Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-XXXX-XXXXX

Bc. Katarína Juhásová

REGISTRÁCIA OBJEKTOV NA MRAČNE BODOV POMOCOU HLBOKÝCH NEURÓNOVÝCH SIETÍ

Priebežná správa o riešení DP2

Vedúci práce: RNDr. Andrej Lúčny, PhD.

Január 2022

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-XXXX-XXXXX

Bc. Katarína Juhásová

REGISTRÁCIA OBJEKTOV NA MRAČNE BODOV POMOCOU HLBOKÝCH NEURÓNOVÝCH SIETÍ

Priebežná správa o riešení DP2

Študijný program:	Inteligentné softvérové systémy
Študijný odbor:	Informatika
Miesto vypracovania:	Ústav informatiky, informačných systémov a softvérového inžinierstva
Vedúci práce:	RNDr. Andrej Lúčny, PhD.

Január 2022

SLOVENSKÁ TECHNICKÁ
UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY
A INFORMAČNÝCH TECHNOLÓGIÍ

Návrh zadania diplomovej práce

Finálna verzia do diplomovej práce [1]

Študent:

Meno, priezvisko, tituly:	Katarína Juhásová, Bc.
Študijný program:	Inteligentné softvérové systémy
Kontakt:	katkajuhasova8@gmail.com

Výskumník:

Meno, priezvisko, tituly:	Andrej Lúčny, RNDr. PhD.

Projekt:

Názov:	Registrácia objektov na mračne bodov pomocou hlbokých neurónových sietí
Názov v angličtine:	Object Registration in 3D Point Clouds Using Deep Neural Networks
Miesto vypracovania:	Ústav informatiky, informačných systémov a softvérového inžinierstva, FIIT STU
Oblasť problematiky:	hlboké učenie

Text návrhu zadania[2]

Mračná bodov sú významnou štruktúrou v oblasti počítačového videnia, avšak výskum ich spracovania pomocou hlbokého učenia je stále v počiatkoch. Pri ich spracovaní je potrebné brať do úvahy komplikácie vyplývajúce zo štruktúry, ktorá sa vyznačuje svojou nepravidelnosťou a nesúmernosťou. Doterajší výskum však ukázal, že hlboké neurónové siete umožňujú vysporiadanie sa s týmito problémami, čím je umožnené vykonávať rôzne úlohy nad mračnami bodov, napríklad detekciu objektov, ich klasifikáciu alebo registráciu, t.j. nájdenie zadaných objektov v danom mračne.

Naštudujte problematiku spracovania 3D skenov a hĺbkových máp v miere potrebnej pre vytvorenie vhodnej sady dát. Spracujte poskytnuté dáta získané anotáciou mračien bodov zodpovedajúcim nasnímaným pneumatikám. Využite na to knižnice OpenCV, prípadne PCL. Postup vytvorenia sady dát popíšte v práci.

Oboznámte sa s problematikou hlbokého učenia. Uveďte práce, ktoré ideovo predchádzali sieťam spracúvajúcim mračná bodov (ako je napríklad PointNet) v miere potrebnej pre pochopenie činnosti týchto sietí.

Vyskúšajte vyriešiť problém registrácie objektov pomocou hlbokej neurónovej siete vhodnej architektúry a zhodnoťte kvalitu tohto riešenia. Porovnajte možnosti riešenia založeného na hlbokom učení s klasickými metódami počítačového videnia. Na implementáciu riešenia použite prostriedky ako GPU, TensorFlow + Keras, Python, OpenCV a ďalšie.

---

[1] Vytlačiť obojstranne na jeden list papiera

[2] 150-200 slov (1200-1700 znakov), ktoré opisujú výskumný problém v kontexte súčasného stavu vrátane motivácie a smerov riešenia

Literatúra[3]

- Brownlee, J., 2019. Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python. Machine Learning Mastery.
- Qi, C.R., Su, H., Mo, K. and Guibas, L.J., 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).

Vyššie je uvedený návrh diplomového projektu, ktorý vypracoval(a) Bc. Katarína Juhásová, konzultoval(a) a osvojil(a) si ho RNDr. Andrej Lúčny, PhD. a súhlasí, že bude takýto projekt viesť v prípade, že bude pridelený tomuto študentovi.

V Bratislave dňa 28.1.2021

Podpis študenta

Podpis výskumníka

Vyjadrenie garanta predmetov Diplomový projekt I, II, III

Návrh zadania schválený: áno / nie[4]

Dňa: .....

Podpis garanta predmetov

[3] 2 vedecké zdroje, každý v samostatnej rubrike a s údajmi zodpovedajúcimi bibliografickým odkazom podľa normy STN ISO 690, ktoré sa viažu k téme zadania a preukazujú výskumnú povahu problému a jeho aktuálnosť (uveďte všetky potrebné údaje na identifikáciu zdroja, pričom uprednostnite vedecké príspevky v časopisoch a medzinárodných konferenciách)

[4] Nehodiace sa prečiarknite

ČESTNÉ PREHLÁSENIE

Čestne vyhlasujem, že som túto prácu vypracovala samostatne, na základe konzultácií a s použitím uvedenej literatúry.

.....

V Bratislave, 3. január 2022 Bc. Katarína Juhásová

POĎAKOVANIE

Ďakujem môjmu školiteľovi RNDr. Andrejovi Lúčnemu, PhD. za ochotu, odbornú pomoc, užitočné pripomienky a cenné rady, ktoré mi poskytol pri vypracovaní diplomovej práce.

Anotácia

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Študijný program:	Inteligentné softvérové systémy
Autor:	Bc. Katarína Juhásová
Diplomová práca:	Registrácia objektov na mračne bodov pomocou hlbokých neurónových sietí
Vedúci diplomovej práce:	RNDr. Andrej Lúčny, PhD.

Január 2022

Využitie hlbokého učenia je vo sfére počítačového videnia veľmi rozšírené, prevažne však ide o spracovanie 2D alebo pravidelných 3D dát, zatiaľ čo komplexnejšie štruktúry ako mračná bodov sú používané zriedkavejšie. Hlavným dôvodom je ich nepravidelnosť, nesúmernosť a nekonzistentná hustota bodov, ktoré nie sú prirodzene utriedené. Vo väčšine prípadov sú spomenuté problémy riešené transformáciou dát na pravidelnú volumetrickú štruktúru ako voxely alebo na sekvenciu 2D snímok. Prelom v tejto oblasti priniesla hlboká neurónová sieť PointNet, ktorá umožňuje extrahovanie lokálnych a globálnych čŕt priamo z mračien bodov reprezentovaných neusporiadanými množinami bodov. V posledných rokoch sieť PointNet inšpirovala vývoj viacerých modelov hlbokého učenia na riešenie rôznych problémov vrátane registrácie objektov. Výnimočne úspešným modelom na registráciu objektov je CorsNet, ktorý používa zlúčené reprezentácie lokálnych a globálnych čŕt na určenie zhody medzi vstupnými mračnami bodov. Predložená priebežná správa o riešení DP2 obsahuje analýzu rôznych prístupov k registrácii objektov pomocou hlbokého učenia, pričom špeciálna pozornosť je venovaná spôsobu získavania reprezentácií čŕt modelom PointNet. Výskumná časť práce zahŕňa opis reimplementácie modelu CorsNet a výsledky experimentov na identifikáciu najvhodnejších metód na predspracovanie a augmentáciu dát. Získané poznatky budú v ďalšej fáze práce použité na registráciou objektov z vlastného datasetu písmen, číslic a ďalších znakov.

Annotation

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course: Intelligent Software Systems
Author: Katarína Juhásová
Master's Thesis: Object Registration in 3D Point Clouds Using Deep Neural Networks
Supervisor: Dr. Andrej Lúčny
2022, January

Deep learning has various applications in the computer vision domain, however, it is primarily used for processing of images or regular 3D data, while more complex structures, such as point clouds, are used less frequently. The main reason is their irregularity, asymmetry and inconsistent density of unordered points, which may cause issues during feature extraction. Therefore, these issues are in most cases solved by transformation of the data into regular volumetric structures, such as voxels or sequences of 2D images. The breakthrough in this area was made by the deep neural network PointNet, which performs local and global feature extraction directly on unstructured point clouds. Recently, PointNet has inspired the development of several deep learning methods addressing a variety of computer vision problems including object registration. An exceptionally successful object registration model is CorsNet, which concatenates global features with local features of the source point cloud to estimate the correspondences of each point. The presented DP2 report contains an analysis of various deep learning methods for object registration, with a separate chapter focused on PointNet feature extraction. The research part of the work includes a description of CorsNet reimplementation and the results of experiments designed to determine the most suitable methods for data preprocessing and augmentation. The findings will be used in the next phase of the work to train the implemented model on our own dataset of letters, numbers and other characters.

Obsah

Zoznam použitých skratiek

CNN	Convolutional Neural Network
FCN	Fully Convolutional Network
LiDAR	Light Detection and Ranging
MLP	Multilayer Perceptron
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
RPN	Region Proposal Network
SVD	Singular Value Decomposition

1 Úvod

Metódy hlbokého učenia v súčasnosti riešia množstvo problémov vo sfére počítačového videnia, a hoci mračná bodov sú v tejto sfére významnou dátovou štruktúrou, výskum ich spracovania hlbokým učením je stále v počiatkoch. Pri práci s mračnami bodov je potrebné brať do úvahy viaceré komplikácie spôsobené ich nepravidelnosťou, nesúmernosťou a nekonzistentnou hustotou bodov, bez jednoznačného daného utriedenia. Hlboké neurónové siete však umožňujú riešiť tieto problémy a následne vykonať rôzne úlohy nad mračnami bodov.

Vo veľkej časti existujúcich modelov hlbokého učenia sú problémy vyplývajúce zo štruktúry mračien bodov riešené transformáciou mračien na 2D obrázky alebo 3D voxely. Pri takýchto transformáciách však často dochádza k zbytočnému zväčšeniu objemu vstupných dát, pričom sa stráca aj určitá presnosť a detaily zachytené v pôvodných mračnách bodov. Alternatívnym prístupom je spracovanie mračien priamo, pričom však musí byť architektúra neurónovej siete prispôsobená na spracovanie takýchto dát a získanie vhodnej reprezentácie čŕt. Príkladom takejto hlbokej neurónovej siete je PointNet [1], ktorá dokáže úspešne extrahovať lokálne globálne črty pre mračná bodov reprezentované ako neusporiadané množiny bodov. Spôsob extrakcie čŕt použitý v modeli PointNet bol základom pre vývoj modelov hlbokého učenia na riešenie rôznych úloh vrátane registrácie objektov. Jedným z týchto modelov je CorsNet [2], ktorý bol spolu s modelom PointNet dôležitý pre výskumnú časť tejto práce.

Cieľom práce je vyriešiť problém registrácie objektov v mračnách bodov pomocou hlbokého učenia. Úloha bola riešená na datasete skenov pneumatík a vzorov na ich stranách. Pri registrácii objektov je známe, aké objekty by sa mali vo vstupných dátach nachádzať a úlohou neurónovej siete je ich nájsť. V prípade, že požadovaný objekt nie je nájdený, znamená to, že sa v mračne nenachádza, prípadne je výrazne zdeformovaný alebo nekompletný. Výsledná neurónová sieť by sa však mala vedieť vysporiadať s deformáciami spôsobenými zakrivením povrchu pneumatiky a mala by byť schopná nájsť aj týmto spôsobom zdeformované objekty.

2 Spracovanie mračien bodov hlbokým učením

Modely hlbokého učenia sú používané na aproximáciu funkcií, ktoré buď nie je možné priamo vyjadriť analyticky, alebo ich vyjadrenie je príliš náročné a neintuitívne. Hlboké neurónové siete je možné opísať ako viacvrstvové výpočtové modely využívajúce nelineárne prechody medzi vrstvami na extrahovanie čŕt zo vstupných dát. Medzi vrstvami platí, že vysokoúrovňové črty sú definované pomocou nízkoúrovňových, čím je tvorený hierarchický vzťah je označovaný ako hlboká architektúra [3]. Vstupné dáta sú pri prechode sieťou transformované na každej vrstve a výstupy z konkrétnej vrstvy sú kombináciou výstupov z predošlej. Týmto spôsobom je na každej vrstve získaná reprezentácia vstupu s rôznymi úrovňami abstrakcie, pričom na vrstvách hlbšie v sieti je miera tejto abstrakcie vyššia. Nad abstraktnými reprezentáciami vstupu je ďalej možné vykonávať rôzne úlohy v závislosti od požadovanej funkcionality siete, napríklad klasifikáciu, segmentáciu alebo detekciu objektov.

