Informatica & Economie

Refinement of GPS

with GPS beaconing for small-scale environments

Odens van der Zwan

Supervisors:
Dr. Mitra Baratchi
Ir. Richard M.K. van Dijk

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl
18/08/2022

Abstract

The Global Positioning System (GPS) is used for location determination in large-scale environments. This includes navigation of cars and other mobile vehicles. More recently is the application of GPS in small-scale environments. In small-scale environments, GPS is, for example, used in sports, health care, location supportive services, and agriculture. However, for GPS to be used in these small-scale environments like a sports field, the accuracy has to be very high, with an accuracy of around 1 meter. This study aims to design an algorithm that corrects GPS coordinates in small-scaled environments. In the study, data were collected in four different locations to measure the influence of these environments on the accuracy of GPS. GPS coordinates are collected using smartwatches. An algorithm computes the corrections for the GPS coordinates by use of smartwatches as GPS beacons that are located continuously in a predetermined position. These beacons are identical smartwatches, as the watch that is moving and needs correction. Because the locations of the beacons are known, we can calculate the error of each beacon for every moment in time. These errors are used as corrections applied to the trajectory a moving watch followed to gain a more accurate positioning. In this study, two different algorithms are developed: the first algorithm calculates the distance to each beacon between the moving smartwatch. It corrects the followed trajectory based on the nearest beacon's error. The second algorithm is a learning algorithm that predicts the corrected location based on all beacon errors and the GPS coordinates of the moving smartwatch. This method used an Ultra Wide Band (UWB) system as ground truth. Therefore it was possible to compare the measured GPS with the UWB and find the best possible correction. This method led to a low mean average percentage error, and low root mean squared error computed from a straight-line trajectory between the corrected trajectory of the moving smartwatch and the ground truth measured with a UWB-tag. We claim carefully that with a learning algorithm, there is an indication that it is possible to refine the positioning of moving GPS receivers based on GPS beacons of which the GPS location is precisely known.

Acknowledgements

First, I would like to give special thanks to my supervisors, Richard van Dijk (daily guidance) and Mitra Baratchi from LIACS, for giving feedback and suggestions and therefore contributing to this study. Next, I would like to thank Rodi Laanen for helping with the UWB experiments and discussing different ideas about the algorithms that are developed. Finally, I want to thank Sportclub KMD at Wateringen for opening their soccer fields we used for the GPS and UWB data collections.

Contents

1 Introduction & Background **6**
1.1 GPS system . . . 6
1.2 Research questions and scope . . . 9
1.3 Definitions of terms . . . 9
1.4 Thesis overview . . . 9

2 Related work **10**
2.1 Beacon based correction methods . . . 10
2.1.1 Outdoor corrections . . . 10
2.1.2 Indoor corrections . . . 11
2.2 Map matching . . . 11
2.3 Multipath correction . . . 11
2.4 Kalman filtering . . . 12
2.5 Developed method . . . 12

3 Methodology **13**
3.1 Materials used . . . 13
3.1.1 GPS data . . . 14
3.1.2 UWB data . . . 15
3.2 Data processing . . . 15
3.3 Algorithms . . . 16
3.3.1 Algorithm 1 . . . 16
3.3.2 Algorithm 2 . . . 17
3.4 Merge UWB and GPS data . . . 19
3.5 Trajectory accuracy scores . . . 19

4 Experiments **21**
4.1 Locations . . . 21
4.1.1 Location 1 . . . 21
4.1.2 Location 2 . . . 21
4.1.3 Location 3 . . . 21
4.1.4 Location 4 . . . 23
4.2 Experimental setup . . . 23
4.3 Experiments . . . 23
4.3.1 Experiments with UWB . . . 24

5 Results **25**
5.1 RQ1 and RQ4: accuracy analysis locations . . . 25
5.2 RQ2: beacon behavior experiments . . . 28
5.3 RQ3: analysis beacon errors . . . 29
5.4 RQ4: corrections with algorithm 1 . . . 32
5.5 RQ5: corrections with learning algorithm 2. . . . 33
5.6 RQ6: corrections with learning algorithm 2 without beacon data . . . 36

6 Conclusions and Further Research **39**
6.1 Conclusion 39
6.2 Further research 39

References **42**

1 Introduction & Background

Over the last decades, the Global Positioning System (GPS) applicability evolved from only space and military to civilian use such as navigation with smartphones and GPS receivers build in cars. This broad introduction in different sectors made it low-cost, and lightweight [gps20]. Since a number of years now, GPS is used for localization and tracking in small-scale environments for different applications: areas of around 500 m^2. One of the many problems in these small areas is the too low accuracy of GPS, which is usually around 4 to 10 meters.
An application where high accurate GPS is needed can be found in the domain of agriculture. The so-called satellite farming [far21] is used to compute a very accurate location for the tractor. This way, the minimum distance between seeds can be calculated, and the number of seeds used in a field can be maximized.
A second application can be found in sports. In professional sports, every small detail can give you an advantage compared with your opponent. GPS can give such an advantage. In many outdoor sports, GPS sensors are used to track motion. GPS is used to determine position, but we can calculate speed and distance traveled by having the position. Taylor [Tay17] gathered data about the game intensity of players to make a better estimation for recovery time.
A third application is found in the observation of the ground vegetation in mountain bike trails [SK17]. Here GPS tracks of smartphones are used to find frequently used bike trails. Small-scale mapping confirmed that widely used trails, widen the path trail.
The next subsection will explain the factors of current GPS which causes its low accuracy, since we need to understand how GPS works to improve its accuracy.

1.1 GPS system

This section is divided into two parts: the first part explains how GPS works, and the second part will talk about the flaws of GPS.
The Global Positioning System, commonly known as GPS, is a satellite-based navigation system that can determine the position by sending out signals to a receiver. The receiver must be in the line of sight of at least three or four different satellites before it can compute a position. Three of these satellites are used for the location, and one is used to correct the receiver's clock error. The satellites orbit in an accurate path around the earth.
Several countries launched Global Positioning Systems. The European Union has Galileo, which went live in 2016. The Chinese government has launched Beidou as their GPS system, and many more systems are now active. All of these systems have at least 24 satellites. These satellites work together to establish an exact position on the ground. A basic GPS system contains three elements. The first element are the satellites. These hover around the globe in an exact orbit. At least three different satellites are needed to compute a location. Figure 1 shows that these satellites are needed to calculate three distances to the GPS receiver. Three different colored circles show the three distances. The area surrounded around the intersections of the circles corresponds to the required position.
The second element is the Ground Control or Control Segment responsible for monitoring the satellites, their orbit routine, and their clock. When an error occurs in a satellite clock, Ground Control will send a time-correction message to the satellite to correct its time.
The third element are the GPS receivers which process GPS signals from several satellites (also

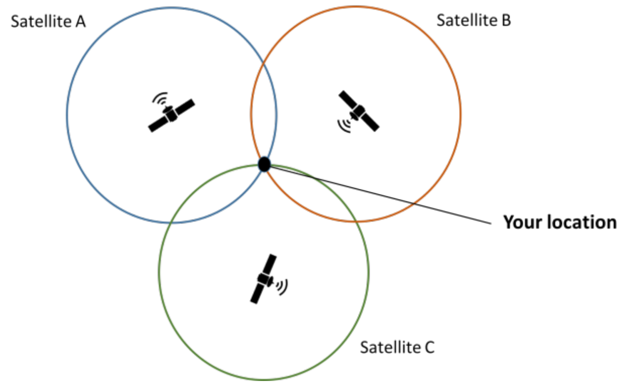

Figure 1: GPS works with at least three satellites to compute a position [NT09].

from different types of GPS systems). Nowadays, watches, cars, smartphones, and many more are equipped with these receivers. In this study, the Samsung Gear Fit 2 Pro is used as the receiver, more on this device in section 3.
Some errors can occur when calculating a position with a GPS. Below are three different types of errors: atmospheric events, errors in the satellite/ receiver, and the multi-path error.

Atmospheric errors is divided into two effects: the tropospheric effect and the ionospheric effect. The troposphere is the layer that is closest to earth. The troposphere is refractive for electromagnetic wave fields. The refraction of a GPS satellite's signal is independent to its frequency. The refraction results in a delay in the arrival of a GPS satellite's signal. It can also be seen as an additional distance to the true distance between receiver and the satellite [SD]. The refraction of the signal is shown in the figure below [MRU22].

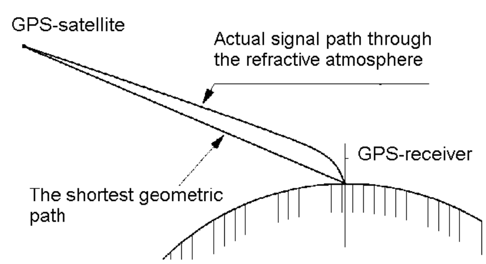

Figure 2: GPS signal gets refracted in the troposphere [MRU22]

Next is the ionospheric effect. The ionosphere is the layer above the troposphere. A GPS signal travels through the ionosphere and gets delayed due to free ions. These ions are created by the ultraviolet light of the sun. Therefore the density of the ions in the layer changes through time. Because the density of the ions changes, the delay of the signal traveling through will also change with time, the peak of error will occur at around 2 PM. Another variable that plays a role in the delay in the ionospheric layer is the distance that needs to be traveled through. A satellite above the receiver will have a shorter signal path than a satellite on the horizon. Because the path traveled in this example is longer, the signal is more sensitive to distortions [SD].

Satellite errors are not caused by the environment or the layers around the earth. These errors occur in the satellite clock or the satellite position. First is the satellite clock. The GPS can compile locations because the time of a sent signal and time of received are compared. The clock in a satellite is atomic and thus very accurate. However, even this type of clock makes tiny mistakes. According to [BRLK14] a clock error varying from -0.00014192 to -0.00014194 seconds lead to a x-coordinate error between 17.02 and -5.939 meters. The errors for y and z were 2.973 to 63.89 and -235.8 to -218.1 meters. Nowadays, these errors are more compensated for. Due to software, the error has now been reduced to an average of 2 meters [Nov15]. Then there is the orbit error. This error occurs due to a false estimation of the satellite's location. Whenever a slight variation in orbits occurs, the ground control station will send a correction to the satellite. Nowadays, these errors result in a 2.5 meters position error [Nov15].

Multipath distortion occurs when a signal of GPS reflects on any substance. The actual error results from two different signals that have traveled paths of different lengths [KMP10]. As said before, GPS calculates the position by combining the lengths of multiple signals sent from the satellites to the receiver. When a signal reflects on an object, it takes a longer path to the receiver. This way, the system determines the position with a longer path; thus, a false position will be given as output. The multipath effect is surrounding dependent, meaning that it will only occur in situations when the receiver is surrounded by an object reflecting the signal. Below is an image explaining the multipath effect [KRK13a].

