testovací generované sadě a na ručně anotované sadě. Počet využitých predikcí je vyjádřen v grafu 7.4.



Obrázek 7.3: První tři obrázky znázorňují špatné napasování modelu. Segmentace scény může obsahovat velké množství šumu, jak je znázorněno v levém obrázku. Ve druhém a třetím obrázku je zachyceno napasování modelu v lokálním minimu, objekt je ale orotovaný o 180°. V třetím obrázku je ukázka pasování 3D odhadu na segment scény, tedy bez použití modelu. Poslední obrázky vpravo zobrazují ideální vstupní data do sítě pro odhad rotace, vlevo je vstup segmentovaný pomocí hlasů pro 2D střed.

Neuronová síť pro odhad rotace byla natrénovaná na ideálních výřezech 3D pozice povrchu objektu, tedy na výřezech z anotací. Ukázka je na obrázku 7.3. Vstupní data se tedy mohou od anotovaných často velice lišit. Přesnějších výsledků by se mohlo docílit při trénování na výřezech z výsledku predikce, kdy vstupní výřez by byl segmentován pomocí hlasů pro daný střed. Data by tedy byla řídká oproti anotaci a blížila by se více vyhodnocovaným vstupům.

Počet detekovaných a vyhodnocených odhadů Jak již bylo zmíněno, srovnání výsledků a anotací probíhá na detekovaných a spárovaných objektech, párování probíhalo na základě 2D středu s rozsahem ± 5 px. Vyfiltrovalo tedy mnoho neporovnaných predikcí. Na obrázku lze vidět například velké množství nevyužitých predikcí, hlavně u objektu metal na scénách s ručními anotacemi. Tento velký nepoměr je způsoben hlavně predikcemi v oblastech se šumem, jak lze vidět na obrázku 7.5, kvůli neznámé podložce je v této oblasti špatně



Obrázek 7.4: Graf vyjadřuje počet detekovaných a vyhodnocených odhadů v rámci všech grafů. Tedy počet spárovaných predikcí versus počet detekovaných ale nespárovaných a nevyužitých v rámci vyhodnocení chybovosti. Velké množství nezahrnutých predikcí může být způsobeno šumem v rámci odhadu 2D středu, špatnou segmentací typu, či velkým množstvím shluků s malým počtem hlasů.

odhadnutý typ objektu. Také je vidět velký nepoměr množství anotací, oproti predikcím u generovaných scén. Toto je zase způsobeno tím, že jsou ukládány všechny snímané objekty, tedy i takové, které jsou téměř celé překryté a nejsou téměř viditelné. Všechny metriky jsou vyhodnoceny na datech dle grafu 7.4 označené jako vyhodnoceno.

7.3 Úspěšnost na reálných datech

Úspěšnost na reálných datech Vyhodnocení na reálných, ručně anotovaných scénách je vyhodnoceno v tabulce 7.3. Hodnoty odhadů jsou pro srovnání zaznamenány v grafech 7.2 společně s odhady na generované testovací sadě. Na obrázku 7.5 a 7.6 jsou vizualizace výstupů ze segmentační sítě a sítě pro odhad 2D a 3D středů objektů. Lze vidět, že jak segmentace tak i odhady pro středy obsahují velké množství šumu a to jak kvůli přítomnosti šumu tak i neznámou podložkou.

model		2D+odhad	ICP+3D		ICP + model		odhad+ICP+model	
Metal	T	23.8 %	48.4 %	46.7 %	54.7 %	68.7 %	59.7 %	81.5%
	R	31.6 %	14.9 %	13.8 %	23.3 %	23.4 %	48.2 %	51.8 %
Opener	T	12.4 %	27.6 %	27.3 %	39 %	38.4 %	46.2 %	51.9 %
	R	21.6 %	12.2 %	13.6 %	25.2 %	23.2 %	45.6 %	48.7 %

Tabulka 7.3: Úspěšnost odhadu translace T a rotace R na reálných scénách pro dané objekty, tolerance pro úspěch je posun 0.005 m a rotace 0,262 rad (cca 15°). Vyhodnocení je na výsledcích pro objekty, jejichž odhadovný střed je v toleranci 5 pixelů od anotace. Parametr *max_correspondence_distance* je 0.015 pro první sloupec s použitím ICP a 0.005 pro výsledky ve druhém sloupci.