Pri riešení spomenutých úloh nad 3D mračnami bodov je potrebné sa vysporiadať s určitými problémami, ako nepravidelnosť dát alebo variabilná hustota bodov v mračnách bez jednoznačného spôsobu usporiadania. Často v závislosti od spôsobu, akým boli dáta získané, sú mračná bodov nepravidelné a miestami veľmi riedke, napríklad pri LiDAR skenoch z ulíc, alebo naopak príliš husté, čo je často prípad pri dátach z industriálneho prostredia [4]. V takej situácii môže hustota bodov skomplikovať identifikáciu hrán medzi segmentami pri segmentácii objektov. Metódy získavania dát vedia taktiež ovplyvniť prítomnosť šumu alebo iných nepresností a chýb spôsobených meraním. Podľa [5] existujú tri hlavné prístupy k riešeniu problému nepravidelnosti mračien bodov a ich spracovaniu pomocou hlbokého učenia:

1. transformácia mračien bodov na voxely alebo sekvenciu obrázkov a následné spracovanie pomocou CNN,
2. spracovanie mračien bodov priamo pomocou MLP,
3. spojením týchto dvoch metód [6, 7].

2.1 Spracovanie transformovaných mračien bodov

Prvým častým prístupom k spracovaniu mračien bodov je ich transformácia na pravidelné 3D voxely. VoxelNet [8] rieši problém detekcie objektov v priestore. Vstupom pre model síce sú mračná bodov, avšak v prvom kroku sú jednotlivé body z mračien rozdelené do 3D mriežky (voxelov), kde pre každý voxel je vypočítaná jeho vektorová reprezentácia opisujúca obsiahnutú priestorovú informáciu. Voxely sú teda reprezentované ako 4D tenzory, z ktorých je ďalej konvolučnými vrstvami získaná agregovaná priestorová informácia a následne pomocou plne konvolučnej RPN (angl. *Region Proposal Network*) vykonaná detekcia.

Podobným spôsobom sú transformované mračná bodov na voxely aj pre model na detekciu objektov Vote3Deep [9]. Každá bunka v 3D mriežke s nenulovým počtom priradených bodov je reprezentovaná vektorom čŕt obsahujúcim štatistiky z bodov patriacich do danej bunky. Okrem týchto štatistík obsahuje aj binárny indikátor prítomnosti jedného a viac bodov a aj pozíciu v 3D mriežke. Pre bunky s nulovým počtom bodov nie je uložená žiadna vektorová reprezentácia, čo má za následok riedku reprezentáciu vstupu.

Podobne aj v prácach [10, 11, 12, 13] sú mračná bodov najskôr transformované na voxely a tie sú ďalej spracované hlbokým učením. V týchto prácach umožnila transformácia mračien bodov na 3D voxely využitie konvolučných neurónových sietí (angl. *Convolutional Neural Network, CNN*) [14]. V mnohých prípadoch však pri tejto transformácii došlo k zbytočnému nafúknutiu objemu vstupných dát, pričom bola stratená určitá presnosť zachytená v pôvodných mračnách bodov.

Alternatívnou možnosťou reprezentácie mračien bodov pomocou štruktúry s fixnou veľkosťou je transformácia na 2D obrázky. Výhodami tohto prístupu je, že môžu čerpať poznatky z oblasti spracovania obrazu, ktorá je výrazne hlbšie preskúmaná na rozdiel od spracovania mračien bodov. Ďalšou výhodou je, že spracovanie 2D dát je menej výpočtovo náročné v porovnaní s 3D dátami, čo je často významná vlastnosť hlavne pri využití v autonómnych vozidlách. Pang a kolektív [4] využili tieto vlastnosti a predstavili CNN na detekciu objektov v mračnách bodov. 3D problém detekcie objektov transformovali na 2D problém tak, že z pôvodného 3D vstupu bolo vytvorených niekoľko hĺbkových máp (angl. *Depth Map*) z rôznych uhlov pohľadu a detekcia bola vykonaná nad každou mapou

zvlášť. Výstupy z jednotlivých hĺbkových máp sú na záver zlúčené na určenie *bounding boxu* v pôvodnom 3D priestore. Transformácie špecificky na hĺbkové mapy sú použité v práci [15], v ktorej bola použitá CNN na klasifikáciou znakov posunkovej reči.

Transformáciu objektu z mračna bodov na 2D snímky s viacerých uhlov pohľadu využíva aj hlboká neurónová sieť na rozpoznanie a klasifikáciu objektov od Su a kolektív [16]. V rámci modelu sú najskôr pomocou paralelných CNN extrahované črty pre každú snímku samostatne. Následne sú tieto črty agregované do globálnej reprezentácie objektu, nad ktorou je vykonaná finálna klasifikácia.

Vstupné dáta sú transformované na obrázky aj pre Complex-YOLO [5], ktorý umožňuje detekciu 3D objektov. Architektúra tohto modelu vychádza zo *state-of-the-art* konvolučnej neurónovej siete (CNN) na detekciu 2D objektov YOLOv2 [17] v reálnom čase. Vstupné dáta pre Complex-YOLO sú získavané LiDAR laserovými skenermi a pred spracovaním sieťou sú transformované na RGB snímky z vtáčej perspektívy, pričom jednotlivé farebné kanály sú mapované na výšku, hustotu a intenzitu vráteného laserového signálu. Detekcia je vykonávaná nad jedným takýmto obrázkom, čo prispelo k efektivite modelu a spracovaniu vstupov v reálnom čase.

Rovnakým spôsobom sú mračná bodov transformované aj pre plne konvolučnú sieť (angl. *Fully Convolutional Network, FCN*) PIXOR [18], ktorá tiež rieši úlohu detekcie objektov na 3D dátach transformovaných do snímky z vtáčej perspektívy. Vylepšením oproti Complex-YOLO je, že výsledné *bounding boxy* určujú nie len pozíciu objektov, ale aj smer ich natočenia.

2.2 Spracovanie mračien bodov priamo

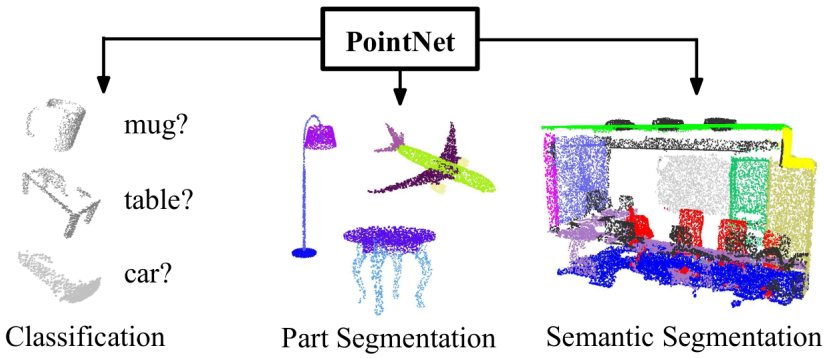
Príkladom modelu pracujúceho priamo s mračnami bodov PointNet++ [1]. Tento hierarchický model rekurzívne delí množinu bodov na menšie podskupiny a z nich získava črty s rôznymi úrovňami abstrakcie. Na každej úrovni je z podmnožiny bodov vytvorená nová skupina obsahujúca menší počet elementov. Extrahovanie každej úrovne abstrakcie sa skladá z troch typov vrstiev: *vzorkovacia vrstva*, *zgrupovacia vrstva* a *PointNet vrstva* inšpirovaná nižšie opísaným modelom PointNet [1]. Nad získanými komplexnými reprezentáciami mračien bodov je následne možné vykonať klasifikáciu alebo segmentáciu. Spôsob získavania reprezentácie čŕt z PointNet++

bol použitý aj v modeli na detekciu objektov opísanom v práci [19].

Ďalším modelom, ktorý umožňuje vykonávať rôzne úlohy priamo nad mračnami bodov je PointNet [1]. Jadrom jeho architektúry sú viacvrstvové perceptróny (MLP) a hoci tento model nepodporuje priamo registráciu objektov, slúžil ako základ na extrakciu čŕt pre viaceré modely vykonávajúce registráciu objektov. Model PointNet je podrobnejšie opísaný v ďalšej kapitole, keďže bol kľúčový aj pri návrhu nášho riešenia pre registráciu objektov.

3 PointNet

PointNet [1] je hlboká neurónová sieť, ktorá umožňuje priamo spracovať mračná bodov. Nad nimi dokáže vykonať úlohy ako klasifikácia 3D objektov, segmentácia častí 3D objektov alebo sémantická segmentácia scény. Príklady výstupov siete je možné vidieť na obrázku 3.1. Architektúra tejto siete bola navrhnutá tak, aby umožnila spracovanie mračien bodov bez transformácie na pravidelnú dátovú štruktúru vo fáze predspracovania. Model PointNet sa zároveň dokáže úspešne vysporiadať s chybami a šumom vo vstupných dátach, podobne ako aj s riedkymi mračnami.



Obr. 3.1: *PointNet funkcionalita: klasifikácia, segmentácia častí objektov a sémantická segmentácia scény [1].*

3.1 Vstupné dáta

Vstupné dáta pre PointNet sú reprezentované ako množina bodov, v ktorej každý bod pozostáva z $[x, y, z]$ súradníc a ďalších čŕt ako farba alebo normála. Táto množina má určité vlastnosti, ktoré bolo potrebné zohľadniť v architektúre siete:

- *množina je neusporiadaná*: pri N počte bodov je potrebné, aby neurónová sieť bola invariantná voči $N!$ permutáciám usporiadaní bodov v rámci množiny.
- *medzi bodmi v množine existujú vzťahy*: body sa nachádzajú v Euklidovskom priestore, teda nie sú izolované, vďaka čomu je možné určiť ich vzájomné

vzdialenosti, pričom blízke body vytvárajú zo sémantického hľadiska podmnožiny. Z toho dôvodu je model navrhnutý tak, aby vedel lokálne zachytiť menšie štruktúry a ich vzájomné vzťahy.

- *množiny bodov sú invariantné vzhľadom na určité transformácie*: transformácie ako rotácia, zrkadlenie alebo škálovanie neovplyvňujú výstupy siete pri klasifikácii ani segmentácii objektov.

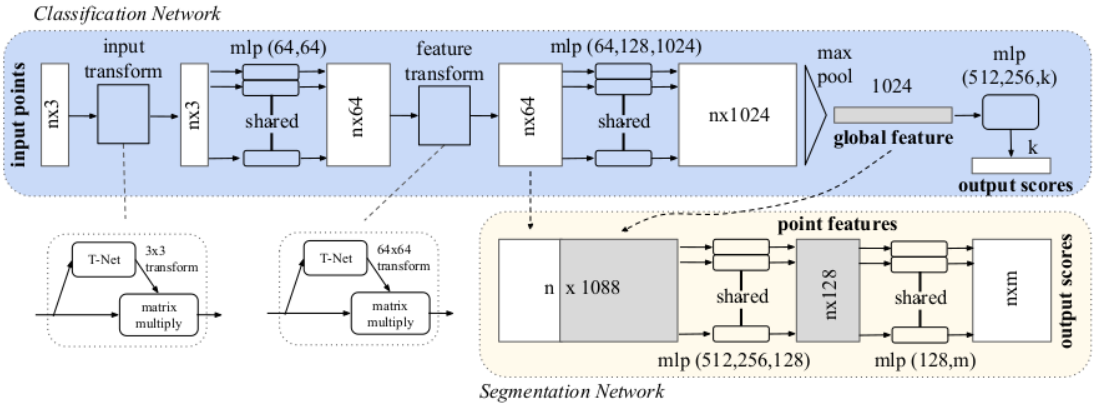
3.2 Architektúra PointNet

Hlboké neurónové siete sú založené na učení reprezentácií. Z toho dôvodu je ich dôležitou vlastnosťou, aby vytvorené reprezentácie boli podobné pre príbuzné objekty, napríklad objekty patriace do jednej triedy, a zároveň aby sa reprezentácie rôznych objektov, teda z rôznych tried, odlišovali. Pri vstupoch s presne definovanou štruktúrou a poradím vstupných elementov, ako sú obrázky, je zachovanie tejto vlastnosti jednoduchšie v porovnaní s neusporiadanými množinami bodov. V tomto prípade je extrahovanie čŕt (reprezentácií) problematickejšie, keďže model sa nemôže spoliehať na vzťahy medzi susediacimi bodmi v množine.