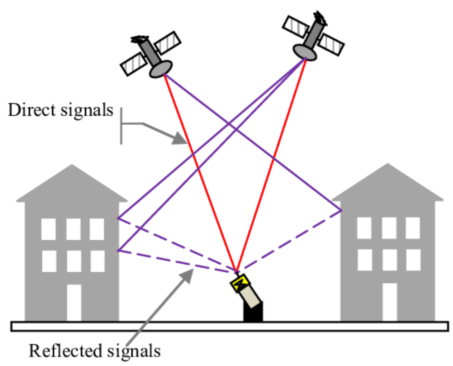

Figure 3: How the multipath effect works [KRK13b].

Table 1: Overview of errors [KEN18].

Error	Error in m
Ionospheric effects	5 m
Troposphere effects	2.5 till 25 m
Satellite clock error	2 m
Ephemeris error(orbit error)	2 m
Multipath error	Up to 100 m in worst case

Table 1 summarizes all the previously explained errors and combines the impact they have on the accuracy of GPS. The table also shows the range of the errors in meters. All errors above come along when using GPS, except for the multipath error. The multipath error depends on the environment the receiver resides. While all these errors occur when using GPS, it should be possible to correct the errors in the same location.

1.2 Research questions and scope

We know that GPS gives an accurate indication of location on larger scales. However, in small-scale environments, it lacks accuracy. This leads to the following research question: How can GPS positioning be refined in small-scale environments? This question will be answered by looking at the following sub-questions:

- Which environment and conditions causes the largest errors?
- How do the created algorithms perform?
- Gives an algorithm based on learning from ground truth data a more accurate prediction of the true GPS?
- Are moving watches more accurate than watches that lay motionless on the ground?

1.3 Definitions of terms

During this study some terms will frequently be used.

- Beacon - Still laying smartwatch with predetermined location.
- Moving watch - Smartwatch that follows a fixed trajectory.

1.4 Thesis overview

First, previously done academic work will be discussed in Section 2. This contains methods of correcting GPS in different settings. Then in Section 3 the data and functionality of the developed algorithms will be described. The information and setup about all the experiments will be found in Section 3. The results and the limitations of these experiments will be discussed in Section 5. Section 6 provides a conclusion and will make suggestions for further research.

2 Related work

In this section, we describe related work for correcting GPS errors. This section is subdivided into beacon-based, map matching, multipath correction, and Kalman filtering methods.

2.1 Beacon based correction methods

In this section, we describe a number of use cases with beacons. The first and most comparable case to this study is outdoor correction, and the next is indoor usage of beacon corrections.

2.1.1 Outdoor corrections

Nirupama Bulusu et al. used two radio-frequencies (RF) to conduct outdoor localization. The first was using the signal strength to compute the location. The second was used to built on connectivity. This means that a connection will be made with a beacon when the position is in range. A location is determined based on all connections made. They found out that the connectivity approach has real potential in determining outdoor positioning. They used beacons which all had an 8.94 m signal connectivity [BHE00]. Their setup is shown in figure 4.

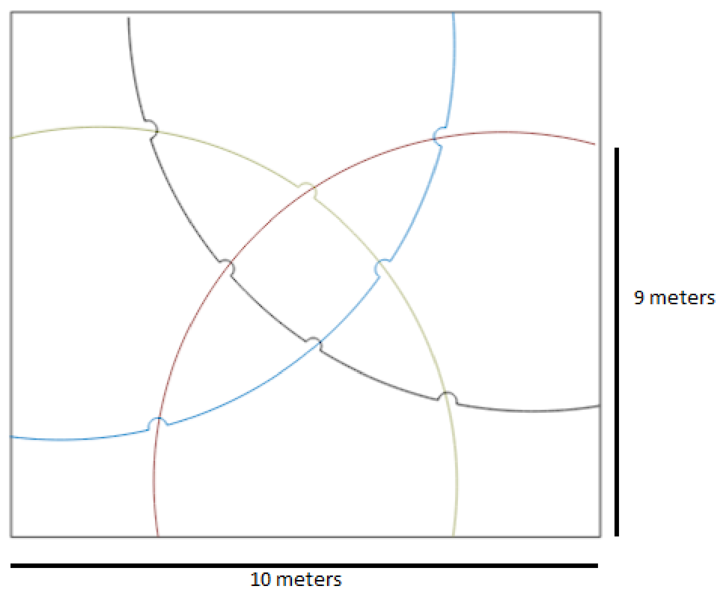

Figure 4: Setup connectivity experiment [BHE00].

We see four different lines. Each line represents the range of one beacon. By combining these lines, a grid is formed. In each area enclosed by different lines, there is a connection with all beacons. This means that when you know which of the beacons you are connected to, you can compute the area in which you are localized. In this approach, they reached a minimum error of 0 and a maximum error of 4.12 meters.
Another possible method of determining position is the usage of an Ultra Wide Band (UWB) system. Beiya Yang et al. deployed a UWB system on Unmanned Aerial Vehicles (UAVs) to get

the precise location in areas where humans can not reach [YYY21]. They combined the position from the UWB with camera images to gain maximum accuracy in their position estimation. This study gained promising results on UWB positioning. They have observed that with this method, a median of the errors of less than 20 cm could be reached. The maximum error was 100 cm.

2.1.2 Indoor corrections

When indoor positioning is required, GPS is not the solution. GPS receivers must see at least three satellites to compute a location. Obstacles such as buildings distort the line of sight, and GPS will not work. However, there are multiple other ways of computing indoor positions. Jan Vascak et al. used a combination of Bluetooth Low Energy (BLE) devices in combination with Kalman Filtering (KF) to create a navigation map for indoor robots [VS18]. This method combines the advantages of BLE devices and tries to overcome the errors by smoothing the trajectory with a KF. This could also be seen as a radio frequency method. The KF removes noise created by reflected parts and other signal disturbances. This approach was tested in three different setups, each with different interference in signals (E1, E2, and E3). The results were promising, the average error in position was respectively 26.67 cm, 37.52cm and 79.12cm for E1,E2 and E3.
Aigerim Mussina et al. used a combination of a Support Vector Machine with Naive Bayes to predict a class. There were three possible classes, indoor, outdoor, and vestibule [MA19]. When these methods are combined, the accuracy of correct predictions was 0.958, which means that the model could predict the correct location in 95% of all examples.

2.2 Map matching

Map matching is fitting GPS positions with the structure of nearby roads. In [MBHW19] three different likelihoods are combined with weights to gather a weighted likelihood. Their idea was to merge three different kinds of already existing map-matching ideas.
The first one is geometric likelihood [NK09]. Here each GPS point is compared with neighboring paths to find the path closest to the measured GPS position coordinates.
The next likelihood is the topological likelihood. This one takes the entire created trajectory into account. It compares the total length of the measured path with the lengths of possible real paths.
The third likelihood used is the temporal likelihood. The temporal likelihood computes the probability of every candidate path by using the velocity the GPS receiver has. The velocity is measured by taking the distance between two successive GPS points within a one second time interval. If the maximum velocity in the street is limited to 50 km/h, and the velocity measured is much higher, the probability that the trajectory is a urban road is low. From all these likelihoods calculated along the route, a weighted average is estimated.
The final route will be the route with the highest weighted likelihood. In this study, they use the Dijkstra Algorithm to compute the shortest route between found GPS coordinates. This approach led to a maximum correct prediction score of 96% and a minimum prediction score of 84%.

2.3 Multipath correction

Mahdi Salarian et al. correct GPS locations based on images of the urban environment. This study uses the input of GPS and other sensors as a magnetometer to determine the direction and, of course, an image of the location to compare with the database [SA15]. The database is filled with images of the urban environment. Next, it goes through the data set of images and calculates which images are near the given location. This limited set is then presented to the RANSAC [SA15] tool to compute the best match for the input image. RANSAC is a tool for comparing images [RR08]. Another way of getting a more accurate position in urban environments was developed by [VD20]. This study introduced 3D mapping-aided correction, which uses over 3850 3D models of buildings to solve the problem caused by multipath errors. Especially, when the given location is on the wrong side of the road, this algorithm can determine which building is blocking the signal by looking at those 3D models. This led to a reduction in wrong-side-of-street occurrences by at least 50%.

2.4 Kalman filtering

Many studies use a variant of the original Kalman filter to improve the accuracy of GPS. Different Kalman filters exist for different systems: the linear Kalman Filter, the Unscented Kalman Filter, and the Extended Kalman Filter. The original method was introduced by [Kal60] in 1960. The method is ideal for correcting dynamic models such as in this study.

Sher Muhammad Nizamani et al. used a Kalman Filter to reduce the error in GPS latitude and longitude data measured with a smartphone [NLAF18]. The filter uses the input GPS data and the input velocity to compute a more accurate location. By using this filter, they achieved to smoothing out the jumps in latitude and longitude. They approximately changed the latitude and longitude with respectively 0.0000199282 and 0.0000103702 on average. This seems like a tiny amount. However, looking at small-scale environments, this can mean some meters away from the actual location.

Another example of a study using a Kalman filter to correct GPS data is found in [DV19]. Here multiple sensor data is fused with an unscented Kalman filter. The sensors measured acceleration and rotation, speed of the vehicle and location based on a GPS receiver built inside the vehicle. All these sensors operated on different data rates. Therefore a multi-rate unscented Kalman filter was introduced to fuse the sensor data.

2.5 Developed method

The algorithms developed in this study fall in the category of beacon-based correction methods for outdoor usage and indoor correction methods.

3 Methodology

This section discusses the data collection, the processing pipeline for this data to come to a refinement of the GPS location measurement, and two different algorithms are described which are used in this pipeline. The pipeline is shown in figure 5.

