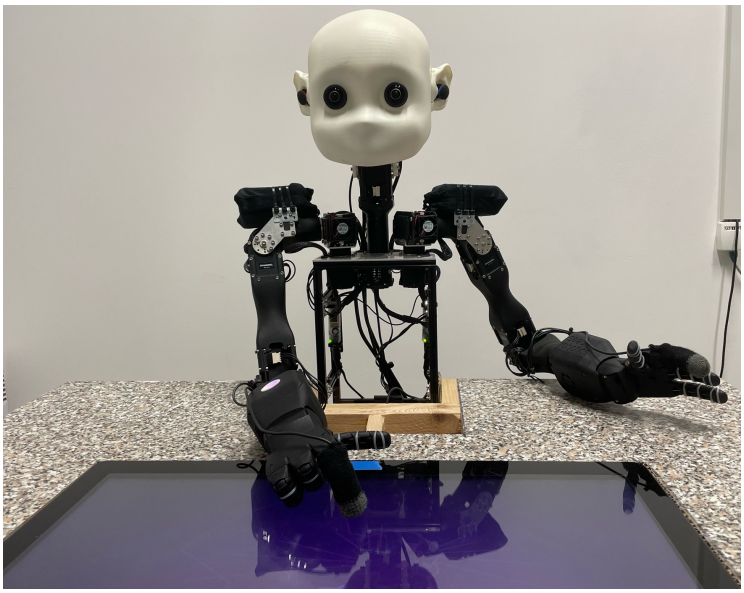
# Strojové učenie - Projekt

Juraj Gavura

27. januára 2025

## 1 Úvod do problematiky

Simulácia je široko využívaná vo svete robotiky vďaka tomu, že umožňuje roboty trénovať lacno, bez potreby prístupu k hardvéru a efektívne, keďže dokáže jednoducho generovať množstvo syntetických dát, ktoré podstatne zrýchlia proces učenia. Z dôvodu nutnej aproximácie a abstrakcie niektorých fyzických javov, však vznikajú nezrovnalosti medzi virtuálnou realitou v simulácii a fyzickou realitou. Tieto nepresnosti nazývame realitná medzera. V tomto projekte sme sa pozreli na prechod zo simulácie do reality pre humanoidného robota NICO, ktorého máme v škole k dispozícii. Tento robot má robotické rameno zakončené ukazovákom, ktorý slúži ako koncový efektor. Pri práci s problémom realitnej medzery, je dôležitým pojmom ground truth, čo predstavuje referenčné údaje, ktoré nám dávajú najpresnejšiu možnú informáciu o reálnom svete. Sú teda kľúčové na porovnanie sveta v simulácii s reálnym svetom. V prípade robota NICO používame ako ground truth dotykový tablet, ktorý leží pred ním. Tento tablet vie zachytávať dotyky koncového efektora a vypísať pozíciu dotyku.



Obr. 1: Humanoidný robot NICO s dotykovým tabletom.

Na obrázku (Obr. 1) môžeme vidieť robota NICO v laboratóriu ako sa dotýka tabletu pred ním. Jednou z možností, ktorými môžeme kvantifikovať realitnú medzeru, je chyba v polohe koncového efektora v karteziánskej sústave, teda v súradniciach $x$, $y$, $z$. V simulácii tieto hodnoty poznáme, vo fyzickej realite môžeme použiť dotykovú obrazovku. Problémom je, že pri prechode nastávajú chyby vo všetkých smeroch, teda aj v $z$-ovej súradnici, výške. Takže bez nejakého trénovania nevieme zaručiť, že sa NICO dotkne ukazovákom tabletu. Kým v simulácii sa dotýka tabletu presne, v realite môže mať prst nad tabletom, alebo priveľmi do ňho tlačiť. Toto je potrebné vyriešiť.

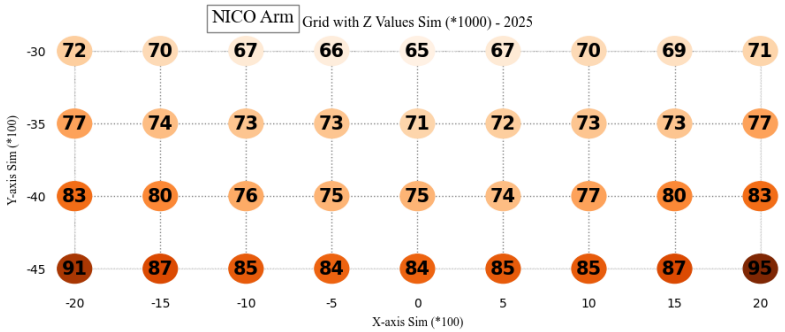
Možným riešením je viesť reálne robotické rameno k tabletu a uložiť si karteziánske súradnice $x$, $y$, $z$ konca ukazováka v simulácii, práve keď sa reálny ukazovák dotýka tabletu. Týmto dostaneme správnu $z$-ovú súradnicu, ktorú potrebujeme robotovi zadať, aby sa dotkol tabletu v bode $x$, $y$. Ak si takýchto bodov uložíme viac a budú dostatočne rozložené po ploche tabletu, môžeme použiť rôzne metódy na aproximovanie $z$-ovej súradnice pre všetky body na tablete.

Tým sa dostávame k formulácii problému pre tento projekt. Máme dáta v podobe bodov v karteziánskej sústave. Teda dvojice súradníc $x$ a $y$ a k nim prislúchajúce $z$. Chceme použiť rôzne metódy strojového učenia na aproximáciu súradnice $z$ pre rôzne body v 2D priestore $x$, $y$. Takže ide o jednoduchý regresný problém.