Pre správne fungovanie modelu PointNet bolo potrebné vytvoriť architektúru, ktorá umožní extrahovať črty a identifikovať vzťahy medzi bodmi susediacimi v Euklidovskom priestore, pričom tieto body nemusia byť aj vo vstupnej množine susedné. Navrhnutá architektúra by mala zároveň umožniť identifikáciu čŕt zo vstupu takým spôsobom, aby aj dáta získané skenovaním rovnakého objektu z rôznych uhlov pohľadu mali takmer zhodné reprezentácie. Architektúru je tiež potrebné prispôsobiť úlohe, ktorú má model riešiť. Na klasifikáciu objektov je potrebná globálna reprezentácia objektu, zatiaľ čo pre segmentáciu častí objektov alebo scény sú potrebné nielen globálne informácie ale aj lokálne. Kompletná architektúra siete PointNet je znázornená na obrázku 3.2.

Architektúra neurónovej siete PointNet je založená na 3 kľúčových častiach:

1. aproximácia symetrickej funkcie,
2. agregácia reprezentácie lokálnych a globálnych čŕt,
3. vnorené siete na zarovnanie vstupu a extrahovaných čŕt.



Obr. 3.2: *PointNet architektúra pre klasifikáciu (Classification Network) a segmentáciu (Segmentation Network). Vstupná množina n bodov je transformovaná, následne sú získané lokálne črty pre body, ktoré sú opäť transformované a použité na získanie globálnych čŕt. Tieto črty agregované pomocou max pooling sú spracované s MLP na získanie skóre pre klasifikáciu do každej z k tried. V rozšírení siete pre segmentáciu sú zlúčené lokálne a globálne črty, ktorých spracovaním je získané skóre pre každú z m možných kategórií pre všetky vstupné body. Čísla v zátvorkách reprezentujú veľkosti vrstiev v MLP [1].*

3.2.1 Aproximácia symetrickej funkcie

Keďže vstupom pre model je neusporiadaná množine bodov, pri N počte bodov musí byť model invariantný voči $N!$ permutáciám vstupu. Vhodným riešením je práve použitie alebo aproximácia symetrickej funkcie. Z definície symetrickej funkcie vyplýva, že pri n vstupných parametroch $x_1, x_2, ..., x_n$ je výstup funkcie rovnaký pre všetky permutácie parametrov.

Cieľom pri návrhu siete bolo, aby neurónová sieť vedela aproximovať funkciu f aplikovaním symetrickej funkcie na transformované body z množiny:

$$f(x_1, ..., x_n) \approx g(h(x_1), ..., h(x_n)), \tag{3.1}$$

kde $f : 2^{\mathbb{R}^N} \to \mathbb{R}$, $h : \mathbb{R}^N \to \mathbb{R}^K$ a $g : \underbrace{\mathbb{R}^K \times ... \times \mathbb{R}^K}_{n} \to \mathbb{R}$ je symetrická funkcia. Funkcia h je aproximovaná pomocou viacvrstvového perceptrónu (angl. *Multilayer Perceptron, MLP*) [20] a pre g je použitá kombinácia symetrickej funkcie max pooling a funkcie s jednou premennou h. Aplikovaním rôznych funkcií h je možné extrahovať rôzne črty zo vstupu.

Model PointNet obsahuje niekoľko viacvrstvových perceptrónov (MLP), ktoré tvoria jadro architektúry. MLP sa skladajú zo vstupnej vrstvy, výstupnej vrstvy a jednej alebo viac skrytých vrstiev neurónov medzi nimi. Vrstvy MLP sú plne prepojené, čiže výstup každého neurónu z vrstvy L_n prispieva do vstupov každého z neurónov na vrstve L_{n+1}. Výstupom neurónu je lineárna kombinácia niekoľkých vstupných reálnych čísel, ktorú je možné modifikovať pomocou nelineárnej aktivačnej funkcie.

3.2.2 Agregácia reprezentácie lokálnych a globálnych čŕt

V časti vyššie je opísané, ako získať pomocou K rôznych funkcií reprezentáciu globálnych čŕt pre vstupné dáta ako vektor $[f_1, ..., f_K]$. Takáto reprezentácia je dostačujúca pre klasifikáciu, avšak pre segmentáciu objektov aj scény je potrebná kombinácia lokálnych a globálnych čŕt, aby bolo možné zachytiť aj informácie o blízkych bodoch vytvárajúcich sémantické podmnožiny. Na obrázku 3.2 je možné vidieť ako je klasifikačná časť siete rozšírená pre segmentáciu objektov. Lokálne a globálne črty získané v rámci pôvodnej klasifikačnej časti sú pre účely segmentácie zlúčené. Následne sú identifikované vlastnosti pre body na základe zlúčených čŕt.

3.2.3 Vnorené siete na zarovnanie vstupu a extrahovaných čŕt

Predchádzajúce dva princípy sú založené na extrakcii viacúrovňových čŕt zo vstupných dát. Pritom však nebol vyriešený problém, že tieto črty by mali byť rovnaké aj keď objekt opísaný bodmi podstúpi rôzne geometrické transformácie ako rotáciu alebo zrkadlenie. Z toho dôvodu sú v modeli PointNet obsiahnuté dve „mini“ siete použité na zarovnanie vstupných bodov a lokálnych čŕt.

Cieľom týchto transformačných sietí je aproximovať maticu pre afínnu transformáciu samostatne pre vstup a neskôr aj pre jeho lokálne vlastnosti. Na obrázku 3.2 sú tieto siete označené ako *T-Net*. T-net časti modelu sa skladajú z konvolučných vrstiev, pooling vrstiev a plne prepojených vrstiev, čím pripomínajú klasické konvolučné neurónové siete (angl. *Convolutional Neural Network, CNN*). CNN sa vyznačujú vlastnosťou, že do aktivácie vypočítanej z konkrétneho bodu neprispieva iba bod sám, ale aj jeho okolie. Týmto spôsobom je možné extrahovať črty z blízkeho okolia bodu na skorších vrstvách a komplexnejšie globálne črty na neskorších

vrstvách.

Na konvolučných vrstvách je výstup počítaný zo vstupu pomocou konvolúcie. Pri konvolúcii je aplikovaný posuvný filter alebo kernel (matica veľkosti $n \times m$) postupne na všetky súvislé podmnožiny veľkosti $n \times m$ zo vstupu. Počet filtrov aplikovaných na jednotlivých vrstvách predstavuje počet neurónov na vrstve a určujú počet dimenzií výstupu. Dôležitou vlastnosťou týchto vrstiev je, že pri procese konvolúcie je rovnaký filter posúvaný po celom vstupe, a teda sa snaží získať rovnakú informáciu zo všetkých častí vstupu. Vďaka tomu je možné identifikovať črty bez ohľadu na to, či sú reprezentované na začiatku alebo na konci vstupu. Veľkosť okolia, z ktorého majú byť črty získané pre konkrétny bod je možné regulovať veľkosťou kernelu. Napríklad pri veľkosti kernelu 3×3 je do výpočtu aktivácie pre konkrétny bod zahrnutá hodnota bodu samotného spolu s jeho ôsmimi susediacimi bodmi. V modeli PointNet sú použité konvolúcie 1×1.

Využitie 1×1 konvolúcie v neurónovej sieti PointNet zabezpečuje extrakciu čŕt samostatne pre jednotlivé body. Po spracovaní vstupu konvolučnými vrstvami je každý bod reprezentovaný ako vektor hodnôt (čŕt), pričom tieto hodnoty boli pre všetky body vypočítané rovnakými kernelmi. Na takúto reprezentáciu vstupu je aplikovaná symetrická funkcia max pooling, vďaka ktorej sú do ďalších vrstiev posunuté vždy rovnaké hodnoty bez ohľadu na zmenu poradia bodov v pôvodnom vstupe. Výstup z pooling vrstvy je ďalej spracovaný plne prepojenými vrstvami a následne použitý na získanie afínnej transformačnej matice veľkosti 3×3, ktorá je v hlavnom modeli použitá priamo na zarovnanie vstupu.

Podobným spôsobom je získaná aj matica na zarovnanie lokálnych čŕt. Transformačná sieť pre lokálne črty je komplexnejšia v porovnaní s tou pre vstupné dáta a transformačná matica má v tomto prípade veľkosť 64×64, čo výrazne zvýšilo výpočtovú náročnosť optimalizácie siete. Z toho dôvodu bola pridaná regularizácia pre výpočet stratovej funkcie (angl. *loss function*):

$$L_{reg} = ||I - AA^T||_F^2, \tag{3.2}$$

kde A je transformačná matica predikovaná „mini“ sieťou *T-Net*. Nastavené obmedzovanie zaručilo, že výsledná transformačná matica pre lokálne črty bude blízka ortogonálnej matici. Ortogonálne matice sú z definície štvorcové matice, ktorých

riadky a stĺpce sú ortogonálne (pravouhlé) vektory. Pre takéto matice platí vzťah:

$$Q^T Q = Q Q^T = I, \tag{3.3}$$

kde Q je pôvodná matica, Q^T je transponovaná matica a I je matica identity. Ortogonálna transformačná matica zachováva skalárny súčin vektorov, vďaka čomu umožňuje vykonanie unitárnych transformácií ako rotácia alebo zrkadlenie.

3.3 Klasifikácia a segmentácia pomocou PointNet

Model PointNet umožňuje klasifikáciu 3D objektov, segmentáciu častí 3D objektov a sémantickú segmentáciu scény. Z architektúry siete na obrázku 3.2 vidno, že architektúra modelu sa líši v závislosti od riešenej úlohy, kde segmentačná sieť predstavuje nadstavbu nad pôvodným čisto klasifikačným modelom.

Štruktúra vstupných dát pre všetky tri úlohy je rovnaká, avšak zatiaľ čo pre klasifikáciu a segmentáciu častí objektu reprezentujú vstupné mračná bodov jeden objekt, pre segmentáciu scény je zachytená v mračnách snímka časti scény. Výstupy siete za líšia v závislosti od úlohy, ktorá má byť riešená. Výstupom klasifikácie objektu je jeden vektor pre celý vstup obsahujúci skóre pre každú z k možných tried. Pre úlohy segmentácie s n množstvom bodov v mračne a m možnými sémantickými kategóriami je výstupom modelu matica veľkosti $n \times m$ obsahujúca skóre pre každú z m možných kategórií pre všetky body.

Pri klasifikácii objektov sú vstupné body najskôr transformované afínnou transformačnou maticou získanou z prvej *T-Net* siete. Zo zarovnaných transformovaných dát sú následne získané lokálne črty. Na obrázku architektúry 3.2 je táto časť siete označaná ako dva viacvrstvové perceptróny s veľkosťou 64 a zdieľanými váhami medzi neurónmi v rámci vrstiev. Formálne sú tieto vrstvy implementované ako 2D konvolučné vrstvy s kernelom veľkosti 1×1, keďže práve tie umožňujú spracovanie každého bodu samostatne rovnakými váhami. Takýmto spôsobom získané lokálne črty sú zarovnané transformačnou maticou z druhej *T-Net* „mini“ siete. Transformované črty sú opäť spracované pomocou vrstiev označených ako MLP so zdieľanými váhami, čím sú extrahované globálne črty, ktoré sú ďalej agregované pomocou max pooling funkcie. Posledná časť klasifikačnej siete je tvorená tromi

MLP, ktoré z globálnych čŕt vstupu určia výstup klasifikácie.

Neurónová sieť na segmentáciu je rozšírením pôvodnej klasifikačnej siete. Na začiatku tejto nadstavby sú zlúčené lokálne a globálne črty získané v rámci klasifikácie. Následne sú použité vrstvy so zdieľanými váhami na extrakciu komplexných čŕt pre body. V závere je vykonaná segmentácia opäť pomocou vrstiev so zdieľanými váhami, po ktorej je možné jednotlivým bodom priradiť sémantickú kategóriu, do ktorej patria.

