

Real Time Fingerprint Positioning Library (RTFPPL) for Retail Phase II

Algorithmic Design Document

Ver 1.2

Document History

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Comment** |
| 17 Mar 2016 | 1.0 | Document created |
| 8 Apr 2016 | 1.1 | RTFPPL Input Data correction |
| 11 Apr 2016 | 1.2 | Added corrections about internal frame designation. |
| 21 June 2016 |  | Added uncertainties usage in RBPF |
| 5 July 2016 |  | Added position increment uncertainties |
| 21 November 2016 |  | Updated position increment uncertainties |
| 29 May 2019 |  | RBPF for 4 biases (mag calibration on robot) |

Table of Contents

[Document History 3](#_Toc10046084)

[Table of Contents 4](#_Toc10046085)

[1 Executive Summary 6](#_Toc10046086)

[2 Algorithms of RTFPPL 7](#_Toc10046087)

[2.1 Coordinate frames 7](#_Toc10046088)

[2.1.1 Device independent frames 7](#_Toc10046089)

[2.1.2 Fingerprint grids 10](#_Toc10046090)

[2.1.3 Device fixed frames 11](#_Toc10046091)

[2.1.4 Euler angles 14](#_Toc10046092)

[2.2 The RTFPPL Input data 15](#_Toc10046093)

[2.2.1 Initialization data – once input data 15](#_Toc10046094)

[2.2.2 Settings – once input data 16](#_Toc10046095)

[2.2.3 FP maps – once input data 17](#_Toc10046096)

[2.2.4 TPN input data 17](#_Toc10046097)

[2.2.5 RF input data 18](#_Toc10046098)

[2.3 The RTFPPL output data 19](#_Toc10046099)

[2.4 RTFPPL algorithm description 20](#_Toc10046100)

[2.4.1 RTFPPL block diagram 20](#_Toc10046101)

[2.4.2 Initialization 21](#_Toc10046102)

[2.4.3 Coordinate converter 21](#_Toc10046103)

[2.4.4 TPN data converter 23](#_Toc10046104)

[2.4.5 Motion model 24](#_Toc10046105)

[2.4.6 Wi-Fi-only position 26](#_Toc10046106)

[2.4.7 BLE-only position 27](#_Toc10046107)

[2.4.8 MFP-based position 27](#_Toc10046108)

[2.4.9 Combined position 28](#_Toc10046109)

[3 Algorithms of Uncertainties Utilization 29](#_Toc10046110)

[3.1 Use attitude angles uncertainties 29](#_Toc10046111)

[3.1.1 Magnetic vector uncertainty estimation 29](#_Toc10046112)

[3.1.2 Estimation of position increment uncertainty 31](#_Toc10046113)

[3.1.3 Algorithm of uncertainty usage in RBPF 33](#_Toc10046114)

[3.1.4 Alternative form of RBPF 38](#_Toc10046115)

[3.1.4.1 Alternative algorithm of RBPF 38](#_Toc10046116)

[3.1.4.2 Sampling from likelihood 39](#_Toc10046117)

[3.1.5 Algorithm of RBPF with 4 biases (for mag calibration on robot) 41](#_Toc10046118)

# Executive Summary

This report is the design documentation for the Real Time Fingerprint Positioning Library (RTFPPL) for the Retail Phase II project. It summarizes the ongoing work and the future work performed/or will be performed starting after the Proof of concept (POC) phase.

The overall architecture of the prospective library with interaction other modules is shown in picture below.



Figure 1 General diagram of Real Time Fingerprint Positioning Library

# Algorithms of RTFPPL

Algorithms of RTFPPL are designed to provide pure Wi-Fi only based position, pure BLE-only based position, MFP-based position, and combined position using data from INVN PDR.

## Coordinate frames

### Device independent frames



NED – North East Down frame with origin in device mass center.

– Filter Frame (FF) defined as: pointed up, axis belong to the horizontal plane and forms angle with North direction, completes right frame. Frame origin is defined so that all fingerprint cells have positive coordinates. The Filter Frame origin point (0, 0, 0) is located at the lowest floor among the floors, which are included in the venue map. All points at this floor in Filter Frame have coordinate value of zero. Note that this floor may be located at any height and it can have any number in the floor numbering system of the venue.

