Initial Pose Estimation using Wi-Fi Signals on a known Map

Motivation

When running the car in autonomous mode the user must set the initial pose using the “2D Pose Estimate” button in RVIZ (Fig. 1). The goal of this project is to render this step redundant by automatically estimating the initial position of the car. This will be done by guessing the approximate area in which the car is positioned based on Wi-Fi signals and spreading the amcl particles throughout that area. The exact location will then be determined using the laser scanner and the particle filter that amcl provides.

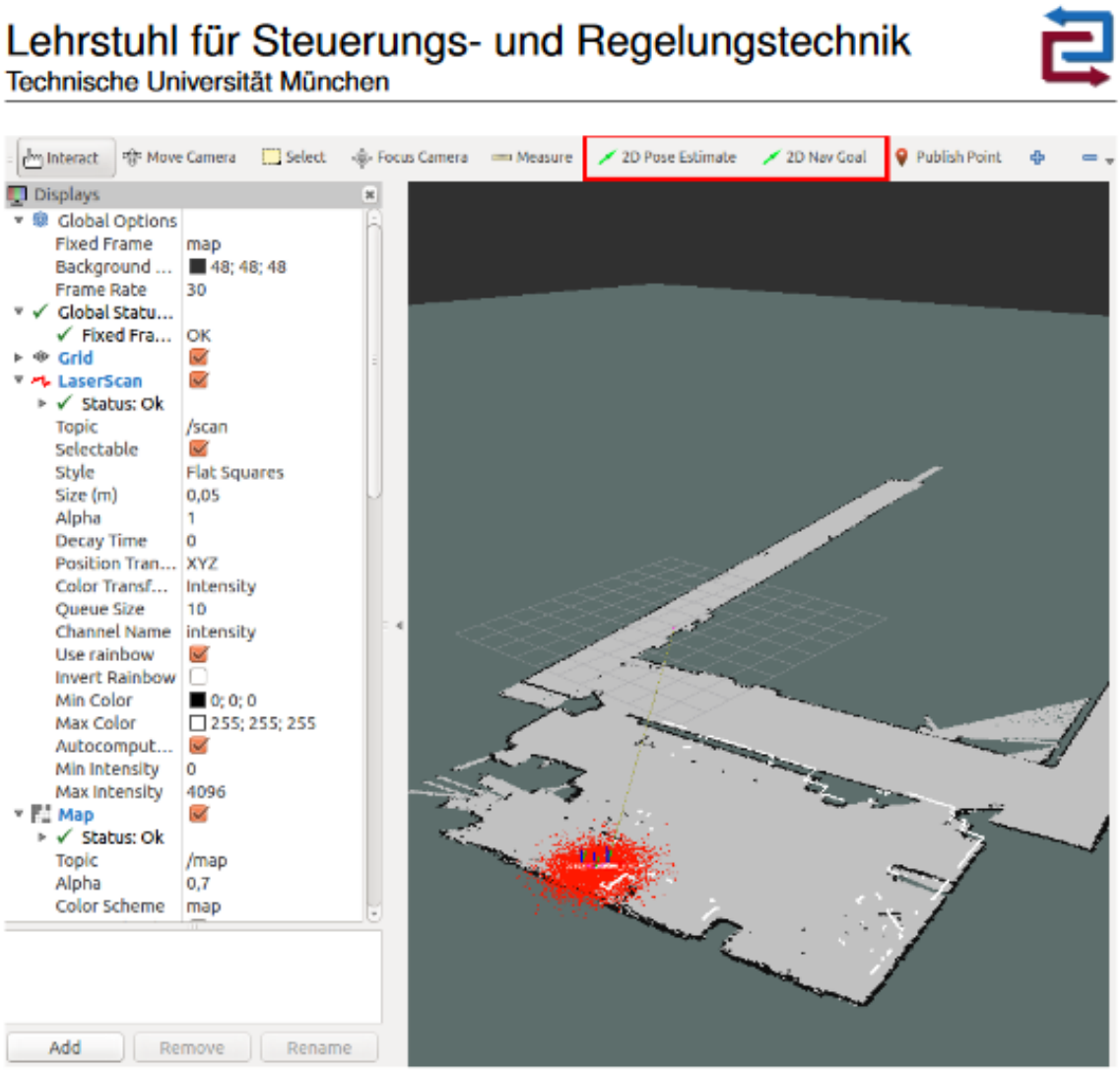


Fig. 1: RVIZ with 2D Pose Estimate Button and the amcl particle cloud.