# 2 Metódy

## 2.1 Dáta

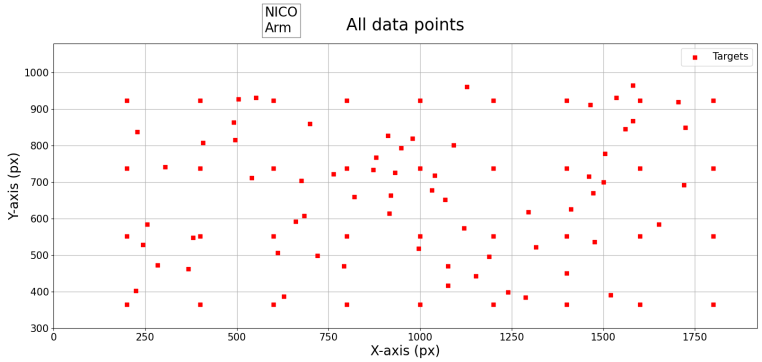
Zbieranie dát pre tento problém si vyžaduje prácu s fyzickým robotom NICO a je pomerne zdĺhavé. Preto sa mi podarilo nazbierať len 100 bodov a riešenia sme prispôsobili k tomuto počtu. Na začiatok sme merali body v pravidelnej mriežke, aby sme dostali približnú informáciu o zmenách $z$-ovej súradnice. Tu sú výsledky:




Obr. 2: Hodnoty $z$-ovej súradnice pre pravidelne rozložené body na tablete.

Môžeme si všimnúť určitý trend zväčšovania sa $z$-ovej suradnice so zväčšujúcou sa vzdialenosťou bodu od základne ramena označenou NICO Arm. V tejto mriežke

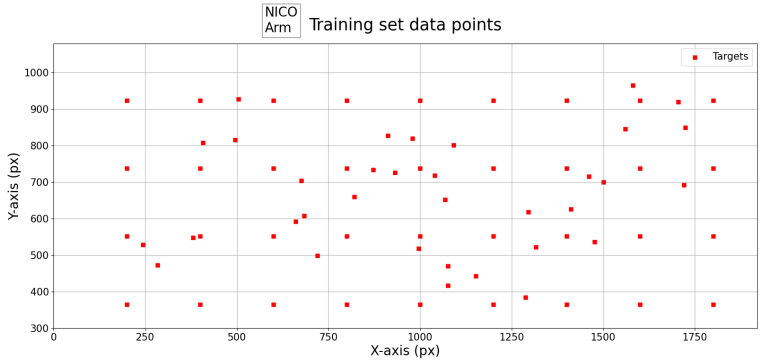
máme dokopy 36 bodov. K tomu sme pridali 64 náhodne rozložených bodov na tablete a získali sme takéto pokrytie:




Obr. 3: Pokrytie dát na dotykovej obrazovke.

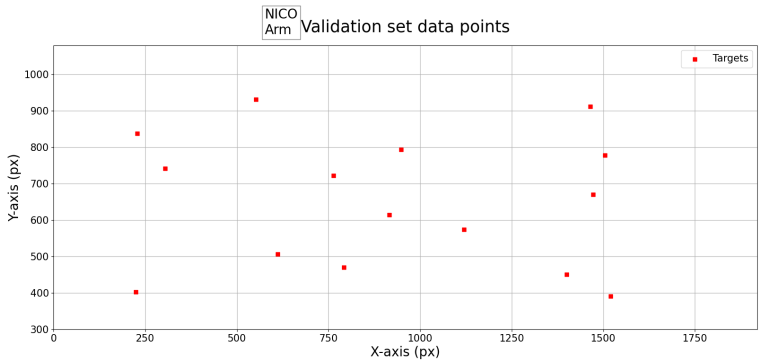
Všimneme si, že tablet má rozlíšenie $1920 \times 1080$ pixelov, no odrezali sme spodných $1920 \times 300$ pixelov, ktoré sú najďalej od robota, pretože do vzdialených rohov tabletu NICO nedočiahne.

Ďalej potrebujeme naše nazbierané dáta vhodne rozdeliť medzi trénovaciu, validačnú a testovaciu množinu. Pre tento problém som si zvolil rozdelenie s pomerom 70 : 15 : 15. Teda trénovacia množina bude mať 70 bodov, validačná 15 a testovacia rovnako 15 bodov. Keďže máme malý počet bodov, tak som tieto body nerozdeľoval náhodne, snažil som sa, aby každá množina mala čo najlepšie pokrytie plochy. Výsledná trénovacia množina vyzerá takto:



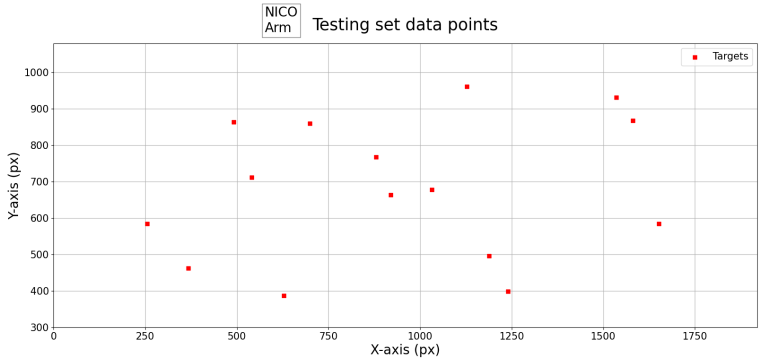

Obr. 4: Pokrytie trénovacích dát na dotykovej obrazovke.