Note that only and coordinates in Filter Frame are in meters, while coordinates are integer floor numbers starting at 0. However, it is convenient to use the system with traditional metric coordinates along each of the 3 axes while calculating coordinate transformations. That’s why is implicitly converted to such system during all the transformations and changed to its true form with integer floor number coordinates only when it is necessary.

The transformation matrix from NED to Filter Frame is:

|  |  |
| --- | --- |
|  | 1 |

Where is the azimuth – angle between north direction and axis. If the Filter Frame is rotated by around axis, then one will coincide with NED frame. Note that this rotation happens around an axis that is pointed upwards in Earth frame. axis of NED and axis of Filter Frame are opposite.

Another frame is used in RTFPPL – quasi-NED frame. Note that this frame orientation relative to NED changes with time, therefore it is not an Earth fixed frame. At any given moment axis of coincides with D axis of NED and evolution in time is defined by frame misalignment angle . The transformation matrix from to NED is:

|  |  |
| --- | --- |
|  | 2 |

Where is misalignment angle between NED and quasi-NED frames.

If the quasi-NED system at the time moment t is rotated around its axis by angle , then the resulting rotated quasi-NED will coincide with NED.

The transformation matrix from quasi-NED to Filter Frame is defined as:

|  |  |
| --- | --- |
|  | 3 |

Quasi Filter Frame (qFF) is used in internal calculations of particle filter. This frame orientation relative to Filter Frame changes with time, like relative to NED. At any given moment axis coincides with axis NED and qFF evolution relative to FF in time is defined by frame misalignment angle . Quasi Filter Frame relates also with frame by azimuth angle that is equal to azimuth angle between Filter Frame and NED as defined in picture below.



The transformation matrix from quasi Filter Frame to Filter Frame is:

|  |  |
| --- | --- |
|  | 4 |

Where is misalignment angle between Filter Frame and quasi-Filter Frame.

|  |  |
| --- | --- |
|  | 5 |

Transformation matrix from quasi-NED to Filter Frame can be also expressed by the misalignment angle and the azimuth angle as:

|  |  |
| --- | --- |
|  | 6 |

According to equations (3) and (6) misalignment angles and are equal.

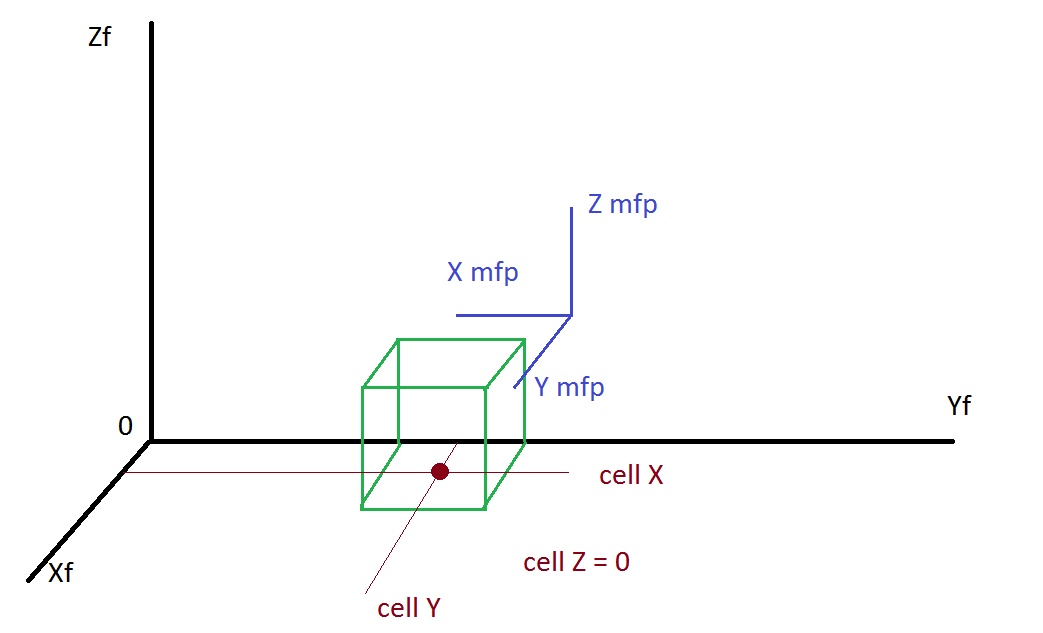
### Fingerprint grids

For each venue and measurement type a corresponding fingerprint database can be used by RTFPPL. The data in a fingerprint database is tied to its grid. Each grid consists of cells. The cell size can be set differently for each venue and fingerprint type.