The Method

For localization without a 2D Pose Estimate, amcl provides a service “global_localization”, wherein all particles are dispersed randomly through the free space in the map (Fig. 2.1). After a little movement of the robot the particles are updated and quickly collapse onto a local minimum. (Fig. 2.2) While this sometimes matches the actual position of the robot, usually this leads to incorrect results. To improve this behavior, I choose not to disperse the particles throughout the whole map but only in a selected area in which the robot is actually located. Then the following parameters will be set to zero in order to update the particle filter without having to manually move the robot:

- `update_min_d` (`double`, default: 0.2 meters)
  Translational movement required before performing a filter update.

- `update_min_a` (`double`, default: π/6.0 radians)
  Rotational movement required before performing a filter update.

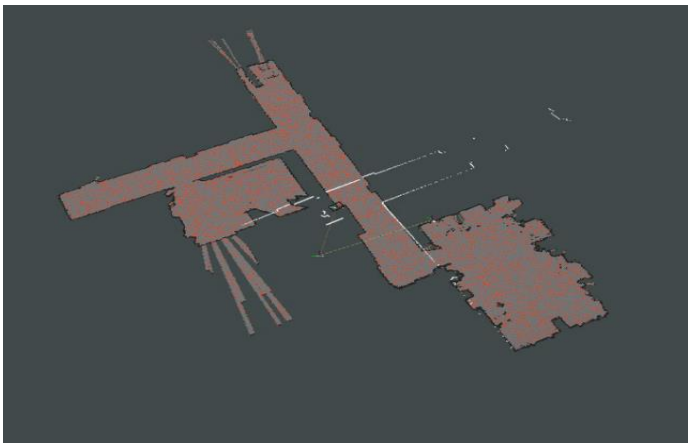


Fig 2.1 (left): The particle cloud is dispersed randomly through the free space in the map.

Fig 2.2 (right): After updating the particle filter the particles have collapsed onto a local minimum. The actual pose of the robot is marked with a green arrow. One can see the crossing in the shape of the laser scanner (white lines).

The approximate location of the robot will be determined based on Wi-Fi signal strength.

The map was separated in seven different areas. (Fig. 3) Based on the received Wi-Fi signals and their strength a classification will be conducted, and the particle cloud dispersed in that area.

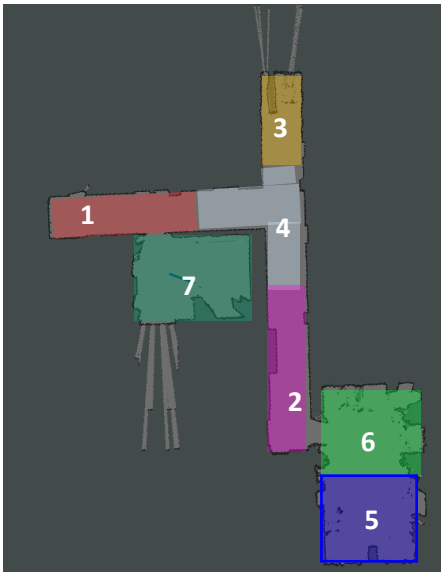


Fig 3: The map was separated in seven areas.

Classification

The data

Wi-Fi signal strength is measured in **dBm** (decibel-milliwatts). This is a logarithmic unit therefore a 10dB increase in level corresponds to a 10-fold increase in signal strength. Furthermore, measurements in **dBm** will display in negative numbers in a scale that runs from -30 dBm to -90 dBm. A value of -30 dBm indicates a perfect connection (usually when you are located directly next to the router, whereas a value of -90 dBm indicates that the service is so weak that no stable / usable connection can be established.

In each of the seven areas hundreds of measurement sample where collected on two different days for a total of over 6.500 samples. The data was then processed so that it is displayed in positive value with a value of 70 for a perfect signal and a value of 10 for a very weak signal. A value of zero stands for no connection.

When plotting the mean values of all samples in the different collection you can get something like a Wi-Fi-fingerprint for the area. In these plots you can quickly see which areas are easier to distinguish and which require more effort. For example Area 5 (dark blue) and Area 1 (red) are located on opposite ends of the map show very different signals. Areas 5 and 6 however are located in the same room and hard to distinguish. In Area 5 however, the signals from other rooms are weaker, since the door is in Area 6, but the router in that room is on a wall in Area 5, therefore these signals are slightly stronger.

Please keep in mind that the plots in Figure 4 are mean values. Therefore, a value below 10 means that sometimes you receive a weak signal whereas at other times no signal is received at all.