Ďalej naša validačná množina bodov:




Obr. 5: Pokrytie validačných dát na dotykovej obrazovke.

Na záver testovacia množina:



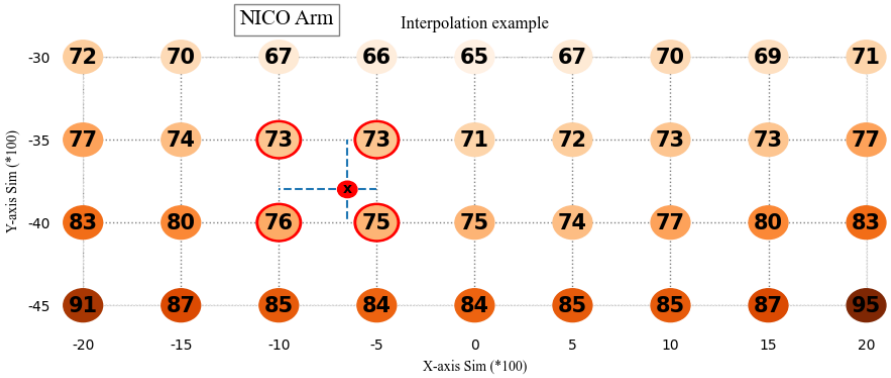

Obr. 6: Pokrytie testovacích dát na dotykovej obrazovke.

Keď sme mali úspešne rozdelené dáta, chýbalo nám ešte spraviť posledný krok predprípravy, a to je normalizácia dát. Pre čo najväčšiu presnosť a efektivitu učenia našich modelov sme si zvolili metódu štandardizácie, teda upravili sme dáta tak, aby mali nulový priemer a jednotkovú smerodajnú odchýlku.

## 2.2 Interpolácia

Ako prvé riešenie nášho problému si ukážeme riešenie interpoláciou. Nejde o metódu strojového učenia, no výsledky z tohto riešenia môžeme použiť na porovnanie iných metód zahŕňajúcich učenie. Toto riešenie pracuje s prvými 36 nazbieranými bodmi, teda s bodmi, ktoré sú pravidelne rozložené po ploche.

Táto metóda funguje tak, že pre akýkoľvek hľadaný bod nájde štyri namerané body z mriežky, ktoré tvoria štvorec, v ktorom sa nachádza hľadaný bod. Podľa týchto štyroch bodov aproximuje $z$-ovú súradnicu hľadaného bodu. Môžeme si to vizualizovať takto:




Obr. 7: Príklad interpolácie pre bod $x$.

Na obrázku (Obr. 7) vidíme hľadaný bod $x$ a štyri vyznačené body, ktoré tvoria štvorec okolo bodu $x$. Postupne aplikujeme lineárne interpolácie a dostaneme výslednú hodnotu $z$-ovej súradnice 74.38. To sedí s obrázkom, keďže bod $x$ je najbližšie k bodu s hodnotou 75.

Ak sa hľadaný bod nachádza mimo nameranej mriežky, tak použije iba najbližšie dva alebo jeden bod.

Takáto interpolácia určite nie je ideálnym riešením pre náš problém, ale slúži ako jednoduchý baseline pre nasledujúce metódy.

## 2.3 Lineárna regresia

Keďže našou úlohou je vyriešiť klasický regresný problém, tak ako prvú metódu z oblasti strojového učenia sme si vybrali lineárnu regresiu. Mohli by sme použiť gradientovú optimalizáciu, no vzhľadom na to, že máme tak malý dataset (100 bodov), tak sme rovno použili uzavretú formu regresie v tvare:

$$\theta = (X^T X)^{-1} X^T y$$

ktorá je presnejšia ako numerické metódy, ale pre veľké datasety časovo náročná, pretože počíta inverz matice $X^T X$, ktorá má rozmer $n \times n$, čo má časovú zložitosť $O(n^3)$. V našom prípade má trénovacia množina spolu s validačnou len 85 bodov, takže je to vhodná metóda pre náš problém.

Uzavretá forma regresie nemá žiadne parametre, ktoré by sme mohli meniť pre dosiahnutie najlepšieho možného výsledku, takže sme v tomto prípade spojili trénovaciu množinu s validačnou. Potom sme pridali do všetkých množín intercept, aby sme vylúčili nežiadúce správanie v prípade, že by mali všetky vstupné premenné hodnotu 0. Spojenú množinu sme hodili do vzorca a dostali sme prvé riešenie metódou strojového učenia pre náš problém.

Tu ale s regresiou nekončíme, dobré riešenie pre našu úlohu si vyžaduje zložitejšie hypotézy. Tie vieme v regresii dosiahnúť pomocou transformácie vstupných at-

ribútov, konkrétne ide o expanziu, takže zvýšenie počtu dimenzií. Tým dostaneme metódu, ktorú nazývame generalizovaná lineárna regresia.

Otázkou je, ako veľmi chceme expandovať vstupné atribúty. Môžeme skúšať rôzne levely transformácie, ale potrebujeme vedieť, ktorá je dobrá. Tu už využijeme validačnú množinu, ktorá nám pomôže detekovať problémy ako underfitting a overfitting. Takže trénovať budeme len na trénovacej množine (70 bodov) a kontrolovať budeme na validačnej množine (15 bodov).

