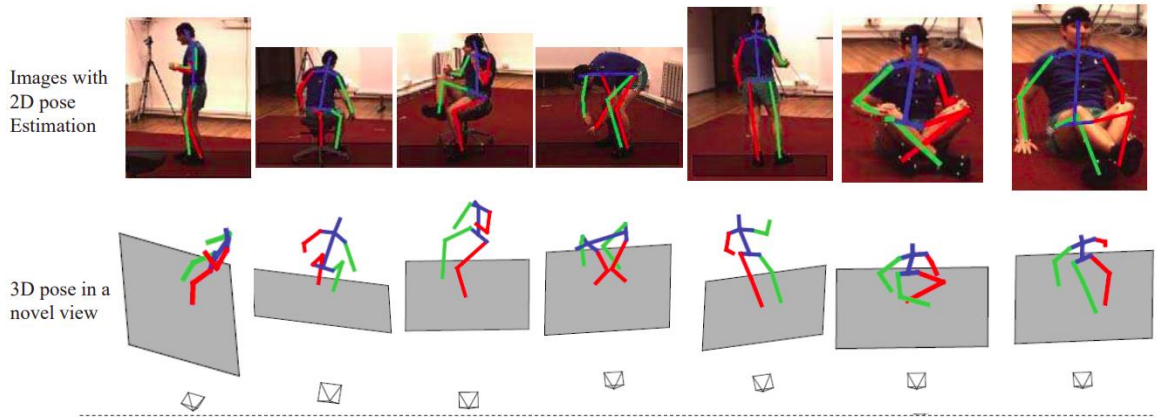
# 2-INF-150/15 Strojové učenie:
# Projekt

31.1.2024 Bc. Erik Róbert Ján Jakubovský

## Úvod do problematiky:

Pózu človeka môžeme reprezentovať ako neorientovaný graf, kde každý vrchol má 2D alebo 3D súradnicu reprezentujúcu pozíciu kĺbu alebo pohyblivej časti človeka.



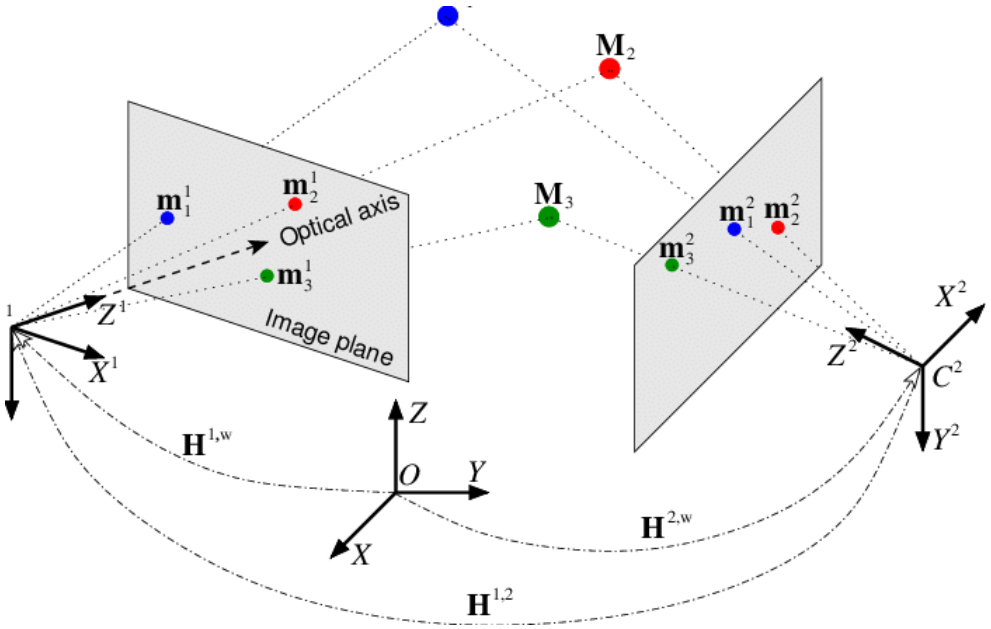

https://openaccess.thecvf.com/content_cvpr_2017/papers/Chen_3D_Human_Pose_CVPR_2017_paper.pdf

Jedným z cieľov odhadu pózy človeka je automaticky vygenerovať takúto kostru z nejakého vstupného obrazu. Zmyslom takejto kostry je, že umožňuje počítaču lepšie rozoznať správanie človeka vo vstupnom obraze. Potenciálne aplikácie sú v oblasti ako AR/VR, dozor, motion capture, self-driving autá, prípadne pri blízkej interakcii človeka so strojom a ďalšie iné.

