Dataset preparation and neural network models development

Aleksandr Barinov

***Abstract*— the goal of this project was to prepare a reasonable dataset for training a machine learning model predicting robot pose given heightmap and additional features. In order to get familiar with Robot Operating System (ROS) and neural networks the observations were collected from a simple robot at first. As a result of the work dataset consisting heightmaps and corresponding labels was produced and an attempt to train a CNN model was made. The predictions of a model were quite good and the switch to the robot with flippers had taken place. The corresponding dataset was prepared as well, but due to misunderstanding of the task by the author the training of a model was not successful this time. Despite the failure at the last part of the project, the main goal, i.e. to create a dataset for training, was achieved, also student got familiar with the ROS and with creating and training neural networks using PyTorch which was the true underlying purpose of this project.**

I. Assignment

Provided with a bag file in ROS extract relevant data and create a data set for a purpose of training a machine learning model. Subsequently create a neural network model able to predict chosen parameters of a state of a robot based on the data set prepared in the previous step.

II. Introduction

The process of gathering data, selecting valuable features and creating the final dataset for training machine learning models could not be neglected. Due to noisy data, wrong choice of features, low variety of samples contained in the final dataset the model trained on these data could be arbitrarily bad not because of wrong chosen model or badly tuned parameters, but simply because it is impossible to learn reasonably well on bad data.

The main contribution of this project lies in computing data sets for training machine learning models for the purpose of predicting parameters of the robot position on the terrain.

III. Methodology

For the purpose of training neural networks height maps of the terrain traversed by the robot had to be computed. For the data from robot without flippers a naive approach of computing height maps was chosen due to the relatively sparse point cloud from robot sensors and not a huge amount of data. A naive approach is to simply compute the mean of all points in the chosen area and assign it as a height of the tile of the final grid. In case of missing data the height of the closest tile was assigned in order to avoid problems with NaN values during model training. The example of a point cloud could be found on the Fig.1 with a corresponding computed heightmap on the Fig.2. The square cut from the point cloud was chosen in a way that the x axis of a heightmap correspond to the x axis of the robot, i.e. yaw of the robot is zero and robot is centered in the square. Therefore, when predicting the robot position we only need to predict its pitch and roll.

Necessary theory on coordinate frames transformations could be found in [1].

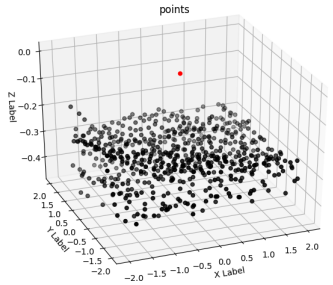

Fig. 1. An example of a point cloud. Red dot is the robot center

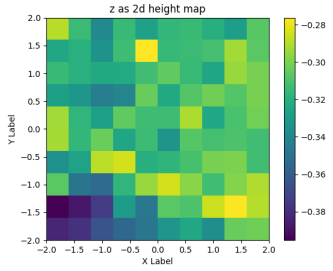

Fig. 2. An example of a heightmap

The convolutional network that was developed for the learning parameters of robot position from the heightmap could be represented as a function $f_w : H -> Y$, where w is a vector of network parameters, $h \in H$ is a heightmap as an input and $y \in Y$ is a vector of predicted values of roll and pitch of the robot.

The theory on convolutional neural networks could be found in [2] and [3].

IV. Experiments

Robot without flippers:

Data set preparation:

At the first stage of the project in order to simplify learning process for the author the observations of a simple robot without flippers were chosen for the task.

The process of creating the dataset was divided into to parts:

- retrieval of the desired data from the bag file
- computation of the heightmap

The retrieval of the desired data here means reading the point clouds, cutting it as stated in the section III and storing it with the corresponding labels that are the parameters of the robot position.

For each task corresponding python script was developed. Scripts could be found in the 'map_cuts' package.

Neural network model development:

The goal of this part was rather not to develop a serious model, but to first of all familiarize the author with the PyTorch and secondly, in order to estimate if the dataset was any good, i.e. if a machine learning model could be trained well enough on these data.

The convolutional neural network with two hidden convolutional layers, two fully connected linear layers at the end, LeakyRelu as an activation function and mean squared error as a loss function was chosen for the purpose of training on the dataset. The implementation could be found in the corresponding Jupiter Notebook document that is a part of submission including this report.

Considering the goal there was no need in cross validation, therefore, for simplicity, dataset was divided into train and test subsets and also completely different dataset was used in order to estimate the final model. The results are presented in the Table.I.