Ďalej potrebujeme vybrať metriky, podľa ktorých budeme kvantifikovať chybu modelu a porovnávať jednotlivé prístupy. Ako hlavnú metriku sme si zvolili strednú kvadratickú chybu (MSE), ktorá je citlivá na veľké chyby a dobre sa s ňou pracuje, takže je vhodná pre regresné modely. Vyzerá takto:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

kde $\hat{y}_i$ sú predikované hodnoty a $y_i$ sú skutočné hodnoty. Pre väčšie množstvo informácií pridáme ešte sekundárnu metriku, strednú absolútnu chybu (MAE), ktorá penalizuje všetky chyby rovnakou váhou, a tak nám poskytne jednoduchú interpretáciu chyby v nezmenenom rozsahu. Má tvar:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Teraz sa môžeme ísť pozrieť na samotné expandovanie atribútov. Naša transformácia bude funkcia v tvare:

$$f : (1, x_1, x_2) \to (1, x_1, x_2, x_1x_2, x_1^2, x_2^2)$$

kde 1 predstavuje intercept, $x_1$ je $x$-ová súradnica bodu a $x_2$ je $y$-ová súradnica bodu. Množinu s takto transformovanými dátami sme vložili do vzorca pre uzavretú formu lineárnej regresie a dostali sme takéto chyby:

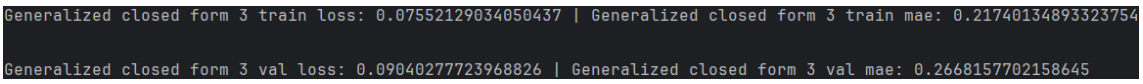
```
Generalized closed form 2 train loss: 0.11187115222428305 | Generalized closed form 2 train mae: 0.27578036381620574

Generalized closed form 2 val loss: 0.12289181847722906 | Generalized closed form 2 val mae: 0.29580628253162305
```

Obr. 8: Generalizovaná uzavretá forma lineárnej regresie (na druhú).

Trénovacia MSE dosiahla 0.11, zatiaľ čo validačná MSE bola 0.12. Čo sa týka MAE, tak pre trénovaciu množinu to bolo 0.28 a pre validačnú množinu dosiahla táto chyba hodnotu 0.30. Vidíme, že MAE dosahuje väčšie hodnoty ako MSE, to je spôsobené tým, že po normalizácii dát sú chyby všeobecne menšie ako 1.

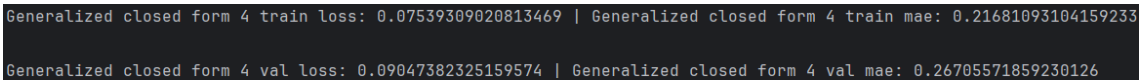
Na porovnanie si skúsime ďalej rozšíriť množinu hypotéz väčšou expanziou vstupných atribútov. Do transformačnej funkcie pridáme $x_1^3$ a $x_2^3$. Dostaneme takéto chyby:




Obr. 9: Generalizovaná uzavretá forma lineárnej regresie (na tretiu).

Všimneme si, že validačná chyba MSE sa zmenšila oproti predošlému modelu, takže sme na správnej ceste. Vidíme ale aj väčší rozdiel medzi validačnými a trénovacími chybami, takže ide o slabý overfitting.

Skúsime ďalej zvýšiť level expanzie, tento krát pridáme aj $x_1^4$ a $x_2^4$. Dostaneme:




Obr. 10: Generalizovaná uzavretá forma lineárnej regresie (na štvrtú).

Tu vidíme veľmi podobné výsledky ako v minulom modeli, dokonca sa validačná chyba o niečo zväčšila. Takže túto expanziu už nechceme.

Najlepšie výsledky mala teda generalizovaná lineárna regresia s transformačnou funkciou:

$$f : (1, x_1, x_2) \to (1, x_1, x_2, x_1x_2, x_1^2, x_2^2, x_1^3, x_2^3)$$

čo je našim ďalším riešením tohto problému pomocou strojového učenia.

## 2.4 Neurónová sieť

Druhou oblasťou strojového učenia, ktorú použijeme pri riešení nášho regresného problému, sú neurónové siete. Sú vhodné pre široké spektrum úloh a majú veľké množstvo parametrov, ktoré je možné meniť pre čo najefektívnejšie riešenie.

Pri hľadaní optimálneho modelu sme sa držali jednoduchej architektúry neurónovej siete. Používali sme sekvenčný model s aktivačnou funkciou relu, ktorá je vhodná pre regresné úlohy. Ako optimalizátor sme zvolili Adam, ktorý je efektívny pri optimalizácii. Samozrejme, vstupná vrstva mala dva neuróny pre dva vstupné atribúty a výstupná vrstva mala jeden pre $z$. Chyba, podľa ktorej sa neurónová sieť optimalizovala, bola MSE a ďalšou metrikou bola MAE. Počas trénovania sme menili počet skrytých vrstiev, počet neurónov v jednotlivých vrstvách, počet epoch a batch size. Prvý základný model vyzeral takto:

```python
# creating a nn model
model = keras.Sequential([
    keras.layers.Input(shape=(2,)),
    keras.layers.Dense(16, activation='relu'),
    keras.layers.Dense(1)
])

# compiling the model
model.compile(optimizer='adam',
              loss='mse',
              metrics=['mae'])

# training the model
history = model.fit(
    x_train_standardized, y_train_standardized,
    validation_data=(x_val_standardized, y_val_standardized),
    epochs=30,
    batch_size=32,
    verbose=1
)
```