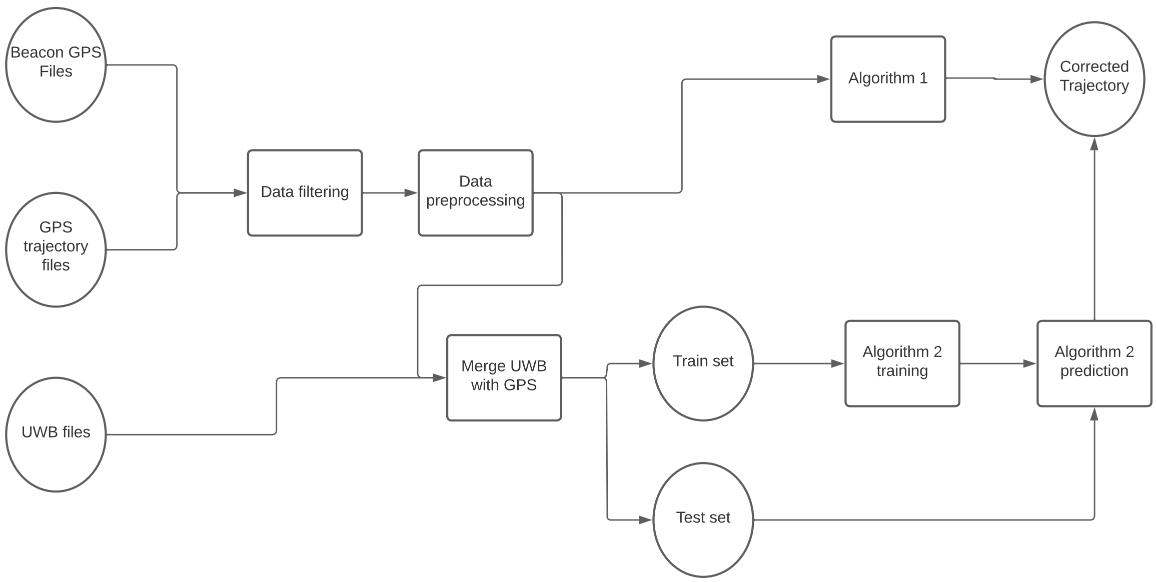

Figure 5: The pipeline from data collection to location refinement: circles are the data components, squares represent the processes implemented with our software.

It the next paragraphs the data components and process steps are described.

3.1 Materials used

The materials are a smartwatch to measure GPS coordinates, acceleration and rotation. Also we used a location system based on Ultra Wide Band to get the ground truth.

3.1.1 GPS data

In this study the Samsung Gearfit 2 Pro was used to gather the GPS data in combination with the WEARDA software package [1] [vDGvL23] developed at LIACS. The sensors and configured sample rates used for the experiments of the Samsung smart watch are shown in Table 2.

Table 2: Sensors and data rate set of Samsung Gearfit 2 Pro.

Sensor	Data rate
GPS	1 second
Accelerometer	10 ms
Barometer	10 ms
Gyroscope meter	10 ms

On the smartwatch, the Sensor application of the WEARDA package was installed and configured with a file. When starting the application, three options are given to the user. The first option is to set a number from 000 to 999. This will be the experiment number merged into the names of the files which will be created. Then there is a button restart, which will restart collecting data based on the experiment number as filename.

The GPS data from the smartwatches are saved in CSV formatted files by the WEARDA software package. The variables in the file are listed below.

- Time, the number of seconds after starting to record.
- Latitude, the measured latitude in GPS degrees.
- Longitude, the measured longitude in GPS degrees.
- Accuracy, the predicted accuracy of the measured GPS point in meters.

Bestand Bewerken Opmaak Beeld Help

```
001 dc81 2022 01 14 11 59 22
time,latitude,longitude,accuracy,private
6.5,52.020123,4.268517,12.0,P
7.2,52.020127,4.268518,8.0,P
8.3,52.020128,4.268520,8.0,P
```

Figure 6: Example data of a GPS CVS file produced by the WEARDA software package.

Figure 6 shows the first lines of an example of the GPS CSV file. We see the first line is the header containing the experiment number, followed by the watch ID, date and time of the measurement. A second header contains the variable names, and the next lines contains their values.

[1] https://www.github.com/liacsprojects/wearda/

Three other identical structured files were created while recording. The first file is for the accelerator sensor, which measures the acceleration at every moment. The second file was to measure the air pressure with a barometer. The final file was used for the rotation velocity. We collected this data as well for enhancing our algorithms but this data is not used in this study.
The beginning of the GPS data records were removed because of startup errors. In the pipeline this is referred to as GPS filtering.

3.1.2 UWB data

For getting the ground truth we used an Ultra Wide Band localization system with an accuracy of around 10 cm, recording the time and position simultaneously. We used the Pozyx creator kit [2] and software implemented by Rodi Laanen, one of our bachelor students to create a file with location data of all tags. This data contained the following relevant variables for our study:

- Timestamp, to get absolute time
- loc(x), measured x location of the UWB in mm.
- loc(y), measured y location of the UWB in mm.

The UWB system expresses its location in Euclidean coordinates. Therefore the GPS coordinates were converted to this coordinate system as well with the Haversine software package [3].
In the first data collection trials, video was recorded to get a kind of ground truth. The location was indicated with numbered traffic pawns put on a predefined route on the ground, which were recorded as well. When filming, we could determine the time and location of the moving smartwatch. However, processing the video files were time consuming because the acquisition of the time and position from the images of the video had to be done by hand since we could not find off-the-shelf software for this task.
We found that the UWB grid is in a 45 degrees angle with the GPS grid. This turn will be seen later on when the results of algorithm 2 are discussed. We have to correct for this turn in order to get the best outcome of the predictive algorithm. In Figure 7 we can see both grids and the turn between them. ere the white lines are the GPS-lines, latitude and longitude. The black lines show the grid of the UWB.

3.2 Data processing

The GPS data was pre-processed to get a nice time alignment. This is because every watch was turned on at a slightly different moment. We took the start of the moving watch as reference time T_{offset} and converted the time and position point of the beacons accordingly. We also computed the time of the GPS positions to seconds counted from the start of the day $T_{position}$. With the following equation the start time T_{start} was calculated:

$$T_{start} = T_{position} - T_{offset} \quad (1)$$

[2] http://store.pozyx.io/
[3] https://pypi.org/project/haversine/

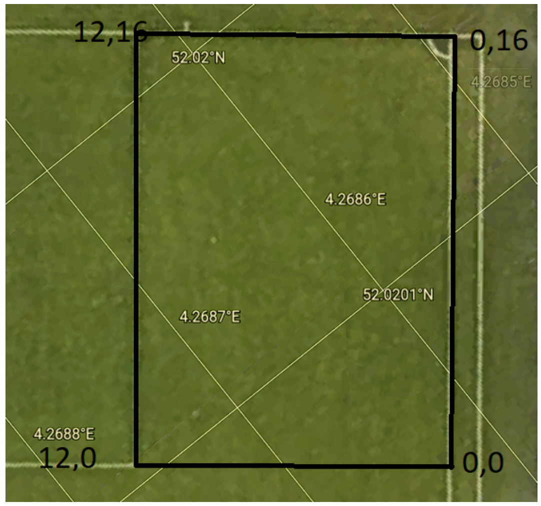

Figure 7: GPS and UWB grids are turned towards each other (source Google Earth).

Also the start of the GPS data was removed since calibration time took 350 seconds to find satellites and compute a location. This was only done when turning on the application. Once the application was calibrated, there was no need for calibration again when restarting with next experiments.
For the first algorithm a Python Pandas data frame was inserted with all possible beacon corrections and moving watch GPS coordinates per time. This data frame is called *corr*.
For the second algorithm also a Python Pandas data frame was inserted with all possible beacon corrections, the ground truth of UWB and the moving watch GPS coordinates. UWB works with an Euclidean coordinate system with a max x length of 11.8 and a max y length of 16 meters. GPS works with latitude and longitude degrees geographic coordinates. Therefore the GPS coordinates and beacon corrections were converted to Euclidean coordinates.

3.3 Algorithms

For this study, two algorithms were developed. The first is a basic heuristic algorithm, the second uses a training set to learn to refine the measured GPS of the moving watch under certain conditions.

3.3.1 Algorithm 1

The idea behind the first algorithm was that a GPS measurement of a moving watch would be influenced by the errors described in Section 1. However, if we place watches at a predetermined position, we should be able to see the error the watch is making in that location. The idea is to correct the moving watch GPS trajectory with use of the error of these GPS watches (we call them beacons further on) they make with their well known position. Then when a moving watch comes close to the beacon, we can apply our error to correct the trajectory. Here, we assume that moving watches around the same beacon position undergo the same error.
The first algorithm needs as input data of the measured trajectory of the moving watch and a list

with all possible corrections corresponding with the closest beacon. It then calculates the distance between the measured GPS point (moving watch) and all the beacons with the Python function *greatcircle.Distance*[4]. This function measures the distance between two GPS points on earth, the units can manually be set on meters. The beacon that is chosen is determined by:

$$Beacon = argmin_i((X_w(t) - X_i(t)) \tag{2}$$

Beacon refers to the beacon with beacon number i, where the distance (Euclidean norm) between the location of that beacon and the measured point is the smallest. $X_w(t)$ is the measured GPS point, and $X_i(t)$ is the location of beacon i. Min refers to taking the minimum distance of the calculated distances.
The corresponding correction is found by taking the nearest beacon.

$$X_{corr}, X_c(t) = X_w(t) - X_i(t) \tag{3}$$

Here the $X_c(t)$ is the correction vector in time t. We can now calculate the corrected position of the moving watch $X_{wcorr}(t)$ with:

$$X_{wcorr}(t) = X_w(t) + X_c(t) \tag{4}$$

3.3.2 Algorithm 2

The second algorithm uses a learning model. This learning model needs the ground truth collected by the Ultra Wide Band (UWB) system. The UWB produces data of the true location of the moving watch as a function of time. The learning model takes the beacon correction vectors and measured location of the moving watch as input and optimizes its output with the true location of the moving watch measured by the UWB system. The model will weight the influence of every input feature. This way, the algorithm can learn the best beacon(s) to minimize the error between the measured path and ground truth.
A Support Vector Regression (SVR) model has been chosen as a learning algorithm. A SVR has proven to be effective in numerical value estimation [AK15] and because it performs well with small amounts of data.
The SVR model's input and output values must be converted with a wrapper since the SVR only has one output variable. We have used two different wrappers. The first wrapper is the MultiOutputRegressor [5]. This wrapper builds different models for the different outputs [Kum21]. The second wrapper is the RegressorChain. Here also, multiple models are built. However, the way that the models are built differs from MultiOutputRegressor. RegressorChain builds a model and uses only dependent variable one to build model 1. This is shown in Figure 8. The difference with the MultiOutput regressor is that the RegressorChain can learn dependencies between the dependent variables. The accuracy of both methods will be compared in Section 5.
In Figure 9 we can see the GPS separated coordinate values of the moving watch and the location the UWB in Euclidean coordinates millimeters as a functon of time. UWB data has been divided

[4]https://geopy.readthedocs.io/en/stable/module-geopy.distance
[5]https://scikit-learn.org/stable/modules/generated/sklearn.multioutput.MultiOutputRegressor.html