Největší úspěšnosti dosáhla metoda využívající jak odhad rotace pomocí sítě, tak i algoritmu ICP pro dopasování point cloudu. Celkově menší úspěšnosti bylo dosaženo na objektu otvírák, nejspíš kvůli menší velikosti objektu a s přítomností segmentace scény

s falešnými hlasy dochází ke špatnému dopasování, dle obrázku 7.3. Dále kvůli tomu že snímané scény s objektem Metal jsou jednodušší a obsahují celkově méně instancí objektu. Jak lze vyčíst z grafu 7.4.

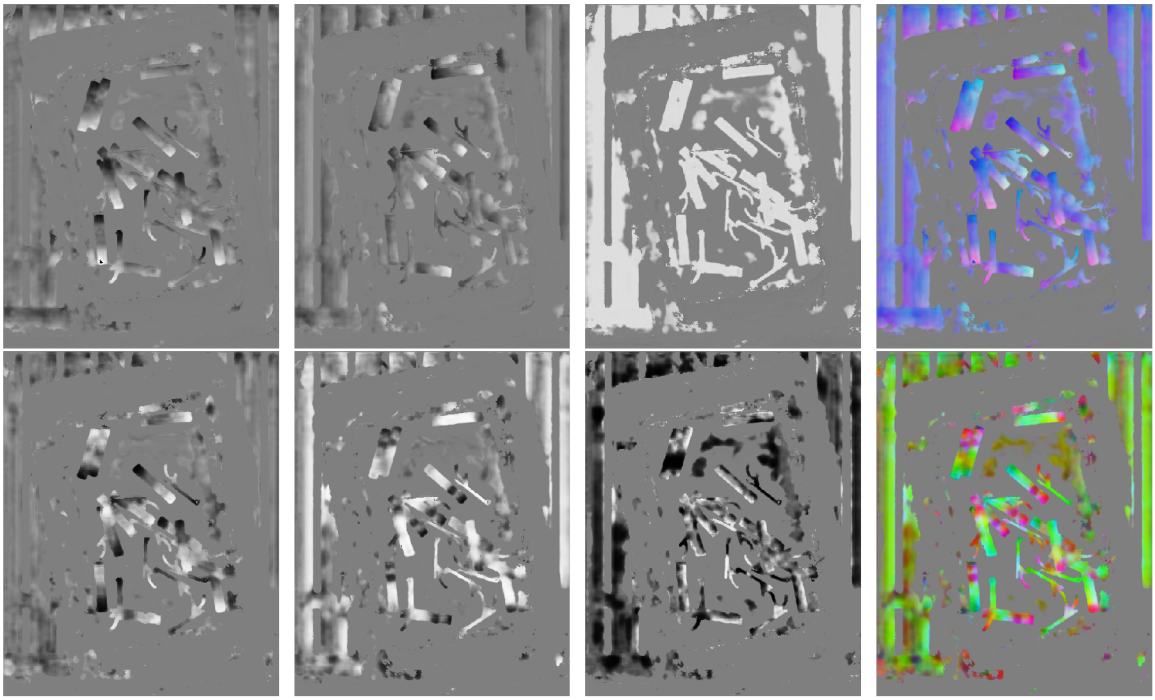
Úspěšnost na generovaných datech Vyhodnocení na generovaných, ručně anotovaných scénách je vyhodnoceno v tabulce 7.4. Hodnoty odhadů jsou pro srovnání zaznamenány v grafech 7.2 společně s odhady pro ručně anotované testovací sadě.

