Article:

Wang, P., Mihaylova, L. orcid.org/0000-0001-5856-2223, Bonnifait, P. et al. (2 more authors) (2021) Feature-refined box particle filtering for autonomous vehicle localisation with OpenStreetMap. Engineering Applications of Artificial Intelligence, 105. 104445. ISSN 0952-1976

https://doi.org/10.1016/j.engappai.2021.104445

Feature-refined Box Particle Filtering for Autonomous Vehicle Localisation with OpenStreetMap

Peng Wang[a,*], Lyudmila Mihaylova[a], Philippe Bonnifait[b], Philippe Xu[b], Jianwen Jiang[c]

[a] *Dept. of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield S10 3JD, United Kingdom*
[b] *Université de Technologie de Compiègne, CNRS, Heudiasyc, 60 203 Compiègne, France*
[c] *Dept. of Automation, University of Science and Technology of China, Hefei, 230027, P.R.China*

Abstract

Vehicle localisation is an important and challenging task in achieving autonomous driving. This work presents a box particle filter framework for vehicle self-localisation in the presence of sensor and map uncertainties. The proposed feature-refined box particle filter incorporates line features extracted from a multi-layer Light Detection And Ranging (LiDAR) sensor and information from OpenStreetMap to estimate the vehicle state. A particle weight balance strategy is incorporated to account for the OpenStreetMap inaccuracy, which is assessed by comparing it to a high definition road map. The performance of the proposed framework is evaluated on a LiDAR dataset and compared with box particle filter variants. Experimental results show that the proposed framework achieves respectively 10% and 53% localisation accuracy improvement with reduced box volumes of 25% and 41%, when compared with the state-of-the-art interval analysis based box regularisation particle filter and the box particle filter.

Keywords: `Localisation, Box particle filtering`, `Autonomous vehicles`, `Information uncertainty`, `OpenStreetMap`

*Corresponding author
Email addresses: `Peng.Wang@Sheffield.ac.uk` (Peng Wang), `l.s.mihaylova@Sheffield.ac.uk` (Lyudmila Mihaylova), `philippe.bonnifait@hds.utc.fr` (Philippe Bonnifait), `philippe.xu@hds.utc.fr` (Philippe Xu), `jjwen@mail.ustc.edu.cn` (Jianwen Jiang)

1. Introduction

The development of reliable autonomous driving solutions is an active research area (Pendleton et al., 2017, Reid et al., 2019). Localisation plays a key role of autonomous systems since it provides the vehicle with self-awareness of its state $\mathbf{x}_k = (x_k, y_k, \theta_k)^T$, which encodes its position (x_k, y_k) and its orientation θ_k relative to a map (Kuutti et al., 2018) at time k.

There are mainly two main types of maps used for localising a vehicle: (1) maps that are incrementally built and maintained along with localisation; (2) accurate commercial digital maps that are built and maintained by companies. *Feature maps* (Holỳ, 2018) and *point-cloud maps* (Javanmardi et al., 2019, Tamas and Goron, 2014) belong to the first group. In general, feature maps represent the environment with geometrical features at various levels; whereas point-cloud maps are usually built by registering point clouds to a geographic information system (Zhang and Singh, 2014). The former is known for its semantic interpretability and low complexity, while the latter is computational resources dependant and holds the promise of high accuracy. These two types of maps are generally components of simultaneous localisation and mapping (SLAM) solutions (Gil et al., 2015, Li et al., 2019). Therefore, they inherit the challenges faced by SLAM approaches, such as localisation accuracy degradation when the uncertainty of sensor measurements increases. In such cases, loop closure and subsequent optimisation techniques are widely adopted to respectively improve the mapping accuracy (Wang et al., 2016) and the localisation results. However, a unified SLAM framework for mapping and maintaining high accuracy is still difficult, causing the inaccuracy of the obtained maps.

The second group of accurate digital maps comes with service charges and often limited access to metadata. The crowdsourced OpenStreetMap (OSM) could be a cheaper replacement for expensive digital maps and can provide flexible solutions since almost all the metadata can be accessed and customised by end-users. OSM has already been applied in urban navigation (Suger and Burgard, 2017). However, the accuracy of OSM remains a challenge (Vargas-

Munoz et al., 2021). It could vary from centimeters to meters from city to city, thus bringing in additional uncertainties apart from those caused by sensors (Brovelli et al., 2016, Senaratne et al., 2017). Hence, it is beneficial to develop robust localisation frameworks when using inaccurate SLAM maps or OSM for autonomous systems.

In this paper, a feature-refined BPF (FRBPF) that stems from the box particle filtering (BPF) approach (Abdallah et al., 2008, Gning et al., 2013) is proposed to achieve accurate and robust localisation results based on an inaccurate yet free OSM. In our case, the OSM serves as a reference map for localising a vehicle. A real-time kinematic (RTK) sensor suite provides the ground truth information. The vehicle is equipped with an Inertial Measurement Unit (IMU) and a LiDAR sensor to fulfill localisation.

In the proposed approach, line features are firstly extracted from raw LiDAR data obtained by the vehicle at time k. The distances and angles of the line features with respect to the vehicle are adopted as measurements and are denoted as $[\mathbf{y}_k]$. The line features are next associated with line features corresponding to building footprints on the OSM. The accuracy of the OSM is assessed with respect to a high-definition map (HDM) maintained by the Université de Technologie de Compiègne (UTC) so that map uncertainties are also considered during localisation. Measurements of the matched line features are fed to the proposed FRBPF for vehicle state updating. With N box particles representing the vehicle states $[\mathbf{x}_k^i]$, $i = 1, \cdots, N$ and upon the arrival of measurements $[\mathbf{y}_k]$, the filter propagates state estimates through the box contraction and update steps as time evolves. In contrast of performing contraction per measurement, a feature-refined contraction merges line features before the contraction step. This is also a way of coping with OSM and sensor data uncertainties, henceforth reduces boxes. As interval analysis based methods do not provide point estimates by nature, this paper takes box centers to achieve point estimates by statistical metrics such as expectation and covariance to evaluate the proposed FRBPF and compares it with the BPF and the box regularised particle filter (BRPF) (Merlinge et al., 2019).

The main contributions of this paper can be summarised as follows: (1) A LiDAR features-refined box particle filter is proposed that is able to deal effectively with OSM and sensor data uncertainties; (2) A contraction algorithm is developed that incorporates the abundant line features from structured urban environments to reduce the volume of box particles; (3) Theoretical proofs about the features-refined contractions are derived; (4) A box particle weight balance strategy is designed to cope with OSM uncertainties and further improves the localisation performance.

The rest of this paper is organised as follows. Section 2 presents an overview of related works. Section 3 gives the necessary theoretical background knowledge. Section 4 elaborates the proposed approach. Section 5 includes validation and discussions of the proposed approach. Finally, conclusions and future works are given in Section 6. Appendix A and Appendix B prove that the feature-refined contraction reduces the particle box volume compared with the traditional contraction.

2. Related Works

2.1. OpenStreetMap based Localisation

OSM is the most well-known crowdsourced map whose metadata is structured by entities such as *nodes*, *ways*, and *relations* (Zheng and Izzat, 2018). *Nodes* represent points of interest. *Ways* are a collection of *nodes* that correspond to buildings and roads, and *relations* indicate the relationships between *nodes* and *ways*. Generally, the exterior surface of buildings can be projected into a two dimensional (2D) plane as line segments or can be approximated by line segments. Hence, the OSM is equivalent to a feature map represented by a set of linear equations.

Compared with highly precise maps, maintained by local authorities, the OSM accuracy needs to be further improved. For instance, building footprints of Milan on OSM show a systematic translation of 0.4 m on the defined X and Y directions in (Brovelli et al., 2016). Furthermore, applications of OSM still suffer from the incompleteness of buildings, roads and other environmental factors (Senaratne et al., 2017).

Nevertheless, OSM has been widely used in vehicle/robot localisation. Suger and Burgard (Suger and Burgard, 2017) present a Markov Chain Monte Carlo approach for autonomous robot navigation, by associating track information from OSM with trails detected by the robot based on three dimensional (3D) LiDAR data. The robustness of the approach is demonstrated with experimental results, which shows the potential of using inaccurate OSM in urban environments. Zheng and Izzat (Zheng and Izzat, 2018) show that by taking OSM as a prior map, one can benefit from road perception by first rendering a virtual street view, and further refining it to provide prior road masks. The road mask can be augmented into drivable space by integrating images or LiDAR point clouds. By taking the road mask as image inputs to a fully convolutional neural network, the authors also discuss the promise of deep learning methods combined with OSM for road perception. Joshi and James (Joshi and James, 2015) propose to combine coarse, inaccurate prior maps from OSM with local sensor information from 3D LiDAR to localise a vehicle. Lane locations are estimated by particle filter variants and then integrated within a map to further improve the localisation accuracy.

2.2. Box Particle Filtering based Localisation

Recently, interval analysis based localisation has shown its potential in dealing with non-Gaussian and biased noise perturbed measurements. The combination of the set-membership framework with particle filtering techniques known as BPF is first introduced by Abdallah et al (Abdallah et al., 2008) to localise a ground vehicle. The application of the BPF to global localisation shows that with only 10 box particles, BPF reaches almost the same accuracy as particle filter with 3,000 particles.

Ever since then, BPF has been applied to different scenarios. Gning et al. (Gning et al., 2012) introduced the Bernoulli BPF and applied it to tracking a single target. It shows that the Bernoulli BPF can track the target accurately and is computationally more efficient compared with the Bernoulli particle filter. A multiple extended object tracking method based on BPF is further proposed by Freitas et al. (Freitas et al., 2018), which benefits from the fact that BPF can well tackle ambiguous observations, which often happens in LiDAR and GPS data. Merlinge et al. (Merlinge et al., 2019) propose the BRPF that outperforms the BPF in terms of Root Mean Square Errors (RMSEs). The BRPF achieves up to 42% improvement in geographical position estimation compared with BPF. The authors also demonstrate that both BRPF and BPF produce lower divergence rate ($\leq$ 1%) than methods such as particle filters. Luo et al. (Luo and Qin, 2018) propose the ball particle filter to deal with issues caused by box subdivision and forward-backward contraction. In the ball particle filter, boxes in BPF are replaced by balls, and a ball contractor is proposed to contract the balls. Applications of the ball particle filter in SLAM show that with 20 particles, the ball particle filter achieves 34.5% and 34.6% position and orientation improvement, respectively. However, the results show that the ball particle filter is about 7% less efficient than the BPF. Nevertheless, all the methods perform contraction when a measurement is obtained, without further integration or refinement. Furthermore, the BPF has not been applied to OSM based localisation.