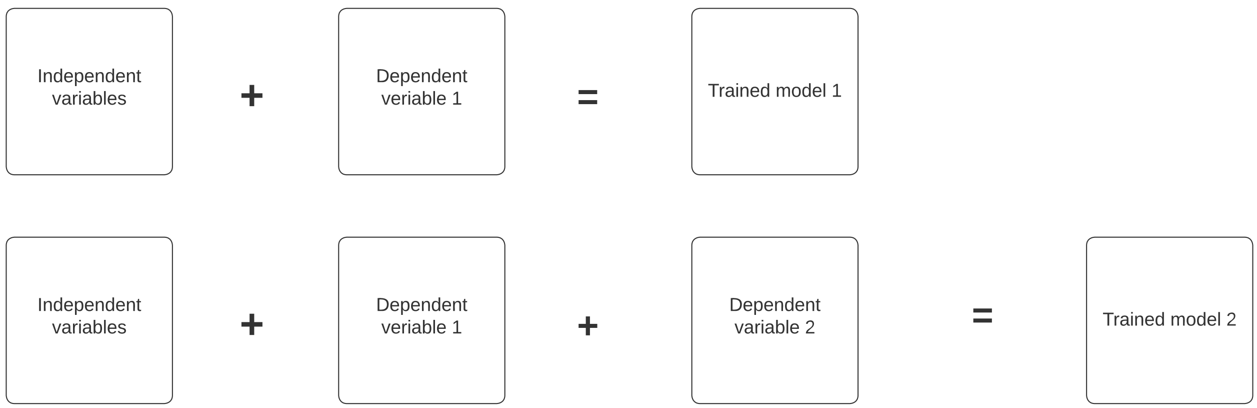

Figure 8: Example of how RegressorChain wrapper builds a model [Cho20].

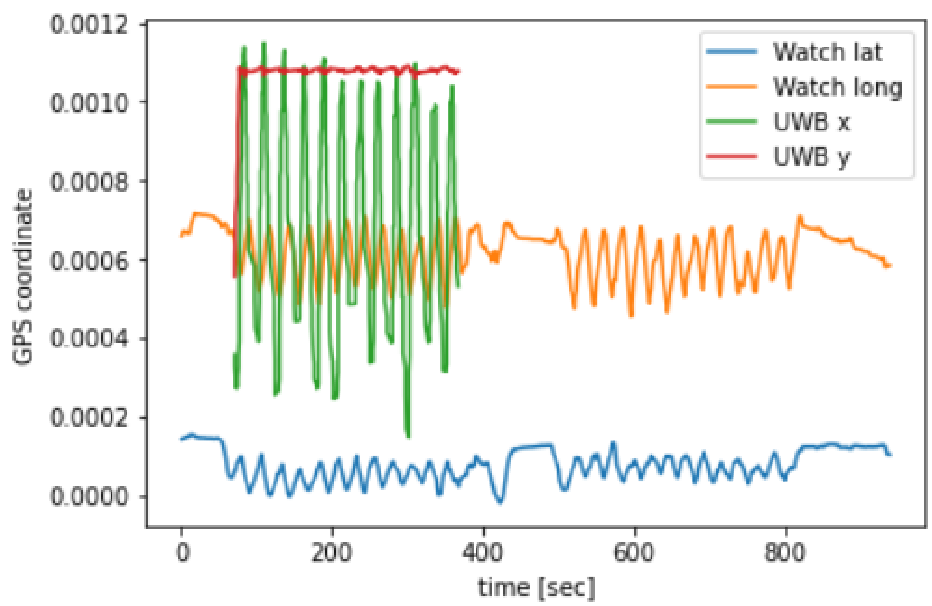

Figure 9: Moving watch GPS coordinates W(lat,lon,t) and UWB U(x,y,t) as function of time. UWB is in millimeters and has been divided by 10,000,000 to align it in range with GPS coordinates.

by 10,000,000 to get it aligned with the UWB coordinate values.
We followed a straight line of 12 meters long. The coordinates systems have an angle of around 45 degrees, so that explains the ripple on latitude and longitude while only the x-coordinate of the UWB location is changing.
We can see that the GPS of the moving watch follows the same path as the UWB-tag. Therefore we think it is possible to correct the GPS trajectory based on a learning algorithm.
We took the first cohort between 70 and 375 seconds to be used as train and test set, and split this data with several ratios between 0.7 to 0.9.

3.4 Merge UWB and GPS data

For algorithm 2, we needed to merge the data of GPS on the correct position with UWB. Herefore, we start working on the GPS file of the moving watch. We take out the rows in the file that correspond with the correct time frame of the UWB file. Next, we go on with the UWB file. Here we first filter on the moving tag and remove all the rows of the master tag. Then we convert the time given in hours, minutes, and seconds to a total of seconds. Next, when the time is in seconds, we search for duplicates and remove them. After this, we fill possible time gaps with its prior values. The gaps are filled in to get values for every second. After the gaps are filled, we merge the GPS file with the UWB file. The GPS of the beacons are treated identical. From all the time data, an interval is selected we feel is good for using it as learning and testset. Then for all beacons, we calculate the correction vector X_{wcorr} so the difference between the measured GPS position and the actual GPS position. This is done with the Haversine formula [6]. Now we are ready to merge all the beacon information with both the UWB and moving watch GPS trajectories and save the data frame to a CSV file. Figure 10 shows an example of the CSV file created, here we see all the column names and some example values, the values are cut-off after b4lat.

```
index,time,wlat,wlon,b1dlat,b1dlon,b2dlat,b2dlon,b3dlat,b3dlon,b4lat,b4lon,uwbx,uwby,b1dmeters,b2dmeters,b3dmeters,b4dmeters
0,71,6.199999999978445e-05,0.0006479999999999819,-47.751563000000004,47.751529,-47.751399,47.751337,-47.751449,47.751435,-47.751618,
1,72,5.999999999772854e-05,0.0006339999999998014,-47.751563000000004,47.751529,-47.751399,47.751337,-47.751449,47.751435,-47.751618,
2,73,6.999999999379725e-05,0.0006140000000005585,-47.751563000000004,47.751529,-47.751399,47.751337,-47.751449,47.751435,-47.751618,
```

Figure 10: Example from CSV file as input for algorithm 2.

3.5 Trajectory accuracy scores

The two algorithms make errors which can be quantified with a number of trajectory accuracy scores. For the algorithms, we cannot measure its accuracy based only on the spatial data [TBS+21]. We need temporal data as well to determine the error distance per time along the true trajectory and the corrected one.

We measure the average error that the beacons are making with:

$$AE_i = \frac{1}{n}\sum_{t=0}^{n}(P_{ti}(t) - P_{mi}(t)) \tag{5}$$

[6] https://pypi.org/project/haversine/

Here AE_i is the average error in meters, P_{ti} is the true location of beacon i, P_{mi} is the measured position of the beacon, and n is the amount of measurements.

The other scores are the Mean Absolute Error (MAE), the Root mean squared error (RMSE), and the Mean Absolute Percentage Error (MAPE) [SRAA21] . These scores are calculated with the following equations:

$$MAE = \frac{1}{n}\sum_{t=0}^{n} ||P_{true}(t) - P_c(t)|| \tag{6}$$

Where P_{true} and P_c are the measured position of UWB (ground truth) and the corrected position of the moving watch by an algorithm respectively. n is the amount of measured points.

$$RMSE = \sqrt{\frac{\sum_{t=0}^{n} ||P_{true}(t) - P_c(t)||^2}{n}} \tag{7}$$

$$MAPE = (\frac{1}{n}\sum_{t=0}^{n} \frac{||P_{true}(t) - P_c(t)||}{||P_{true}(t)||}) * 100 \tag{8}$$

The MAPE [KK16] is the normalized version of the MAE. It gives the accuracy of a forecast values compared with its true value. The best forecast accuracy percentage is 0%, the worst 100%.

4 Experiments

In this section, the experiments are described. An experiment is mainly a data collection event in a selected location. The experiments were done in four locations.

4.1 Locations

4.1.1 Location 1

For location 1, a location was chosen with the least amount of environmental disturbances. Therefore an open soccer field was chosen. Another reason for this location was the ability to use the line work on the field as ground truth. The soccer club which was chosen is soccer club Klein Maar Dapper in Wateringen. The location is shown in Figure 24. The orange trail path in the figure shows the trajectory that is followed. The blue circles are the locations of the beacons.

Figure 11: Overview of location 1 (source Google Map).

4.1.2 Location 2

For location 2, a park was chosen to study the effects of surrounding trees on GPS. Hereby, a park in The Hague at De Uithof, was chosen. Also here we set out a trail path to make it easier to walk the same route multiple times. The trail path is indicated in orange in Figure 12.

4.1.3 Location 3

Location 3 is located at the Jan Gresshofplatsoen in The Hague. This is a square surrounded by houses. The square is chosen because, here, the effect of small buildings around a GPS trajectory could be measured. Also, because the square is lined out, there is a more accurate way of finding the ground truth. The location overview is shown in Figure 13.

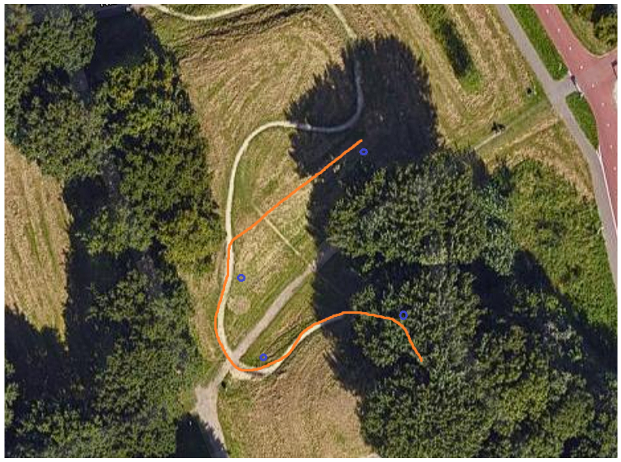

Figure 12: Overview of location 2 (source Google Map).

Figure 13: Overview of location 3 (source Google Map).

4.1.4 Location 4

And the final location is expected to give the most amount of environmental disturbances. This place is located at the Menno ter Braakstraat in The Hague. At the end of the street, two flats are build. The height of the flats could cause problems with finding enough satellites to compute an accurate location. The location is shown in Figure 14.

Figure 14: Overview of location 4 (source Google Map).