Mnoho riešení používa farebné obrázky na odhady 2D pózy človeka v obraze pomocou konvolučných neurónových sietí. Tieto metódy zväčša predikujú heatmapy, ktoré zohľadňujú s akou pravdepodobnosťou pixel zodpovedá 2D pozícii kĺbu, tieto prístupy môžu mať vysokú presnosť, avšak nemožno z nich jednoznačne určiť pozíciu človeka v 3D priestore. Odhad 3D pózy človeka je náročnejšia úloha z viacerých dôvodov. Po prvé sa projekciou z 3D priestoru do 2D priestoru pomocou kamery stráca informácia o hĺbke a po druhé je manuálna anotácia obrazov takmer nemožná a vyžaduje použitie motion capture systému. Kvôli spomenutým nedostatkom rgb obrazov sa chcem zamerať na inú formu obrazu, konkrétne na mračná bodov. Zatiaľ čo v rgb obrázkoch máme pixely v 2D mriežke, ktorá istým spôsobom reprezentuje topológiu pixelov, tak mračná bodov sú množina 3D bodov, kde každý bod reprezentuje bod v 3D priestore zaznamenaný kamerou.

## Množina dát a predspracovanie:

Ako dataset som si zvolil AMASS. Je to dataset, ktorý zjednocuje 3D kostry z viacerých iných datasetov a aplikuje ich na parametrický SMPL model človeka. Tento model simuluje realistické správanie ľudskej kože a pohyb rúk. Pre jednoduchosť a časové obmedzenie som si zvolil iba CMU časť datasetu. Vygenerované trojuholníkové siete (mesh) som prevzal z druhej ruky. Každý snímok kostry reprezentovala trojuholníková sieť vygenerovaná z náhodnej selekcie parametrov.




https://www.researchgate.net/publication/224331619_Online_distributed_calibration_of_a_large_network_of_wireless_cameras_using_dynamic_clustering/figures?lo=1

Na generovanie štruktúrovaných mračien bodov som použil proprietárny softvér, ktorý simuluje správanie 3D skenerov. Správanie softvéru možno opísať modelom štrbinovej kamery:

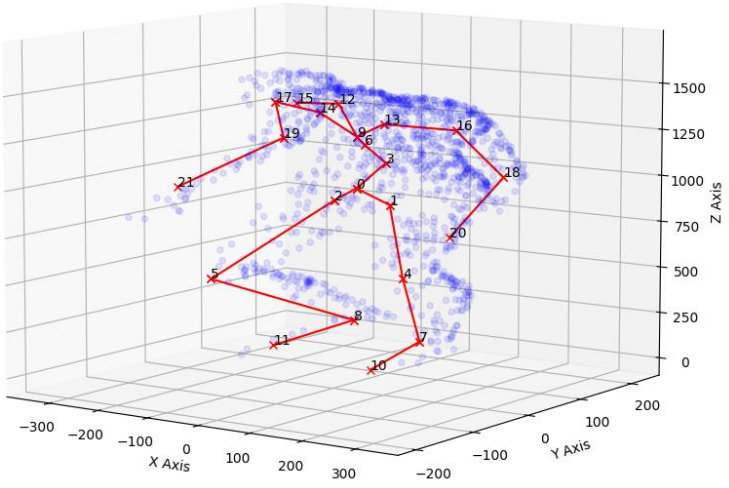
$$w\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K[R|t]\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Kde X, Y, Z sú súradnice bodu v homogénnom súradnicovom priestore (vo svete), K je matica vnútorných parametrov kamery, matica [R|t] sú vonkajšie parametre kamery a zodpovedajú transformácii bodu vo svete do súradnicového priestoru kamery, (u, v) sú obrazové súradnice pixelu a w zodpovedá hĺbke, teda vzdialenosti bodu od kamery. Maticu vnútorných parametrov tvoria ohniskové vzdialenosti v smeroch x a y a zodpovedajú vzdialenosti zobrazovacej roviny v kamerovom priestore, cx a cy sú hlavný bod a zodpovedajú stredu zobrazovacej roviny. Nakoľko je štandard, že najvrchnejší a najviac ľavý pixel obrazovky definujeme so súradnicami (0, 0), tak $c_x$ a $c_y$ zodpovedajú polovici šírky a výšky obrázku. Matica vonkajších parametrov je menej priamočiara, ale možno si ju predstaviť nasledovne riadky R, tvoria jednotkové vektory, pričom sú vzájomne kolmé a každý reprezentuje os kamerového súradnicového systému, tretí riadok reprezentuje hĺbku, druhý riadok smer hore a prvý riadok smer vbok. Nakoľko kamera neleží v strede súradnicového systému treba každý bod (X, Y, Z) posunúť o t = –RC, kde C je pozícia kamery vo svete.