		2D+odhad	ICP+3D		ICP + model		odhad+ICP+model	
Metal	T	66.3 %	61.1 %	64.8 %	58.1 %	75.1 %	62.4 %	89.7 %
	R	62.1 %	15.4 %	14.5 %	19.9 %	20.1 %	65.9 %	76 %
Opener	T	44.1 %	49.2 %	42.9 %	58.6 %	64.1 %	64.0 %	78 %
	R	41.1 %	26.8 %	23.5 %	40.1 %	36 %	73.3 %	77.2 %
mix	T	59.8 %	62.6 %	64.7 %	67 %	78.1 %	69.8 %	90.6 %
	R	56 %	24.8 %	21.4 %	30.9 %	30.7 %	76.3 %	82.4 %

Tabulka 7.4: Úspěšnost odhadu translace T a rotace R na generovaných scénách pro jednotlivé objekty, tolerance pro úspěch je pokud chyba posunu Te $<$ 0.005 m a pro rotaci pokud chyba rotace Re $<$ 0,262 rad (cca 15°). Vyhodnocení je na výsledcích pro objekty, jejichž odhadovaný střed je v toleranci 5 pixelů od anotace. Parametr *max_correspondence_distance* je 0.015 pro první sloupec s použitím ICP a 0.005 pro výsledky ve druhém sloupci.

Generované scény celkově obsahují malé množství šumu a tedy úspěšnost na těchto datech je výrazně lepší i při větším množství instancí. Při použití ICP s modelem objektu a s větším prahem *max_correspondence_distance* byla úspěšnost odhadu posunu na obou objektech srovnatelná. Avšak při zvýšení prahu, tedy zlepšení přesnosti výsledku algoritmu, se odhad posunu výrazně zlepšil jen pro větší objekt Metal. Odhad rotace je ale pořád úspěšnější pro model Opener. To bude pravděpodobně způsobeno tím, že objekt Metal je relativně plochý a zarovnání pro obě rotace je velmi podobné.

Obrázek 7.5: Vizualizace výstupu ze segmentační sítě vyhodnocené na reálné scéně. První obrázek znázorňuje kategorii None, druhý kategorii Opener a třetí Metal. Na třetím obrázku lze vidět chybně odhadnutou kategorii, kvůli velkému množství šumu a neznámé podložce.



Obrázek 7.6: Vizualizace výstupů ze sítě pro odhad středů na reálné scéně. První obrázek znázorňuje směr ke středu v ose x, druhý v ose y a třetí v ose z. Jak je vidět tak tyto obrázky obsahují značné množství šumu a také odhady pro objekty v oblasti podložky. Proto vzniká ve scéně velké množství chybných detekcí.

Kapitola 8

Zhodnocení a možnosti budoucího vývoje

Samotná navržená metoda pro odhad posunu a rotace nebyla moc vhodná pro tento typ scény, na objektu Metal dosáhla úspěšnost odhadu pouze 23.8 % pro translaci a 31.6 % úspěšnost pro odhad rotace, pro druhý objekt Opener byla úspěšnost pouze 12.4 % pro odhad translace a 21.6 % pro odhad rotace, přičemž tolerance pro správný odhad je 5 mm a 15°. Použitím algoritmu ICP na odhadnuté výsledky je dosažena úspěšnost odhadu translace 81.5 % a rotace 51.8 % pro objekt Metal a pro objekt Opener je úspěšnost odhadu translace 51.9 % a 48.7 % úspěšnost odhadu rotace. Pro větší objekty je tato metoda jednou z možných variant, ale pro menší objekty není odhad dostatečně přesný.

Odhad rotace by se dal zlepšit buď tím, že by byla pro každý typ objektu zvláštní vrstva pro odhad rotace nebo, jak jsem zmiňovala, natrénováním na odhad z vrstvy vyhodnocující hlasování, ne tedy na anotacích, ale na výstupech sítě a algoritmu, tak jak to bylo použito přímo v PoseCNN, takže můj návrh řešení nebyl moc vhodný.