4.2 Experimental setup

To be able to do the experiments that were necessary to gather enough information for this study, different trails were set out sometimes using rope and pins to attach the rope to the ground.
For the first location, the lines of the soccer field are ideal to use. This way, the path can easily be compared to the ground truth. Before beginning to walk, a sketch of the location is made, drawing in the locations' figures to ensure the experiment is precise and reproducible. Next, we place all anchors for the UWB experiments in the correct positions. The same goes for the GPS beacons for GPS experiments and UWB experiments. When all devices are in place, we can set up the digital environment of the UWB system. This system calibrates the anchors and the tags. When the devices are calibrated, we can start the GPS beacons. These need some calibration time as well. Therefore, we should wait a minimum of 300 seconds in sunny weather before beginning to walk the trajectories. When the calibration time of the smartwatches is over, we can start walking the trajectories. Once calibrated, the watches do not need to calibrate again when starting over with a new ID. This way, multiple runs can be done after calibration.

4.3 Experiments

To gather enough data about the performance of the algorithms, 13 experiments were done. Table 3 shows all the experiments.
Experiments 1 to 9 are needed to answer research sub-question 1 and 4. Experiments 10 to 12 give insight into questions 2 and 3. When combining all these experiments, we can say something about

the accuracy in small-scale environments of GPS. In the third column, 'walking a fixed path' points at the trajectories drawn in Section 4.1. When 'laying all watches down' is noted, all the watches are laid on the ground to see the standard error over time. From experiment 10, there was chosen to have a different experiment type. This followed from in between results, where trajectories were more accurate than the beacons. This resulted in wrong corrections. To investigate these errors in the beacons, an experiment was made that gained insight into the performance of watches over a more extended period. This is recorded as experiments 10 and 11. While doing the experiment, different kinds of weather came along. This influenced the accuracy of GPS. If there is a sunny day, then the accuracy of GPS will probably be higher.

Table 3: Overview of the data collection experiments.

Experiment number	Location	Description of experiment	Conditions
1	1	Walking fixed path	sunny
2	1	Walking fixed path	cloudy
3	1	Walking fixed path	sunny
4	2	Walking fixed path	sunny
5	2	Walking fixed path	sunny
6	3	Walking fixed path	cloudy
7	3	Walking fixed path	sunny
8	4	Walking fixed path	sunny
9	4	Walking fixed path	sunny
10	1	Laying all watches at same point	sunny
11	1	Laying all watches at same point	sunny
12	1	UWB as ground truth for algorithm 2	sunny

4.3.1 Experiments with UWB

Around May 2022 we got the availability of a Ultra Wide Band location system to measure spatial and temporal location data of the moving watch. This data can be used as ground truth since the accuracy is 10 cm and the time interval 5Hz. The UWB system has a limited range of 12-20 meters so we could not repeat prior experiments to get the ground truth. However, other, smaller trails were created to make the measurements possible with this UWB system. We only collected data in location 1 with the UWB system. We followed a number of trajectory shapes.

5 Results

In this section, the outcomes of the different experiments will shown how the algorithm performs. Results of the experiments will answer the research questions discussed in Section 1.2.

5.1 RQ1 and RQ4: accuracy analysis locations

To answer research question 1 about how the environment influences the measured location accuracy, we measured the trajectory compared with the true original trajectory for every location (error). Further we calculated the accuracy per location as was also measured by the GPS device (accuracy). This is done by taking the minimum, maximum and average accuracy for every beacon in every location. We only selected the experiments with sunny weather to keep the circumstances for the experiments as much as identical.

Table 4: Average errors of the GPS beacons per location.

Location	# beacons	Avg error (m)	Min Avg error (m)	Max Avg error (m)
1	5	3.5	0.96	7.0
2	4	34.6	11.24	77.0
3	4	13.0	8.25	20.0
4	3	6.0	2.95	12.0

To understand which environment disturbs the GPS signal most, we look at Table 4. We can see that for each location, an average error, min average error, and max average error have been computed. The min average error is the lowest average error of all beacons in that location. The Max average error is the highest of averages of all beacons.
We see some surprising results. The expectations were that the errors in location 4 were significantly higher because of the multipath effect. A side note on the result in location 4 is that the lowest error is found in beacon three, which was the beacon that was farthest away from the flats. However, we see that location 1 gives the most accurate measurements, followed by location 4, location 3, and worst is location 2. However, we must say that the average of location 2 is influenced by one beacon performing poorly, and therefore the average comes out higher.
These results could indicate that if such errors do not influence the trajectory in all places, then algorithm 1 will not be able to make improvements based on the beacon errors.

Another way to determine the accuracy is to take the accuracy the GPS beacons calculate next to its GPS coordinates, and we are curious to see if the found ranking with the prior method corresponds. Table 5 shows the GPS accuracies per location, and the ranking of the accuracy in decending order seems identical as the prior method.

To gain more information about the low accuracy occurring in locations 2 and 3. We decided to take a closer look at that to see the possible effects of the low accuracy on the corrections done

Table 5: GPS accuracy measured per GPS beacon per location.

Location	Beaconnr	Avg accuracy (m)	Worst accuracy (m)	Best accuracy(m)
1	1	5.2	32.0	1.5
1	2	10.1	48.0	1.5
1	3	12.5	48.0	1.5
1	4	5.0	32.0	1.5
1	5	5.2	16.0	1.0
1	avg	7.6	35.2	1.4
2	1	191.9	400.0	24.0
2	2	27.0	48.0	6.0
2	3	34.5	200.0	12.0
2	4	15.8	24.0	6.0
2	avg	67.3	168.0	12.0
3	1	15.0	128.0	3.0
3	2	17.4	192.0	12.0
3	3	16.1	96.0	12.0
3	4	15.3	96.0	3.0
3	avg	16.0	128.0	7.5
4	1	7.7	16.0	3.0
4	2	4.1	16.0	3.0
4	3	16.0	16.0	16.0
4	avg	9.3	16.0	7.3

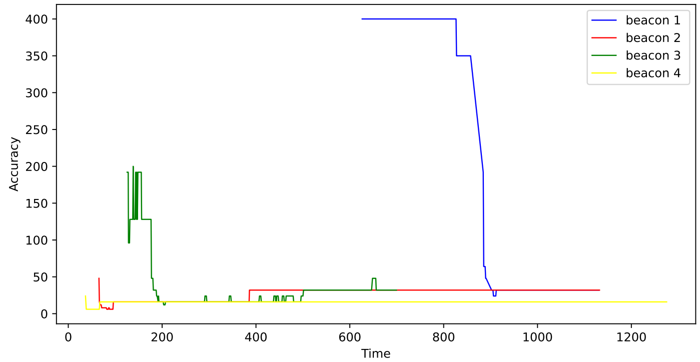

Figure 15: Accuracy's of beacons over time in location 2.

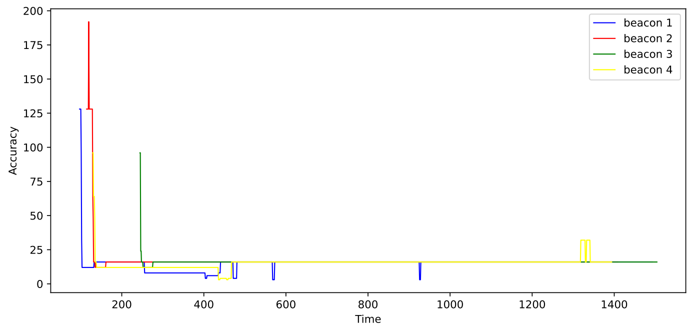

Figure 16: Accuracy's of beacons over time in location 3.

later. Therefore, we created Figures 15 and 16. Here the accuracies of the beacons are plotted over time to see if the accuracy is lowest in calibration time or if it might spike. In the second case, where the accuracy drops after the calibration time, algorithm one will correct with low accuracy and likely get a wrong output. First, we look at location 2. Here we see that beacon 1 does not find the satellites to compute a position in the first 600 seconds. This while the watch was turned on simultaneously as the other watches. Therefore, the set-up of the calibration time will not do anything with beacon 1, and false corrections will probably be made. The late start of beacon 1 is because it is located under a tree, and therefore it is hard to be in sight of at least three satellites. We see that the accuracy has no big spikes after calibration time for the rest of the beacons. However, despite having no jumps, we see that the accuracy also does not increase in time. It stays at 25.0, which means that the corrections made will be done with lower accuracy than preferred.
Now, we analyze the accuracies of the beacons in location 3. Here we find a little lag in beacon 3, yet, we do not see major upwards jumps in the accuracies. All the beacons hover on 16.0 with a few small jumps upwards to 32.0 for beacon 4 and some increases in accuracy in beacon 1. The accuracies hovering around the same level could mean that we can correct the trajectory if the trajectory has the same accuracy.

To answer research question 4, we can compare these accuracies of the beacons with the trajectories' accuracies. The accuracies of the trajectories are displayed below in Table 6. Now we can compare the averages of the accuracies of the trajectories with the beacon averages. We see that the beacons in locations 1 and 4 are more accurate than the trajectories in these locations. For location 2, the average of beacon 1 is worse, but the average of the other three beacons is around the same as the trajectory. For location 3, the average of all beacons is the same as for the trajectory. However, we see that the beacons become more accurate in time, and the trajectory accuracy stays at 16.0. This means that the beacons become more accurate than the trajectory in time.

Table 6: Accuracies of trajectories.

Location	Avg accuracy	Best accuracy	worst accuracy
1	14.97	3.0	16.0
2	29.76	16.0	32.0
3	16.0	16.0	16.0
4	16.3	16.0	32.0

5.2 RQ2: beacon behavior experiments

Research question 2 is about the behavior of the GPS beacons when they are positioned in the same location. We expect that all beacons indicate an identical error between the true GPS coordinates and what they measure. However, these so-called "still laying experiments", experiment 10 and 11, see Table 3 shows:

1) The beacons have a startup time in which they tend to move to a stable position.
2) The beacons behave differently: every beacon measures its GPS position differently, but the error between true and measured position decreases.

The detailed results are shown in the graphs in Figures 17, 18, 22 and 21. The graphs show the measured GPS positions in time: the position moves to a stable point. Figure 17 shows us that all the beacons measure their GPS point around $(52.0202, 4.2694)$. However, we see the true location of the beacons indicated by a light blue star. This indicates that the assumption that smartwatches in the same position make the same error is true. Therefore we could use the GPS beacons as corrections for the moving watch GPS trajectory. Figure 18 also shows that almost every beacon hovers around the same position. Nevertheless, we also see that the red differs more from the rest. Therefore, we may think that sometimes a larger error will be applied to correct the GPS trajectory. However, by looking at Figures 17 and 18, we believe that the beacons can be of use to correct GPS trajectories measured close to the beacons.