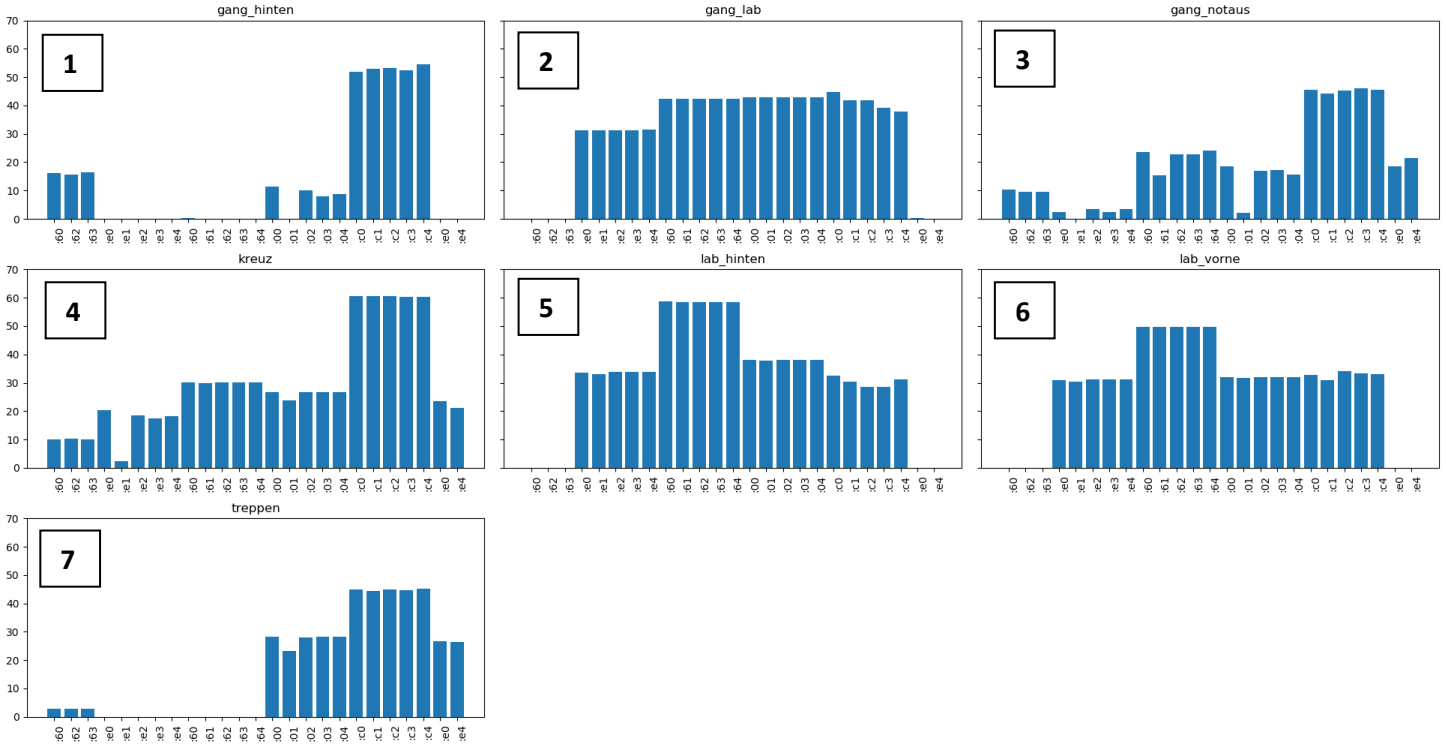


Fig 4: Mean values of the samples collected in the seven areas of the map. A higher value means a stronger signal whereas a value of 0 means no signal at all.

Preprocessing

For the following techniques some preprocessing is necessary. The first step was already mentioned: The data was made positive s.t. a higher value represents a stronger signal and a value of zero shows that no signal was received at all.

Furthermore, the data gets standardized by removing the mean and scaling to unit variance. The standard score of a sample x is calculated as:

$$z = \frac{x - u}{s}$$

where u is the mean of the training samples and s is the standard deviation of the training samples.

This step is necessary since many machine learning estimators might behave badly if the individual features do not more or less look like standard normally distributed data. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

The data was then randomly split into training set (90%) and test set (10%).

K-Nearest Neighbors

The principle behind nearest neighbor methods is to find a predefined number of training samples (in our case: three) closest in distance to the new point and predict the label from these. A pure KNN resulted in a 93% - 97% success rate, depending on which samples where in the test set (different seeds for the pseudo random number generator where tried). A change in the number of neighbors did not result in a measurable performance gain.

Principal Component Analysis

A dimensionality reduction via PCA to 5 dimensions increased the performance of the KNN algorithm significantly. It now returns a success rate of 95.5% - 98.5%. I tried reductions to one dimension up to 25 dimensions and 5 dimensions consistently returned the best result. If I had to guess I would say that these 5 dimensions correspond to the five routers that distribute these 26 Wi-Fi signals that I used.

Multi-Layer Perceptron Classifier

A multi-layer perceptron is a class of feedforward artificial neural network. An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Through trial and error, I conducted that two hidden layers with (160, 60) nodes and **L-BFGS** as solver returned the best ratio between success-rate, consistency and effort. Depending on the sample set the MLP Classifier performs slightly better or slightly worse than the PCA + KNN. Since it also requires more effort to train the Network and more knowledge to efficiently and correctly tune the hyperparameters I choose to implement the PCA + KNN method in the final node.