As we can see model does not suffer from overfitting and on average error of the estimation of the roll is around 33% of the ground true value and for the pitch it is around 22%. This was enough to understand that the prepared dataset could be used for the purpose of training neural networks.

Robot with flippers:

Data set preparation:

This part for the robot with flippers was pretty much the same as for the robot without flippers with few exceptions.

New python script (flippers_extract_data.py) was developed for the goal of the data retrieval from the bag file. Its purpose is to read the latest available point cloud, store it and then every 0.2 seconds read a position of the robot (x,y,z,roll,pitch,yaw) and flippers states (angles and velocities of 4 flippers), create a cut from point cloud as described in the section III and then save the data. Consequentially the height map is computed and stored along with the data mentioned above (add_heightmap.py script is responsible for it).

Neural network model development:

This is the point when the misunderstanding by the author of the task to be completed has taken place. The true goal was to create and train a CNN model that would take as an input a heightmap and a state of flippers (angles, velocities) as additional features in order to predict roll and pitch of a robot. The goal author was pursuing was a creation and training of a CNN model that would take as an input a heightmap and predict roll, pitch of a robot along with flippers position (angles).

The structure of the CNN was the same as for the robot without flippers except of 6 features on the output instead of 2. Implementation could be found in the Jupiter Notebook 'flippers_nn.ipynb'.

The results achieved by the model are contained in the Table.II. It could be seen immediately that trained model fails awfully when tested on the unseen data. With the knowledge of how data on which dataset for training was computed were gathered we could understand the results better. Bag file was recorded for a robot run in a simulation on a flat surface with random movements of flippers that did not correspond to the terrain. Therefore it was not actually possible to learn flippers positions based only on the heightmaps.

But the huge error in predictions of pitch and roll of a robot is not following from the previously mentioned problem with data. This error could be explained as a consequence of the changes in a robot relative to the terrain position in a vertical dimension (z axis). For the robot without flippers we assumed, that the robot z coordinate stay the same relative to the terrain, i.e. robot can't move in a vertical direction on a flat terrain. But for robot with flippers it could not be assumed, because due to the possibility to rise on flippers the z coordinate could vary significantly.

V. Discussion

In case of robot without flippers computed data set was proven to be suitable for training machine learning models with a goal of predicting roll and pitch of a robot.

Despite the fact there was no success in training decent CNN model for the robot with flippers, what should have been the point of transition to even more complicated tasks, the flaw was not in the data set prepared for training. The actual problem was that this data set was used for the purpose it could not serve as described in section IV.

It could not be stated though, that the data set computed on the observations from the robot with flippers is correct and could be used for the task of training ML model predicting roll and pitch of the robot. Farther elaboration is necessary to prove this. But because of manual control of the samples from the computed data set the author could state that it is unlikely that the data set is flawed.

VI. Conclusion

The goal of extracting relevant features from the robot observations is not trivial and should be approached with a

TABLE I

Results [roll, pitch] in degrees

dataset	errors mean	values mean
Train subset	[1.33 1.17]	[3.69 6.52]
Test subset	[1.31 1.25]	[3.79 6.67]
Different dataset	[1.59 1.59]	[4.53 6.44]

TABLE II

Results [roll, pitch, flipper1, flipper2, flipper3, flipper4] in degrees

dataset	errors mean	values mean
Train subset	[0.25 3.48 10.12 10.08 5.58 5.59]	[0.41 11.58 53.44 53.43 34.02 34.04]
Test subset	[0.34 10.38 30.45 30.49 18.82 18.78]	[0.32 11.96 41.48 41.48 29.49 29.51]

good understanding of how the data were collected and for which purpose the dataset to be computed based on these observations will be used.

It is also crucial to test the computed dataset before the start of the process of developing complicated machine learning model. If dataset fails to serve the purpose of training simple model due to some flaws in computations, wrongly chosen features or unnoticed errors during the dataset preparation there is no point in spending time on developing the final model, because the results of training could be arbitrarily bad due to flaws in the dataset, as was shown in the section IV.

REFERENCES

[1] Karel Zimmermann.: *Robot, lidar, Eulidean tranformations*, Department for Cybernetics Faculty of Electrical Engineering Czech Technical University in Prague.

[2] Karel Zimmermann.: *Learning for vision III. Convolutional networks*, Department for Cybernetics Faculty of Electrical Engineering Czech Technical University in Prague.

[3] Petr Posik.: *Deep Learning*, Czech Technical University in Prague, Faculty of Electrical Engineering, Dept. of Cybernetics.