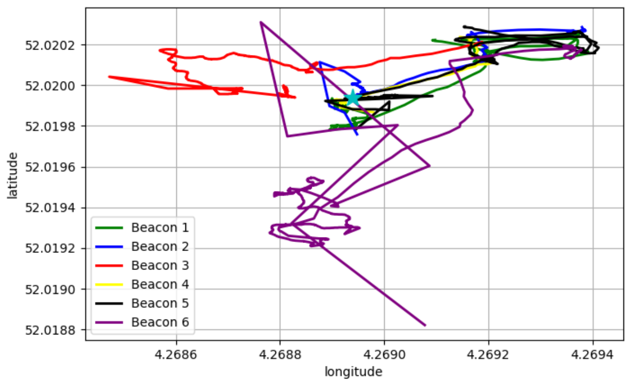

Figure 17: Still laying beacons experiment 1, location 1: light blue star indicates true position

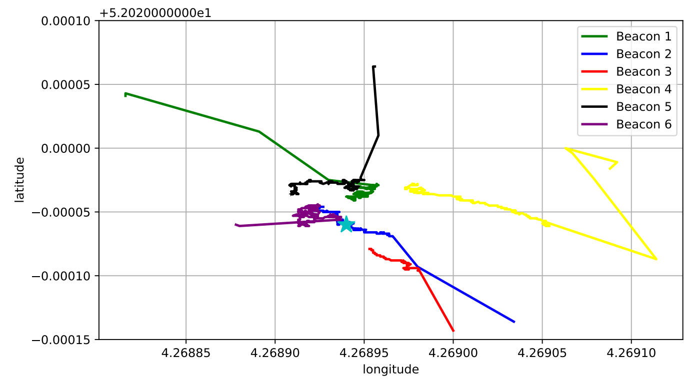

Figure 18: Still laying beacons experiment 2, location 1: light blue star indicates true position

5.3 RQ3: analysis beacon errors

To gain more insight into the errors beacons make, we divided the error into a latitude error and a longitude error. Figure 21 shows the latitude and longitude errors that were made in location 3, experiment 7. We can compare this with another location to see if there are differences in how the error per coordinate type behave. Figure 22 shows the latitude and longitude beacon error in location 1, experiment 3. We see that both latitude and longitude errors are mostly horizontal lines in time, meaning that they do not change much in time as it seems.
If we compare location 1 with location 3, we see that errors in longitude are higher, almost a factor 3 more compared to location 1:

Table 7: Longitude and latitude errors in location 1 and 3.

Location	longitude error	latitude error
1	0.000 - 0.0001 = 0.0001	-0.00004-0.00004 = 0.00008
3	-0.00020 - 0.00010 = 0.00030	-0.00010 - 0.00015 = 0.00005

Now we wanted to find out what the effect on the total GPS error was. This is calculated with the Haversine formula. Figure 23 has been created for this purpose. This figure shows us per beacon that the overall GPS error is ranging from 1.5 meters (beacon 1 and 2) and 4 meters (beacon 3,4,5) over a time period of 700 seconds. The beacon errors paddle below a certain upper error boundary in time.

Above observations show that for the beacon behavior:

- an open field location such as location 1 results in smaller longitude GPS error - almost a factor of 3 less - than urban environments such as in location 3.

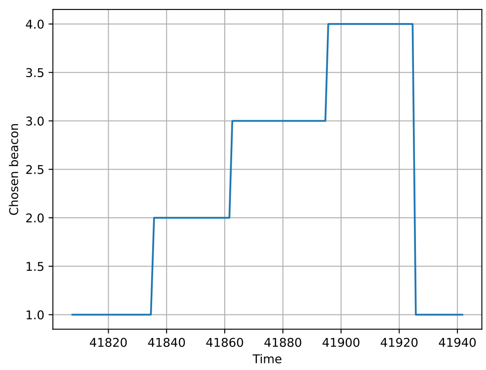

Figure 19: Chosen beacon number as function of time, location 3

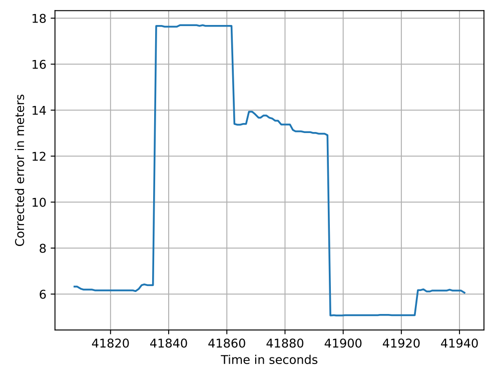

Figure 20: Applied correction (error) in time, location 3

- an open field location such as location 1 results in an identical latitude GPS error than urban environments such as in location 3.
- the longitude coordinate is more sensitive to GPS errors than the latitude error - almost a factor of 3 higher.
- the beacon errors are beacon dependent.
- the beacon errors vary slowly (low frequency of 10-30 around minutes) between certain values in a range of 1.5 meters, others in a range of 4 meters, in a short time frame of around 10 minutes.

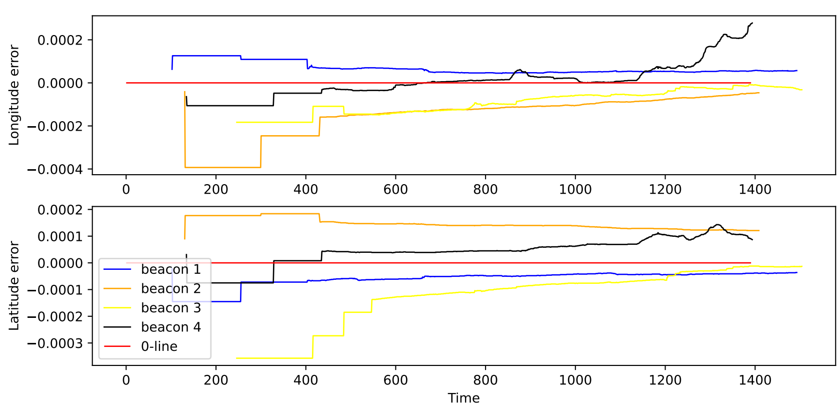

Figure 21: Latitude and longitude beacon error as function of time, location 3

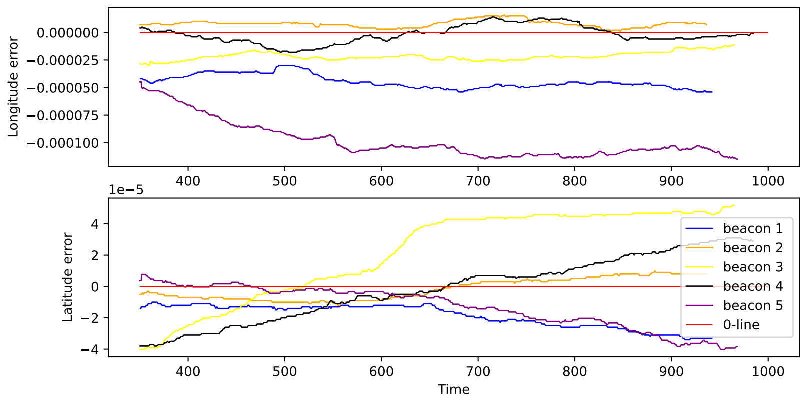

Figure 22: Latitude and longitude beacon error as function of time, location 1

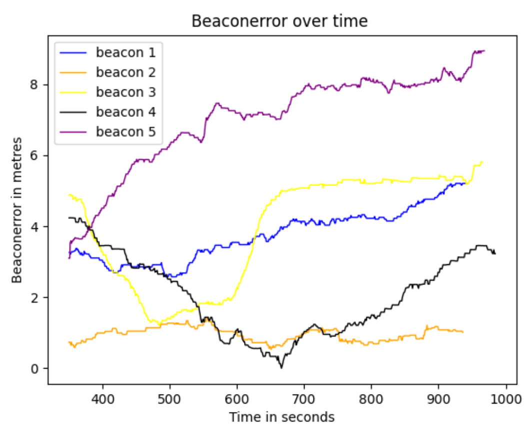

Figure 23: Beacon error in meters as a function of time, location 1, experiment 3

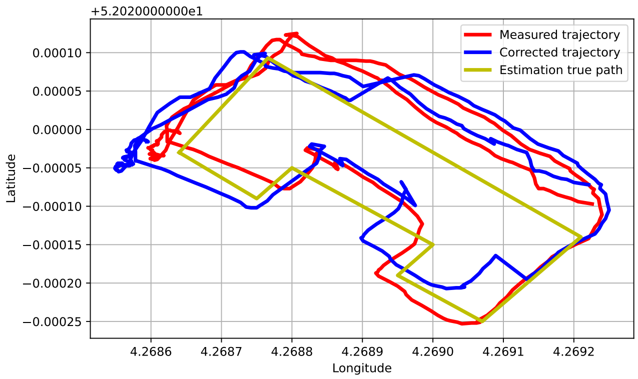

Figure 24: Correction result algorithm 1 in location 1

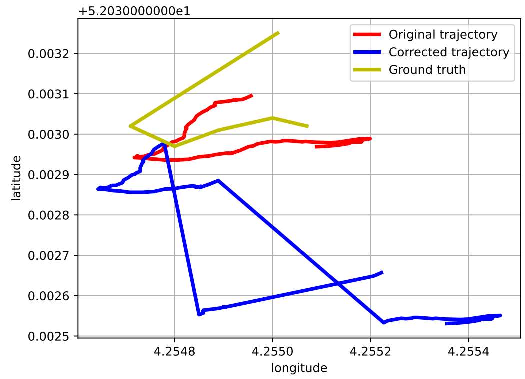

Figure 25: Correction result algorithm 1 in location 2

5.4 RQ4: corrections with algorithm 1

To analyze algorithm one, we need to look at the corrected trajectories in every location. This is done by plotting the ground truth together with the original and corrected trajectory. For every location, one graph is shown.
For every location, we see corrections that shifted the measured coordinates closer to the ground truth, as well as corrections that negatively changed the measured points. A possible explanation for the false corrections is the estimation of the position of beacons. Though we know that the location of a beacon is fixed, we cannot compute this position's GPS coordinates with 100% certainty. This may cause unexpected jumps in the corrected trajectory.
In Figure 25, we see some strange things happen. We see big jumps in the GPS trajectory. This is the result of high errors which are made in the beacons. We can look up the errors in Table 4 and see that the maximum error in this location is actually over 70 meters. This will result in false corrections, as seen in Figure 25. When we go into the data file of this beacon, we actually see that in most lines, the predicted accuracy is lower than 400 meters, meaning that it is really inaccurate. This beacon lies around tree. Therefore it will probably be out of the line of sight and unable to compute an accurate position.
In two out of four example paths, we see that some big jumps appear in the corrected trajectory. We can analyze this by taking a closer look at what the errors of the chosen beacons were.