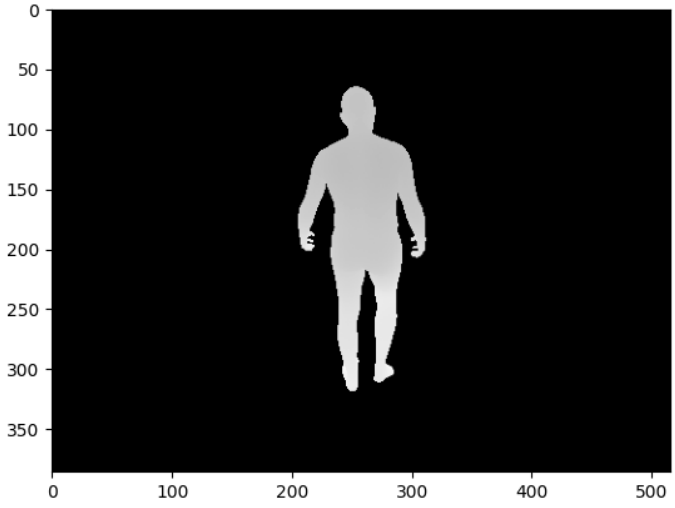
Takto definovaný model kamery využíva mnoho metód na odhad 3D pózy. Najprv sa odhadnú 2D pozície v heatmape a pre každú pozíciu sa regresuje hĺbka, potom ak poznáme parametre kamery môžeme robiť reprojekciu z 2D do 3D invertovaním matíc K a R. V praxi sa ukázalo, že možno docieliť týmto spôsobom lepšie výsledky ako priamo regresiou 3D pozície.

Štruktúrované mračno bodov si možno predstaviť ako 2D obrázok, kde pre každý pixel máme aj jeho 3D súradnicu. Štruktúrované mračná bodov boli generované pre každý mesh zo 4 rôznych pohľadov. Ak chceme vygenerovať neštruktúrované mračno bodov z viacerých pohľadov, stačí z každej mriežky zobrať všetky zaznamenané body a vložiť ich do jednorozmerného poľa, nakoľko pre hardvérové obmedzenie potrebujeme, aby v dávke boli mračná bodov s rovnakým počtom bodov, každé mračno sa podvzorkuje, algoritmom farthest point sampling. V princípe z každého mračna bodov sa vyberie podmnožina fixnej dĺžky, kde body sú si vzájomne najviac vzdialené.

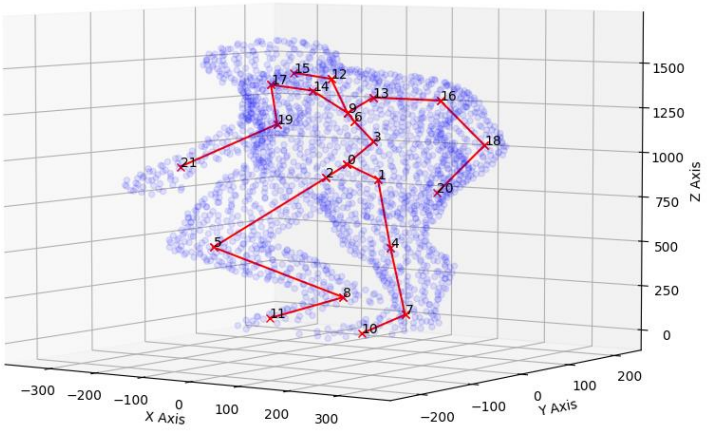
Štrukturované mračno bodov vyprojektované v priestore:




Tie isté dáta reprezentované ako hĺbková mapa:



Podvzorkované neštruktúrované mračno bodov vygenerované zo 4 pohľadov:




Pre každý dátový formát boli vyrátané priemerné body a minimálne a maximálne body po odčítaní priemeru. Pri trénovaní dáta boli normalizované centrovaním na nulu a min-max normalizáciou.

Bod sa centruje na nulu nasledovne:

$$x' = x - \bar{x}$$

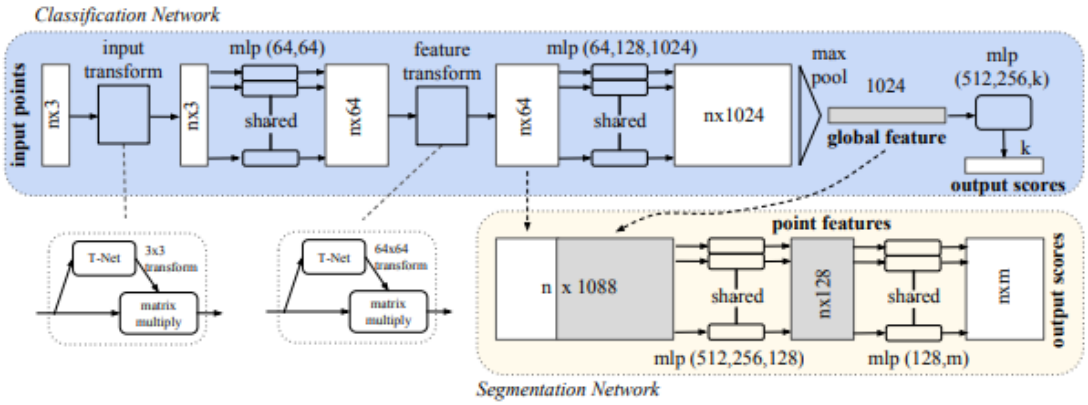
Kde $\bar{x}$ je priemerný bod. Bod sa škáluje do intervalu <a, b> nasledovne:

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$

V prípade odhadu pózy z neštruktúrovaných mračien bodov, boli transformované aj ground-truth kostry.

**PointNet:**

PointNet bola jedna z prvých neurónových sietí určená na prácu s neštruktúrovanými mračnami bodov, bez ďalšieho predspracovania. Architektúru možno použiť na klasifikáciu mračien bodov alebo segmentáciu bodov. Nakoľko matematickú množinu reprezentujeme poľom, sú body nevyhnutne usporiadané, čo môže predstavovať problém, nakoľko nechceme aby sa pri trénovaní zohľadňovalo poradie bodov. Predísť by sa tomu dalo augmentáciou, konkrétne permutovaním poradia bodov, čo vôbec nie je efektívne. Každé mračno bodov okrem iného môže sémanticky reprezentovať ten istý objekt, pri rôznych natočeniach. Tieto problémy adresuje nasledujúca architektúra:




https://arxiv.org/abs/1612.00593

Architektúru tvoria takzvané T-Net-y, ktorých úlohou je predikovať transformačné matice, ktoré transformujú vstup do rovnakého kanonického priestoru, čím sa adresuje druhý vyššie spomenutý problém. Prvý problém je vyriešeným tým, že namiesto viacvrstvových perceptrónov sa používajú takzvané zdieľané perceptróny, ktoré sú v skutočnosti reprezentované jednorozmernými konvolúciami s veľkosťou kernelu 1 × veľkosť príznakového vektora. Tieto konvolúcie generujú príznaky, bez zohľadňovania poriadia bodov, príznaky sú agregované do globálneho príznaku pomocou max-pooling-u pozdĺž prvej osi. Tento globálny príznak sa buď používa priamo na klasifikáciu alebo segmentáciu, ako je znázornené v grafe.

**Augmentovanie dát:**

Dáta boli augmentované troma spôsobmi: Každé neštruktúrované mračno bodov možno náhodne podvzorkovať, v praxi sa vždy náhodne vyberie polovica bodov zo vstupu. Prípadne pripočítať náhodný šum generovaný z normálnej distribúcie, alebo otočiť mračno bodov o náhodný uhol okolo os x, y, z rotačnými maticami. Pri experimentoch sa ako rozptyl zvolilo 5mm, otočenie okolo osi z náhodne z intervalu <–180, 180>, osi x a y z intervalu <–30, 30> .

**Riešenie:**

Z grafu je zrejmé, že PointNet nebol určený na regresiu, napriek tomu sa dá upraviť klasifikačná vetva na regresnú, rozdiel je, že ako aktivačná funkcia sa na poslednej vrstve použije lineárna aktivačná funkcia a za k sa zvolí 3j, kde j je počet kĺbov a 3 reprezentuje polohu v smere x, y, z. Dropout regularizácia bola odstránená nakoľko pri regresii môže robiť problémy a zhoršovať schopnosť regresovať. Architektúra bola reimplementovaná v Pytorch framework-u podľa pôvodnej publikácie. Stratová funkcia bola definovaná ako suma L2 normy medzi predikciou a ground truth a straty pre maticu A generovanú druhým T-Net-om:

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

Ako optimalizačná metóda sa zvolil Adam s krokom 0,001, pričom každých 20 epoch bol tento krok zmenšený o polovicu. Sieť sa trénovala 100 epoch.