Each fingerprint grid has a 2d array of cells for each floor in the venue map. Each cell has coordinates in the Filter Frame, which are tied to the center of the cell.

The MFP data for each cell is a 3d vector in a Magnetic Fingerprint Frame (see 2.1.1).

Note that the MFP cells still have coordinates in Filter Frame, while Magnetic Fingerprint Frame is only used to store 3 components of magnetic vector measurements in the cell. The following figure shows a MFP cell and the axis directions for each frame.



The transformation matrix from Filter Frame to MFP frame is defined as:

|  |  |
| --- | --- |
| = | 7 |

The transformation matrix from NED to Magnetic Fingerprint Frame is defined as:

|  |  |
| --- | --- |
|  | 8 |

### Device fixed frames

All sensor measurements are provided by OS sensor API in Sensor Coordinate System.

|  |  |
| --- | --- |
|  |  |
| Device Frame | Universal Device Fixed Frame |

XdYdZd – Sensor Coordinate System (Device Frame). Axes of the frame usually match to measurement axes of sensors. The frame defined as: origin in device mass center, Yd axis pointed to device top side, Zd axis pointed up from device screen, Xd – completes right coordinate frame. This frame is not used by TPN PDR, the transformation happens before the data reach the TPN library.

XYZ – Universal Device Fixed (UDF) frame. The frame defined as: origin in device mass center, X axis pointed to device top side, Z axis pointed down from device screen, Y – completes right coordinate frame. UDF Frame is used for all input and output of TPN PDR.

Transition from Device Frame to UDF frame is defined as the following:

|  |  |
| --- | --- |
|  | 9 |

Hire and below in document liter denotes transformation matrix, liter denotes transformation quaternion.

To avoid ambiguity of heading angle, TPN PDR provides Euler angles between quasi-NED frame and one of three different Internal Device frames. Frame ID is also given. These frames are defined as described below.

1. Horizontal frame. This frame is equal to Universal Device Fixed Frame.

|  |  |
| --- | --- |
|  |  |
| Universal Device Fixed Frame | Horizontal frame |

Transition matrix and frame ID defined as:

|  |  |
| --- | --- |
|  | 10 |

1. Vertical Up frame. This frame is obtained by 90 degrees clockwise rotation of Universal Device Fixed Frame around Y axis.

|  |  |
| --- | --- |
|  |  |
| Universal Device Fixed Frame | Vertical Up frame |

Transition matrix and frame ID defined as:

|  |  |
| --- | --- |
|  | 11 |

1. Vertical Down frame. This frame is obtained by 90 degrees counter clockwise rotation of Universal Device Fixed Frame around Y axis.

|  |  |
| --- | --- |
|  |  |
| Universal Device Fixed Frame | Vertical Down frame |

Transition matrix and frame ID defined as:

|  |  |
| --- | --- |
|  | 12 |

### Euler angles

Internal Device Frames (Horizontal, Vertical Up, or Vertical Down) relates with quasi NED frame by three Euler angles:

– is roll angle, counterclockwise rotation around X axis;

– is pitch angle, counterclockwise rotation around Y axis;

– is heading angle, counterclockwise rotation around Z axis.

This angles define orientation of Internal Device Frame relative to quasi-NED frame.

Rotation of Internal Device Frame (Horizontal, Vertical Up, or Vertical Down) to quasi-NED frame is performed in following order:

1-st rotation – roll around X, counter clockwise

2-nd rotation – pitch around Y, counter clockwise

3-d rotation – heading around Z, counter clockwise

Transformation matrix from Internal Device Frame to qNED frame is defined as:

|  |  |
| --- | --- |
|  | 15 |

Conversion between UDF frame and quasi NED frame can be performed by following transition matrix:

|  |  |
| --- | --- |
|  | 16 |

## The RTFPPL Input data

### Initialization data – once input data

* + Initial position and floor with uncertainties
    - Latitude, degrees
    - Longitude, degrees
    - Height from sea level, m
    - Floor
    - Latitude/Longitude covariance matrix (2x2)
    - Floor deviation – floor ambiguity
    - Floor ambiguity flag (-1: floor down only, 1: floor up only; 0 – floor up and down)
    - Height standard deviation, m
  + Initial heading with uncertainty
    - Initial heading of person, degrees – angle between N and User X axis (counter clockwise)
    - Initial heading standard deviation, degrees