3. Theoretical Background

3.1. Boxes and Inclusion Functions

In interval analysis, intervals or boxes are used as basic operands for modeling and calculation, etc. An interval or box is defined as

$$[\mathbf{x}] = \big([x_1], \cdots, [x_i], \cdots, [x_d]\big)^T \in \mathbb{IR}^d,$$

where $[x_i] = [\underline{x}_i, \overline{x}_i]$ with $\underline{x}_i$, $\overline{x}_i \in \mathbb{R}$, and $\forall x_i \in [x_i]$, $\underline{x}_i \leq x_i \leq \overline{x}_i$ stands. $\mathbb{IR}^d$ and $\mathbb{R}$ are respectively the $d \in \mathbb{N}^+$ dimensional real interval space and the real number space (Alefeld and Mayer, 2000). When $[\mathbf{x}]$ is one dimensional, it is usually called an interval, and it is called a box when the dimension is two or above. This paper adopts 'box' to refer to both intervals and boxes hereafter for brevity. The volume of a box is defined as $|[\mathbf{x}]| = \prod_{i=1}^{d} |[x_i]|$, where $|[x_i]| = \overline{x}_i - \underline{x}_i$ (Ilog, 1999). Note when $d = 1$, 'volume' refers to the size of the one dimensional interval, and when $d = 2$, it refers to the area of the two dimensional box. For brevity and generality, this paper uses 'volume' to refer to all the scenarios, unless otherwise specified.

Given boxes $[\mathbf{x}]$, $[\mathbf{y}]$, and an operator $\diamondsuit \in \{+, -, \cdots, /\}$, $[\mathbf{x}]\diamondsuit[\mathbf{y}]$ is defined as the smallest box in terms of volume that contains all feasible values of $\mathbf{x}\diamondsuit\mathbf{y}$. For a given box $[\mathbf{x}]$, its center is defined as $\mathbf{c_x} = \big((\underline{x}_1 + \overline{x}_1)/2, \cdots, (\underline{x}_d + \overline{x}_d)/2\big)^T$ (Drevelle and Bonnifait, 2013).

In general, when applying a function $\mathbf{f} : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ $(d_1, d_2 \in \mathbb{N}^+)$ that is defined in the real number space directly to manipulate a box $[\mathbf{x}]$, one cannot guarantee that $\mathbf{f}([\mathbf{x}])$ is still a box. In interval analysis, the inclusion function $[\mathbf{f}]$ is taken as a counterpart of $\mathbf{f}$ to ensure that $[\mathbf{f}]([\mathbf{x}])$ is still a box. The inclusion function is normally defined as $\mathbf{f}([\mathbf{x}]) \subset [\mathbf{f}]([\mathbf{x}])$, $\forall[\mathbf{x}] \subset \mathbb{IR}^d$ (Jaulin and Desrochers, 2014, Jaulin et al., 2001).

3.2. Constraint Satisfaction Problems

When a box propagates through an inclusion function, its volume could increase dramatically. This reveals the 'conservative' nature of interval analysis based methods, i.e. expanding box volumes to guarantee that no feasible solutions are excluded. This, however, can cause overestimation problems as non-feasible solutions could be included as well when a box is expanded. The Constraint Satisfaction Problem (CSP) is exploited to help reducing box volumes. The CSP aims at finding a subset X of the feasible domain $[\mathbf{x}]$, which satisfies

$$X = \{\mathbf{x} \in [\mathbf{x}] | \mathbf{h}(\mathbf{x}) = 0\}, \tag{1}$$

where $\mathbf{h}(\mathbf{x}) = 0$ indicates the constraint. Finding X is computationally demanding. In interval analysis, instead of finding X, one can apply a contractor $\mathcal{C}$ to reduce the volume of $[\mathbf{x}]$ and get $[\mathbf{x}_c] = \mathcal{C}([\mathbf{x}])$, such that $X \subset [\mathbf{x}_c] \subseteq [\mathbf{x}]$ (Drevelle and Bonnifait, 2013).

The forward-backward contractor is broadly accepted in literature due to its efficiency and effectiveness. Given a set of constraints in the form of $\mathbf{h}(\mathbf{x}) = \mathbf{y}$, with $\mathbf{x}$ and $\mathbf{y}$ measurable quantities, the contraction is achieved (Jaulin, 2009a) by propagating from $\mathbf{x}$ to $\mathbf{y}$ in the first step (*forward propagation*). The constraints are next propagated inversely from $\mathbf{y}$ to $\mathbf{x}$ (*backward propagation*). The process is repeated until no more significant box volume reduction can be observed. Jaulin gives some examples to make the process easy to understand in (Jaulin, 2009a).

3.3. The q-satisfied Intersection

For a given set of $Q \in \mathbb{N}^+$ boxes $\{[\mathbf{x}]_i, i = 1, \cdots, Q\}$, the computation of their intersection

$$[\mathbf{x}] = \bigcap_{i=1}^{Q} [\mathbf{x}]_i \tag{2}$$

is frequently required. However, outliers cause empty intersections, which can lead to early termination or even divergence of algorithms.

The q-satisfied intersection (Wang et al., 2015, 2018) along with the q-relaxed intersection proposed in (Jaulin, 2009b) are used to find a subset of $\{[\mathbf{x}]_i, i = 1, \cdots, Q\}$, such that their intersection is not empty. The difference between the two is that the q-satisfied method searches for the maximum number q of boxes with non-empty intersection, where q is not determined at the beginning. While in the q-relaxed intersection, q is normally determined according to the application. In the q-satisfied intersection, q is defined as

$$q = \max \left\{ \mathrm{card}(A) \,\middle|\, A \subseteq \{1, \ldots, Q\}, \bigcap_{j \in A} [\mathbf{x}]_j \neq \emptyset \right\}, \tag{3}$$

with $\mathrm{card}(A)$ indicates the cardinality of set A. Subsequently a q-satisfied intersection is defined as

$$[\mathcal{A}_i] = \bigcap^{\{q\}} [\mathbf{x}]_{1,\ldots,Q} = \bigcap_{j \in A} [\mathbf{x}_j], \;\; \mathrm{card}(A) = q. \tag{4}$$

Usually, one can get $K \in \mathbb{N}^+$ q-satisfied intersections $[\mathcal{A}_1], \ldots, [\mathcal{A}_K]$. An approximation to (2) is then denoted as

$$[\mathbf{x}] = \bigcap_{i=1}^{Q} [\mathbf{x}_i] = \mathcal{B}(\{[\mathcal{A}_1], \ldots, [\mathcal{A}_K]\}), \tag{5}$$

where $\mathcal{B}(\cdot)$ indicates the minimum box that encloses $\{[\mathcal{A}_1], \ldots, [\mathcal{A}_K]\}$.

In this paper, q is found by decreasing Q by 1 each step and check whether (3) is satisfied. In scenarios where real-time performance is critical, one can decreasing Q by a greater than 1 step to accelerate the process.

4. Feature Refined Box Particle Filter for Localisation

4.1. Problem Description

The motion of a vehicle is usually described by an evolution model $\mathbf{f}$ and an observation model $\mathbf{g}$. The former represents dynamics of the vehicle, and the latter reveals what measurements the vehicle can incorporate to locate itself. They are separately represented as

$$\begin{cases} \mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \boldsymbol{\mu}_k, \\ \mathbf{y}_k = \mathbf{g}(\mathbf{x}_k, \mathbf{m}) + \boldsymbol{\nu}_k, \end{cases} \tag{6}$$

where $\mathbf{x}_{k-1}$ and $\mathbf{x}_k$ are vehicle states at $k-1$ and k, $\mathbf{y}_k$ denotes the measurement, $\mathbf{m}$ is the reference map, $\mathbf{u}_k = [v_k, \omega_k]^T$ is the input with v_k the vehicle speed and $\omega_k = \dot{\theta}_k$ the yaw rate, and $\boldsymbol{\mu}_k$ and $\boldsymbol{\nu}_k$ are separately the system and observation noises.

In the Bayesian framework, the objective of localising a vehicle is to estimate the posterior distribution over the current vehicle pose $\mathbf{x}_k$ denoted as

$$\begin{aligned} & p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, \mathbf{m}) \\ = \; & \tfrac{1}{\chi_k} p(\mathbf{y}_k \mid \mathbf{x}_k, \mathbf{m}) p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{m}), \end{aligned} \tag{7}$$

where

$\chi_k = \int p(\mathbf{y}_k \mid \mathbf{x}_k, \mathbf{m}) p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{m}) d\mathbf{x}_k$

is the evidence distribution. Equation (7) can be decomposed into two components besides $\frac{1}{\chi_k}$. The predictive distribution is defined as

$$\begin{aligned} & p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, \mathbf{m}) \\ = & \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k-1}, \mathbf{m}) d\mathbf{x}_{k-1}, \end{aligned} \tag{8}$$

where $p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k)$ indicates the state transitional density, and the prior distribution $p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k-1}, \mathbf{m})$ at time $k-1$ is essentially the posterior distribution of $\mathbf{x}_{k-1}$. The second component $p(\mathbf{y}_k \mid \mathbf{x}_k, \mathbf{m})$ is the measurement density given the state $\mathbf{x}_k$ and the reference map $\mathbf{m}$. It is also known as the likelihood of observing $\mathbf{y}_k$ at state $\mathbf{x}_k$.

The BPF falls into the same Bayesian localisation framework. One of the major differences is that variables become boxes. This paper uses $[\mathbf{m}]$ to indicate an inaccurate OSM. The evolution and observation models are, therefore, rewritten as

$$\begin{cases} [\mathbf{x}_k] = [\mathbf{f}]\big([\mathbf{x}_{k-1}], \mathbf{u}_k\big) + [\boldsymbol{\mu}_k], \\ [\mathbf{y}_k] = [\mathbf{g}]\big([\mathbf{x}_k], [\mathbf{m}]\big) + [\boldsymbol{\nu}_k], \end{cases} \tag{9}$$

where $[\mathbf{f}]$ and $[\mathbf{g}]$ are the corresponding inclusion functions.

This paper develops a BPF based localisation framework with evolution and observation models given in (9) within the Bayesian framework.

4.2. Bayesian Paradigm of Box Particle Filter for Localisation

The BPF employs a set of N weighted boxes $\big\{(w_k^i, [\mathbf{x}_k^i])\big\}_{i=1}^N$ to approximate the point-wise state estimation. For clarity, this paper decomposes the BPF based localisation into the following four steps.