4 Registrácia objektov v mračnách bodov

Pri registrácii objektov je známe, aké objekty by sa mali vo vstupných dátach nachádzať a úlohou neurónovej siete je ich nájsť. Vstupom pre sieť sú hľadaný vzor a mračno bodov, v ktorom má byť daný vzor nájdený. Pri úspešnej registrácii je možné pomocou afínnej transformácie zarovnať mračno vzoru na vstupné mračno a identifikovať tak hľadaný vzor v dátach.

Opäť aj pre registráciu objektov existujú riešenia, ktoré transformujú riešený problém do 2D priestoru. Príkladom je model z práce [21] vykonávajúci registráciu objektov na základe porovnávania lokálnych 2D čŕt. Medzi ďalšie práce zamerané na registráciu objektov v mračnách bodov patria [22, 23, 24, 25, 26] a stručný prehľad o existujúcich metódach spracovali Zhang a kolektív [27].

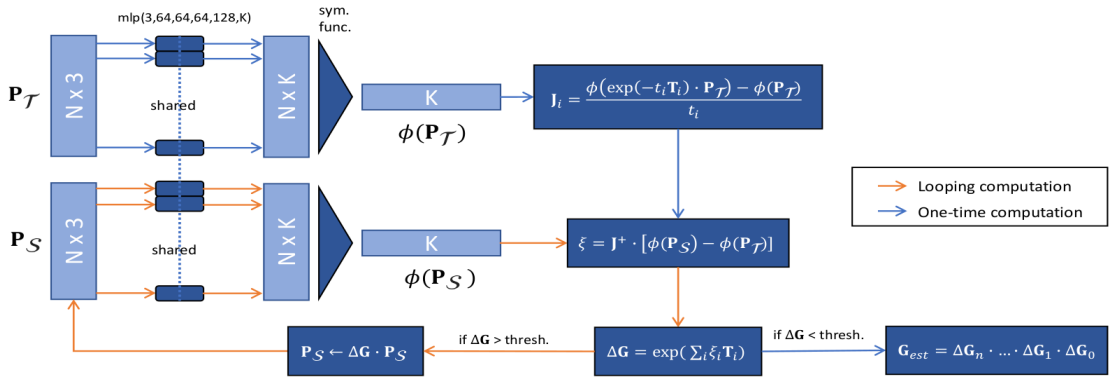
Problematika registrácie objektov v mračnách bodov je výrazne menej riešená v porovnaní s klasifikáciou alebo detekciou objektov v mračnách. V tejto kapitole je opísaných niekoľko existujúcich metód hlbokého učenia na registráciu objektov, pričom špeciálny dôraz je kladený na model CorsNet [2], ktorý je dôležitý pre výskumnú časť práce.

4.1 PointNetLK

Výpočtovo nenáročným modelom na registráciu objektov je PointNetLK [28]. Architektúra tohto modelu je znázornená na obrázku 4.1. PointNetLK využíva na získanie reprezentácie čŕt prístup založený na poznatkoch z PointNet a na registráciu je použitý modifikovaný Lucas-Kanade (LK) algoritmus implementovaný ako rekurentná neurónová sieť.

Lucas-Kanade algoritmus [29] slúži na určenie vektoru posunutia (u, v) porovnaním dvoch obrázkov zachytávajúcich scénu alebo objekt v rôznom čase, pričom sa predpokladá, že časový rozdiel Δt medzi obrázkami je malý. Tento vektor posunutia je vypočítaný pre vybrané významné body obrázku. LK algoritmus využíva okolie významných bodov na určenie ich vektoru posunutia, čo však má za následok komplikácie v prípade, že gradient v okolí je blízky nule, prípadne že bod sa nachádza

na hrane a gradient je výrazne väčší v jednom smere. Pri sieti PointNetLK by však táto vlastnosť nemala spôsobovať problémy, keďže namiesto obrázkov sú použité 2D vektorové reprezentácie mračien bodov, v ktorých je výskyt takéhoto správania málo pravdepodobný.

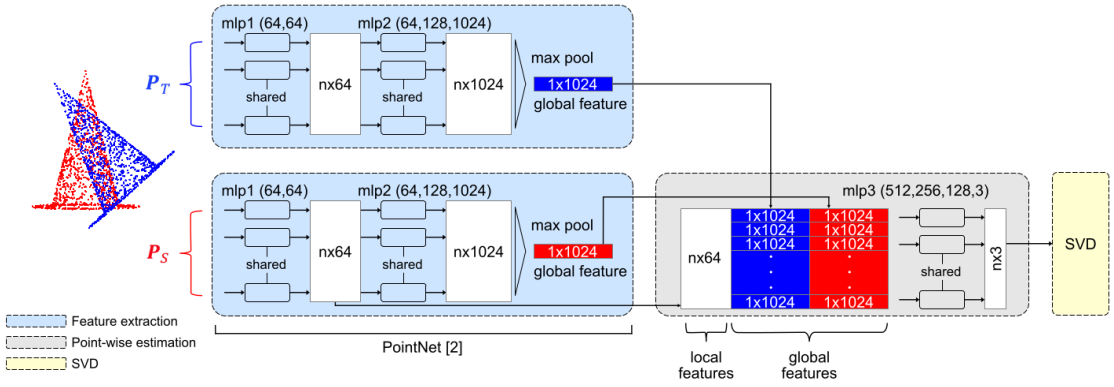


Obr. 4.1: *PointNetLK architektúra. Z mračien bodov pre šablónu (template)* $\mathbf{P}_\mathcal{T}$ *a vstupný objekt (source)* $\mathbf{P}_\mathcal{S}$ *sú extrahované vektorové reprezentácie globálnych čŕt* $\phi(\mathbf{P}_\mathcal{T})$ *a* $\phi(\mathbf{P}_\mathcal{S})$*. Jacobiho matica je pre vzor vypočítaná iba raz a ďalej je používaná pri výpočte parametrov otočenia. Pozícia pôvodného vstupného objektu* $\mathbf{P}_\mathcal{S}$ *je následne iteratívne upravovaná, pričom prebieha aj prepočet jeho vektorovej reprezentácie [28].*

4.2 CorsNet

Lepšie výsledky ako PointNetLK dosahuje end-to-end hlboká neurónová sieť CorsNet [2]. Na rozdiel od modelu PointNetLK, ktorý pracuje iba s globálnou reprezentáciou čŕt, tento model používa zlúčenú reprezentáciu lokálnych a globálnych čŕt. Vstupom pre CorsNet sú dve mračná bodov reprezentujúce vstupný objekt a šablónu, pričom úlohou modelu je predikovať transformačnú maticu umožňujúcu zarovnanie vstupného objektu na šablónu. Architektúra siete CorsNet (obr. 4.2) pozostáva z troch častí:

- získanie globálnych čŕt z oboch vstupných mračien bodov,
- zlúčenie reprezentácií globálnych čŕt s lokálnymi črtami vstupného objektu a odhad súhlasnosti extrahovaných čŕt,
- určenie transformačnej matice pomocou singulárneho rozkladu (angl. *Singular Value Decomposition, SVD*).



Obr. 4.2: *CorsNet architektúra. Mračná bodov pre vzor (template) a vstupný objekt (source) sú označené ako* $\mathbf{P}_T$ *a* $\mathbf{P}_S$*. Architektúra je rozdelená na 3 segmenty: získanie globálnych čŕt (Feature extraction), odhad súhlasnosti (Point-wise estimation) a metódy singulárneho rozkladu (SVD) [2].*

Extrakcia globálnych čŕt. Mračná bodov pre vstupný objekt ($\mathbf{P}_S$) a šablónu ($\mathbf{P}_T$) majú rovnakú veľkosť a sú reprezentované maticami o veľkosti $n \times 3$, kde n predstavuje počet bodov opisujúcich objekt a každý bod je daný trojicou $[x, y, z]$ súradníc. Na získanie lokálnych a globálnych čŕt z takto reprezentovaných mračien bodov sú použité dve PointNet siete. Kompletná PointNet architektúra na klasifikáciu objektov je opísaná v kapitole 3.2, avšak pri registrácii objektov je potrebná iba časť na extrakciu čŕt končiaca *max pooling* vrstvou.

Odhad súhlasnosti extrahovaných čŕt. Prvá vrstva v druhej časti siete slúži na zlúčenie reprezentácií oboch vstupných objektov. Lokálne črty pre každý bod zo vstupného objektu sú zlúčené s globálnou reprezentáciou šablóny aj vstupného objektu. Následne je použitá sekvencia MLP na odhadnutie posunutia medzi šablónou a zdrojovým objektom. Výstupná matica z posledného MLP má opäť veľkosť $n \times 3$ a ak ju označíme ako $\Delta\mathbf{P}_S$, mal by platiť vzťah

$$\hat{\mathbf{P}_T} = \mathbf{P}_S + \Delta\mathbf{P}_S, \tag{4.1}$$

kde $\hat{\mathbf{P}_T}$ je predikovaná transformácia zdrojového objektu na šablónu. Výstup druhej časti siete $\Delta\mathbf{P}_S$ však nie je žiadnym spôsobom obmedzený a transformovaný objekt $\hat{\mathbf{P}_T}$ môže byť zdeformovaný a úlohou modelu nemá byť nájdenie ľubovoľného posunutia pre každý bod zvlášť tak, aby bol v konečnom dôsledku každý bod

vstupného objektu namapovaný na šablónu, ale aby celý objekt ako taký bol premietnutý na šablónu, pričom nemusí platiť, že každý bod z $\mathbf{P}_S$ sa premietne na špecifický bod v $\mathbf{P}_T$. Z toho dôvodu je predikovaná transformácia $\hat{\mathbf{P}_T}$ použitá na určenie rigidnej transformačnej matice $\mathbf{G} \in SE(3)$, pre ktorú by malo platiť

$$\mathbf{G} \cdot \mathbf{P}_S = \hat{\mathbf{P}_T}. \tag{4.2}$$

SVD. Výpočet predikovanej transformačnej matice $\mathbf{G}_{est}$ je v poslednej časti siete vykonaný pomocou SVD. Ako prvé sú určené centroidy pre $\mathbf{P}_S$ a $\hat{\mathbf{P}_T}$ ako

$$\overline{\mathbf{P}_S} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{P}_S \quad \text{a} \quad \overline{\hat{\mathbf{P}_T}} = \frac{1}{n}\sum_{i=1}^{n}\hat{\mathbf{P}_T}. \tag{4.3}$$

Kovariančná matica $\mathbf{H}$ daná vzťahom

$$\mathbf{H} = \sum_{i=1}^{n}(\hat{\mathbf{P}_T} - \overline{\hat{\mathbf{P}_T}})(\mathbf{P}_S - \overline{\mathbf{P}_S})^T \tag{4.4}$$

je následne rozložená SVD algoritmom na získanie $\mathbf{U}, \mathbf{V} \in SO(3)$:

$$[\mathbf{U}, \mathbf{S}, \mathbf{V}] = SVD(\mathbf{H}). \tag{4.5}$$

Predikovaná rotácia $\mathbf{R}_{est} \in SO(3)$ a translácia $\mathbf{t}_{est} \in \mathbb{R}^3$ sú definované vzťahmi

$$\mathbf{R}_{est} = \mathbf{V}\mathbf{U}^T \quad \text{a} \quad \mathbf{t}_{est} = -\mathbf{R}_{est} \cdot \overline{\hat{\mathbf{P}_T}} + \overline{\mathbf{P}_S}, \tag{4.6}$$

pričom kompletná predikovaná rigidná transformácia $\mathbf{G}_{est} \in SO(3)$ je daná ako

$$\mathbf{G}_{est} = \begin{pmatrix} \mathbf{R}_{est} & \mathbf{t}_{est} \\ 0 & 1 \end{pmatrix}. \tag{4.7}$$

Hoci na výpočet transformačnej matice pomocou SVD sú použité čisto lineárne operácie bez prítomnosti trénovateľných parametrov, pre ucelenosť riešenia je aj tento výpočet implementovaný ako súčasť end-to-end neurónovej siete CorsNet.