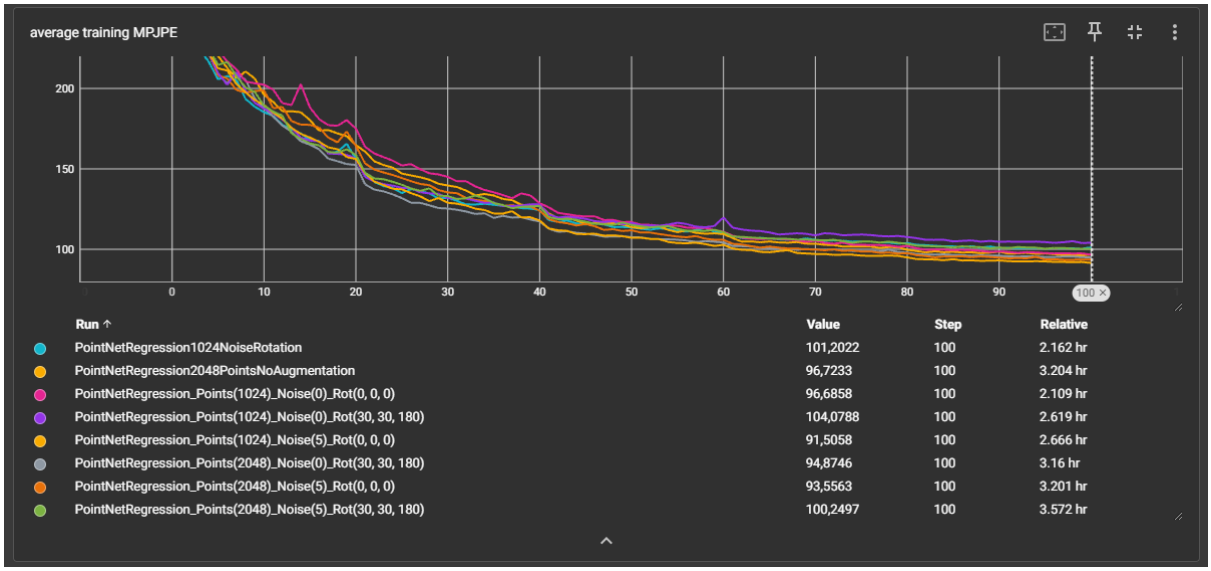
**Výsledky:**

Sieť sa trénovala 100 epoch na trénovacej množine a vyhodnocovala na validačnej podmnožine. Ako metrika sa zvolila MPJPE:

$$MPJPE = \frac{1}{NJ}\sum_{i=1}^{N}\sum_{k=1}^{J}\left\|y_k^i - \widehat{y_k^i}\right\|_2$$

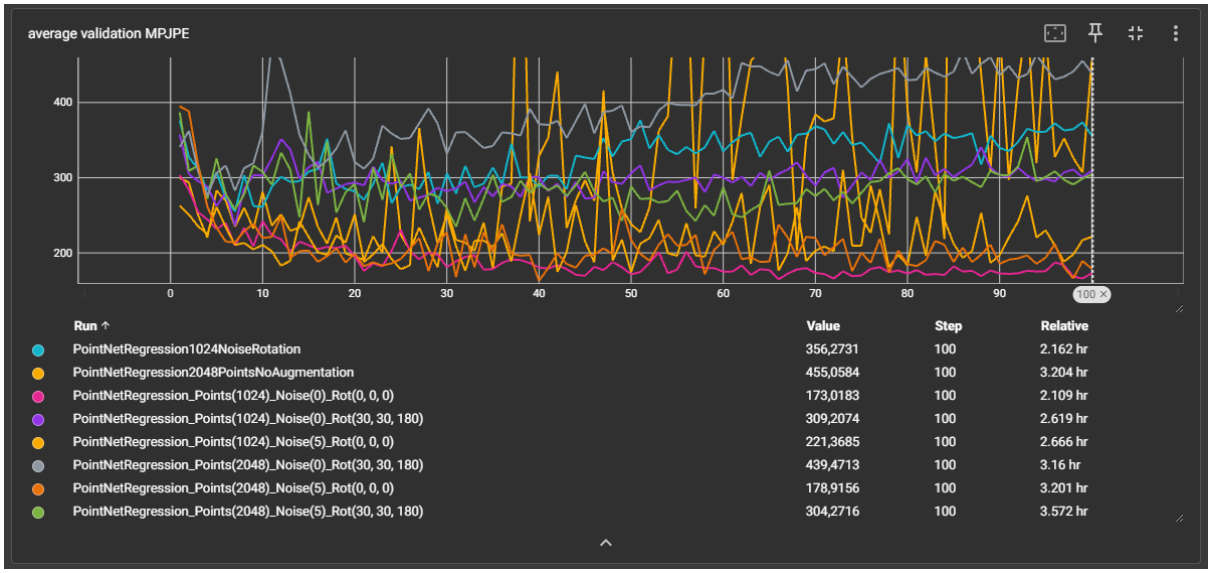
Kde $y_k^i$, je skutočná 3D pozícia kĺbu k vo vzorke i a $\widehat{y_k^i}$ je predikcia. Metrika vlastne ráta priemernú euklidovskú vzdialenosť medzi predikciou a skutočnou hodnotou. Pri súčasných state of the art metódach sa táto metrika pohybuje okolo 50 mm.