4.2.1. The Predictive Distribution

The equivalent prior distribution at time $k-1$ as in (8) is defined as

$$p(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k-1}, [\mathbf{m}]) \approx \sum_{i=1}^N w_{k-1}^i \mathcal{U}_{[\mathbf{x}_{k-1}^i]}(\mathbf{x}_{k-1}), \tag{10}$$

where $\mathcal{U}_{[\mathbf{x}]}(\cdot)$ denotes the multivariate uniform probability density function (pdf) with the interval $[\mathbf{x}]$ as support. The predictive distribution is now given as

$$\begin{aligned} & p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, [\mathbf{m}]) \\ \approx\ & \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \textstyle\sum_{i=1}^N w_{k-1}^i \mathcal{U}_{[\mathbf{x}_{k-1}^i]}(\mathbf{x}_{k-1}) d\mathbf{x}_{k-1} \\ =\ & \textstyle\sum_{i=1}^N w_{k-1}^i \int_{[\mathbf{x}_{k-1}^i]} p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \mathcal{U}_{[\mathbf{x}_{k-1}^i]}(\mathbf{x}_{k-1}) d\mathbf{x}_{k-1}, \end{aligned} \tag{11}$$

The integral in (11) indicates the distribution of the predicted state after propagating the i-th box $[\mathbf{x}_{k-1}^i]$ through $[\mathbf{f}]$. This leads to

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \mathcal{U}_{[\mathbf{x}_{k-1}^i]}(\mathbf{x}_{k-1}) = \mathbf{0}, \tag{12}$$

$\forall \mathbf{x}_k \notin [\mathbf{f}]\big([\mathbf{x}_{k-1}^i], \mathbf{u}_k\big) + [\boldsymbol{\mu}_k]$. This limits the distribution of the predicted state $\mathbf{x}_k$ to

$$\begin{aligned} & \int_{[\mathbf{x}_{k-1}^i]} p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_k) \mathcal{U}_{[\mathbf{x}_{k-1}^i]}(\mathbf{x}_{k-1}) d\mathbf{x}_{k-1} \\ \approx\ & \mathcal{U}_{[\mathbf{f}]\left([\mathbf{x}_{k-1}^i], \mathbf{u}_k\right) + [\boldsymbol{\mu}_k]}(\mathbf{x}_k) = \mathcal{U}_{[\mathbf{x}_{k|k-1}^i]}(\mathbf{x}_k). \end{aligned} \tag{13}$$

By substituting (13) into (11), the predictive distribution becomes

$$p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, [\mathbf{m}]) \approx \sum_{i=1}^N w_{k-1}^i \mathcal{U}_{[\mathbf{x}_{k|k-1}^i]}(\mathbf{x}_k). \tag{14}$$

4.2.2. The Posterior Distribution

The likelihood component $p(\mathbf{y}_k \mid \mathbf{x}_k, [\mathbf{m}])$ is critical in getting the posterior distribution $p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, [\mathbf{m}])$. In BPF, the likelihood is defined as

$$p(\mathbf{y}_k \mid \mathbf{x}_k, [\mathbf{m}]) = \mathcal{U}_{[\mathbf{y}_k]}\big(\mathbf{g}(\mathbf{x}_k, [\mathbf{m}])\big). \tag{15}$$

The definition indicates how predicted measurement $\mathbf{g}(\mathbf{x}_k, [\mathbf{m}])$ is distributed within the support determined by $[\mathbf{y}_k]$, where the observation noise $[\boldsymbol{\nu}_k]$ is considered.

The posterior distribution is now given as

$$\begin{aligned} & p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, [\mathbf{m}]) \\ =\ & \tfrac{1}{\chi_k} p(\mathbf{y}_k \mid \mathbf{x}_k, [\mathbf{m}]) p(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}, \mathbf{u}_{1:k}, [\mathbf{m}]) \\ =\ & \tfrac{1}{\chi_k} \mathcal{U}_{[\mathbf{y}_k]}\big(\mathbf{g}(\mathbf{x}_k, [\mathbf{m}])\big) \textstyle\sum_{i=1}^N w_{k-1}^i \mathcal{U}_{[\mathbf{x}_{k|k-1}^i]}(\mathbf{x}_k) \\ =\ & \tfrac{1}{\chi_k} \textstyle\sum_{i=1}^N w_{k-1}^i \mathcal{U}_{[\mathbf{x}_{k|k-1}^i]}(\mathbf{x}_k) \mathcal{U}_{[\mathbf{y}_k]}\big(\mathbf{g}(\mathbf{x}_k, [\mathbf{m}])\big), \end{aligned} \tag{16}$$

in which, the last two terms imply a CSP problem

$$X_k^i \subseteq [\mathbf{x}_k^i] = \left\{\mathbf{x}_k^i \in [\mathbf{x}_{k|k-1}^i] \mid \mathbf{g}(\mathbf{x}_k^i, [\mathbf{m}]) \in [\mathbf{y}_k]\right\}, \tag{17}$$

i.e. X_k^i is a subset of the predicted state $[\mathbf{x}_{k|k-1}^i]$ that satisfies the measurement constraint (also refer to (1) for understanding). When a contractor is applied, the updated state $[\mathbf{x}_k^i]$ that satisfies $X_k^i \subseteq [\mathbf{x}_k^i]$ can be obtained. Hence, the following relationship holds according to (Gning et al., 2013)

$$\begin{aligned} \mathcal{U}_{[\mathbf{x}_k^i]}(\mathbf{x}_k) = &\ \mathcal{U}_{[\mathbf{x}_{k|k-1}^i]}(\mathbf{x}_k)\mathcal{U}_{[\mathbf{y}_k]}\big(\mathbf{g}(\mathbf{x}_k, [\mathbf{m}])\big) \\ = &\ \tfrac{1}{|[\mathbf{x}_{k|k-1}^i]|}|[\mathbf{x}_k^i]|\mathcal{U}_{[\mathbf{x}_k^i]}(\mathbf{x}_k)\tfrac{1}{|[\mathbf{y}_k]|}, \end{aligned} \tag{18}$$

and the posterior distribution in (16) can be simplified as

$$\begin{aligned} &p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, [\mathbf{m}]) \\ = &\ \tfrac{1}{\chi_k}\textstyle\sum_{i=1}^N w_{k-1}^i \tfrac{1}{|[\mathbf{y}_k]|}\tfrac{1}{|[\mathbf{x}_{k|k-1}^i]|}|[\mathbf{x}_k^i]|\mathcal{U}_{[\mathbf{x}_k^i]}(\mathbf{x}_k) \\ \propto &\ \textstyle\sum_{i=1}^N w_{k-1}^i \tfrac{|[\mathbf{x}_k^i]|}{|[\mathbf{x}_{k|k-1}^i]|}\mathcal{U}_{[\mathbf{x}_k^i]}(\mathbf{x}_k). \end{aligned} \tag{19}$$

4.2.3. Weight Update and Re-sampling

In BPF, particle weights are updated via

$$w_k^i \propto w_{k-1}^i * L_k^i, \tag{20}$$

with $L_k^i = \frac{|[\mathbf{x}_k^i]|}{|[\mathbf{x}_{k|k-1}^i]|}$, and $0 \leq L_k^i \leq 1$.

When relation (17) is absolutely or strongly violated (measurements are not compatible with the prediction), $|[\mathbf{x}_k^i]|$ becomes zero or negligible. This leads the updated weight w_k^i to be zero or negligible as well. It will cause the particle degeneracy phenomenon where only a few particles are with prominent weights. The re-sampling procedure is then triggered when the following N effective criterion meets

$$\frac{1}{\sum_i^N {w_k^i}^2} < \eta_{eff} N. \tag{21}$$

Re-sampling is done by subdividing boxes of high weights from randomly selected dimensions (Gning et al., 2013), or from the most pessimistic state dimensions (the longest box edge corresponded dimension) (Merlinge et al., 2019).

4.2.4. Point State Estimate and Covariance

By nature, interval analysis based methods do not provide point estimates. To provide statistical metrics such as expectation and covariance, in accordance with (Merlinge et al., 2019), this paper defines the point expectation as

$$\hat{\mathbf{x}}_k \triangleq \mathbb{E}\big[\mathbf{x}_k \sim p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{1:k}, [\mathbf{m}])\big] \approx \sum_i^N w_k^i \mathbf{c}_k^i, \tag{22}$$

which is used as point state estimate at time instant k, with $\mathbf{c}_k^i$ indicates the center of $[\mathbf{x}_k^i]$, and $\mathbb{E}[\cdot]$ is the statistical expectation.

4.3. Features-refined Box Particle Filter

4.3.1. Features-refined Contraction

As shown in (1) and (17), contraction accounts for merging innovations into predicted states to make them accurate and reliable. For a given state $[\mathbf{x}]$ and measurements $\{[\mathbf{y}_i] \in \mathbb{IR}^d, i = 1, \cdots, n\}$ of a feature, there are two ways to accomplish contraction. The first follows a step-wise paradigm, i.e. doing contraction upon the arrival of each measurement. The step-wise contraction is widely accepted and has been applied in (Abdallah et al., 2008) and (Gning et al., 2013). Alternatively, one can integrate if not all but several measurements before contraction. It is, therefore, named features-refined contraction in this paper. **Note** that this paper omits the time stamp k for the purpose of a general description. This paper also denotes $\{[\mathbf{y}_k^i], i = 1, \cdots, n\}$ as $\{[\mathbf{y}_i], i = 1, \cdots, n\}$ for brevity.

Jaulin (Jaulin, 2009a) has proved that the order of variables being contracted does not affect the convergent boxes. However, there lacks research works in literature demonstrating that the features-refined contraction is equivalent to the step-wise counterpart. The problem will be formulated and their equivalence will be proved as follows.

Step-wise contraction: Given a state $[\mathbf{x}]$ and measurements $\{[\mathbf{y}_i] \in \mathbb{IR}^d, i = 1, \cdots, n\}$ of a feature, the step-wise contraction result is obtained by solving a

CSP problem

$$[s\mathbf{x}_i] = \left\{s\mathbf{x}_i \in [\mathbf{x}_i] \mid \mathbf{h}(s\mathbf{x}_i) = \mathbf{0}\right\}, \tag{23}$$

where

$$\mathbf{h}(s\mathbf{x}_i) = s\mathbf{x}_i - \mathbf{g}^{-1}([\mathbf{y}_i]), \tag{24}$$

$[s\mathbf{x}_{i-1}] = [\mathbf{x}_i]$ holds for $i \in \{2, \cdots, n\}$, and $[\mathbf{x}_1] = [\mathbf{x}]$.