For location 3, Figures 19 and 20 show the chosen beacon number and the error which was applied to the trajectory for every moment in time. This experiment (7) was done on a sunny day. We can see that beacon 1 has a small error for which it is correcting. Beacons 2 and 3 are actually far off. The errors which are applied are almost 18 meters in beacon 2 and close to 14 meters in beacon 3. This is a case where the trajectory is actually more accurate than the beacons! Therefore algorithm 1 is not able to refine the GPS coordinates.

We see that the assumption we made about correcting high errors with beacon errors is true. Algorithm 1 is however not able to handle such large beacon errors. We see that locations 1 and 4 have conditions which let algorithm 1 perform better because of the smaller beacon errors. These

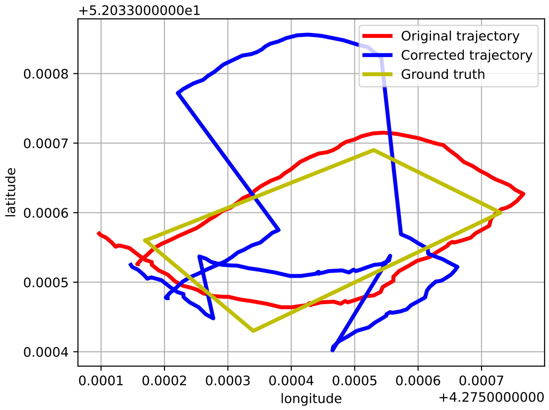

Figure 26: Correction result algorithm 1 in location 3

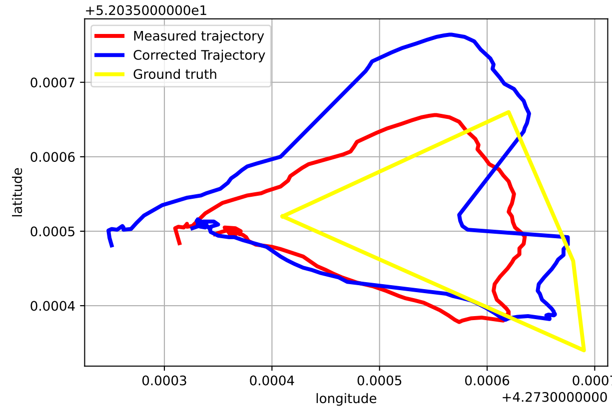

Figure 27: Correction result algorithm 1 in location 4

are also the locations with the smallest average error and the smallest min and max average errors. Therefore we can conclude that algorithm 1 is not able to work with high errors measured by the beacons.

5.5 RQ5: corrections with learning algorithm 2.

The performance of algorithm 2 is measured using the accuracy metrics and the train set composition for the straight line ground truth as described in Section 3.

The beacon error were expressed in the numerical part of the GPS coordinates and in meters.

Best results were achieved with a SVR with the RegressorChain wrapper when the beacon errors were expressed in fractions of the GPS coordinates. The MAPE was 20% with a train/test ratio of 0.2. This result can be found in Table 8.

Best results were achieved with a SVR with the multi output regressor wrapper when the beacon errors were expressed in meters by converting the fractions of the GPS coordinates with the Haversine package. The MAPE was 16% with a train/test ratio of 0.3.

More details follow now.
To see how algorithm 2 is performing, we need to check the UWB data. Therefore Figure 28 has been created to display the straight line trajectory. We see that y does not change significantly and that x is periodic. This corresponds with the x and y of walking a straight line. Because, when walking a straight line, only 1 of the coordinates should change.
After we checked the UWB data, we can plot the corresponding GPS data. Figure 29 shows the raw GPS data plotted. The background is an indication of what scale we are talking about. This area is 16 by 12 meters, and is the same area described in Section 3. Now we can compare this

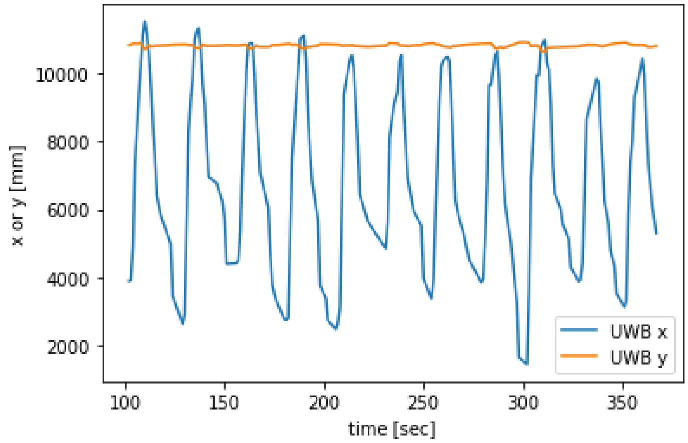

Figure 28: UWB straight line trajectory

with the UWB data and the predicted positions of algorithm 2. The line that has been walked here was almost 12 meters long. This is the same trajectory as from Figure 28. Comparing the two figures gives us insight into the accuracy of GPS in small-scale environments. We see that UWB can provide a more precise position. GPS gives a less accurate indication. We here see the trajectory measured with GPS. We see a difference between the x and y values of the grid. This means that the measured points are off in two dimensions. Here algorithm 2 will try to correct this.

Figure 29: Raw GPS trajectory on straight. line

Now we can take a look at how the second algorithm is performing. We divided the results into two different Tables. The first table gives insight into the performance of the RegressorChain as a wrapper, these results are in Table 8. The second table shows the results of the MultiOutputRegressor as a wrapper, these can be found in Table 9. In both tables, there are multiple variables that differ, which are the test size and the wrapper. However, to compare both tables, we tested each variable with both wrappers. Note that a random state has been used. Therefore outcomes may differ when run again.

Table 8: Accuracy scores of algorithm 2 with RegressorChain with beacon errors in GPS degrees.

Trajectory	Wrapper	test size	MAE	MSE	RMSE	MAPE
Straight line	RegressorChain	0.05	0.00013	3.533exp-08	0.000188	25.74
Straight line	RegressorChain	0.1	0.00022	1.165exp-07	0.00034	28.900
Straight line	RegressorChain	0.2	0.000126	4.651exp-08	0.000216	20.535
Straight line	RegressorChain	0.4	0.000257	1.13exp-07	0.000337	32.726

Table 9: Accuracy scores of algorithm 2 with MultiOutputRegressor with beacon errors in GPS degrees.

Trajectory	Wrapper	test size	MAE	MSE	RMSE	MAPE
Straight line	MultiOutputRegressor	0.05	0.000234	1.38exp-07	0.000371	34.863
Straight line	MultiOutputRegressor	0.1	0.000242	1.34exp-07	0.000366	31.673
Straight line	MultiOutputRegressor	0.2	0.0003297	1.64exp-07	0.000405	39.761
Straight line	MultiOutputRegressor	0.4	0.000159	6.23exp-08	0.000250	38.017

These numbers are generated with a $randomstate = 40$, meaning that they could differ when another seed is used. Based on this, we could take the average of all MAPE scores to filter out coincidence. The average MAPE of RegressorChain is then 26.98. Against 36.08 from MultiOutputRegressor. For this seed, we may say that the RegressorChain is performing slightly better than the MultiOutputRegressor.

To show what the algorithm is able to do, we plot the UWB data with the predicted positions. Examples of these predictions can be found in Figures 30 and 31.

We also made a change in the input of algorithm 2. As explained before, we also calculate the latitude and longitude as a combined difference expressed in meters. These results are now discussed. Here we did more runs with the algorithm to validate its accuracy. The test size is now fixed on 0.1. This means that 10 percent of the data set is used for testing. Which eventually will be one time forth and one time back on the straight line because the line is 11 meters long. If we go 1 meter per second, we will take 11 seconds to go on the line. If we also go back, it will result in 22 seconds. 10 percent of 296 measurements conclude in 30 measurements.

Table 10: Accuracy scores of algorithm 2 with MultiOutputRegressor and beacon errors in meters.

Trajectory	Wrapper	test size	MAE	MSE	RMSE	MAPE
Straight line	MultiOutputRegressor	0.1	0.0001606	6.66exp-08	0.000258	36.116
Straight line	MultiOutputRegressor	0.1	0.000141	4.91exp-08	0.0002215	31.075
Straight line	MultiOutputRegressor	0.1	0.0001411	4.57exp-08	0.000214	30.505
Straight line	MultiOutputRegressor	0.1	0.000115	3.997exp-08	0.000199	15.956
Straight line	MultiOutputRegressor	0.1	0.000132	5.102exp-08	0.000226	18.054

Table 11: Accuracy scores of algorithm 2 with RegressorChain and beacon errors in meters.

Trajectory	Wrapper	test size	MAE	MSE	RMSE	MAPE
Straight line	RegressorChain	0.1	0.000112	3.04exp-08	0.000174	22.410
Straight line	RegressorChain	0.1	0.0001116	2.84exp-08	0.000169	19.459
Straight line	RegressorChain	0.1	0.000152	5.09exp-08	0.000225	32.189
Straight line	RegressorChain	0.1	0.000105	3.029exp-08	0.000174	17.035
Straight line	RegressorChain	0.1	0.000141	4.44exp-08	0.000211	29.869

The average of the MAPE with the MultiOutputRegressor is 26.341 and the average of the RegressorChain is 24.192. Meaning that the RegressorChain wrapper seems to perform slightly better then the MultiOutputRegressor. We see that the maximum RMSE of the RegressorChain is lower then with the MultiOutputRegressor. Therefore, we can say that the RegressorChain makes smaller mistakes in the prediction compared to the MultiOutputRegressor.

5.6 RQ6: corrections with learning algorithm 2 without beacon data

In the previous paragraph the performance of learning algorithm 2 was described where the beacon errors were part of the input feature set. The question came up what the performance would be without beacon errors as input feature to investigate what its influence would be on the performance. This analysis is done with experiment 12.

It appears that the SVR model with the multi output regressor wrapper and with beacons performed with a MAPE of 17%, while the same model with regressor chain wrapper without beacons performed with a MAPE of 17% as well.

The details will follow now.