5 Návrh riešenia

Cieľom práce je preskúmať možnosti použitia registrácie 3D objektov pomocou neurónových sietí pre využitie v gumárenskom priemysle. Riešenou úlohou je registrácia objektov z konkrétneho datasetu obsahujúceho skeny pneumatík, na ktorých hľadáme vzorové písmená, čísla a ďalšie znaky. Vstupom pre neurónovú sieť sú dve mračná bodov reprezentujúce vstupný objekt a vzor, pričom úlohou siete je predikovať transformáciu umožňujúcu zarovnanie vstupného objektu na vzor. V prechádzajúcej kapitole boli opísané existujúce modely na registráciu 3D objektov a ich postup je možné zovšeobecniť do troch krokov:

1. získanie reprezentácií oboch vstupných objektov pomocou modelu PointNet,
2. spracovanie zlúčených reprezentácií objektov plne prepojenými vrstvami,
3. transformácia výstupu poslednej vrstvy do požadovaného formátu.

Opísaný postup viedol vo viacerých prípadoch k úspešným výsledkom a z toho dôvodu bol použitý ako základ aj pri návrhu nášho riešenia. Navrhnutá architektúra vychádza z modelu CorsNet. Kľúčovou časťou tejto práce je otestovanie správania modelu CorsNet na našich dátach a následná modifikácia architektúry v závislosti od priebežných zistení a výsledkov dosiahnutých pre rôzne úrovne komplexnosti transformácií. Nakoľko však neexistuje verejne dostupná implementácia, v prvom kroku bolo potrebné reimplementovať a natrénovať tento model. S takto vytýčeným jadrom riešenia sme plán práce na výskumnej časti rozdelili na nasledovné úlohy:

1. Reimplementácia modelu PointNet.
2. Reimplementácia modelu CorsNet.
3. Predspracovanie datasetu ModelNet40 pre CorsNet.
4. Natrénovanie a evaluácia modelov na pôvodných dátach.
5. Predspracovanie našich dát na formát podobný pôvodným dátam.
6. Vývoj modelu CorsNet na našom datasete.
7. Iteratívne zvyšovanie komplexnosti datasetu a vylepšenie modelu.
8. Vyhodnotenie chyby medzi vzorom a naskenovanými dátami.

CorsNet reimplementácia a trénovanie. Prvé štyri body zahŕňajú reimplementáciu modelov na základe dostupných zdrojov a ich natrénovanie na pôvodnom datasete ModelNet40. Architektúry oboch modelov sú opísané v kapitolách 3.2 a 4.2. Implementovaním rovnakého predspracovania dát a použitím rovnakých nastavení hyperparametrov ako bolo opísané v článkoch [2, 28] sme sa snažili dosiahnuť podobné výsledky a overiť tak fungovanie modelu a správnosť našej implementácie. Modely boli trénované na pôvodných dátach, čím sme chceli overiť do akej miery je model schopný vykonávať registráciu a nad akými vstupnými dátami a transformáciami. Metódy predspracovania a augmentácie dát prispievajúce k presnosti predikcií boli v ďalším krokoch použité aj pri práci s našim datasetom.

Predspracovanie dát. Po úspešnej registrácii objektov z datasetu ModelNet40 nasledovalo predspracovanie našich dát a pre ich ďalšie využitie na trénovanie vlastného modelu. V tejto fáze boli dáta predspracované takým spôsobom, aby reprezentácie objektov aj výstupné transformácie (translácia a rotácia) čo najvernejšie kopírovali charakter pôvodných trénovacích dát. Vstupný objekt a vzor boli reprezentované množinou 3D bodov, zatiaľ čo výstupom bola transformačná matica umožňujúca premietnutie vstupného objektu na vzor.

Vývoj modelu CorsNet na našom datasete. Keďže nami vygenerovaný dataset sa výrazne nelíšil od pôvodného datasetu ModelNet40, v ďalšom kroku bolo cieľom preskúmať, či je implementovaný model schopný úspešne vykonať registráciu aj nad našimi dátami. V závislosti od výsledkov trénovania boli vykonané malé zmeny v architektúre a optimalizácia hyperparametrov.

Iteratívne zvyšovanie komplexnosti datasetu a vylepšenie modelu. Po úspešnej registrácii jednotlivých písmen a znakov boli na vstupné mračná bodov aplikované komplexnejšie transformácie a mierne deformácie. Nakoľko pôvodnou myšlienkou za vznikom tejto diplomovej práce bola možnosť registrovať vzory na zakrivenom povrchu pneumatiky, podobnú situáciu sme chceli simulovať aj vo vstupných dátach. Okrem rozšírenia transformácií o deformácie boli zvýšené komplexnosti samotných objektov. Namiesto vstupných objektov tvorených jednotlivými znakmi boli vo fáze predspracovania generované objekty ako dvojice alebo trojice písmen a neskôr aj celé slová. Komplexnosť transformácií aj objektov bola zvyšovaná postupne a v závislosti od priebežne dosahovaných výsledkov boli vykonávané potrebné zmeny v architektúre siete.

Ďalším spôsobom na zvýšenie komplexnosti objektov pri predspracovaní bolo odstránenie častí objektu, napríklad odrezanie nožičky písmena *A* alebo vynechanie časti písmena *O*, pričom však vstupné mračná bodov mali stále rovnakú veľkosť zodpovedajúcu veľkosti vstupu siete. V praxi sa stáva, že vzor vytlačený na pneumatike nie je dokonalý a týmto spôsobom boli simulované možné perturbácie vyskytujúce sa v reálnych naskenovaných mračnách. Cieľom tejto fázy práce bolo otestovať schopnosť siete vykonať registráciu aj nad chybnými dátami a pozorovať, s akou presnosťou dokáže zarovnať objekty s chýbajúcimi časťami na kompletné vzory.

Vyhodnotenie chyby medzi šablónou a naskenovanými dátami. V rámci postprocessingu bola implementovaná funkcia na porovnanie vstupného objektu a vzoru a určenie ich percentuálnej zhody. Porovnanie nie je možné vykonať priamo nad vzorom a transformovaným vstupným objektom, nakoľko oba objekty sú opísané bodmi v 3D priestore, ktoré neumožňujú priamy výpočet objemu alebo percentuálnej miery prekryvu. Riešením tohto problému bolo vytvorenie polygónovej siete (angl. *mesh*) z bodov, následné vypočítanie objemu prieniku vzniknutých dvoch objektov a určenie pomeru tohto objemu voči celkovému objemu vzoru. V prípade, kedy je vstupný objekt reprezentovaný mračnom bodov bez chýbajúcich časti a po transformácii je presne zarovnaný na vzor, vypočítaný pomer by mal byť blízky 100%. Ak bol správne zarovnaný vstupný objekt s chýbajúcou časťou, z výsledného pomeru vieme zistiť aká veľká časť objektu chýba.

6 Implementácia modelu a evaluácia

Základom implementovaného riešenia je neurónová sieť na registráciu 3D objektov CorsNet. Jadrom práce na projekte v zimnom semestri bola práve reimplementácia modelu a jeho natrénovanie na datasete ModelNet40. PointNet časť modelu na získanie reprezentácie vstupných objektov a rovnako aj kompletný CorsNet model boli úspešne implementované, pričom výsledky dosiahnuté v oboch prípadoch sa blížili k výsledkom odprezentovaným v článkoch [1, 2]. V nasledujúcich častiach sú uvedené detaily reimplementácie a evaluácia modelov.

6.1 PointNet implementácia

Hlboká neurónová sieť PointNet je podrobne opísaná v kapitole 3.2. Základná architektúra siete umožňuje extrahovanie lokálnych a globálnych čŕt zo vstupu reprezentovaného neusporiadanými množinami bodov a následne využíva vektorové reprezentácie čŕt na vykonanie klasifikácie. Na registráciu objektov sú potrebné iba časti na extrakciu čŕt, avšak nakoľko sme chceli mať možnosť použiť *transfer learning* pri trénovaní modelu CorsNet, bolo potrebné implementovať a natrénovať celý klasifikačný model.

Architektúra reimplementovaného modelu PointNet kopíruje schému z obr. 3.2. Vstupný objekt je reprezentovaný $n \times 3$ maticou, kde n predstavuje počet bodov, pričom každý bod je daný trojicou $[x, y, z]$ súradníc. Po vstupnej vrstve nasleduje prvá transformačná časť *T-Net*, ktorej úlohou je určenie transformačnej matice na zarovnanie vstupných bodov. *T-Net* spracováva jednotlivé body pomocou MLP implementovaných ako 2D konvolúcie s veľkosťou kernelu 1×1. Použité sú tri takéto vrstvy s počtami kernelov 64, 128 a 1024. V rámci *T-Net* je na ďalšej vrstve aplikovaná funkcia *max pooling*, ktorá zabezpečuje rovnaký výstup pri ľubovoľnej permutácii vstupných bodov. Výstup pooling vrstvy je spracovaný tromi plne prepojenými vrstvami s veľkosťami 512, 256 a 9. Výsledný vektor je transformovaný na 3×3 maticu, ktorou je vynásobená vstupná $n \times 3$ matica. Nasledujú dva MLP so zdieľanými váhami implementované pomocou konvolučných

vrstiev a ich výstup je opäť zarovnaný transformačnou maticou, ktorá je výstupom druhej "mini" siete *T-Net*. Transformačná matica má tentokrát veľkosť 64×64 a slúži na zarovnanie reprezentácie lokálnych čŕt, ktorá sú výstupom posledného MLP pred *T-Net*. Nasledujú tri MLP a pooling vrstva na získanie globálnych čŕt. Touto vrstvou končí časť na extrakciu čŕt a ďalšie tri plne prepojené vrstvy slúžia len na klasifikáciu objektu. Po všetkých konvolučných aj plne prepojených vrstvách aj aplikovaná normalizácia a až potom nasleduje aktivačná funkcia ReLU. Medzi posledné MLP na klasifikáciu boli pridané *dropout* vrstvy za účelom regularizácie trénovania.

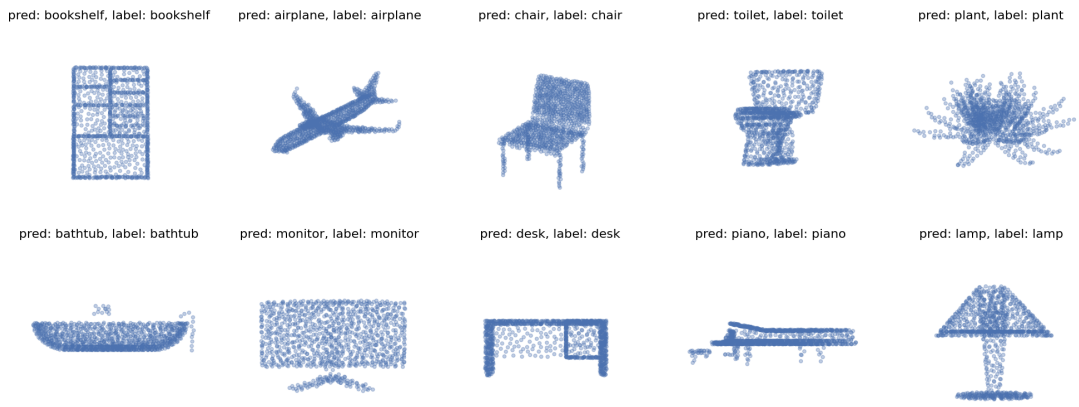
Reimplementácia modelu PointNet bola vykonaná pomocou Python knižnice Keras a vychádzala z verejne dostupného zdrojového kódu pre pôvodný model, pri ktorom bola použitá priamo knižnica TensorFlow.

6.2 PointNet trénovanie a evaluácia

Implementovaný model PointNet bol natrénovaný na klasifikáciu objektov na datasete ModelNet40. Dataset obsahuje 12 311 objektov patriacich do 40 tried a rozdelených na trénovaciu sadu (9 843 objektov) a testovaciu (2 468 objektov). Vstupné mračná bodov s veľkosťou 1024 sú rovnomerne navzorkované z povrchu objektov a normalizované do jednotkovej kocky. V rámci augmentácie počas trénovania bol pridaný šum a objekty boli rotované okolo zvislej osi.