For localisation, (23) reduces to

$$\begin{aligned}
[s\mathbf{x}_n] &= [s\mathbf{x}_{n-1}] \bigcap \mathbf{g}^{-1}\big([\mathbf{y}_n]\big) \\
&= [s\mathbf{x}_{n-2}] \bigcap \mathbf{g}^{-1}\big([\mathbf{y}_{n-1}]\big) \bigcap \mathbf{g}^{-1}\big([\mathbf{y}_n]\big) \\
&\ \ \vdots \\
&= [s\mathbf{x}_1] \bigcap\nolimits_{i=2}^{n} \mathbf{g}^{-1}\big([\mathbf{y}_i]\big) \\
&= [\mathbf{x}] \bigcap\nolimits_{i=1}^{n} \mathbf{g}^{-1}\big([\mathbf{y}_i]\big),
\end{aligned} \tag{25}$$

where $\mathbf{g}^{-1}$ is an arbitrary function that is piece-wisely monotonic (Rohou et al., 2018). The final result $[s\mathbf{x}_n]$ can be abbreviated as $[s\mathbf{x}]$ without causing confusions.

Features-refined contraction: Given $[\mathbf{x}]$ and measurements $\{[\mathbf{y}_i] \in \mathbb{IR}^d, i = 1, \cdots, n\}$ of a feature, the features-refined contraction result $[b\mathbf{x}]$ can be obtained through

$$[b\mathbf{x}] = \left\{b\mathbf{x} \in [\mathbf{x}] \mid \mathbf{h}(b\mathbf{x}) = \mathbf{0}\right\}, \tag{26}$$

where

$$\mathbf{h}(b\mathbf{x}) = b\mathbf{x} - \mathbf{g}^{-1}\big(\bigcap [\mathbf{y}_i]\big). \tag{27}$$

Similarly, for localisation, (26) reduces to

$$[b\mathbf{x}] = [\mathbf{x}] \bigcap \mathbf{g}^{-1}\big(\bigcap\nolimits_{i=1}^{n} [\mathbf{y}_i]\big). \tag{28}$$

Corollary 1: Given a state $[\mathbf{x}]$ and measurements $\{[\mathbf{y}_i] \in \mathbb{IR}^d, i = 1, \cdots, n\}$ of a feature, $[s\mathbf{x}] = [b\mathbf{x}]$ stands, i.e.

$$\bigcap\nolimits_{i=1}^{n} \mathbf{g}^{-1}\big([\mathbf{y}_i]\big) \cap [\mathbf{x}] = \mathbf{g}^{-1}\big(\bigcap\nolimits_{i=1}^{n} [\mathbf{y}_i]\big) \cap [\mathbf{x}]. \tag{29}$$

The proof of the corollary is given in Appendix A. When $\mathbf{g}^{-1}$ is nonlinear, the inclusion function $[\mathbf{g}^{-\mathbf{1}}]$ is usually used instead to deal with the contraction problem. The disadvantage is that it degenerates **Corollary** 1 because inclusion functions usually overly enlarge (or shrink) box volumes. This paper proposes **Corollary** 2 to show results when it comes to inclusion function cases.

Corollary 2: Given a state $[\mathbf{x}]$ and measurements $\{[\mathbf{y}_i] \in \mathbb{IR}^d, i = 1, \cdots, n\}$, and a piece-wisely monotonic function $\mathbf{g}^{-1}$ with the corresponding inclusion function $[\mathbf{g}^{-1}]$, the following equation stands.

$$[\mathbf{g}^{-1}]\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big) \cap [\mathbf{x}] \subseteq \bigcap\nolimits_{i=1}^{n}[\mathbf{g}^{-1}]\big([\mathbf{y}_i]\big) \cap [\mathbf{x}]. \tag{30}$$

This implies when inclusion functions are used, results from a features-refined contraction are finer than those from the step-wise contraction, which means more non-feasible solutions are excluded by the features-refined contraction.

The proof of **Corollary** 2 is given in Appendix B. Combining **Corollary** 1 and **Corollary** 2, one can conclude that the features-refined contraction produces finer results than step-wise contraction despite incorporating the same measurements. This helps in mitigating the 'conservative' aspect of interval analysis based methods that involve contraction to refine results.

Example 1: Fig. 1 gives an example where $n = 3$ to show the difference between the step-wise and features-refined contractions, in scenarios where $\mathbf{g}^{-1}$ and its inclusion function counterpart $[\mathbf{g}^{-1}]$ are used, respectively. Each step in Fig. 1 is explained separately as follows.

The first step: The upper sub-column shows that given three measurements $[\mathbf{y}_1]$, $[\mathbf{y}_2]$, and $[\mathbf{y}_3]$, one can get their intersection $[\mathbf{y}] = \bigcap_{i=1}^{3}[\mathbf{y}_i]$ as shown in the lower sub-column.

The second and third step: Given $\mathbf{g}^{-1}$ (rather than $[\mathbf{g}^{-1}]$) and $[\mathbf{y}_1]$, $[\mathbf{y}_2]$, $[\mathbf{y}_3]$, it is intuitive to begin with calculating $\mathbf{g}^{-1}([\mathbf{y}_1])$, $\mathbf{g}^{-1}([\mathbf{y}_2])$, and $\mathbf{g}^{-1}([\mathbf{y}_3])$ for refining $[\mathbf{x}]$ to get $X \in [\mathbf{x}]$. This is usually achieved by following either (25) or (28). Step 2 shows the step-wise contraction achieved by following (25). Note that the line width is varied to show that the contraction is done by using $\mathbf{g}^{-1}([\mathbf{y}_1]$ to $\mathbf{g}^{-1}([\mathbf{y}_3]$ step by step. Step 3, on the other hand, demonstrating

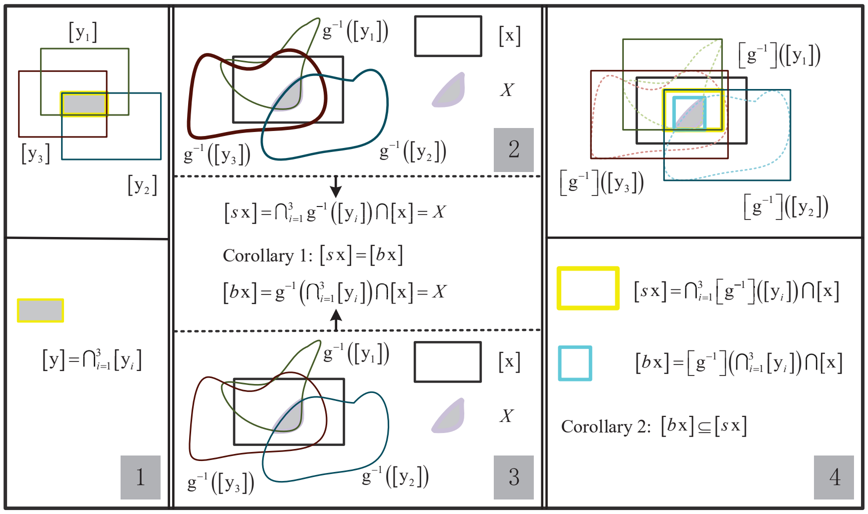


Figure 1: An example illustrating the difference between the step-wise and features-refined contractions. Step 1 gives the intersection of three measurements $[\mathbf{y}] = \bigcap_{i=1}^{3}[\mathbf{y}_i]$. Step 2 and 3 together demonstrate Corollary 1. Note that the line width varies in Step 2 to show the contraction is done step-wisely. Step 4 demonstrates Corollary 2.

that $\mathbf{g}^{-1}(\bigcap_{i=1}^{3}[\mathbf{y}_i]$ is calculated first, which is next used to refine $\mathbf{x}$ to get X. Step 2 and 3 together constitute Corollary 1.

Note that when $\mathbf{g}^{-1}$ is used, X is not necessarily a box. It can be of any shape as shown by the shaded area in the second and third step. The disadvantage is the computation of X is usually complex. Furthermore, if one gets another $\tilde{X}$ through other measurements, the calculation of the intersection between X and $\tilde{X}$ is complex as well.

The fourth step: To simplify the computation, the inclusion function $[\mathbf{g}^{-1}]$ of $\mathbf{g}^{-1}$ is introduced. It converts $\mathbf{g}^{-1}([\mathbf{y}_1])$, $\mathbf{g}^{-1}([\mathbf{y}_2])$, and $\mathbf{g}^{-1}([\mathbf{y}_3])$ into three boxes denoted by $[\mathbf{g}^{-1}]([\mathbf{y}_1])$, $[\mathbf{g}^{-1}]([\mathbf{y}_2])$, and $[\mathbf{g}^{-1}]([\mathbf{y}_3])$. It is intuitive that the operation on the latter three boxes is simplified comparing to the operation on $\mathbf{g}^{-1}([\mathbf{y}_1])$, $\mathbf{g}^{-1}([\mathbf{y}_2])$, and $\mathbf{g}^{-1}([\mathbf{y}_3])$.

One can now either follow the step-wise contraction to refine $[\mathbf{x}]$ to get $[s\mathbf{x}]$, or use the features-refined contraction to refine $[\mathbf{x}]$, resulting in $[b\mathbf{x}]$. They are separately shown in Step 4 in Fig. 1, with $[s\mathbf{x}]$ depicted by the yellow rectangle and $[b\mathbf{x}]$ in cyan rectangle in the lower sub-column. One can see that $[b\mathbf{x}] \subseteq [s\mathbf{x}]$ stands. This demonstrates the feature-refined contraction yields 'finer' results compared with step-wise contraction, which would help to mitigate the 'conservative' aspect of interval analysis based methods.

In real applications, $\bigcap_{i=1}^{n}[\mathbf{y}_i]$ could result in an empty intersection despite a subset with non-empty intersection of the measurements $\{[\mathbf{y}_i] \in \mathbb{IR}^d, i = 1, \cdots, n\}$ can still help in contraction. Therefore, (5) is exploited to find a q-satisfied intersection to approximate $\bigcap_{i=1}^{n}[\mathbf{y}_i]$. As $q \leqslant n$ holds, for a given predicted state $[\mathbf{x}]$, the following equation stands,

$$[q\mathbf{x}] = \big\{q\mathbf{x} \in [\mathbf{x}] \mid \mathbf{h}(q\mathbf{x}) = \mathbf{0}\big\}, \tag{31}$$

where

$$\mathbf{h}(q\mathbf{x}) = q\mathbf{x} - [\mathbf{g}^{-1}]\big(\bigcap_{j\in A}[\mathbf{y}_j]\big), \tag{32}$$

and $[q\mathbf{x}]$ is the result obtained by applying contraction to $[\mathbf{x}]$ with measurement achieved through q-satisfied intersection. One can directly see the following condition stands.

$$[\mathbf{g}^{-1}]\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big) \cap [\mathbf{x}] \subseteq [\mathbf{g}^{-1}]\big(\bigcap_{j\in A}[\mathbf{y}_j]\big) \cap [\mathbf{x}], \tag{33}$$

where $\bigcap_{j\in A}[\mathbf{y}_j] = \bigcap^{\{q\}}[\mathbf{y}]_{1,\ldots,n}$ with $A \subseteq \{1, \cdots, n\}$ as defined in (5).