### Settings – once input data

* + Venue data
    - Venue ID
    - Origin latitude – bottom left corner latitude, degrees
    - Origin longitude, – bottom left corner longitude, degrees
    - Origin altitude, m
    - Venue azimuth degrees
    - Floor count – floor number in venue
    - X-size of venue, m – maximal available x coordinate in Filter Frame
    - Y-size of venue, m – maximal available y coordinate in Filter Frame
    - Latitude scale factor
    - Longitude scale factor
    - Floor height separation, m (array, TBD in future)
  + Magnetic sensor uncertainties (TBD)
  + Wi-Fi and BLE measurement uncertainties (TBD)
  + Wi-Fi/BLE/Mag/Mix position enable flags (TBD)
  + Combined position flag (MFP+Wi-Fi+BLE), (TBD)
  + MFP position flag – enables MFP position output and accounting of MFP in combined position
  + WFP position flag – enables WiFi FP position output and accounting of WiFi FP in combined position
  + BLE position flag – enables BLE FP position output and accounting of BLE FP in combined position

### FP maps – once input data

* + MFP map, (format - TBD), now mfp3 file is used
  + Wi-Fi FP map, (format - TBD), now wfp3 file is used
  + BLE FP map, (format - TBD), now bea3 file is used

### TPN input data

TPN input data include motion, attitude and magnetic data. TPN data are periodical data with sample rate of 20 Hz. These data are provided by TPN PDR and include the following:

* + Time stamp, sec
  + TPN position data
    - Latitude, degrees
    - Longitude, degrees
    - Position standard deviation in north direction, m
    - Position standard deviation in east direction, m
    - User heading angle, degrees
    - Misalignment angle, degrees
    - User heading standard deviation, degrees; misalignment standard deviation proposed as: , - empirical coefficient
    - Floor number
    - Height from sea level, m
    - Height standard deviation, m
    - Fidgeting, flag
  + TPN attitude data
    - Euler angles, i.e. roll, pitch, heading of device for conversion from Internal Device Frame to initial NED, degrees
    - Roll, pitch, heading standard deviation, degrees, provided for roll, pitch, heading of device for conversion from Internal Device Frame to initial NED
    - Internal Device Frame identifier: (0 – Horizontal frame, 1 – Vertical Up frame, -1 – Vertical Down frame)
  + TPN magnetic data
    - Uncalibrated magnetic vector in Universal Device Fixed Frame:

A dimension of each vector element is .

* + - Covariance matrix of magnetometer bias estimation errors:

Index *l* denotes different sample rate of these data with sample rate of magnetic vector.

A dimension of each matrix element is .

* + - Standard deviations of magnetometer measurements noise. A dimension is .
    - Magnetic data validity flag

### RF input data

* Wi-Fi observation, sample rate: 0.3-5Hz
  + Time stamp of observation
  + Wi-Fi observation vector = <Wi-Fi observation item>

Wi-Fi observation item

* + - BSSID
    - RSSI, dBm
    - Central channel frequency, MHz
    - Time stamp\* of measurement (optionally)
* BLE data (0.3-5Hz)
  + Time stamp of observation
  + BLE observation vector = < BLE observation item >

BLE observation item

* + - UUID
    - Major
    - Minor
    - txPower - tx Pover level for 1m distance, dBm
    - BSSID validity flag
    - BSSID
    - RSSI, dBm
    - Central channel frequency, MHz

Time tags of Wi-Fi and BLE data should be the same as of PDR data

## The RTFPPL output data

The RTFPPL calculates and outputs following data items:

* Pure BLE position
* Pure Wi-Fi position
* MFP based position
* Combined position

Each output item contains data in following format:

* Position validity
* Latitude, in degree
* Longitude, in degree
* Latitude standard deviation
* Longitude standard deviation
* Floor number
* Floor number uncertainty

## RTFPPL algorithm description

### RTFPPL block diagram

Figure shows a block diagram of RTFPPL.



Figure 2 Block Diagram of Real Time Fingerprint Positioning Library

### Initialization

The algorithm input is data according section 2.1.

The algorithm can be summarized in the following steps:

Step 1:

Step 2:

Step 3:

The algorithm output is

### Coordinate converter

Main function of Coordinate Converter is a conversion geographical coordinates to coordinates of a local 2D frame and vice versa. Additionally, CC provides conversion horizontal plane projections (2D) of vectors and covariance matrixes between the both frames.