Hyperparametre boli nastavené rovnakým spôsobom ako vo verejne dostupnej PointNet implementácii s batch veľkosťou 32, optimalizátorom Adam s hodnotou *learning rate* nastavenou na 0.001 a počtom epoch limitovaným na 250.

Celková presnosť (angl. *accuracy*) najlepšieho natrénovaného modelu bola 86.89%, čo sa blížilo k hodnote 89.22%, ktorú dosiahli Qi a kolektív [1]. Nakoľko našim cieľom nie je klasifikácia objektov a model PointNet má byť využitý iba na extrakciu čŕt, dosiahnuté výsledky potvrdili správnosť reimplementácie a boli dostatočné pre ďalšiu prácu na projekte. Príklady klasifikácie natrénovanej siete je možné vidieť na obr. 6.1.



Obr. 6.1: *PointNet klasifikácia.*

6.3 Predspracovanie dát pre CorsNet

Kurobe a kolektív [2] uviedli, že dataset pre CorsNet bol vytvorený z ModelNet40, avšak spôsob predspracovania nebol veľmi detailne opísaný. Na trénovanie modelu bolo potrebné vytvoriť dvojice vstupných objektov (šablónu a vstupný objekt) a k nim prislúchajúcu transformačnú maticu $\mathbf{G}_{gt}$, ktorá umožňuje premietnutie vstupného objektu na šablónu.

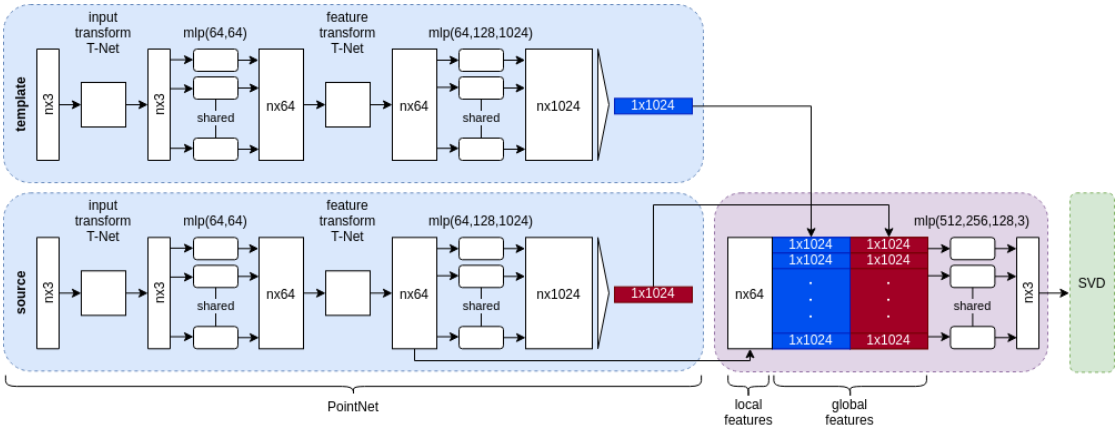
Mračná bodov pre vstupné objekty boli vygenerované rovnakým spôsobom ako pre PointNet. Z povrchu objektov bolo rovnomerne navzorkovaných 1024 bodov, ktoré boli následne normalizované do jednotkovej kocky. Transformačné matice boli náhodne vygenerované s rotáciami okolo ľubovoľných osí z intervalu $< 0, 45 >$ stupňov a transláciami z intervalu $< 0, 0.8 >$ pre všetky tri osi. Šablóny pre príslušné vstupné objekty boli vytvorené ako skalárny súčin šablóny a transformačnej matice $\mathbf{G}_{gt}$. Takto pripravené trojice boli následne použité na natrénovanie modelu na registráciu objektov.

6.4 CorsNet implementácia

Na rozdiel od modelu PointNet, pre CorsNet nebola verejne dostupná implementácia a preto sme primárne vychádzali z náčrtu architektúry, ktorý je spolu s ďalšími

detailami o fungovaní modelu opísaný v kapitole 4.2. Model sa skladá z dvoch PointNet sietí, jednej vrstvy na zjednotenie reprezentácie oboch vstupných objektov, niekoľko MLP na predikciu transformácie ako rozdielu medzi objektami a SVD vrstvy, ktorej výstupom je predikovaná transformačná matica. Podobne ako pôvodný CorsNet, aj naša reimplementácia je end-to-end neurónová sieť s SVD časťou implementovanou ako samostatnou vrstvou bez trénovateľných parametrov.

Naša neurónová sieť až na niekoľko detailov kopíruje architektúru zobrazenú na obr. 4.2. Zo schémy CorsNet modelu sa javí, že model PointNet nebol použitý kompletný a transformačné časti *T-Net* boli vynechané. V našom modeli je na extrakciu lokálnych a globálnych čŕt použitá nezmenená PointNet architektúra ako je opísaná v časti 3.2. Schéma implementovanej neurónovej siete je znázornená na obr. 6.2.



Obr. 6.2: *CorsNet architektúra (reimplementácia). Architektúra pozostáva z troch segmentov: získanie globálnych čŕt (**modrá**), odhad súhlasnosti (**fialová**) a SVD (**zelená**).*

6.5 CorsNet trénovanie

Na trénovanie modelu boli použité ako vstupy predspracované dvojice mračien bodov reprezentované maticami o veľkosti 1024×3 a úlohou modelu bolo predikovať rigidnú transformačnú maticu na premietnutie mračna bodov pre vstupný objekt na šablónu.

Na určenie chyby predikovanej transformačnej matici $\mathbf{G}_{est}$ voči *ground truth* matice $\mathbf{G}_{gt}$ bola použitá vlastná stratová funkcia (angl. *loss function*). Dáta boli

predspracované tak, aby pre vstupný objekt $\mathbf{P}_S$ a šablónu $\mathbf{P}_T$ platilo

$$\mathbf{P}_T = \mathbf{G}_{gt} \cdot \mathbf{P}_S. \tag{6.1}$$

Pre CorsNet [2] bola stratová funkcia definovaná ako

$$loss = ||(\mathbf{G}_{est})^{-1} \cdot \mathbf{G}_{gt} - \mathbf{I}_4||_F. \tag{6.2}$$

Táto funkcia bola prebraná z modelu PointNetLK [28], pre ktorý je verejne dostupná implementácia, kde stratová funkcia vyzerá nasledovne:

$$loss = 16 \cdot ||(\mathbf{G}_{est})^{-1} \cdot \mathbf{G}_{gt} - \mathbf{I}_4||_F^2, \tag{6.3}$$

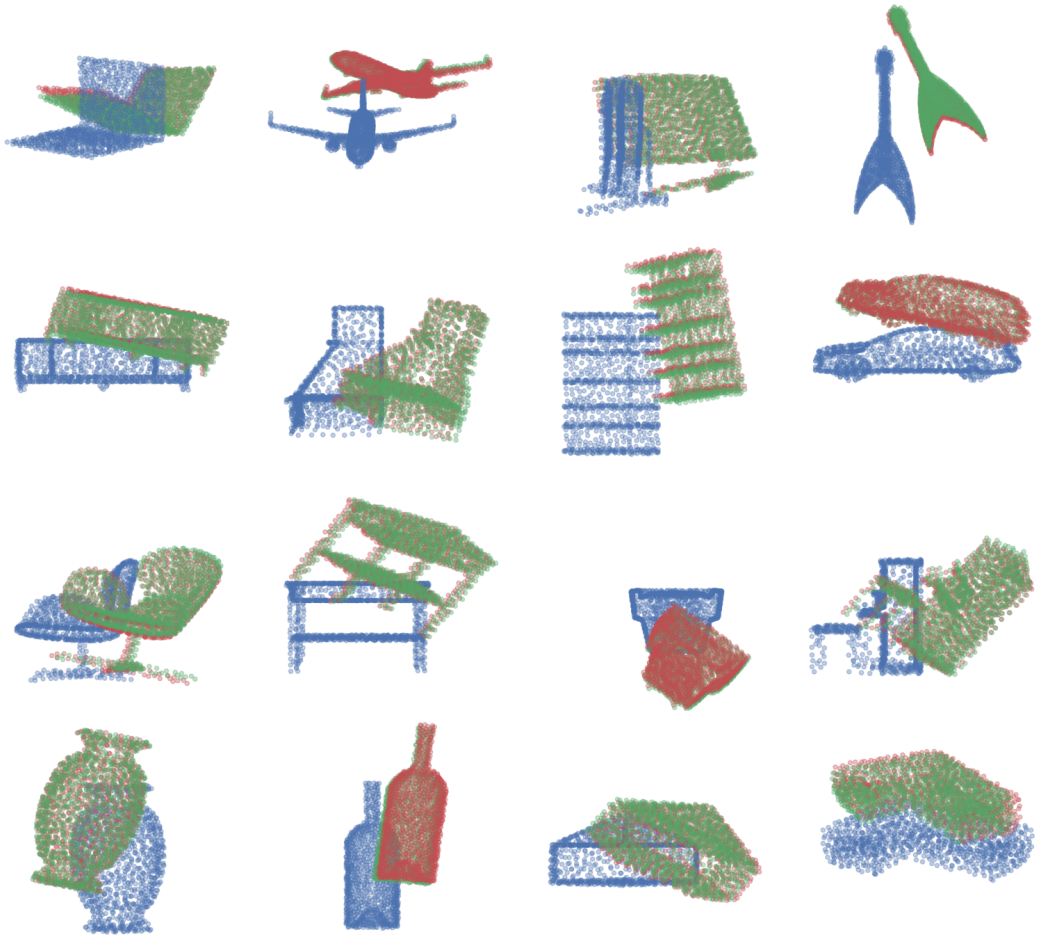
hoci autori v článku uviedli rovnakú definíciou ako pre CorsNet. Použitie MSE namiesto RMSE a vynásobenie výsledku konštantou pri našich experimentoch viedlo v rýchlejšej konvergencii funkcie a preto bola použitá definícia 6.3.

Úspešnosť predikcií bola meraná predovšetkým implementovanou stratovou funkciou. Okrem jej hodnoty boli sledované aj metriky presnosť (angl. *accuracy*) a MSE. Nastavenia hyperparametrov uvedené v článku [2] zahŕňalo použitie optimalizátora Adam s počiatočnou hodnotou *learning rate* 0.0001. Počet epoch bol nastavený na 300, pričom hodnota *learning rate* bola pri 75., 150. a 200. epoche vydelená 10.

6.6 CorsNet experimenty

Pre originálny model CorsNet neboli dostupné žiadne informácie o augmentácií dát počas trénovania alebo prípadnom použití *transfer learning* s predtrénovanými PointNet váhami a práve tieto charakteristiky boli prvým predmetom výskumu. Testované boli rôzne spôsoby augmentácie dát počas trénovania a drobné zmeny v predspracovaní objektov. Sledovaný bol taktiež vplyv použitia predtrénovaných váh na extrakciu čŕt vstupných objektov a obmedzenie trénovania váh na určitých vrstvách. Pri nastavení hyperparametrov sme sa snažili držať detailov uvedených v časti 6.5, avšak pri viacerých experimentoch bolo potrebné tieto nastavenia pozmeniť predovšetkým z dôvodu hardvérových obmedzení. Modely boli trénované

na zariadení s 16 GB RAM a NVIDIA GeForce GTX 1650 Max-Q GPU 4GB.



Obr. 6.3: *Výsledky registrácie objektov (**modrá**: vstupný objekt, **zelená**: šablóna, **červená**: transformácia).*

V tejto fáze projektu nebolo cieľom dosiahnuť čo najlepšie výsledky siete. Dôraz bol kladený na overenie fungovania modelu a identifikáciu najvhodnejšej kombinácie predspracovania a augmentácie dát. Nájdená optimálna kombinácia bude slúžiť ako základ pri predspracovaní vlastného datasetu. Čas trénovania bol skrátený z pôvodných 300 epoch na 200. Napriek tomuto obmedzeniu dosahovali viaceré natrénované modely úspešné výsledky. Na evaluáciu boli použité metriky presnosť, MSE počítaná nad celou predikovanou maticou $\mathbf{G}_{est}$ a podobne ako pre originálny

CorsNet aj RMSE samostatne pre rotáciu $\mathbf{R}$ a transláciu $\mathbf{t}$. Pre použité metriky platí:

$$16 \cdot MSE(\mathbf{G}) = 9 \cdot RMSE(\mathbf{R})^2 + 3 \cdot RMSE(\mathbf{t})^2. \tag{6.4}$$

6.6.1 Experimenty - predspracovanie dát

Spôsob generovania transformačných matíc bolo ponechaný z pôvodného predspracovania s náhodnými rotáciami z intervalu $< 0, 45 >$ stupňov a transláciami z intervalu $< 0, 0.8 >$ pre všetky tri osi. Modifikovaný bol spôsob získavania mračien bodov z 3D objektov. Pôvodne boli vstupný objekt aj šablóna tvorené rovnakým mračnom bodov a líšili sa iba transformáciou v priestore. Model PointNet by však mal byť schopný vytvoriť podobné reprezentácie globálnych čŕt pre rôzne mračná bodov opisujúce rovnaký objekt a preto by malo byť možné vykonať aj registráciu nad vstupným objektom a šablónou tvorených rôznymi bodmi z povrchu jedného objektu.