4.3.2. Weight Balance

In BPF, particle weights are updated through (20), which indicates that given w_{k-1}^i, the weight w_k^i at time k is proportional to the likelihood L_k^i. One can generalise (20) by writing

$$w_k^i = w_{k-1}^i * \exp(L_k^i - L_{\max}), \tag{34}$$

where $L_{\max} = \max\{L_k^i \mid i = 1, \cdots, N\}$, and $\exp(L_k^i - L_{\max})$ is a factor accounting for weight updating. This is because when $0 \leq L_k^i \leq 1$ stands, $\exp(L_k^i - L_{\max})$ can be approximated by $(L_k^i - L_{\max})$, which still matches the proportional relationship given by (20).

This paper proposes to balance the weight updating formula (34) by

$$w_k^i = w_{k-1}^i * \big(\exp(L_k^i - L_{\max}) * \alpha + \exp(L_{\mathrm{med}} - L_k^i) * (1-\alpha)\big), \tag{35}$$

where L_{med} is the median value of $\{L_k^i \mid i = 1, \cdots, N\}$, α is the balance parameter, and $\exp(L_k^i - L_{\max}) * \alpha + \exp(L_{\mathrm{med}} - L_k^i) * (1-\alpha)$ is the weight updating factor (WUF).

By comparing (34) with (35), one can see that the latter keeps $\exp(L_k^i - L_{\max})$, meaning that a high likelihood box particle will maintain a high weight after it is updated through (35). Meanwhile, the term $\exp(L_{\mathrm{med}} - L_k^i) * (1-\alpha)$ is added to account for low likelihood box particles that are consistent with the real vehicle state but unlikely due to map errors. This helps also to mitigate the negative effects where the high likelihood is caused by inaccurate OSM features.

Fig. 2 shows how WUF changes when α decreases from 1.0 to 0. When one investigates the curves along the left vertical axis, the blue curve is generated by setting $\alpha = 1.0$. The subsequent nine light blue curves from bottom to top are separately generated by setting $\alpha = \{0.9, 0.8, \cdots, 0.2, 0.1\}$. The magenta curve is generated by setting $\alpha = 0$. When α is set to 1.0, one can see that (35) becomes equivalent to (34). When it keeps decreasing, WUF tends to balance between high and low likelihood box particles. When α reaches 0, WUF is solely determined by $\exp(L_{\mathrm{med}} - L_k^i)$, which tends to put trust on low likelihood box particles. The values for generating Fig. 2 are $L_{\mathrm{med}} = 0.3$ and $L_{\max} = 1.0$.

4.4. OpenStreetMap Accuracy Evaluation

4.4.1. Definition of Coordinate Systems

Entities on OSM are encoded by geodetic coordinates, i.e. latitude and longitude. This paper chooses the local East, North, and Up (ENU) coordinate system to achieve localisation, which makes transforming the geodetic coordinates into the local coordinate system necessary. Compared with other Cartesian coordinate systems such as the Earth-Centered, Earth-Fixed (ECEF) coordinate system, the ENU system provides simple 2D planar projections of

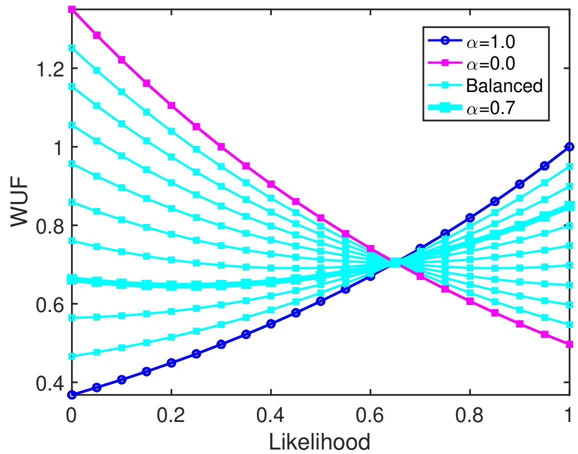


Figure 2: The relationship between likelihood and WUF, with α decreasing from 1.0 to 0. When one focuses on the curves along the left vertical axis, the blue curve is generated by setting $\alpha = 1.0$. The nine light blue curves from bottom to top are separately generated by setting $\alpha = \{0.9, 0.8, \cdots, 0.2, 0.1\}$. The curve with $\alpha = 0.7$ is highlighted in bold, which is used in this paper. The magenta curve is generated by setting $\alpha = 0$.

geodetic coordinates of interest. Also, the transformation from 3D geodetic coordinates to the ENU coordinate system is invertible, which makes it easy for transforming localisation results to the geodetic coordinate system if needed. The transformation between different coordinate systems is shown in Fig. 3. In this paper, as the campus is roughly flat, the 'Up' dimension is omitted for brevity. Fig. 4 shows how one line feature extracted from LiDAR perception is represented in the OSM and vehicle coordinate systems. The $O_G\_x_Gy_G$ indicates the OSM (and the HDM) coordinate system. The $O_R\_x_Ry_R$ is the vehicle coordinate system. p_j and α_j are separately the distance and angle of the line feature with respect to $O_G\_x_Gy_G$. r_i and ψ_i are the distance and angle of the line feature with respect to $O_R\_x_Ry_R$, respectively.

4.4.2. Accuracy Evaluation of OpenStreetMap

A customised HDM serves as the local ENU coordinate system in this paper and the OSM is aligned to it for OSM evaluation, as shown in Fig. 4. Aligning

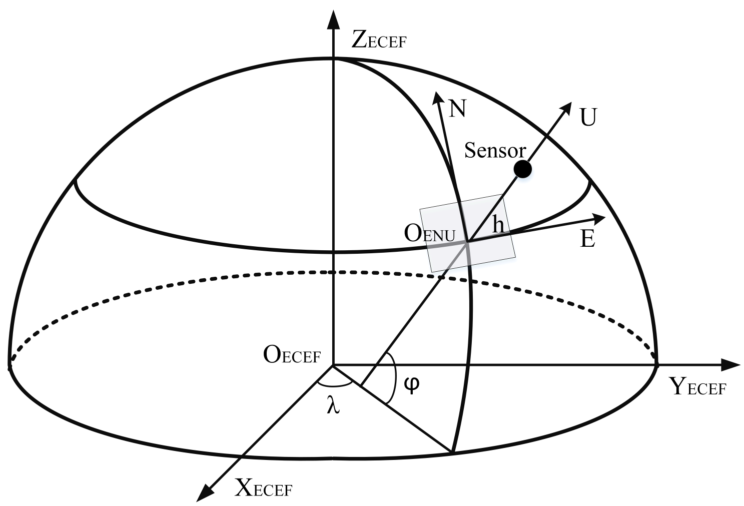


Figure 3: The ENU coordinate system used and the transformation with other systems, λ indicates the longitude, φ indicates the latitude, and h is the ellipsoidal height.

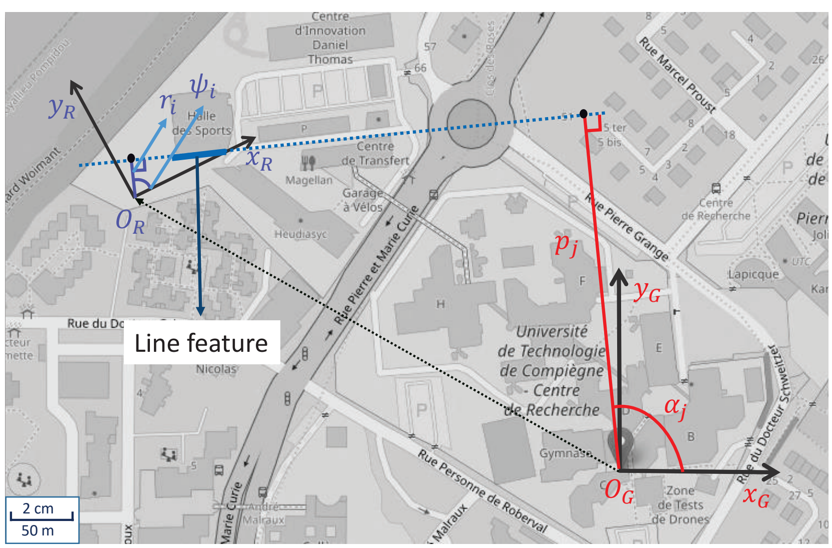


Figure 4: Coordinate systems used for line feature representation.

OSM to the HDM coordinate system can lead to negative coordinates of OSM data, which is caused by projecting geodetic coordinates of OSM into the local HDM coordinate system. The HDM provides accurate ENU coordinates of points along the road. There are points along centers of the roads, and points that mark the boundaries of the roads, as shown in Fig. 5(a). This paper considers only OSM features within the UTC campus are considered as shown in Fig. 5(b).

This paper adopts the distance from HDM points to the corresponding OSM roads as a measure of the OSM accuracy. A whole accuracy evaluation of OSM is out the scope of this paper. Instead, three places that are roughly in the center of the UTC campus have been chosen for evaluation. These three places are marked as *Road set 1*, *Road set 2*, and *Road set 3*, which are shown in Fig. 5(a).

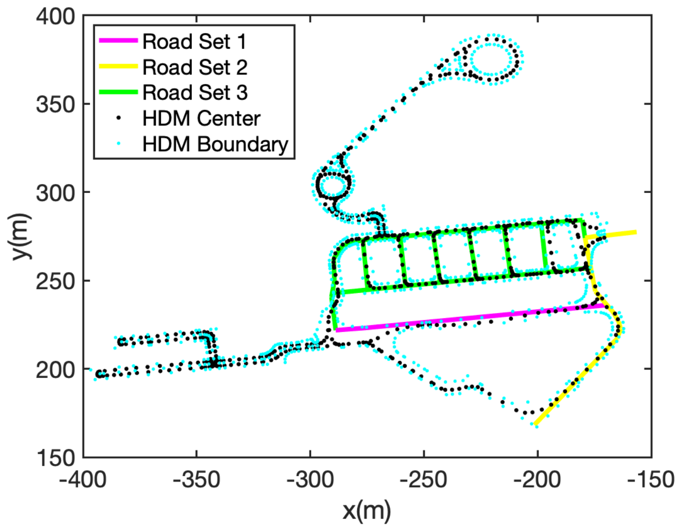
A total number of 153 samples from the three places shown in Fig. 5(a) are used for OSM accuracy evaluation, and the results are summarised in Table I. It shows that the average distances and the standard deviations from each *Road set*. Column *Road set all* shows results by aggregating distances from all three places. One can conclude that the OSM accuracy varies even within the UTC campus. For generality, the results from *Road set all* are taken as evaluation results. This provides an accuracy of around 0.726 m, with a standard deviation of 0.778 m.