Úpravou generátoru a definováním stejné podložky jako je použita v reálných snímcích, by se mohl odhad výrazně zlepšit, dále také přidáním a simulací většího množství šumu. Dalším vývojem by tedy mohlo být i zlepšení simulátoru například přidáním simulace chování materiálu nebo využitím většího rozlišení. Díky tomu by mohl být i odhad posunu přesnější.

Kapitola 9

Závěr

Cílem této práce bylo prostudovat problematiku lokalizace a odhadu pozice objektu pro bin picking a navrhnout řešení pro hledání objektu a odhad pózy objektu na datech snímaných z 3D kamery. Tyto data zachycující krabici s množstvím malých, lesklých a překrývajících se předmětů. Cílem je tedy získat pozici objektů v krabici tak, aby bylo možné tento předmět uchopit pomocí robota. Tato práce shrnuje přehled současně používaných metod pro lokalizaci objektů a odhad pózy objektů. Dále jsou podrobněji probrány metody neuronových sítí, aplikovaných na problém detekce a segmentace objektů a odhad pozice. V rámci práce byl vytvořen simulátor scény, napodobující reálnou scénu, který je vhodný ke generování trénovacích dat. Byla navržena metoda inspirovaná sítí PoseCNN, která zpracovává vstupní point cloud. Síť je natrénována na generované trénovací datové sadě. Také byly vytvořeny ruční anotace scén, použité pro vyhodnocení přesnosti odhadu. Samotná navržená metoda pro odhad posunu a rotace nebyla moc vhodná pro tento typ scény. Na jednom z objektů dosáhla úspěšnost odhadu pouze 23.8 % pro translaci a 31.6 % úspěšnost pro odhad rotace, u druhého objektu byla úspěšnost pouze 12.4 % pro odhad translace a 21.6 % pro odhad rotace, přičemž tolerance pro správný odhad je 5 mm a 15°. Použitím algoritmu ICP na odhadnuté výsledky je dosažena úspěšnost odhadu translace 81.5 % a rotace 51.8 % pro první objekt a pro druhý je úspěšnost odhadu translace 51.9 % a 48.7 % úspěšnost odhadu rotace. V rámci další práce by mohl být simulátor vylepšen, aby věrněji simuloval scénu.

Literatura

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Dostupné z: https://www.tensorflow.org/.

[2] Alismail, H., Baker, L. D. a Browning, B. Continuous trajectory estimation for 3D SLAM from actuated lidar. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Květen 2014, s. 6096–6101. DOI: 10.1109/ICRA.2014.6907757.

[3] Besl, P. a McKay, N. D. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1992, sv. 14, č. 2, s. 239–256. DOI: 10.1109/34.121791.

[4] Bisong, E. *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. Leden 2019. ISBN 978-1-4842-4469-2.

[5] Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J. et al. Learning 6D Object Pose Estimation Using 3D Object Coordinates. In: Fleet, D., Pajdla, T., Schiele, B. a Tuytelaars, T., ed. *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014, s. 536–551. ISBN 978-3-319-10605-2.

[6] Brachmann, E., Michel, F., Krull, A., Yang, M. Y., Gumhold, S. et al. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In:. červen 2016, s. 3364–3372. DOI: 10.1109/CVPR.2016.366.

[7] Bradski, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. 2000.

[8] Brégier, R., Devernay, F., Leyrit, L. a Crowley, J. L. Symmetry Aware Evaluation of 3D Object Detection and Pose Estimation in Scenes of Many Parts in Bulk. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct 2017.

[9] Chen, Y. a Medioni, G. Object modelling by registration of multiple range images. *Image and Vision Computing*. 1992, sv. 10, č. 3, s. 145 – 155. DOI: https://doi.org/10.1016/0262-8856(92)90066-C. ISSN 0262-8856. Range Image Understanding. Dostupné z: http://www.sciencedirect.com/science/article/pii/026288569290066C.