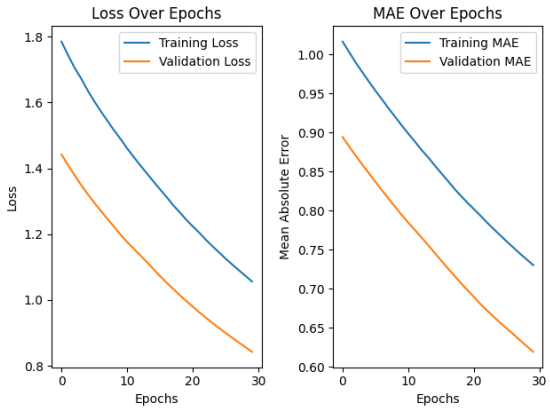
Obr. 11: Prvá architektúra modelu neurónovej siete.

Tento model obsahuje iba jednu skrytú vrstvu so 16 neurónmi a trénovali sme ho 30 epoch s batch size 32. Dostali sme takéto výsledky:

```
Train loss: 1.0427697896957397 | Train mae: 0.7248780727386475

Validation loss: 0.8428727388381958 | Validation mae: 0.6190980672836304
```

Obr. 12: Prvá neurónová sieť (underfitting) - MSE a MAE.




Obr. 13: Prvá neurónová sieť (underfitting) - MSE a MAE za čas.

Z obrázku (Obr. 12) môžeme vyčítať, že trénovacie aj validačné chyby sú veľmi vysoké a v grafoch (Obr. 13) vidíme, že sme zvolili nízky počet epoch, teda mohli sme trénovať dlhšie. Celkovo ide o pomerne silné podučenie, takže treba spraviť zložitejší model a dlhšie ho trénovať.

Zo zaujímavosti sme sa druhý model pokúsili čo najviac preučiť, zvýšili sme počet skrytých vrstiev na 6 s počtom neurónov 256 na každú vrstvu a trénovali sme tento model až 500 epoch s batch size 1, takže sa po jednom spracovával každý jeden bod. Takto vyzerala architektúra:

```python
# creating a nn model
model = keras.Sequential([
    keras.layers.Input(shape=(2,)),
    keras.layers.Dense(256, activation='relu'),
    keras.layers.Dense(256, activation='relu'),
    keras.layers.Dense(256, activation='relu'),
    keras.layers.Dense(256, activation='relu'),
    keras.layers.Dense(256, activation='relu'),
    keras.layers.Dense(256, activation='relu'),
    keras.layers.Dense(1)
])

# compiling the model
model.compile(optimizer='adam',
              loss='mse',
              metrics=['mae'])

# training the model
history = model.fit(
    x_train_standardized, y_train_standardized,
    validation_data=(x_val_standardized, y_val_standardized),
    epochs=500,
    batch_size=1,
    verbose=1
)
```

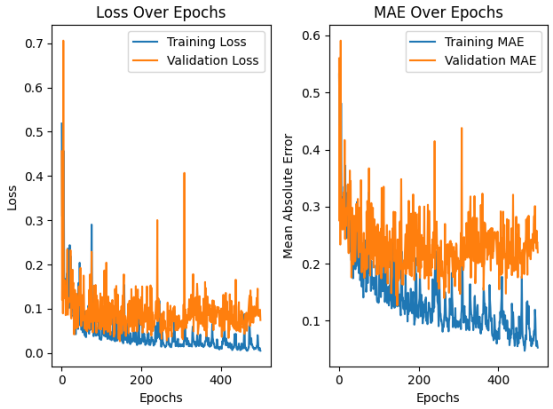
Obr. 14: Druhá architektúra modelu neurónovej siete.

A tu sú výsledné chyby po natrénovaní modelu:

```
Train loss: 0.0022149665746837854 | Train mae: 0.032223958522081375

Validation loss: 0.0743151381611824 | Validation mae: 0.21977652609348297
```

Obr. 15: Druhá neurónová sieť (overfitting) - MSE a MAE.




Obr. 16: Druhá neurónová sieť (overfitting) - MSE a MAE za čas.

Podarilo sa nám dosiahnúť nižšiu validačnú chybu ako pri metóde lineárnej regresie, ale zároveň je rádovo 10 krát vyššia ako trénovacia chyba. To je základný príznak preučenia. V grafoch (Obr. 16) je pekne vidieť, ako trénovacia chyba klesá, zatiaľ čo validačná chyba stúpa. Takže určite potrebujeme spraviť kompromis medzi zložitosťou prvého a druhého modelu.

Pre tretí pokus na architektúru neurónovej siete sme znížili počet skrytých vrstiev na 4 s klesajúcim počtom neurónov. Prvá skrytá vrstva má 128 neurónov, druhá 64,

tretia 32 a posledná 16. Toto do určitej miery brzdí preučenie, pretože núti model vyberať dôležité príznaky a tým generalizovať. Tento model sme trénovali 200 epoch s batch size 16. Architektúra vyzerala takto:

```
# creating a nn model
model = keras.Sequential([
    keras.layers.Input(shape=(2,)),
    keras.layers.Dense(128, activation='relu'),
    keras.layers.Dense(64, activation='relu'),
    keras.layers.Dense(32, activation='relu'),
    keras.layers.Dense(16, activation='relu'),
    keras.layers.Dense(1)
])

# compiling the model
model.compile(optimizer='adam',
              loss='mse',
              metrics=['mae'])

# training the model
history = model.fit(
    x_train_standardized, y_train_standardized,
    validation_data=(x_val_standardized, y_val_standardized),
    epochs=200,
    batch_size=16,
    verbose=1
)
```