Table 1: The mean and standard deviation of OSM accuracy evaluation

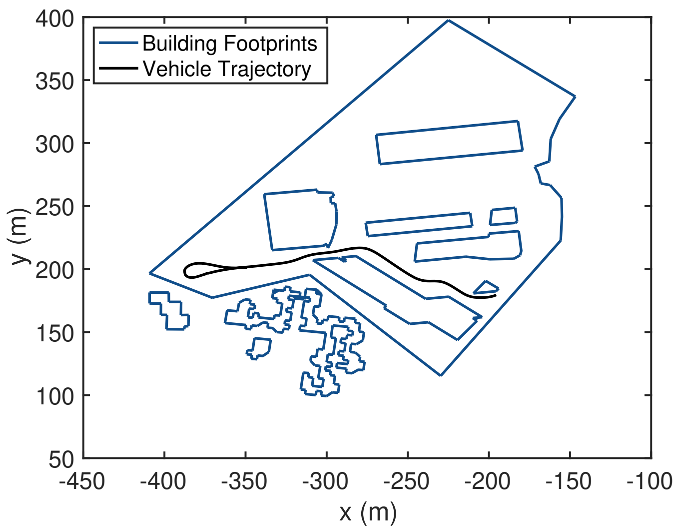
	Road set 1	Road set 2	Road set 3	Road set all
Mean (m)	1.866	0.536	0.556	0.726
Std (m)	1.0870	0.297	0.573	0.778

4.5. Measurement and Uncertainty Representation

While a vehicle is navigating in an urban environment, various features can be captured by exteroceptive sensors. This paper only focuses on line features



(a)



(b)

Figure 5: The HDM and building footprints from OSM used for OSM accuracy evaluation. The vehicle trajectory is also given: (a) The HDM and the three sets of data used for OSM accuracy evaluation; (b) The building footprints from OSM of UTC with vehicle trajectory.

extracted from LiDAR data, given the fact that they are not only abundant in structured urban environments but also the fundamental components of OSM.

4.5.1. Measurement and Innovation

The line feature in Fig. 4 in the $O_G\_x_Gy_G$ is represented as

$$x_G \cos \beta_j + y_G \sin \beta_j = p_j, \tag{36}$$

where j indicates the line feature is associated with the j-th OSM line feature, β_j is the angle between the x_G-axis and the line normal vector, and p_j is the orthogonal distance between O_G and the line.

In the vehicle coordinate system $O_R\_x_Ry_R$, the line feature is represented as

$$x_R \cos \psi_i + y_R \sin \psi_i = r_i, \tag{37}$$

with i marking the i-th line feature in the vehicle coordinate system, ψ_j is the angle between the O_R-axis and the line normal vector, and r_i is the orthogonal distance between O_R and the line. Note that (36) and (37) represent the same line feature in the two different coordinate systems.

Distances and angles are taken as feature measurements. By concatenating the n_R measurements in $O_R\_x_Ry_R$ at time k, the measurement vector is formulated as

$$\mathbf{y}_k = \left(r_1, \psi_1, r_2, \psi_2, \cdots, r_{n_R}, \psi_{n_R}\right)^T. \tag{38}$$

The method proposed in (Teslić et al., 2011) is exploited to associate the measurements with OSM features. Without loss of generality, one can assume that a feature denoted by (r_i, ψ_i) in $O_R\_x_Ry_R$ is associated with a feature denoted by (p_j, β_j) in $O_G\_x_Gy_G$. Now, given the predicted vehicle state at time k as

$$\mathbf{x}_{k|k-1} = (x_{k|k-1}, y_{k|k-1}, \theta_{k|k-1})^T,$$

the feature denoted by (p_j, β_j) in the OSM is transformed into $O_R\_x_Ry_R$ by

$$\begin{bmatrix} \tilde{r}_i \\ \tilde{\psi}_i \end{bmatrix} = \begin{bmatrix} |C_j| \\ \beta_j - (\theta_{k|k-1} - \frac{\pi}{2} + (-0.5 \cdot \mathrm{sign}(C_j) + 0.5)\pi) \end{bmatrix}, \tag{39}$$

with

$$C_j = p_j - x_{k|k-1} \cos \beta_j - y_{k|k-1} \sin \beta_j. \tag{40}$$

By aggregating all the n_R results in (39), the measurement prediction corresponding to (38) is denoted as

$$\tilde{\mathbf{y}}_k = \left(\tilde{r}_1, \tilde{\psi}_1, \tilde{r}_2, \tilde{\psi}_2, \cdots, \tilde{r}_{n_R}, \tilde{\psi}_{n_R}\right)^T, \tag{41}$$

and the measurement innovation, which is usually defined as the difference between the measurement in (38) and the measurement prediction given in (41), is denoted as

$$\mathbf{I}_k = \left(\Delta r_1, \Delta\psi_1, \Delta r_2, \Delta\psi_2, \cdots, \Delta r_{n_R}, \Delta\psi_{n_R}\right)^T, \tag{42}$$

with $\Delta r_i = r_i - \tilde{r}_i$, and $\Delta\psi_i = \psi_i - \tilde{\psi}_i$. The innovation $\mathbf{I}_k$ is used to update the state estimate in filtering techniques such as Kalman filter and particle filter (Wang et al., 2018).

4.6. Measurement and Innovation within Interval Analysis

While uncertainties of the line parameters are often taken into account statistically (Teslić et al., 2011), boxes are used here to represent uncertainties to the line feature parameters. According to Section 4.4.2, building footprints in OSMs are shifted (or biased). An interval is added to each endpoint of the line features in the map to account for the inaccuracy of the OSM. This leads to an intervalised OSM, which is denoted as $[\mathbf{m}]$.

When the OSM is intervalised as $[\mathbf{m}]$, line features in both coordinate systems are intervalised consequently as

$$[x_G] \cos[\beta_j] + [y_G] \sin[\beta_j] = [p_j], \tag{43}$$

and

$$[x_R] \cos[\psi_i] + [y_R] \sin[\psi_i] = [r_i]. \tag{44}$$

This equals to adding an box to each measurement (r_i, ψ_i), turning the measurement in (38) into

$$[\mathbf{y}_k] = \left([r_1], [\psi_1], [r_2], [\psi_2], \cdots, [r_{n_R}], [r_{n_R}]\right)^T, \tag{45}$$

and the measurement prediction into

$$[\tilde{\mathbf{y}}_k] = \left([\tilde{r}_1], [\tilde{\psi}_1], [\tilde{r}_2], [\tilde{\psi}_2], \cdots, [\tilde{r}_{n_R}], [\tilde{\psi}_{n_R}]\right)^T. \tag{46}$$

The innovation then becomes

$$[\mathbf{I}_k] = \left([\Delta r_1], [\Delta \psi_1], \cdots, [\Delta r_{n_R}], [\Delta \psi_{n_R}]\right)^T, \tag{47}$$

with $[\Delta r_i] = [r_i] \bigcap [\tilde{r}_i]$, and $[\Delta \psi_i] = [\psi_i] \bigcap [\tilde{\psi}_i]$. $[\mathbf{I}_k]$ is used to perform the contraction. Please note that, as innovations and measurements are directly related, measurements (and not innovations) are used in formulating and solving CSPs, in accordance with the literature.

4.7. The Features-refined Box Particle Filter based Localisation Algorithm

The proposed FRBPF follows a Bayesian approach similar to the BPF described in Section 4.2, and the new contraction and weight balance method are incorporated in FRBPF as Algorithm 1. Fig. 6 gives a graphical representation of the FRBPF.

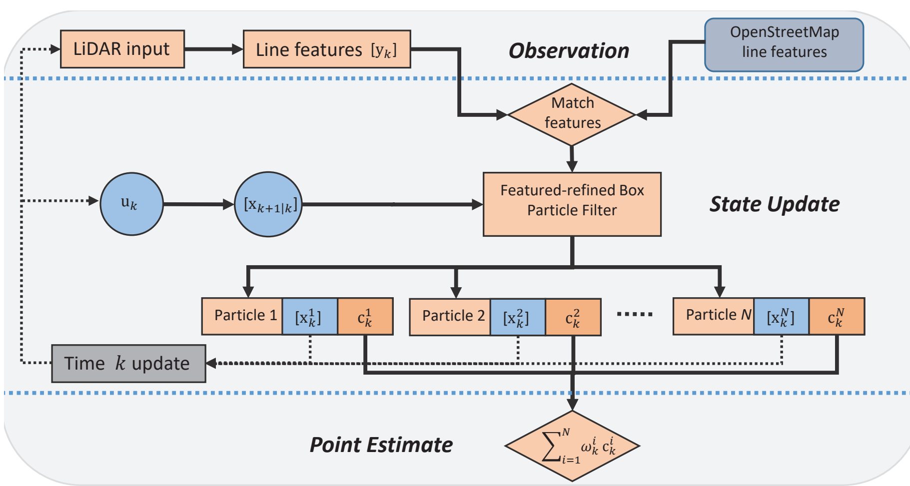


Figure 6: Flowchart of the proposed approach, where k and $k+1$ are time stamps, $[\mathbf{x}_k^i]$ is the state maintained by the i-th particle, $\mathbf{c}_k^i$ is the center of $[\mathbf{x}_k^i]$, w_k^i is the weight of $[\mathbf{x}_k^i]$, and N is the number of particles.

Algorithm 1 The Features-refined Box Particle Filter

Input: N box particles $\{[\mathbf{x}_0^i]\}_{i\in\{1,\cdots,N\}}$ of empty intersection, whose weights are initiated as $w_0^i = 1/N$, and an OSM $[\mathbf{m}]$.

Output: Point-wise state estimates and box volumes.

1: **for** each time-step k **do**

2: Propagate box particles using (9).

3: Calculate innovation using (46, 47).

4: Contract box particles using (31), when an innovation is available.

5: Calculate likelihood and update weights using (35).

6: Weight normalisation.

7: Estimate point state $\hat{\mathbf{x}}_k$ and box volumes.

8: **if** (21) is satisfied **then**

9: Re-sampling: choose a set of particles with the highest weights and determine the new box number n^i per existing box particle.

10: Subdivide each chosen box into n^i new boxes along the most pessimistic dimension and do regularisation by randomly moving the box particles suggested in (Merlinge et al., 2019)

11: Reset all weights to $w_k^i = 1/N$.