[10] Chollet, F. et al. *Keras* [https://keras.io]. 2015.

[11] Community, B. O. *Blender - a 3D modelling and rendering package*. Stichting Blender Foundation, Amsterdam: Blender Foundation, 2018. Dostupné z: http://www.blender.org.

[12] Do, T.-T., Pham, T., Cai, M. a Reid, I. D. LieNet: Real-time Monocular Object Instance 6D Pose Estimation. In: *BMVC*. 2018.

[13] Drost, B., Ulrich, M., Bergmann, P., Härtinger, P. a Steger, C. Introducing MVTec ITODD — A Dataset for 3D Object Recognition in Industry. In: *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2017, s. 2200–2208. DOI: 10.1109/ICCVW.2017.257.

[14] Fischler, M. A. a Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*. 1981, sv. 24, s. 381–395.

[15] GmbH, I. I. D. S. *Ensenso N35* [online]. Online; Accessed: 2020-01-20. Dostupné z: https://en.ids-imaging.com/ensenso-n35.html.

[16] Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P. et al. Array programming with NumPy. *Nature*. Springer Science and Business Media LLC. září 2020, sv. 585, č. 7825, s. 357–362. DOI: 10.1038/s41586-020-2649-2. Dostupné z: https://doi.org/10.1038/s41586-020-2649-2.

[17] He, K., Gkioxari, G., Dollar, P. a Girshick, R. Mask R-CNN. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct 2017.

[18] Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G. et al. Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In: Lee, K. M., Matsushita, Y., Rehg, J. M. a Hu, Z., ed. *Computer Vision – ACCV 2012*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, s. 548–562. ISBN 978-3-642-37331-2.

[19] Hodaň, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M. et al. T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects. *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017.

[20] Hodaň, T., Matas, J. a Obdržálek, Š. On Evaluation of 6D Object Pose Estimation. In: Hua, G. a Jégou, H., ed. *Computer Vision – ECCV 2016 Workshops*. Cham: Springer International Publishing, 2016, s. 606–619. ISBN 978-3-319-49409-8.

[21] Hodan, T., Michel, F., Brachmann, E., Kehl, W., Buch, A. G. et al. BOP: Benchmark for 6D Object Pose Estimation. *CoRR*. 2018, abs/1808.08319. Dostupné z: http://arxiv.org/abs/1808.08319.

[22] Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E. et al. BOP Challenge 2020 on 6D Object Localization. *European Conference on Computer Vision Workshops (ECCVW)*. 2020.

[23] Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E. et al. *BOP: Benchmark for 6D Object Pose Estimation - Datasets* [online]. 2020. Dostupné z: https://bop.felk.cvut.cz/datasets/.

[24] Hough, P. V. C. Machine Analysis of Bubble Chamber Pictures. *Conf. Proc. C.* 1959, sv. 590914, s. 554–558.

[25] Hunter, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering.* IEEE COMPUTER SOC. 2007, sv. 9, č. 3, s. 90–95. DOI: 10.1109/MCSE.2007.55.

[26] Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J. et al. *Unity: A General Platform for Intelligent Agents.* 2020.

[27] Kaskman, R., Zakharov, S., Shugurov, I. a Ilic, S. HomebrewedDB: RGB-D Dataset for 6D Pose Estimation of 3D Objects. *International Conference on Computer Vision (ICCV) Workshops.* 2019.

[28] Kehl, W., Milletari, F., Tombari, F., Ilic, S. a Navab, N. Deep Learning of Local RGB-D Patches for 3D Object Detection and 6D Pose Estimation. In: Leibe, B., Matas, J., Sebe, N. a Welling, M., ed. *Computer Vision – ECCV 2016.* Cham: Springer International Publishing, 2016, s. 205–220. ISBN 978-3-319-46487-9.