We constructed a baseline method of predicting the position to see if there is an improvement. We used the same method (SVR) to predict a position again. However, we do not give the beacon-errors as input to the learning algorithm this time. This will predict a position only based on the GPS input. Suppose the accuracy of this prediction is worse than the accuracy of algorithm 2. In that

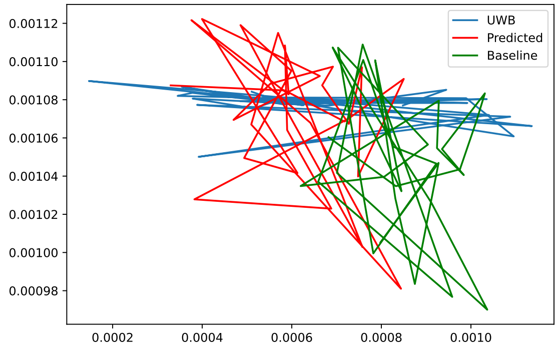

Figure 30: Baseline, predicted and UWB trajectory with RegressorChain with difference in meters

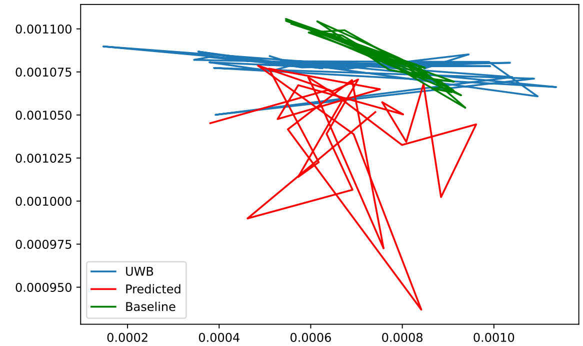

Figure 31: Baseline, predicted and UWB trajectory with MultiOuputRegressor with difference in meters

case, we can carefully say that algorithm 2 is possibly able to predict a position based on the errors of beacons and raw trajectory input. We chose test size of 0.1 again, while this resulted in one complete path from left to right and back. This resulted in Tables 12 and 13.

Table 12: Accuracy scores of multi output regressor with and without beacons.

Trajectory	Beacons	test size	MAE	MSE	RMSE	MAPE
Straight line	beacons errors	0.1	0.00013	4.96exp-08	0.00022	17.25
Straight line	no beacons errors	0.1	0.000279	1.69exp-07	0.000411	41.62

Table 13: Accuracy scores of regressor chain with and without beacons.

Trajectory	Beacons	test size	MAE	MSE	RMSE	MAPE
Straight line	beacon errors	0.1	0.00017	7.19exp-08	0.00027	37.78
Straight line	no beacon errors	0.1	0.00011	3.58exp-08	0.00019	16.57

These tables show that we do not see a significant performance difference between the baseline and the extended methods.

In Table 12, the baseline is performing worse, but in Table 13, the extended method is performing worse.
However, looking at Figures 30 and 31, we can see that the actual predicted path is turned towards the UWB trajectory with a vector shift. This could mean that the predictive model can predict the positions better after rotation and offset correction. As explained before, the grids of UWB and GPS are turned. This may cause the turn we see in the figures. However, the figures are zoomed in to make the graphs fit. This zoom causes a difference in scale for the x- and y-axis, and therefore a

different look. If we zoom out on these figures we see that the predicted point come closer to a straight line on the measured UWB.

6 Conclusions and Further Research

6.1 Conclusion

We can carefully claim that a learning algorithm can refine the position of a moving GPS receiver based on data from a set of location-fixed GPS beacons. On a small data set, our learning algorithm shows a number of more refined position points, but we do not see a convincing improvement between the baseline algorithm developed and the learning algorithm. This indicates that we can predict an accurate position based on a learning algorithm, but not explicitly caused by the data from the location-fixed GPS beacons (the beacon errors). The MAPE trajectory accuracy metric went from 39% to 16%, and so indicates that the learning model can predict a better position based on the location-fixed GPS beacons.

Further, the learning algorithm output gives a trajectory that better maps with the ground truth if the plot in this area is zoomed out and shifted to the right side. For the baseline algorithm, we saw that it could not handle inaccuracies in the beacon files. The trajectory was more accurate without the use of the location-fixed GPS beacon data, and when used the algorithm made the position accuracy worse.

Other findings from the accuracy analysis are that trees influence GPS accuracy of receivers negatively and that the longitude error was almost a factor 3 larger in an urban location (location 3) than in an open location (location 1). Additionally, we noticed that the beacon errors fluctuate between certain boundaries during the day.

6.2 Further research

Further research for the learning algorithm approach to refine positions with location-fixed GPS beacons is to enlarge the amount of the data combined with automated annotation with the UWB position system. It appears that the position error measured by the location-fixed GPS beacons varies during the day. The error is not drifting away but fluctuates within certain boundaries. Interesting could be to investigate the existence of generally applicable trends, applied to correct GPS position measures within shorter intervals.

Another improvement would be to extend the learning algorithm with data such as the orientation and velocity of the moving GPS receivers. When these variables are implemented, a Kalman filter could be used to compare both results and even refine the position further without feeding it to the learning model.

Another possibility for additional research is the use of the difference in latitude and longitude error. In this study, we observed a larger error in longitude compared to latitude. One could study these differences and make position predictions based on only one coordinate axis, to see what is best to compute an accurate position.

In Figure 27, we saw that the measured trajectory has a fixed vector different from the ground truth position grid. This could mean that a shift of the whole measured trajectory could lead to a trajectory closer to the ground truth. A study could examine this rotation and shift, and search for examples for which this method would work.

Finally, we have also found that beacons around trees tend to have significant inaccuracies in their measurements. Further research could investigate the relationship between environment and GPS accuracy.

References

[AK15] Mariette Awad and Rahul Khanna. *Support Vector Regression*, pages 67–80. Apress, Berkeley, CA, 2015.

[BHE00] Nirupama Bulusu, John Heidemann, and Deborah Estrin. Gps-less low cost outdoor localization for very small devices. *Personal Communications, IEEE*, 7:28 – 34, 11 2000.

[BRLK14] Bharati Bidikar, Gottapu Rao, Ganesh Laveti, and Mnvs Kumar. Satellite clock error and orbital solution error estimation for precise navigation applications. *Positioning*, 05:22–26, 01 2014.

[Cho20] Kaushik Choudhury. Chained multi-output regression solution with scikit-learn. 08 2020.

[DV19] M. De Vries. Multi-rate unscented kalman filtering for pose estimation. Master Thesis TU Delft, 02 2019.

[far21] Gps agriculture: How satellite farming is helping precision agritech. 03 2021.

[gps20] History of gps satellites and commercial gps tracking. 06 2020.

[Kal60] R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, pages 35–45, 1960.

[KEN18] Malek Karaim, Mohamed Elsheikh, and Aboelmagd Noureldin. Gnss error sources. In Rustam B. Rustamov and Arif M. Hashimov, editors, *Multifunctional Operation and Application of GPS*, chapter 4. IntechOpen, Rijeka, 2018.

[KK16] Sungil Kim and Heeyoung Kim. A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting*, 32(3):669–679, 2016.

[KMP10] Tomislav Kos, Ivan Markezic, and Josip Pokrajcic. Effects of multipath reception on gps positioning performance. In *Proceedings ELMAR-2010*, pages 399–402, 2010.

[KRK13a] G. Kumar, Gottapu Rao, and M. Kumar. Gps signal short-term propagation characteristics modeling in urban areas for precise navigation applications. *Positioning*, 04:192–199, 01 2013.

[KRK13b] G. Kumar, Gottapu Rao, and M. Kumar. Gps signal short-term propagation characteristics modeling in urban areas for precise navigation applications. *Positioning*, 04:192–199, 01 2013.

[Kum21] Satyam Kumar. Essential guide to multi-class and multi-output algorithms in python. 11 2021.

[MA19] Aigerim Mussina and Sanzhar Aubakirov. Improving indoor positioning via machine learning. pages 190–195, 01 2019.

[MBHW19] Adam Millard-Ball, Robert C. Hampshire, and Rachel R. Weinberger. Map-matching poor-quality gps data in urban environments: the pgmapmatch package. *Transportation Planning and Technology*, 42(6):539–553, 2019.

[MRU22] Peep Miidla, Kalev Rannat, and Penal Uba. Some problems in water vapor tomography. 05 2022.

[NK09] Paul Newson and John Krumm. Hidden markov map matching through noise and sparseness. pages 336–343, 01 2009.

[NLAF18] Sher Muhammad Nizamani, Noor-U-Zaman Laghari, Muhammad Ali, and Muhammad Farrukh. In this research i have reduced gps error by applying kalman filter. 09 2018.

[Nov15] Novatel. *An Introduction to GNSS,SecondEdition An Introduction to GNSS GPS, GLONASS, BeiDou, Galileo and other Global Navigation Satellite Systems.* NovAtel Inc, 2015.

[NT09] Masashi Nomura and Toshiyuki Tanaka. Gps positioning method under condition of only three acquired satellites. *SICE Journal of Control, Measurement, and System Integration*, pages 206–212, 2009.

[RR08] Marc Pollefeys Rahul Raguram, Jan-Micheal Frahm. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. *European Conference on Computer Vision*, pages 500–513, 2008.

[SA15] Mahdi Salarian and Rashid Ansari. Efficient refinement of gps-based localization in urban areas using visual information and sensor parameter. 02 2015.

[SD] Jan Van Sickle and John A. Dutton. Gps and gnss for geospatial professionals. The Tropospheric effect,dtrop.

[SK17] Susanna Lehvävirta Silviya Korpilo, Tarmo Virtanen. Smartphone gps tracking—inexpensive and efficient data collection on recreational movement. *Landscape and Urban Planning*, 157:608–617, 01 2017.

[SRAA21] Upma Singh, Mohammad Rizwan, Muhannad Alaraj, and Ibrahim Alsaidan. A machine learning-based gradient boosting regression approach for wind power production forecasting: A step towards smart grid environments. *Energies*, 14:5196, 08 2021.

[Tay17] Anadi James Taylor. The advantages of using gps data in football. 11 2017.

[TBS+21] Yaguang Tao, Alan Both, Rodrigo I. Silveira, Kevin Buchin, Stef Sijben, Ross S. Purves, Patrick Laube, Dongliang Peng, Kevin Toohey, and Matt Duckham. A comparative analysis of trajectory similarity measures. *GIScience & Remote Sensing*, 58(5):643–669, 2021.

[VD20] Frank Van Diggelen. Google to improve urban gps accuracy for apps. 12 2020.

[vDGvL23] Richard van Dijk, Daniela Gawehns, and Matthijs van Leeuwen. Wearda: recording wearable sensor data for human activity monitoring. preprint in arXiv:2303.00064v1, 02 2023.

[VS18] Jan Vascak and Igor Savko. Radio beacons in indoor navigation. pages 283–288, 08 2018.

[YYY21] Beiya Yang, Erfu Yang, and Leijian Yu. Vision and UWB-based anchor self-localisation system for UAV in GPS-denied environment. *Journal of Physics: Conference Series*, 1922(1):012001, may 2021.