For a line feature from OSM, suppose a set of measurements denoted as $\{[\mathbf{y}_i] \in \mathbb{R}^d, i = 1, \cdots, n\}$ are obtained. The corresponding predicted measurements and innovations are next calculated following equations (46, 47). When the feature-refined contraction and q-satisfied intersection are adopted, (17) becomes (31) to represent the feature-refined contraction problem. As seen earlier, the weight updating strategy uses (35) instead of (20), to balance between low and high likelihood box particles to mitigate the localisation uncertainties caused by OSM and measurement uncertainties. Please note that for Step 10, one can either follow the approach in (Merlinge et al., 2019) to subdivide a box along the most pessimistic dimension for re-sampling, or follow the random subdivision approach used in (Abdallah et al., 2008, Gning et al., 2012). Both approaches are studied and their comparison is given in the next section.

5. Performance Evaluation

5.1. Models and Experiment Settings

The evolution model $[\mathbf{f}]$ uses the measured speed v_k and yaw rate ω_k and is given as

$$\begin{cases} [x_{k+1}] = [x_k] + T \cdot v_k \cdot \cos([\theta_k] + T \cdot \dfrac{\omega_k}{2}) + [\mu_k^x], \\ [y_{k+1}] = [y_k] + T \cdot v_k \cdot \sin([\theta_k] + T \cdot \dfrac{\omega_k}{2}) + [\mu_k^y], \\ [\theta_{k+1}] = [\theta_k] + T \cdot \omega_k \ + [\mu_k^\theta], \end{cases} \tag{48}$$

where $([x_k], [y_k], [\theta_k])^T \triangleq [\mathbf{x}_k]$ is the interval vehicle state, and $([\mu_k^x], [\mu_k^y], [\mu_k^\theta])^T \triangleq [\boldsymbol{\mu}_k]$ is the interval evolution noise.

The measurement model $[\mathbf{g}]$ is defined as

$$\begin{cases} [r_k] = \sqrt{([x_k] - [x_R])^2 + ([y_k] - [y_R])^2} + [\nu_k^r], \\ [\psi_k] = \operatorname{atan2}([y_k] - [y_R], [x_k] - [x_R]) - [\theta_k] + [\nu_k^\psi], \end{cases} \tag{49}$$

where $[r_k]$ and $[\psi_k]$ are separately the interval distance and angle of a line feature indicated by $([x_R], [y_R])$ with respect to the vehicle, $([\nu_k^r], [\nu_k^\psi])^T \triangleq [\boldsymbol{\nu}_k]$ is the interval measurement noise.

LiDAR data collected by a Velodyne® VLP-16 sensor mounted on the roof of a vehicle are processed. Sixteen layers of point clouds are obtained. This paper extracts line segments from these layers directly and they are next associated with OSM line features. Please note that it is possible that one can extract line features from point clouds reflected by trees, but they will be filtered out by data association (no line features corresponding to the tree exist on the OSM) and q-satisfied intersection (line features corresponding to the same footprint on the OSM tend to be 'closer' to each other than the features from the trees, hence features extracted from the trees will be filtered out).

An abundant number of line segments can be extracted from LiDAR point-clouds that correspond to a single line feature in OSM. The abundance enables feature-refined contraction and makes the framework proposed meaningful. Ground truth locations are obtained through a RTK sensor suite. Building footprints of the UTC campus are extracted from OpenStreetMap as shown in

Fig. 5(b). The FRBPF, BRPF, and BPF are implemented in Matlab® 2018a programs. The PC configuration includes an Intel® Core(TM) i7-7800X CPU and 16.0GB RAM. The box particles do not mutually intersect, and are scattered around the initial state of the vehicle provided by the real-time kinematic sensor suite. η_{eff} is set to 0.7 for FRBPF, BRPF, and BPF, which is a common choice (Merlinge et al., 2019). The weight balance parameter α is set to 0.7 here. The OSM inaccuracy is incorporated by adding a box $[-0.73 \text{ m}, 0.73 \text{ m}]$ (the bounds correspond to the average evaluation error given in Section 4.4) to the distance measurement r, and a box $[-0.5 \text{ rad}, 0.5 \text{ rad}]$ to the angle measurement ψ.

5.2. Localisation Performance

For general and reliable performance evaluation, $N_{MC} = 100$ times Monte Carlo runs have been carried out for FRBPF, BRPF, and BPF. The point-wise estimation errors and average box volumes are both calculated for performance evaluation. The estimation errors are calculated by $RMSE_{\mathcal{X}}(k) = \sqrt{\frac{1}{N_{MC}} \sum_{run=1}^{N_{MC}} ||\hat{\mathcal{X}}_{k,run} - \mathcal{X}_{k,run}||^2}$, with $\hat{\mathcal{X}}_{k,run}$ stands for the estimate at time k and $\mathcal{X}_{k,run}$ is the ground truth. The terms 'area' and 'size' will be separately used for position and orientation estimation instead of 'volume' to avoid ambiguities. 'Volume' will be kept for generic descriptions.

Fig. 7 and Fig. 8 show the position and orientation estimation results of FRBPF, BRPF, and BPF, respectively. Both the average box volumes and point-wise estimation errors are given. One can see that FRBPF and BRPF show prominent advantages in terms of both average box volumes and point-wise estimation errors. When compare FRBPF with BRPF, one can see that the former still shows better performance in general, i.e. smaller average box volumes and smaller point-wise estimation errors. It is worth mentioning that there are cases where BRPF slightly outperforms FRBPF. This is due to the reason that when q in q-satisfied intersection is small or around 1, FRBPF degenerates to BRPF, hence leading to similar performance to BRPF.

Table 2: Experimental results of FRBPF, BRPF, and BPF. The first two columns are separately the position and orientation errors, and the last two columns are the position box area and orientation box size, respectively.

	Position (m)	Orientation (rad)	Position Area (m^2)	Orientation Size (rad)
FRBPF	0.368	0.010	1.050	0.387
BRPF	0.409	0.012	1.400	0.439
BPF	0.783	0.034	1.781	0.518

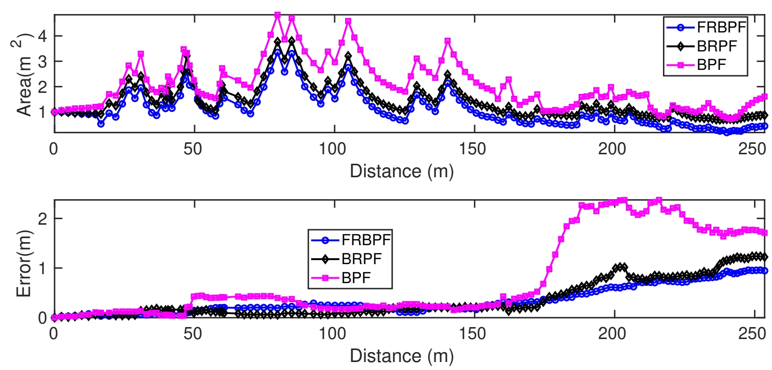


Figure 7: Average position box areas and point-wise position estimation errors.

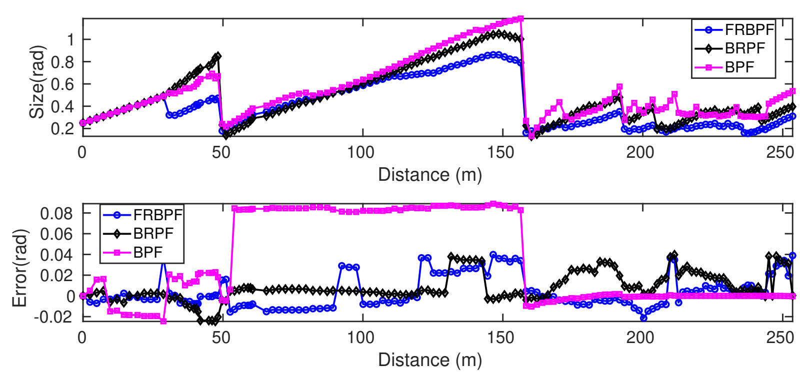


Figure 8: Average orientation box size and point-wise orientation estimation errors.

The overall localisation trajectories of FRBPF, BRPF, and BPF are also compared with the ground truth, which is given in Fig. 9. One can see that FRBPF on average achieves the best point-wise localisation results. Fig. 10 zooms in the three areas indicated by rectangles to make Fig. 9 easier to read. When compared with BRPF and BPF, FRBPF also performs the best in box volumes reduction. This can be further observed from Table II, which also shows the average point-wise estimation along the full trajectory.

The efficiency of FRBPF, BRPF and BPF are at the same level. In particular, FRBPF takes 673 ms in average per step, BRPF takes around 667 ms per step, and BPF takes 647 ms per step on average. FRBPF takes longer partially because of the q-satisfied intersection. This is intuitive as finding the q-satisfied intersection needs extra computational efforts. One can accelerate the q-satisfied intersection by decreasing the box number by a greater step than 1. In addition, this paper uses Matlab for FRBPF implementation, which is generally slower than implementations by languages such as C++. It is also worth mentioning that compared with Merlinge et al. (2019), Abdallah et al. (2008), this paper counts time consumed by extracting line features from LiDAR data and associating them with OSM, etc., which would also contribute to the total execution time.

5.3. Discussions

5.3.1. Box Area Reduction

Figs. 7 and 8 show that the proposed FRBPF has a reduced average box volumes compared with BRPF and BPF. To make it easier to understand, the box hull is adopted as an additional indicator for visualisation and comparison.

Fig. 11 shows boxes and the corresponding box hulls from one iteration of FRBPF, BRPF, and BPF, respectively. To be precise, the box-hull area of FRBPF as given in Fig. 11(a) is 43.75 m^2, which implies an average box area of 0.68 m^2. In contrast, the box-hull areas of BRPF and BPF are separately 46.46 m^2 and 56.94 m^2, as shown in Fig. 11(b) and Fig. 11(c). The corresponding average box areas are 0.73 m^2 and 0.89 m^2, respectively. One can therefore

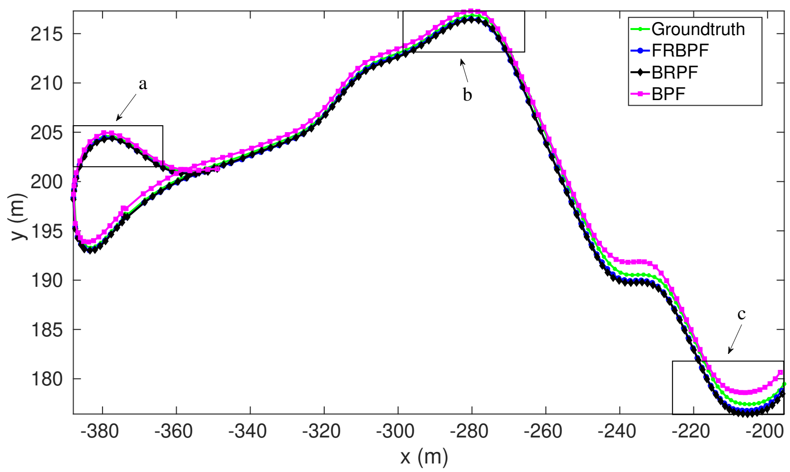
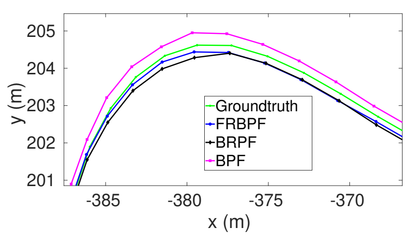
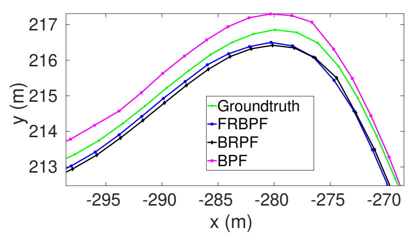


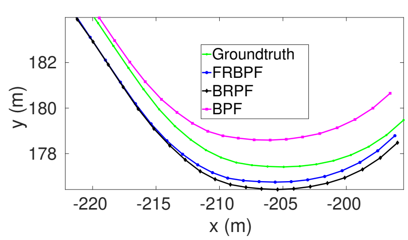
Figure 9: Comparison of FRBPF results with BRPF, BPF, and the ground truth.