However, I am certain that with more time to tune the parameters and some experience this method can be further improved and help with some of the challenged below.

Challenges

This project is still facing some challenges and requires some more work to consistently return the best results. While in testing the algorithms consistently return success rates of more than 95%. The actual error when applied in practice seems to be higher. This is due to the fact, that the Wi-Fi signal strength changes throughout the day. I assume that this is due to interferences from other devices and local hotspots that are not always present. Figure 5 shows a comparison between the Wi-Fi fingerprints of the locations on 2 different days.

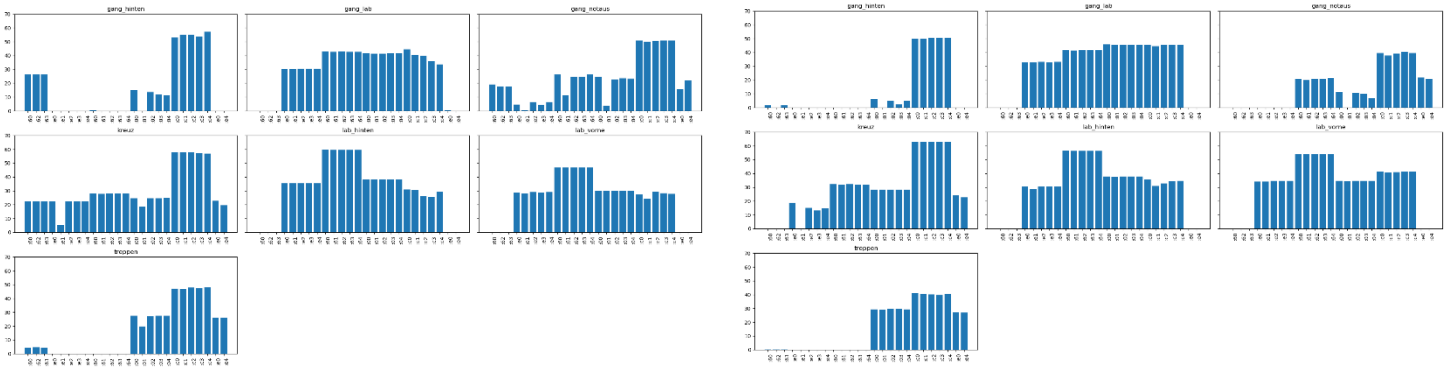
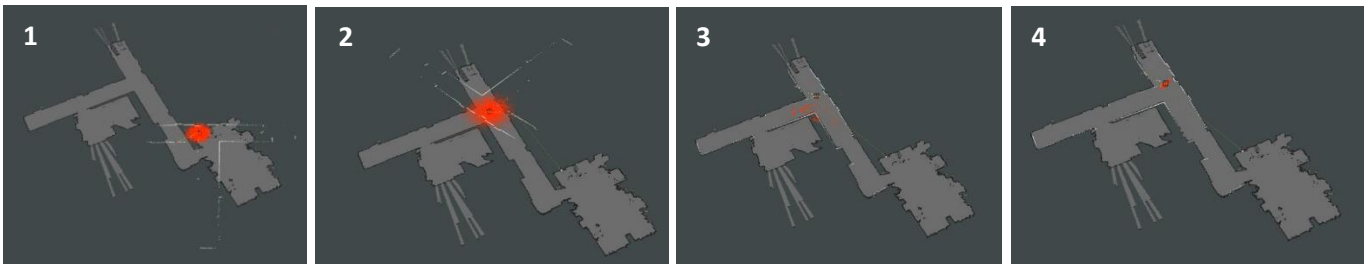


Fig 5: The Wi-Fi fingerprint (mean values of the samples) on two different days. While the areas are still recognizable there are quite some changes (possibly due to interferences).

To improve this behavior, I would recommend collecting more samples on different days, so that the algorithms learn to know the changes that can happen. I also have the impression that the MLP Classifier performs better in practice than the other two methods, but I lacked the time to provide evidence for this assumption.

Another problem is that, the closer you get to the borders of the areas it gets harder to distinguish between the areas. To solve this one could take into consideration the uncertainty of the classification to modulate the distribution of the particles.

In Pictures



1. The position of the robot is unknown. The robot collects a sample of all available Wi-Fi Signals and the corresponding Signal Strength.
2. A classification algorithm determines the approximate location of the robot based on the collected sample and its database. Then the particle cloud gets distributed in the determined area.
3. The parameters `update_min_d` and `update_min_a` are set to zero so that amcl can update the particle cloud without movement of the robot. The particles start to collapse.
4. The particle cloud has collapsed onto a minimum which should accurately represent the robot's position.