Vectors

Local frame is defined as:

* frame origin sets in point with latitude and longitude ;
* axis belong to the horizontal plane and forms angle with North direction;
* axis belong to the horizontal plane and forms angle 90 degrees with ;
* Coefficients and define scaling and axis directions.

Parameters , , , , define the conversion and have to be previously specified.

Parameters , are calculated by default if they are specified as zeroes:

|  |  |
| --- | --- |
|  | 17 |

Where and are major and minor Earth semi-axes respectively.

Conversion of latitude and longitude to local frame coordinates performed as following:

|  |  |
| --- | --- |
|  | 18 |

Latitude and longitude are calculated by local frame coordinates as it is defined bellow:

|  |  |
| --- | --- |
|  | 19 |

Vectors are converted as:

|  |  |
| --- | --- |
|  | 20 |

Where - vector in local frame;

- vector in geographical frame, are north and east vector projection;

- is transformation matrix between geographical coordinates and local coordinates defined as:

|  |  |
| --- | --- |
|  | 21 |

Covariance matrix transformation is performed as:

|  |  |
| --- | --- |
|  | 22 |

Where – covariance matrix in local frame;

– covariance matrix in geo frame, *n* and *e* indexes denote north and east components.

According to definitions above CC is used in RPFPPL for geographical coordinates and its covariance matrix conversion to coordinates of a Filter Frame and quasi Filter Frame and vice versa. Local frame represents Filter Frame or quasi Filer Frame accordingly.

### TPN data converter

TPN data converter provides conversion TPN input data to internal RTFPPL data format.

A diagram of TPN converter is given below.



Position data conversion is provided according to equations given in item 2.5.3.

Attitude data conversion consists calculation of transformation quaternions:

|  |  |
| --- | --- |
|  | 23 |

Where – is transformation quaternion from quasi NED to quasi Filter Frame that can be calculated from transition matrix (equation ),

– is transformation quaternion from UDF to current Internal Device frame, that that can be calculated from transition matrix (equations 10-12 ),

– is transformation quaternion from Internal Device frame to quasi NED, that that can be calculated from transition matrix (equation 15).

Euler angles are defined as:

|  |  |
| --- | --- |
|  | 24 |

Magnetic data conversion provides as follow

|  |  |
| --- | --- |
|  | 25 |

Where - is relation factor between [mG] and [uT] dimensions of magnetic field representation.

### Motion model

Motion model provides updates to the position and heading of each particle. Those updates are made in the propagation function.

Motion model propagation function inputs are:

Coordinate increments estimations,

– floor change estimation as a floating point value.

Standard deviation of -

Error covariance matrix for :

The coordinate increments are received in the quasi-NED frame. One of the fusion filter estimated parameters is the angle between quasi-NED frame and the Filter Frame. If the X axis of FF is rotated by around Z axis of FF, then it will coincide with North in quasi-NED.

The transformation matrix from quasi-NED frame to Filter Frame is:

Noise is added to the input coordinate increments:

=

Where is calculated based on .

Then 2x2 submatrix of is used to convert to Filter Frame:

The propagation function yields the following updates for each particle:

Here is random value generated from a 1d normal distribution with 0 mean and variance equal to 1.

### Wi-Fi-only position

Position estimation is obtained from the Wi-Fi FP by comparing current Wi-Fi RSSI measurements to the data in the FP. The main problem is choosing the most probable position.

At first the likelihood function values are calculated for each cell in the fingerprint. Those values represents the likelihood of getting measurements vector if the device was located inside that cell.

Consider an example where a Wi-Fi RSSI measurements vector is received for access points. The goal is to calculate the value of likelihood function for the FP cell - .

Two Gaussian mixture model is used to calculate the likelihood function for one RSSI measurement .

Here values , are taken from Wi-Fi FP database.

The final value of likelihood function for the whole vector of RSSI measurements and for cell is calculated as follows:

Here parameter is some constant parameter.

Then “most probable” cells are chosen, which all have higher value of the likelihood function than the rest of the cells. Then a position estimate is calculated based on the values of likelihood function for those cells. The weight of each cell is calculated as follows:

If , then position estimate validity flag will be set to *false.* Otherwise it will be set to *true.*

The position estimate is:

Current floor number estimate is floor number where the largest number of cells from among the “most probable” cells are located.