Predspracované dáta, ktoré použili Qi a kolektív [1] na trénovanie modelu PointNet obsahujú objekty opísané 2048 bodmi rovnomerne navzorkovanými z povrchu objektov. Pôvodne boli pre trénovanie vstupné mračná bodov tvorené iba polovicou týchto bodov. My sme navrhli alternatívny spôsob predspracovania, pri ktorom je vstupný objekt tvorený prvými 1024 bodmi a šablóna zvyškom.

Predspracovanie	**Presnosť**	**MSE**	**RMSE (R)**	**RMSE (t)**
Rovnaké mračná bodov	95.60%	0.00574	0.09015	0.06725
Rôzne mračná bodov	**96.10%**	**0.00501**	**0.08627**	**0.05537**

Tabuľka 6.1: *Evaluácia rôznych metód predspracovania.*

Dve neurónové siete boli natrénované s rovnakým nastavením hyperparametrov a augmentácie a jediný rozdiel bol v predspracovaní dát. Evaluácia bola vykonaná nad datasetom tvoreným rovnakými mračnami bodov pre oba vstupné objekty a z tabuľky 6.1 je vidno, že pri použití rôznych mračien bodov pre šablónu a vstupný objekt boli celkovo dosiahnuté lepšie výsledky.

6.6.2 Experimenty - augmentácia dát

Inšpirovaní augmentáciou aplikovanou pre PointNet, aj v tomto prípade bol počas trénovania k obom vstupným mračnám bodov pridaný šum z normálnej distribúcie s hodnotou $\sigma = 0.01$ a hornou hranicou 0.05. Táto perturbácia dát sa už pri prvých natrénovaných modeloch ukázala ako užitočná a bola použitá pri všetkých modeloch spomenutých vo výsledkoch.

PointNet architektúra by však mala zabezpečiť minimálne zmeny v reprezentácii objektov bez ohľadu na usporiadanie bodov vo vstupe. Na potvrdenie tohto predpokladu bolo v rámci augmentácie dát pri trénovaní otestované zamiešanie bodov reprezentujúcich vstupné objekty nezávisle od seba. Týmto spôsobom sa aj pri použití rovnakých mračien bodov pre šablónu aj vstupný objekt malo zabrániť situáciám, kedy by na registráciu objektov nebola potrebná neurónová sieť, ale postačoval by iba lineárny výpočet aplikovaný na poslednej vrstve siete.

Augmentácia	**Presnosť**	**MSE**	**RMSE (R)**	**RMSE (t)**
Bez premiešania bodov	95.51%	**0.00507**	**0.08355**	0.06900
Premiešanie bodov	**95.60%**	0.00574	0.09015	**0.06725**

Tabuľka 6.2: *Evaluácia rôznych metód augmentácie dát.*

Z výsledkov na tabuľke 6.2 nie je možné jednoznačne tvrdiť, či premiešanie bodov prispelo k zmenšeniu odchýlky predikovaných transformácií. Porovnávané metriky však ukázali, že táto forma augmentácie dát výrazne nezhoršilo výsledky siete a prispelo k univerzálnosti funkcie siete pre dáta z reálneho sveta.

6.6.3 Experimenty - transfer learning

S predtrénovaným modelom PointNet na klasifikáciu bolo možné otestovať využitie *transfer learning* a jeho vplyv na predikcie siete. Natrénované boli tri modely s rôznymi kombináciami nastavení (tab. 6.3). Ak bola použitá metóda *transfer learning*, váhy v PointNet častiach modelu CorsNet boli inicializované na hodnoty z najlepšieho natrénovaného modelu dosahujúceho presnosť klasifikácie 86.89%. Otestovaná bola aj alternatíva, kedy boli PointNet váhy fixné a trénovateľné boli iba váhy v časti sieti na určenie posunutia medzi vstupným objektom a šablónou.

Výsledky z tabuľky 6.4 ukazujú, že obmedzenie trénovateľných parametrov viedlo k najhorším výsledkom. Prekvapivým správaním bolo, že aplikácia metódy *transfer learning* neprispela k lepším výsledkom siete v porovnaní s náhodnou inicializáciou váh.

Model	**Transfer learning**	**Trénovateľný PointNet**
CorsNet-v1	✓	✓
CorsNet-v2	✓	-
CorsNet-v3	-	✓

Tabuľka 6.3: *Nastavenie modelov na testovanie prínosu* transfer learning.

Model	**Presnosť**	**MSE**	**RMSE (R)**	**RMSE (t)**
CorsNet-v1	95.51%	0.00507	0.08355	0.06900
CorsNet-v2	85.75%	0.03894	0.20493	0.28242
CorsNet-v3	**96.56%**	**0.00336**	**0.07041**	**0.03570**

Tabuľka 6.4: *Evaluácia prínosu* transfer learning.

6.7 Vstupné dáta a predspracovanie

Vstupne dáta pre túto prácu obsahujú písmená, číslice a ďalšie znaky získané z 58 obojstranných skenov pneumatík. Celkovo je k dispozícii 116 vyrovnaných skenov, z ktorých boli detekciou objektov získané prítomné znaky. Príklady detegovaných písmen a číslic sú znázornené na 6.4. Zároveň sú k dispozícii ďalšie súbory obsahujúce 2D hĺbkovú mapu pôvodných skenov vyrovnaných do obdĺžnikov a súradnice určujúce umiestnenie každého zo znakov na vyrovnanom povrchu pneumatiky. Tieto informácie budú dôležité pre prípadné skladanie slov z písmen v neskorších fázach projektu, prvotné zameranie však bude na registráciu jednotlivých písmen a číslic.

Na separáciu jednotlivých znakov bola použitá detekcia na základe hrán, čo má za dôsledok mierne nedokonalosti vo vstupných dátach. Príkladom sú chýbajúce bodky nad písmenami *i* a *j* alebo rozdelenie grafiky zo strán pneumatík na jednotlivé geometrické útvary. Dostupné dáta obsahujú okrem písmen a číslic aj ďalšie ASCII znaky ako matematické symboly alebo zátvorky. Prítomné sú však aj kompletné

menšie obrázky alebo ich časti a ďalšie non-ASCII znaky. Príklady týchto objektov sú uvedené na obr. 6.5.

Obr. 6.4: *Vstupné dáta - ASCII znaky.*

Dostupné dáta už boli vopred spracované a pre účely projektu nebude potrebné zložité predspracovanie. Celkový postup pozostáva zo štyroch krokov:

1. odstránenie objektov opisujúcich non-ASCII znaky,
2. normalizácia objektov,
3. navzorkovanie dvojíc šablóna-vstup z každého objektu,
4. vygenerovanie transformačnej matice a transformácia šablóny.

Obr. 6.5: *Vstupné dáta - non-ASCII znaky.*

Na odstránenie non-ASCII objektov bude potrebné manuálne pretriediť celý dataset a odstrániť problematické objekty, ktorých príklady sú uvedené na obr. 6.5. Prvotným plánom je odstrániť všetky non-ASCII objekty, je však možné, že bude zvážené aj ponechanie kompletných obrázkov. Samostatné bodky, geometrické tvary a bližšie nešpecifikovateľné objekty nebudú použité pri trénovaní modelu.

Po odstránení nevhodných objektov bude aplikovaná rovnaká forma normalizácie ako pri doterajšom výskume a vstupné objekty budú zarovnané do jednotkovej kocky. Následne budú z každého objektu náhodne navzorkované dvojice šablóna-vstup, ktoré budú reprezentovať vstupné mračná bodov pre neurónovú sieť. V časti 6.6 bolo ukázané, že model model PointNet skutočne prispieva k úspešnej registrácii vstupných objektov, ktoré síce so šablónou opisujú identický útvar, avšak

body použité na jeho reprezentáciu nemusia byť rovnaké. Táto vlastnosť modelu bude využívaná naďalej v snahe simulovať situácie z reálneho prostredia, kedy nie je možné očakávať, že mračná bodov získané zo skenov toho istého objektu budú sú rovnaké. Z toho dôvodu budú pre vstupné dvojice objektov navzorkované dve rôzne mračná bodov z rovnakého znaku alebo písmena. Doteraz vyvíjané modely spracovávali komplexnejšie objekty a na vstupe očakávali mračná bodov o veľkosti 1024. V našom prípade však ide o jednoduchšie vstupy a experimenty budú vykonané aj nad menšími vstupmi.

Predspracovanie bude zakončené vygenerovaním transformačnej matica a je použitím na transformáciu mračna bodov reprezentujúceho šablónu. V prvej fáze budú transformácie generované rovnakým spôsobom ako pri doterajšej práci, kedy boli použité rotácie okolo ľubovoľných osí z intervalu $< 0, 45 >$ stupňov a translácie z intervalu $< 0, 0.8 >$. Ďalšie kroky predspracovania ako mierne deformovanie objektov a vytváranie komplexnejších vstupov z viacerých objektov bude navrhnuté v závislosti od priebežne dosiahnutých výsledkov.

7 Zhodnotenie

Cieľom tejto práce je poskytnutie analýzy problematiky registrácie objektov na mračnách bodov metódami hlbokého učenia a implementácia vlastného modelu na registráciu objektov z vlastného datasetu písmen, číslic a ďalších znakov. Na riešenie daného problému bola dôležitá analýza rôznych metód predspracovania a reprezentácie mračien bodov, pričom špeciálny dôraz bol kladený na hlbokú neurónovú sieť PointNet, ktorej bola venovaná celá kapitola 3. Jej hlavným prínosom je, že umožňuje extrahovanie lokálnych a globálnych čŕt priamo z mračien bodov reprezentovaných neusporiadanými množinami bodov. Spôsob extrakcie čŕt z modelu PointNet bol základom pre viaceré modely hlbokého učenia na registráciu objektov, medzi ktoré patrí aj hlboká neurónová sieť CorsNet.

V tejto práci sú poskytnuté detaily reimplementácie modelov PointNet a CorsNet. Cieľom výskumu v zimnom semestri bolo overenie fungovania modelu CorsNet a identifikácia najvhodnejších metód predspracovania a augmentácie dát pre reimplementovanú neurónovú sieť CorsNet. Získané poznatky budú v ďalšej fáze riešenia práce použité na predspracovanie vlastného datasetu a vývoj modelu na registráciu ASCII znakov a našich dát.

Experimenty ukázali, že reprezentácia vstupného objektu a šablóny rôznymi mračnami bodov opisujúcimi rovnaký objekt vo fáze predspracovania viedlo zlepšeniu robustnosti siete a k úspešnejším výstupom registrácie. Ďalším faktorom prispievajúcim k robustnosti modelu a čiastočne aj k jeho úspešnosti je náhodné premiešanie bodov vstupných objektov v rámci augmentácie počas trénovania. Využitie *transfer learning* pri trénovaní modelu sa naopak preukázalo slabšími výsledkami v porovnaní s náhodnou inicializáciou váh, preto v ďalších fázach projektu neplánujeme používať predtrénované váhy na extrakciu čŕt.

Keby boli dosiahnuté výsledky uspokojivé, navrhnutú neurónovú sieť bude možné využiť priamo v produkčnom prostredí v gumárenskom alebo automobilovom priemysle. V súčasnosti používaný softvér na kontrolu povrchu pneumatík (angl. *Tire Surface Inspection*) vrátane kontroly kvality textu na stranách je komplikovaný a výpočtovo náročný a navrhnuté riešenie by mohlo priniesť zlepšenie.