Vývoje metriky pri trénovaní:




| Run | Value | Step | Relative |
|---|---|---|---|
| PointNetRegression1024NoiseRotation | 101,2022 | 100 | 2.162 hr |
| PointNetRegression2048PointsNoAugmentation | 96,7233 | 100 | 3.204 hr |
| PointNetRegression_Points(1024)_Noise(0)_Rot(0, 0, 0) | 96,6858 | 100 | 2.109 hr |
| PointNetRegression_Points(1024)_Noise(0)_Rot(30, 30, 180) | 104,0788 | 100 | 2.619 hr |
| PointNetRegression_Points(1024)_Noise(5)_Rot(0, 0, 0) | 91,5058 | 100 | 2.666 hr |
| PointNetRegression_Points(2048)_Noise(0)_Rot(30, 30, 180) | 94,8746 | 100 | 3.16 hr |
| PointNetRegression_Points(2048)_Noise(5)_Rot(0, 0, 0) | 93,5563 | 100 | 3.201 hr |
| PointNetRegression_Points(2048)_Noise(5)_Rot(30, 30, 180) | 100,2497 | 100 | 3.572 hr |

Vývoje metriky pri validovaní:



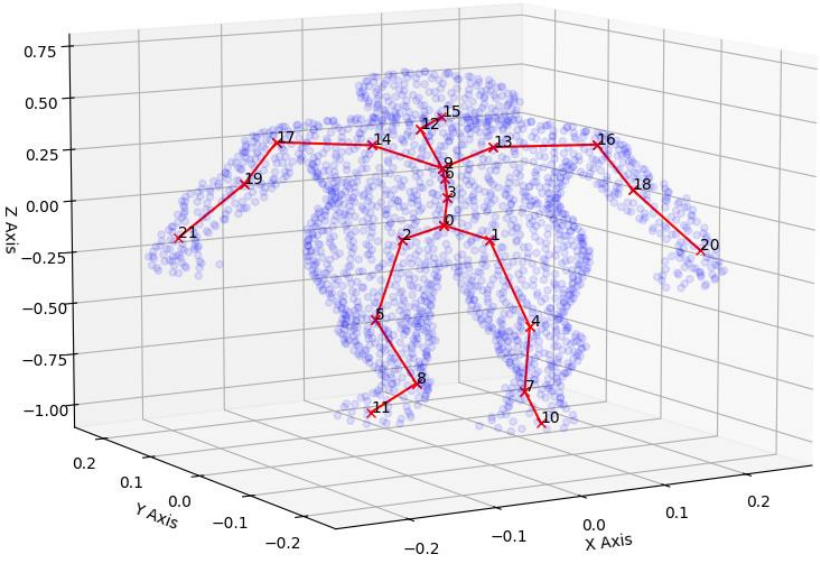

| Run | Value | Step | Relative |
|---|---|---|---|
| PointNetRegression1024NoiseRotation | 356,2731 | 100 | 2.162 hr |
| PointNetRegression2048PointsNoAugmentation | 455,0584 | 100 | 3.204 hr |
| PointNetRegression_Points(1024)_Noise(0)_Rot(0, 0, 0) | 173,0183 | 100 | 2.109 hr |
| PointNetRegression_Points(1024)_Noise(0)_Rot(30, 30, 180) | 309,2074 | 100 | 2.619 hr |
| PointNetRegression_Points(1024)_Noise(5)_Rot(0, 0, 0) | 221,3685 | 100 | 2.666 hr |
| PointNetRegression_Points(2048)_Noise(0)_Rot(30, 30, 180) | 439,4713 | 100 | 3.16 hr |
| PointNetRegression_Points(2048)_Noise(5)_Rot(0, 0, 0) | 178,9156 | 100 | 3.201 hr |
| PointNetRegression_Points(2048)_Noise(5)_Rot(30, 30, 180) | 304,2716 | 100 | 3.572 hr |

| Predspracovanie: | Najlepšia MPJPE | epocha |
|---|---|---|
| žiadne | 178,9 | 35 |
| Náhodná podmnožina | 164,8 | 66 |
| Náhodná rotácia | 276 | 4 |
| Šum | 163 | 40 |
| Náhodná podmnožina + náhodná rotácia | 235,9 | 7 |
| Náhodná podmnožina + šum | 174,4 | 50 |
| náhodná rotácia + šum | 307,4 | 5 |
| Náhodná podmnožina + náhodná rotácia + šum | 255,1 | 7 |