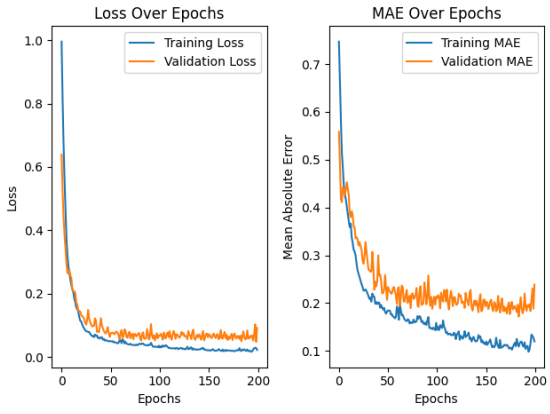
Obr. 17: Tretia architektúra modelu neurónovej siete.

Dostali sme výsledky:

```
Train loss: 0.04462382569909096 | Train mae: 0.15876884758472443

Validation loss: 0.0917629525065422 | Validation mae: 0.23863142728805542
```

Obr. 18: Tretia neurónová sieť - MSE a MAE.




Obr. 19: Tretia neurónová sieť - MSE a MAE za čas.

Dosiahli sme nízku validačnú chybu, a menší rozdiel oproti trénovacej chybe, no stále je približne dvojnásobne väčšia, takže stále ide o mierne preučenie.

Do finálneho modelu sme preto pridali dropout vrstvy, ktoré ďalej redukujú overfitting, a taktiež sme pridali funkciu, ktorá dokáže zastaviť trénovanie modelu

skôr ako uplynie daný počet epoch, ak sa začne validačná chyba zväčšovať. Naša finálna architektúra:

```python
# creating a nn model
model = keras.Sequential([
    keras.layers.Input(shape=(2,)),
    keras.layers.Dense(128, activation='relu'),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(64, activation='relu'),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(32, activation='relu'),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(16, activation='relu'),
    keras.layers.Dense(1)
])

# compiling the model
model.compile(optimizer='adam',
              loss='mse',
              metrics=['mae'])

# setting up early stopping
early_stopping = keras.callbacks.EarlyStopping(monitor='val_loss', patience=30, restore_best_weights=True)

# training the model
history = model.fit(
    x_train_standardized, y_train_standardized,
    validation_data=(x_val_standardized, y_val_standardized),
    epochs=200,
    batch_size=16,
    verbose=1,
    callbacks=[early_stopping]
)
```

Obr. 20: Štvrtá, finálna architektúra modelu neurónovej siete.

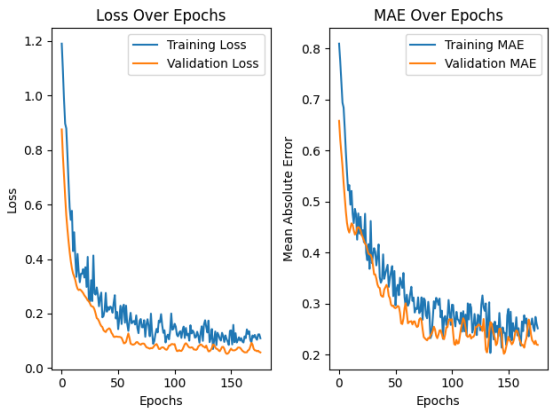
Po 177 epochoch trénovanie skončilo, a tým sa nám podarilo získať takéto výsledky:

```
Epoch 177/200
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - loss: 0.1085 - mae: 0.2464 - val_loss: 0.0578 - val_mae: 0.2189

Train loss: 0.0730404481291771 | Train mae: 0.20570041239261627

Validation loss: 0.052644167095422745 | Validation mae: 0.20140238106250763
```

Obr. 21: Štvrtá, finálna neurónová sieť - MSE a MAE.




Obr. 22: Štvrtá, finálna neurónová sieť - MSE a MAE za čas.

Dosiahli sme menšiu validačnú chybu ako použitím generalizovanej lineárnej regresie, a dokonca je táto validačná chyba menšia ako trénovacia chyba rovnakého modelu, takže sa nám podarilo úplne znegovať overfitting.

# 3 Výsledky

Na samotné testovanie sme si dokopy pripravili 4 rôzne metódy, ktoré teraz porovnáme podľa chyby tvorenej na testovacej množine. Naše metódy sú:

- Interpolačná metóda - baseline
- Lineárna regresia - uzavretá forma
- Generalizovaná lineárna regresia - uzavretá forma
- Neurónová sieť

Začneme s interpolačnou metódou, ktorá je najjednoduchšia z nich, neobsahuje strojové učenie a tak pôsobí ako baseline pre tento projekt. S touto metódou sme získali testovacie chyby:

```
Interpolation loss: 0.0840087722050798 | Interpolation mae: 0.25496333825552375
```

Obr. 23: Testovacia chyba interpolačnej metódy - MSE a MAE.

Prvá metóda s použitím strojového učenia je klasická lineárna regresia v uzavretej forme. S ňou sme získali takéto výsledky:

```
Closed form reg test loss: 0.07094041536427352 | Closed form reg test mae: 0.2000573184713866
```

Obr. 24: Testovacia chyba lineárnej regresie - MSE a MAE.

Už len pri jednoduchej lineárnej regresii bez expandovania parametrov môžeme vidieť zníženie chyby MSE z 0.084 na 0.071. Podobne chyba MAE sa znížila z 0.255 na 0.200.