7.1 Ďalšia práca

V kapitole 5 je opísaných osem bodov práce na projekt, z ktorých bola prvá polovica riešená počas zimného semestra. Najbližšími krokmi bude teda predspracovanie našich dát a vývoj modelu na vlastnom datasete. Plány pre ďalší vývoj modelov CorsNet a jeho súčasti PointNet sú opísané v samostatných častiach.

7.1.1 PointNet - ďalší vývoj

Pre vylepšenie modelu samotného modelu PointNet nie je veľa možností pre vylepšenie. PointNet je vo všeobecnosti považovaný za *state-of-the-art* riešenie pre získavanie vektorovej reprezentácie priamo z neusporiadaných mračien bodov a veľa iných modelov v oblasti registrácie objektov v 3D mračnách bodov využíva tento model podobným spôsobom ako CorsNet, práve na extrakciu lokálnych a globálnych čŕt. Predpokladáme, že aj na optimalizáciu hyperparametrov bolo vykonané veľké množstvo experimentov a týmto spôsobom výrazné zlepšenie nedosiahneme. Pre naše použitie siete je však možné rozšíriť augmentáciu dát a okrem pridávania šumu a rotácie okolo zvislej osi by mohlo byť vhodné pridať rotáciu okolo ľubovoľnej osi, prípadne rotáciu naraz vo viacerých smeroch. Vývoj samotného modelu PointNet však nie je cieľom tejto práce a spomenuté zmeny môžu byť aplikované aj pre model CorsNet, ktorého vylepšenie bude jadrom práce v nasledujúcom semestri.

7.1.2 CorsNet - ďalší vývoj

Okrem optimalizácie hyperparametrov a trénovania modelu na vlastnom datasete bude otestované ešte niekoľko možností vylepšenia siete. Model CorsNet bol trénovaný so štyrmi rôznymi verziami stratových funkcií, pričom v doterajších experimentoch sme pracovali iba s prvým typom. V ďalších krokoch bude určite zvážené použitie aj zvyšných typov. Podobne ako aktuálne používanú stratovú funkciu, aj zvyšné tri bude potrebné implementovať.

Ďalšia možná modifikácia siete vychádza z obrázku architektúry modelu CorsNet. Pri porovnaní PointNet časti tohto modelu a originálnej PointNet architektúry je možné vidieť, že pôvodný PointNet obsahuje 2 transformačné siete *T-Net*, zatiaľ čo v nákrese architektúry modelu CorsNet chýbajú. Úlohou týchto mini sietí je

aproximovať afínnu transformačnú maticu, čo má pomôcť eliminovať geometrické transformácie objektu ako rotácia alebo zrkadlenie. Pri klasifikácii je táto časť siete esenciálna, nakoľko je cieľom správne klasifikovať objekt bez ohľadu na jeho otočenie v priestore, avšak prínos tejto časti siete pre CorsNet však nemusí byť taký podstatný. Keďže cieľom CorsNet modelu je predikcia transformačnej matice, ktorá pozostáva aj z rotácie, transformovaním objektu v rámci PointNet časti sa môžu strácať informácie o rotácii a translácii v priestore. Odstránením *T-Net* častí siete by sa výrazne zmenšil počet trénovateľných parametrov, čo by umožnilo použitie väčších *batch* a celkové urýchlenie trénovania.

Okrem vývoja samotného modelu budeme určite nadväzovať na poznatky získané z predošlej evaluácie a porovnaní rôznych metód predspracovania a augmentácie dát pričom ďalší čas bude venovaný vylepšovaniu textu s cieľom lepšej zrozumiteľnosti.

Literatúra

1. QI, C. R. et al.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, s. 652–660.
2. KUROBE, A. et al.: CorsNet: 3D point cloud registration by deep neural network. *IEEE Robotics and Automation Letters*. 2020, vol. 5, no. 3, s. 3960–3966.
3. DENG, L., YU, D.: Deep learning: methods and applications. *Foundations and Trends in Signal Processing*. 2014, vol. 7, no. 3–4, s. 197–387.
4. PANG, G., NEUMANN, U.: 3D point cloud object detection with multi-view convolutional neural network. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. 2016, s. 585–590.
5. SIMONY, M. et al.: Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 2018, s. 0–0.
6. QI, C. R. et al.: Frustum pointnets for 3d object detection from rgb-d data. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, s. 918–927.
7. WANG, Y. et al.: Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*. 2019, vol. 38, no. 5, s. 1–12.
8. ZHOU, Y., TUZEL, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, s. 4490–4499.
9. ENGELCKE, M. et al.: Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, s. 1355–1361.

10. HUANG, J., YOU, S.: Point cloud labeling using 3d convolutional neural network. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. 2016, s. 2670–2675.

11. ZHOU, Y. et al.: End-to-end multi-view fusion for 3d object detection in lidar point clouds. In: *Conference on Robot Learning*. 2020, s. 923–932.

12. LI, Y. et al.: MVF-CNN: Fusion of multilevel features for large-scale point cloud classification. *IEEE Access*. 2019, vol. 7, s. 46522–46537.

13. ZHI, S. et al.: Toward real-time 3D object recognition: A lightweight volumetric CNN framework using multitask learning. *Computers & Graphics*. 2018, vol. 71, s. 199–207.

14. BROWNLEE, J.: *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Machine Learning Mastery, 2019.

15. KANG, B., TRIPATHI, S., NGUYEN, T. Q.: Real-time sign language fingerspelling recognition using convolutional neural networks from depth map. In: *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. 2015, s. 136–140.

16. SU, H. et al.: Multi-view convolutional neural networks for 3d shape recognition. In: *Proceedings of the IEEE international conference on computer vision*. 2015, s. 945–953.

17. REDMON, J., FARHADI, A.: YOLO9000: better, faster, stronger. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, s. 7263–7271.

18. YANG, B., LUO, W., URTASUN, R.: Pixor: Real-time 3d object detection from point clouds. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, s. 7652–7660.

19. QI, C. R. et al.: Deep hough voting for 3d object detection in point clouds. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, s. 9277–9286.

20. PANCHAL, G. et al.: Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers. *International Journal of Computer Theory and Engineering*. 2011, vol. 3, no. 2, s. 332–337.

21. LIN, C.-C. et al.: A novel point cloud registration using 2D image features. *EURASIP Journal on Advances in Signal Processing*. 2017, vol. 2017, no. 1, s. 1–11.

22. LU, W. et al.: DeepICP: An end-to-end deep neural network for 3D point cloud registration. *arXiv preprint arXiv:1905.04153*. 2019.

23. GROSS, J., OŠEP, A., LEIBE, B.: Alignnet-3d: Fast point cloud registration of partially observed objects. In: *2019 International Conference on 3D Vision (3DV)*. 2019, s. 623–632.

24. PEREZ-GONZALEZ, J., LUNA-MADRIGAL, F., PIÑA-RAMIREZ, O.: Deep learning point cloud registration based on distance features. *IEEE Latin America Transactions*. 2019, vol. 17, no. 12, s. 2053–2060.

25. ELBAZ, G., AVRAHAM, T., FISCHER, A.: 3D point cloud registration for localization using a deep neural network auto-encoder. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, s. 4631–4640.

26. WANG, Y., SOLOMON, J. M.: Deep closest point: Learning representations for point cloud registration. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, s. 3523–3532.

27. ZHANG, Z., DAI, Y., SUN, J.: Deep learning based point cloud registration: an overview. *Virtual Reality & Intelligent Hardware*. 2020, vol. 2, no. 3, s. 222–246.

28. AOKI, Y. et al.: Pointnetlk: Robust & efficient point cloud registration using pointnet. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, s. 7163–7172.

29. ROJAS, R.: Lucas-kanade in a nutshell. *Freie Universit at Berlinn, Dept. of Computer Science, Tech. Rep*. 2010.

Príloha A: Plán práce na riešení projektu

A.1 Priebežná správa o riešení DP1

- 01.03.2021 - 07.03.2021: Zber a triedenie ďalších materiálov k téme diplomovej práce.
- 08.03.2021 - 14.03.2021: Inštalácia a spustenie modelu PointNet.
- 15.03.2021 - 21.03.2021: Podrobná analýza architektúry PointNet z vydanej práce a dostupného zdrojového kódu.
- 22.03.2021 - 04.04.2021: Podrobný opis funkcionality, architektúry a spôsobu reprezentácie vstupných dát pre model PointNet.
- 05.04.2021 - 18.04.2021: Predspracovanie dát pre vlastný model na registráciu objektov v mračnách bodov.
- 19.04.2021 - 02.05.2021: Písanie častí práce zameraných na spracovanie mračien bodov hlbokým učením a registrácie objektov v mračnách.
- 03.05.2021 - 09.05.2021: Pridanie kapitol 1, 5, 7 a plánu práce pre jednotlivé fázy riešenia diplomovej práce.
- 10.05.2021 - 16.05.2021: Finalizácia priebežnej správy o riešení DP1 a zapracovanie pripomienok.
- 17.05.2021: Odovzdanie priebežnej správy o riešení DP1.

V úvode semestra bol harmonogram práce dodržiavaný podľa plánu, čo sa však postupne s nárastom iných študijných povinností skomplikovalo. Písanie práce sa celkovo presunulo až do druhej polovice semestra. Oproti pôvodnému plánu boli vykonané určité obmeny, v rámci ktorých bolo vynechané predspracovanie dát, avšak napísaná bola rozsiahlejšia analýza prístupov k spracovaniu mračien bodov hlbokým učením, než bolo na začiatku predpokladané. Aj napriek výskytu niekoľkých komplikácií a posunu časového harmonogramu bola priebežná správa o riešení DP1 úspešne odovzdaná v požadovanom termíne.

A.2 Priebežná správa o riešení DP2

- 07.10.2021: Vytýčenie plánu práce na zimný semester.
- 08.10.2021 - 17.10.2021: Reimplementácia a trénovanie modelu PointNet.
- 18.10.2021 - 25.10.2021: Predspracovanie datasetu ModelNet40 pre CorsNet.
- 26.11.2021 - 28.11.2021: Reimplementácia a trénovanie modelu CorsNet.
- 29.11.2021 - 02.01.2022: Zhrnutie výstupov modelu a získaných poznatkov do priebežnej správy o riešení DP2.
- 03.01.2022: Odovzdanie priebežnej správy o riešení DP1.

Po vytýčení plánu práce na zimný semester boli reimplementácia modelu PointNet na základe dostupných zdrojov a prvé predspracovanie dát pre CorsNet dokončené podľa plánovaného časového harmonogramu. Reimplementácia modelu CorsNet sa však o týždeň predĺžila oproti pôvodnému plánu, nakoľko v tejto fáze projektu bol vyvíjaný nie len samotný model, ale bol testovaný aj vplyv zmien v predspracovaní dát na úspešnosť jeho predikcií. Hoci zhrnutie výsledkov práce do priebežnej správy bolo o týždeň posunuté, upravením pôvodného termínu odovzdania práce zo 17. decembra na 3. január bol odklon od dôvodného harmonogramu vyrovnaný a zároveň bol poskytnutý dodatočný čas na písanie práce.

A.3 Diplomová práca

Na základe výsledkov dosiahnutých pri riešení DP2 bude náplň práce v poslednom semestri zahŕňať nasledovné body:

- 01.02.2022 - 06.02.2022: Zapracovanie pripomienok k častiam práce odovzdaným v rámci priebežnej správy o riešení DP2.
- 07.02.2022 - 20.02.2022: Predspracovanie nášho datasetu písmen, číslic a ďalších znakov.
- 21.02.2022 - 20.03.2022: Vývoj modelu CorsNet na predspracovaných dátach.
- 21.03.2022 - 14.04.2022: Iteratívne zvyšovanie komplexnosti datasetu a vylepšenie modelu.
- 15.04.2022 - 17.04.2022: Vyhodnotenie chyby medzi šablónou a naskenovanými dátami.
- 18.04.2022 - 08.05.2022: Zhrnutie výstupov výskumu do diplomovej práce.
- 09.05.2022 - 15.05.2022: Doplnenie príloh a finalizácia písania diplomovej práce.
- 16.05.2022: Odovzdanie finálnej verzie diplomovej práce.