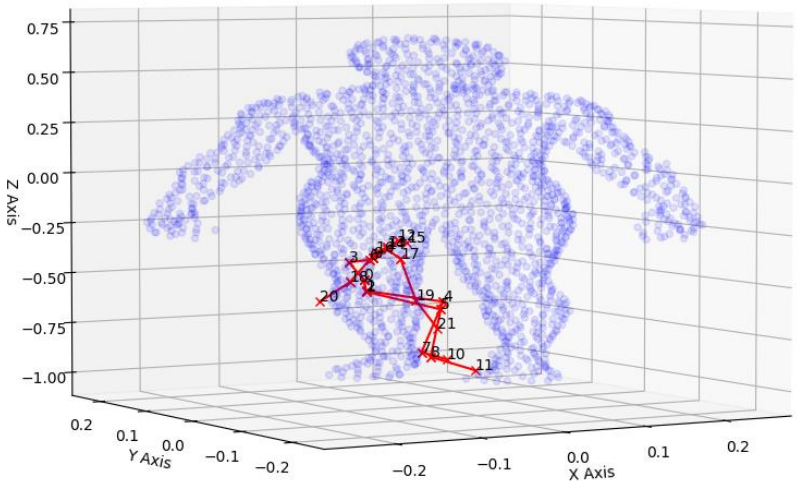
Prekvapivo je najlepšia augmentácia, len tá čo pridáva vstupným dátam náhodný šum.
V prípade náhodnej rotácii, trénovacia množina rýchlejšie overfituje.

Testovacia metrika vyšla na modely trénovanom iba augmentáciou šumom ako 169,715 mm.

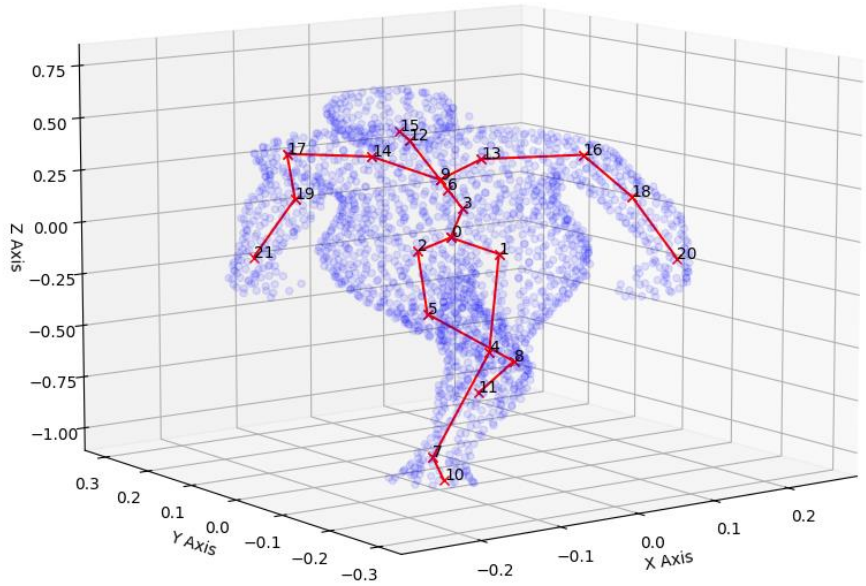
Ground Truth:



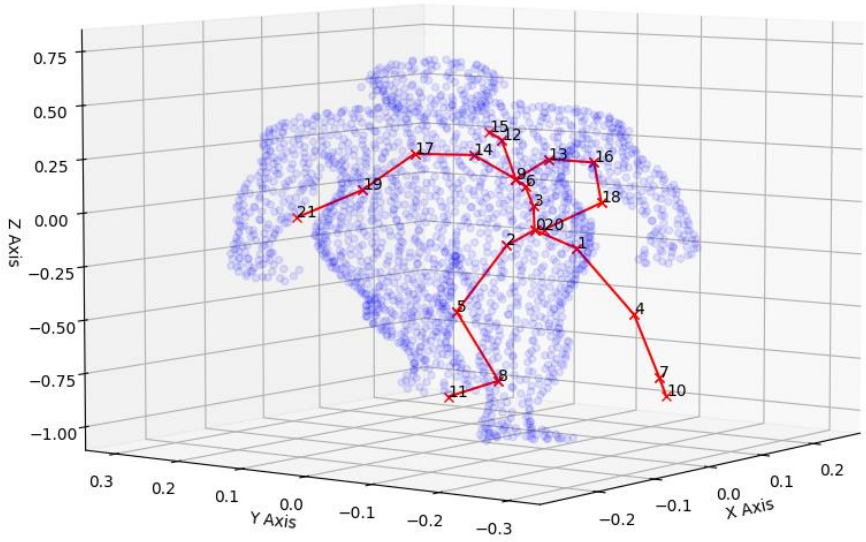

Predikcia:



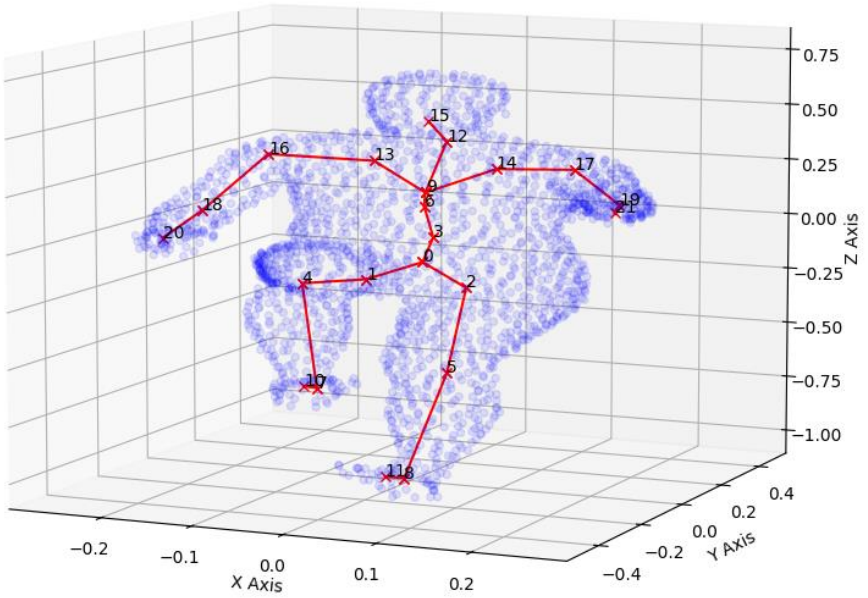
Ground Truth:



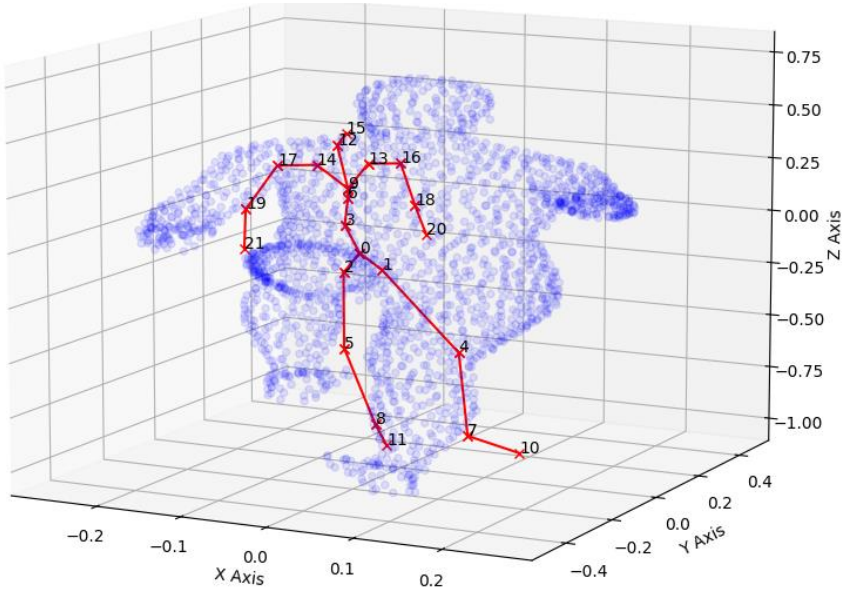

Predikcia:



Ground truth:




Predikcia:




Z obrázkov a z metriky môžeme usúdiť, že takto navrhnutá architektúra nie je vhodná na odhad pózy človeka.

**Komentár:**

Pôvodným zámerom tejto práce bolo porovnať presnosť odhadu pózy pomocou CenterNet-u, na štruktúrovaných mračnách bodov, a PointNet-u, na neštruktúrovaných mračnách bodov. Žiaľ kvôli časovej tiesni sa mi nepodarilo úspešne reimplementovať CenterNet. Cieľom malo byť odhadnúť 2D pozíciu a hĺbku kĺbov a následne ich reprojektovať do 3D. Oficiálna implementácia síce existuje ale využíva veľmi starú verziu pytorch-u a nepodarilo sa mi ju spojazdniť a skompilovať. Dáta aj implementácie sú priložené, ale v architektúre je

pravdepodobne chyba, nakoľko sieť sa neučí a veľmi zle odhaduje hĺbku, ktorá pravdepodobne znemožňuje trénovanie.