### BLE-only position

BLE based positioning is provided by same way as WiFi based positioning.

### MFP-based position

(info from doxygen, TODO: correct this)

3D vector magnetic updater implementation.

The likelihood function looks as follows: where - is a true magnetic induction vector and is a PF state vector.  
We assume the following magnetometer measurement model: , where - measured vector, is orientation matrix and - sensor bias.  
We decompose orientation matrix as follows

Where - phone to the map plane transformation matrix and - in plane rotation, which depends on heading hypothesis.  
So the final

and assuming independency of the field components

It is assumed that after bias compensation, each of components is a Gaussian distributed value with zero mean and standard deviation defined by estimated bias uncertainty.

### Combined position

# Algorithms of Uncertainties Utilization

## Use attitude angles uncertainties

### Magnetic vector uncertainty estimation

Definitions:

|  |  |
| --- | --- |
| Definition | Meaning |
|  | MFP vector |
|  | Roll angle (rotation around X axis) |
|  | Pitch angle (rotation around Y axis) |
|  | Misalignment angle between Filter Frame and quasi-NED (rotation around Z axis) |
|  | Variance of attitude angles |
|  | User heading (misalignment angle plus heading) |
|  | Variance of user heading |

Assume: ,

Consider DCM

,

where is a non-random part of DCM rotating from MFP frame to UDF, defined as:

matrices are defined in section 2.1.

is a random part of DCM, which describes angular errors.

For small error angles (substituting sine by angle, considering cosines as unities, and ignoring terms of second order) we can write:

(26)

Rotated magnetic vector from MFP frame to UDF

(27)

where

Mathematical expectation of is

(28)

Covariance of is

(29)

Where

= , (30)

Finally, formula (4) can be rewritten as follows:

(31)

As soon input uncertainties are given in internal frames, we transform to the UDF them according to the table ( see below):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Angles | Uncertainties in internal frames | | | Uncertainties in UDF frame |
| Horizontal frame | Vertical Up frame | Vertical Down frame |
| Roll |  |  |  |  |
| Pitch |  |  |  |  |
| Heading |  |  |  |  |

### Estimation of position increment uncertainty

Position increment can be defined as follows:

where is a distance travelled from time to ,

is user (or platform) heading.

We consider and as independent and normally distributed variables:

where and are mean values of the distance and the user heading, and are their variances.

Position increment uncertainty, i.e. covariance of , is defined as follows:

where .

Due to independency of and , the term (1,1) of the covariance matrix can be computed as follows:

Terms containing can be easily determined as follows:

Terms containing can be determined using a known integral:

Using this equation, we can achieve:

Two forms of position increment covariance will be considered further. First, this is a case of small user heading variance , the second is the case of a big user heading variance .

First of all, we consider the first case of a small user heading variance. Substituting the last expressions into formula for and accounting that

for , we can get:

 Analogously we can determine other terms of the covariance matrix. Finally, we can get the covariance matrix of position increment:

In the case or a big user heading variance, terms with negative user heading variance in the exponent became small and we can neglect them. Thus, we can get the following position increment covariance matrix, which expectedly does not depend on user heading:

In the particular case of user stopping, when , the last equation is transformed to the following covariance matrix:

Thus, it is possible to adapt the position increment covariance matrix based on the following input variables:

* user heading standard deviation ,
* user (or platform) heading ,
* indicator of user stopping.

### Algorithm of uncertainty usage in RBPF

The state vector consists of two parts

where the denotes the state variable with linear dynamics and denotes nonlinear state variable.

The former state variable contains three biases of a magnetometer:

The latter state variable contains coordinates of a user:

where and are Cartesian coordinates, is a misalignment angle between Filter Frame and quasi-NED, and is a floor.

At time , the -th particle, is the number of particles, is defined as , where is a covariance of .

At time , the input variables are as follows:

and are increments of Cartesian coordinates (in fingerprint frame),

and are position standard deviation in NED (ENU) frame,

are raw magnetometer readings, ; is the number of readings between two consecutive moments of time, and ,

are roll, pitch, and heading correspondingly,

, , and are standard deviation of heading, pitch and roll correspondingly,

is a variance of magnetometer measurement noise (assumed to be equal for each axis),

is the MFP grid; every cell contains a mean vector and a covariance matrix of normal distribution. The covariance matrix is diagonal:

The state model (in BF) can be defined by the following equations:

,

,

,

Where:

is a noise of magnetometer bias estimate,

is a noise of a magnetic fingerprint map,

is noise of a magnetometer,

is DCM rotating from MFP to UDF.