[29] Lowe, D. G. Object recognition from local scale-invariant features. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision.* Sep. 1999, sv. 2, s. 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410. ISSN null.

[30] Maturana, D. a Scherer, S. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* Sep. 2015, s. 922–928. DOI: 10.1109/IROS.2015.7353481.

[31] Pang, G. a Neumann, U. 3D point cloud object detection with multi-view convolutional neural network. *2016 23rd International Conference on Pattern Recognition (ICPR).* 2016, s. 585–590.

[32] Park, K., Patten, T. a Vincze, M. Pix2Pose: Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation. In: *The IEEE International Conference on Computer Vision (ICCV).* Oct 2019.

[33] Pavlakos, G., Zhou, X., Chan, A., Derpanis, K. G. a Daniilidis, K. 6-DoF Object Pose from Semantic Keypoints. *CoRR.* 2017, abs/1703.04670. Dostupné z: http://arxiv.org/abs/1703.04670.

[34] Qi, C. R., Litany, O., He, K. a Guibas, L. J. Deep Hough Voting for 3D Object Detection in Point Clouds. In: *Proceedings of the IEEE International Conference on Computer Vision.* 2019.

[35] Qi, C. R., Su, H., Mo, K. a Guibas, L. J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CoRR.* 2016, abs/1612.00593. Dostupné z: http://arxiv.org/abs/1612.00593.

[36] Rad, M. a Lepetit, V. BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth. *CoRR.* 2017, abs/1703.10896. Dostupné z: http://arxiv.org/abs/1703.10896.

[37] Redmon, J. a Farhadi, A. YOLOv3: An Incremental Improvement. *ArXiv.* 2018.

[38] Ronneberger, O., P.Fischer a Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 2015, sv. 9351, s. 234–241. LNCS. (available on arXiv:1505.04597 [cs.CV]). Dostupné z: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a.

[39] Rosenfeld, A. a Thurston, M. Edge and Curve Detection for Visual Scene Analysis. *IEEE Transactions on Computers*. 1971, C-20, č. 5, s. 562–569. DOI: 10.1109/T-C.1971.223290.

[40] Shi, S., Wang, X. a Li, H. PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[41] Tekin, B., Sinha, S. N. a Fua, P. Real-Time Seamless Single Shot 6D Object Pose Prediction. *CoRR*. 2017, abs/1711.08848. Dostupné z: http://arxiv.org/abs/1711.08848.

[42] Truebenbach, E. Is fully automated bin picking finally here? *The Robot Report*. 2019. Dostupné z: https://www.therobotreport.com/fully-automated-bin-picking-finally-here/.

[43] Umesh, P. Image Processing in Python. *CSI Communications*. Citeseer. 2012, sv. 23.

[44] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T. et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*. 2020, sv. 17, s. 261–272. DOI: 10.1038/s41592-019-0686-2.

[45] Xiang, Y., Schmidt, T., Narayanan, V. a Fox, D. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *CoRR*. 2017, abs/1711.00199. Dostupné z: http://arxiv.org/abs/1711.00199.

[46] Zhou, Q.-Y., Park, J. a Koltun, V. Open3D: A Modern Library for 3D Data Processing. *ArXiv:1801.09847*. 2018.

Příloha A

Obsah DVD

- `anotator/` - adresář obsahující nástroj pro editor Blender
  - `original_data/` - ukázka originálních dat
  - `Models/` - modely pro anotační nástroj
- `dataset/`
  - `test/` - generovaná testovací sada
  - `train/` - generovaná trénovací sada
  - `val/` - anotovaná testovací sada
- `generátor` - adresář obsahující simulátor scén a generátor snímků v prostředí Unity
- `generator-build/` - build generátoru
- `scripts/` - veškeré skripty pro načítání dat, trénování sítí a vyhodnocení
  - `models/` - modely reprezentované point cloudem
  - `weights/` - váhy pro natrénované sítě
- `texts/` - text diplomové práce a plakát