Ďalej sme sa pokúsili rozšíriť množinu hypotéz transformáciou vstupných atribútov do viac dimenzií, teda použitím generalizovanej lineárnej regresie. Dosiahli sme takéto výsledky:

```
Generalized closed form 3 test loss: 0.06040287026289145 | Generalized closed form 3 test mae: 0.21356322195971122
```

Obr. 25: Testovacia chyba generalizovanej lineárnej regresie - MSE a MAE.

Úspešne sa nám podarilo ďalej zlepšiť testovaciu chybu. Síce chyba MAE sa jemne zvýšila, no znížili sme chybu MSE na 0.060, čo je hlavné.

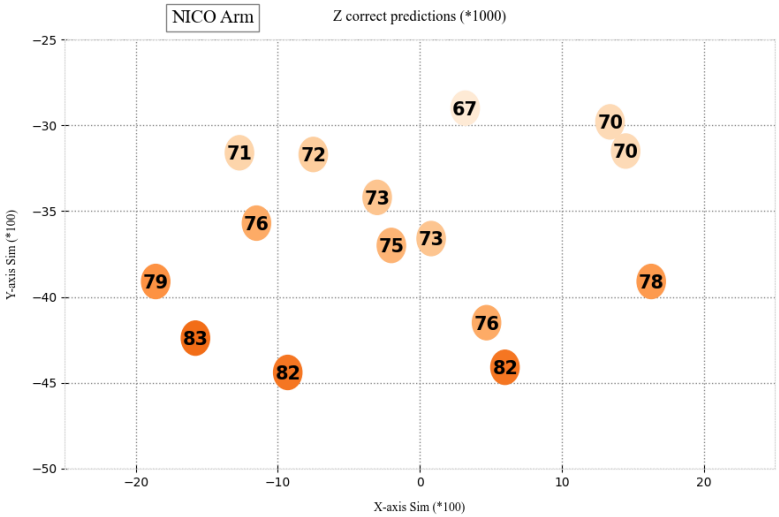
Poslednou a najzložitejšou metódou v tomto projekte bola neurónová sieť so štyrmi skrytými vrstvami. Pozreli sme sa na testovaciu chybu tohto modelu:

```
NN test loss: 0.03883150592446327 | NN test mae: 0.17372801899909973
```

Obr. 26: Testovacia chyba neurónovej siete - MSE a MAE.

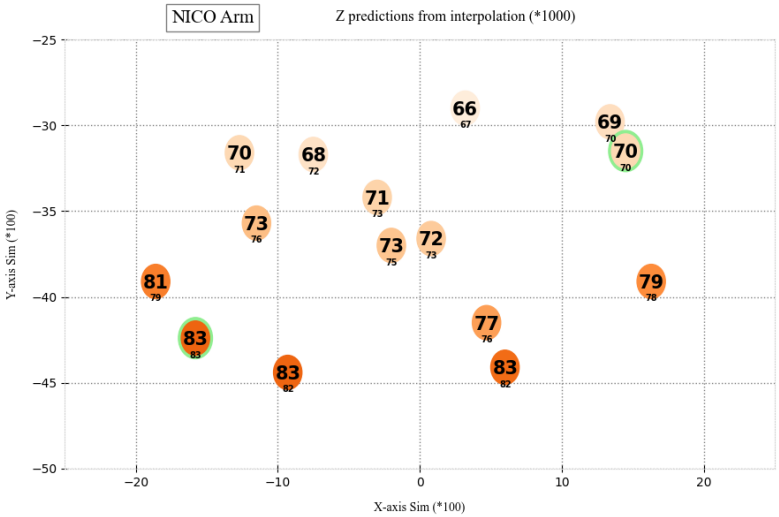
Namerali sme výborných 0.039 MSE a 0.174 MAE. To je ďalšie zlepšenie oproti minulému prístupu generalizovanej lineárnej regresie, a hlavne je to menej ako polovica testovacej chyby interpolačnej metódy. Takže môžeme určite povedať, že sme úspešne dvojnásobne zlepšili riešenie definovaného problému.

Čísla nám dávajú dobrú kvantifikáciu chybovosti a presnosti našich metód, ale na záver by sme si chceli ešte vizualizovať, ako reálne tieto metódy aproximujú hľadanú $z$-ovú súradnicu. To sme spravili vykreslením výsledných predikcií testovacej množiny a porovnaním so správnymi predikciami, teda hodnoty $z$ v testovacej množine. Takto vyzerajú správne predikcie:




Obr. 27: Správne predikcie $z$.

Teraz sa pozrieme na predikcie vytvorené našim baselinom, teda interpolačnou metódou:

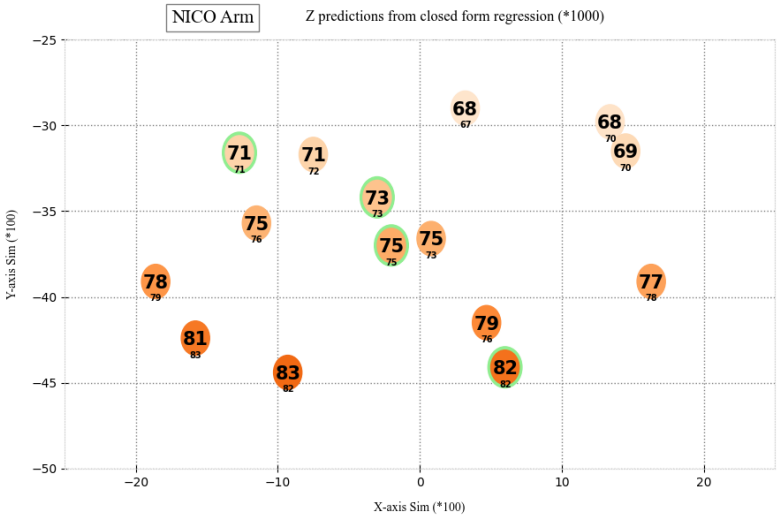



Obr. 28: Predikcie $z$ interpolačnou metódou.