The algorithm of RBPF is as follows (without floor estimation):

1. Compute position uncertainties in the Filter Frame (FF):

where:

is the position increment covariance in the NED frame (see the previous section),

.

The distance can be either computed as travelled from time to or set as averaged step length. The distance variance can be set as a constant.

1. Prediction of in PF:

,

,

,

,

,

.

1. Prediction of in KF:

,

(predicted linear state),

(predicted state covariance),

(predicted measurement covariance),

where the covariance matrix is defined as follows:

(predicted measurement),

where is a fingerprint of a cell corresponding to coordinates of -th particle,

.

1. Update of in PF:

,

where averaging is applied for all .

A simplification is possible if we assume that the predicted measurement covariance does not depend on , i.e. . Then

where is covariance of .

Neglecting the last term, we have the following equation:

,

where and are averaged of .

1. Normalize weights .
2. Resample
3. Update of in KF:

,

,

.

1. Output positon and uncertainty estimate:

where .

1. Simplification of averaging in step 4 is possible if matrix is rewritten in the following form:

Due to a special form of matrix , we can swap two last matrixes:

In the right part, only the last matrix depends on index of a particle , whereas four preceding matrixes depend on index of readings . Let us insert the following depictions:

As a result, we achieve:

Now we can rewrite predicted measurement as follows:

In the last equation, only matrix depends on . This allows simplifying averaging by (see step 4):

Thus, we only need to average by magnetic measurements and the matrix . These operations can be realized in advance, before call of the particle filter.

1. Adaptive RBPF serves for estimating the measurement error matrix . Now a particle is defined as follows: .

Innovations are used for adaptation:

Covariance matrix of innovations includes covariance of the bias noise, covariance of the measurement noise, and MFP map uncertainty:

Estimate of measurement error covariance matrix can be derived from the previous equation:

where is the length of moving window for estimation of the innovation covariance matrix.

To prevent obtaining non-positive definite matrix due to the matrix subtraction, minimal values of diagonal elements of must be limited by some positive values.

### Alternative form of RBPF

### Alternative algorithm of RBPF

An alternative algorithm of RBPF is as follows:

*For m = 1:M*

*End*

*For m = 1:M*

*If*

*Else*

*End*

*End*

In this algorithm, definitions are like in previous sections with the following exceptions:

is a part of particles that are generated from likelihood (mixing ratio),

.

This algorithm generates particles traditionally from a motion model whereas particles are generated alternatively, from likelihood. can be 0.05-0.1.

### Sampling from likelihood

The most difficult part of the alternative algorithm is sampling from likelihood. The likelihood can be written as follows:

where

We use a simplified sampling procedure:

1. Set an area around weighted average of , say 10 by 10 cells of MFP map, and number every cell of the area,
2. Find heading for every cell:

where are coordinates of the center of the cell ,

,

is a weighted average bias of a magnetometer.

1. Calculate likelihood for every cell,

where

1. Normalize set of :
2. Generate particles using normalized weights . Particle sampled with weight has the following components:

Further modifications of the algorithms can be as follows:

* Blur of and using uniform distribution with spread of cell size; blur using normal distribution with as a mean value; variance of will be tuned by simulation.
* For economy of computations, execute the second part of the algorithm (sampling) once per several calls.

### Algorithm of RBPF with 4 biases (for mag calibration on robot)

This section describes difference from a basic RBPF algorithm of section 3.1.3.

The state vector consists of two parts

where the denotes the state variable with linear dynamics and denotes nonlinear state variable.

The linear state variable contains three biases of a magnetometer and one bias of Z component of MFP DB:

The biases of a magnetometer are in device frame whereas the bias of Z component of MFP is in FP frame.

1. Prediction of in KF:

,

(predicted linear state),

(predicted state covariance, 4x4),

(predicted measurement covariance),

where (3x4 matrix)

We assume that the predicted measurement covariance does not depend on , i.e. .

(predicted measurement),

Applying averaging as was described before, we get the following equation:

1. Update of in PF:

,

where and are averaged magnetometer reading and averaged.

1. Update of in KF:

, (KF gain, 4x3 matrix)

,

.