Vidíme, že skoro všetky predikcie sa líšia. Našli sme iba dve, ktoré táto metóda

predikovala správne, sú vyznačené zeleným okrajom kruhu. Maximálna chyba tejto množiny je 3.

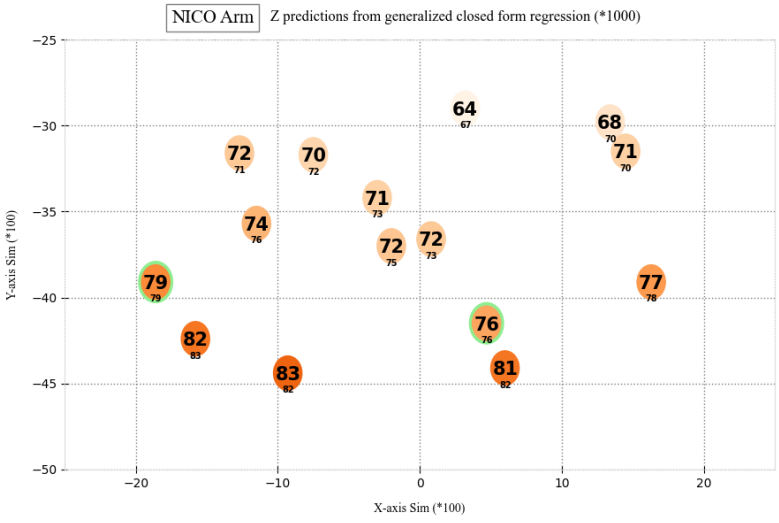
Pozrime sa ďalej na prvú metódu za použitia strojového učenia, uzavretú formu lineárnej regresie. Tu sú vizualizované predikcie:




Obr. 29: Predikcie $z$ lineárnou regresiou.

Táto metóda aproximovala správne štyri hodnoty $z$ a maximálna chyba je iba 2, čo je výrazné zlepšenie oproti interpolácii.

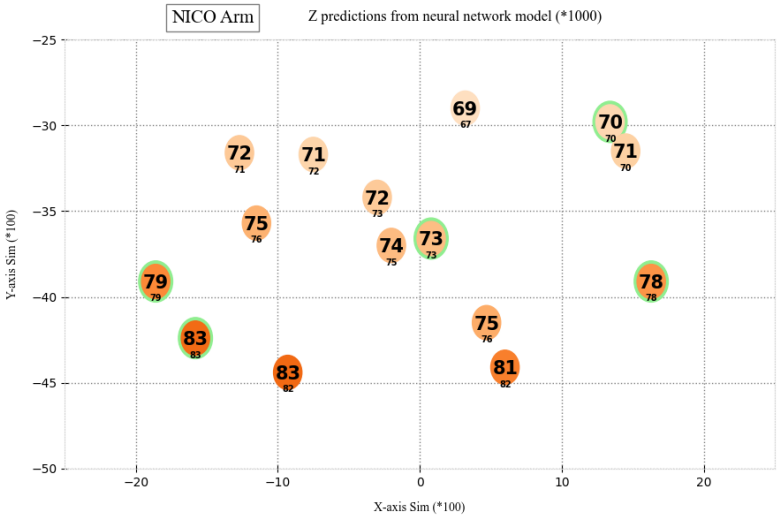
Pozrime sa teraz na generalizovanú lineárnu regresiu, ktorá je zložitejšia ako obyčajná lineárna regresia v predošlom obrázku. Tu sú vizualizované predikcie:




Obr. 30: Predikcie $z$ generalizovanou lineárnou regresiou.

Môžeme si všimnúť, že generalizovaná regresia správne aproximovala iba dva body, čo je horšie ako klasická lineárna regresia aj napriek tomu, že testovacia chyba generalizovanej regresie bola menšia. Je to asi spôsobené zaokrúhlením hodnôt. Dokonca maximálna chyba sa vrátila späť na 3.

Na záver sa pozrime na predikcie vytvorené neurónovou sieťou. Tu sú vizualizované:




Obr. 31: Predikcie $z$ neurónovou sieťou.

Vidíme, že táto metóda je nazoaj najlepším riešením daného problému. Správne aproximovala päť bodov a chyba v ostatných bodoch je väčšinou 1 okrem jedného bodu, v ktorom je chyba 2.

# 4 Záver

Uplatnili sme mnoho poznatkov zo strojového učenia. Vytvorili a porovnali sme viacej metód na riešenie nášho regresného problému. Podarilo sa nám znížiť chybu predošlého riešenia interpoláciou na polovicu pomocou použitia neurónovej siete. Tým sme výrazne zlepšili presnosť robota NICO pri práci s dotykovou obrazovkou, takže tento projekt môžeme považovať úspešný. Pre ďalšiu optimalizáciu by bolo možné nazbierať väčšie množstvo trénovacích dát a možno lepšie doladiť parametre neurónovej siete.