(a)

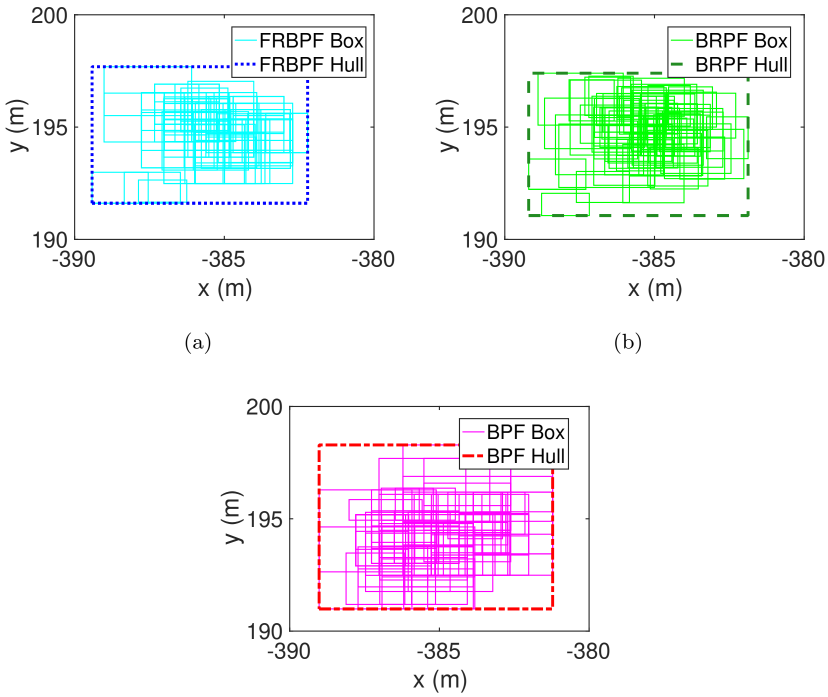


(b)



(c)

Figure 10: Zoomed figures indicated by rectangles in Fig. 9.



(a) (b)

(c)

Figure 11: Boxes and box hulls of FRBPF, BRPF, and BPF from one iteration. The dashed rectangles represent box hulls, and rectangles within the box hulls are boxes from each algorithm.

conclude that FRBPF helps in box areas reduction, which holds when q in the q-satisfied intersection equals or slightly smaller than the number of line features. In the worst case, i.e. $q = 1$ and one randomly selects one measurement for contraction, it could lead to the increase of the box-hull area. Alternatively, if all the measurements are used for contraction one by one, then FRBPR degenerates to BRPF, hence the box-hull area would be similar to BRPF. This can be observed from Figs. 7 and 8.

5.3.2. The Impacts of Weight Balance

Let consider the case shown in Fig. 12, where the footprint of a building on OSM does not align with the real surface due to the inaccuracy of the mapping. Let's suppose there are only two predicted boxes denoted by $[\mathbf{x}^i_{k|k-1}]$ and $[\mathbf{x}^j_{k|k-1}]$, with areas $|[\mathbf{x}^i_{k|k-1}]|$ and $|[\mathbf{x}^j_{k|k-1}]|$, respectively. Without loss of generality, let's also assume that $|[\mathbf{x}^j_{k|k-1}]| = |[\mathbf{x}^i_{k|k-1}]|$ with the same weights. In this example, the vehicle is located within $[\mathbf{x}^i_{k|k-1}]$. When the vehicle gets a LiDAR point cloud reflected by the real surface, a set of measurements $\mathbf{y}_k = ([r_1], [\psi_1])^T$ is obtained from the extracted line feature. In the meantime, a corresponding set of predicted measurements can be calculated based on the state prediction and the OSM feature for each particle. The predicted measurements for the two particles in Fig. 12 are denoted as $\tilde{\mathbf{y}}^i_k = ([\tilde{r}^i_1], [\tilde{\psi}^i_1])^T$ and $\tilde{\mathbf{y}}^j_k = ([\tilde{r}^j_1], [\tilde{\psi}^j_1])^T$. Innovations $([\Delta r^i_1], [\Delta\psi^i_1])^T$ and $([\Delta r^j_1], [\Delta\psi^j_1])^T$ are next calculated following (47) for contraction.

Without loss of generality, let assume that $[\Delta\psi^i_1] = [\Delta\psi^j_1]$, which makes the contraction solely depending on $[\Delta r^i_1]$ and $[\Delta r^j_1]$. When $|[\Delta r^j_1]| > |[\Delta r^i_1]|$, it means that the measurement $([r_1], [\psi_1])^T$ is more compatible with $([\tilde{r}^j_1], [\tilde{\psi}^j_1])^T$ than with $([\tilde{r}^i_1], [\tilde{\psi}^i_1])^T$. This leads to higher likelihood for particle $[\mathbf{x}^j_k]$ than for $[\mathbf{x}^i_k]$ according to (20). Given that both $([r_1], [\psi_1])^T$ and OSM are identical to each particle, the only reason lies in the difference of the predicted states. As $|[\mathbf{x}^j_{k|k-1}]|$ is assumed to be equal to $|[\mathbf{x}^i_{k|k-1}]|$, one can image the center of $[\mathbf{x}^j_{k|k-1}]$ is further away from the real surface than $[\mathbf{x}^i_{k|k-1}]$, hence further away from the real location of the vehicle. This contradicts with the likelihood of $[\mathbf{x}^j_k]$ is higher than $[\mathbf{x}^i_k]$. Hence, the weight balance is incorporated in the FRBPF to mitigate such problems.

The importance of weight balance when an OSM is studied further with four settings of α (0.0, 0.5, 0.7, and 1.0). The estimation errors are given in Fig. 13. One can see that when α is used to balance the weight, the point-wise estimation performance of FRBPF is improved as given in Fig. 13(a) and 13(b). It is worth mentioning that when using the weight balance, one should still emphasise on the high likelihood particles by setting α above 0.5. Indeed, when α is set to be small (such as 0 or 0.5), the estimation errors remain high. Based on these results, the value of α=0.7 in the best choice with the used OSM.

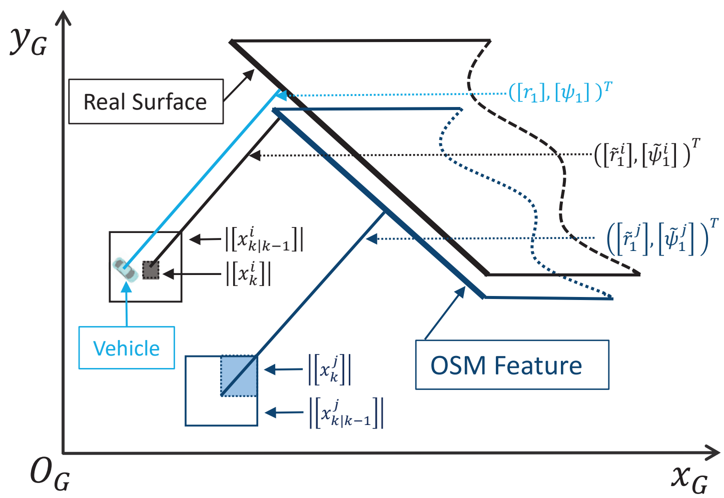
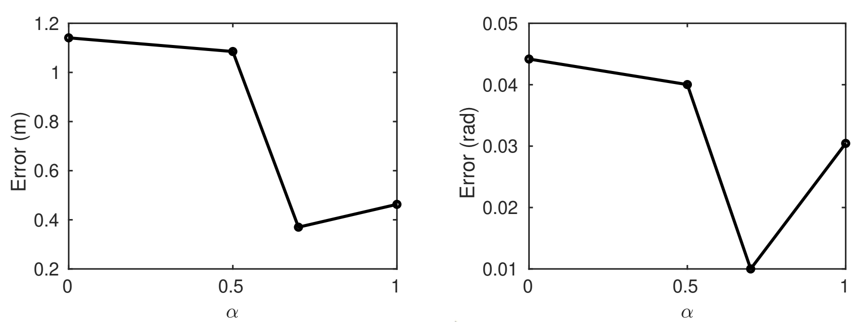


Figure 12: An example demonstrating the necessity of weight balance because of a map error.



(a) (b)

Figure 13: The effect of α on the performance of FRBPF: (a) Average position errors; (b) Average orientation errors.

6. Conclusions

A features-refined box particle filter framework has been proposed. The theoretical proofs are derived first - about the contraction step which is a key for the reduction of the size of the box particles. Next, the effectiveness of the features-refined box particle filter for vehicle localisation based on OpenStreetMap has been demonstrated. Line features extracted from LiDAR point-clouds are associated with OSM line features to enable features-refined contraction and so improve localisation accuracy. A weight balance strategy has been proposed to improve the performance of the proposed features-refined box particle filter when dealing with the uncertainty present in the map.

The proposed framework successfully localises a vehicle using LiDAR and OSM, with better point-wise state estimation accuracy and smaller box volumes compared with the generic box particle filter and the state-of-the-art interval analysis based box regularisation particle filter. The future work will continue in two directions: 1) Fusion of multiple types of sensor data within the box particle filtering approach; 2) Evaluation of the accuracy of OSM in large scale environments, hence focusing on expanding the scalability of the approach.

Appendix A. Proof of Corollary 1

Proof: Suppose there exists an x, such that

$$x \in \bigcap\nolimits_{i=1}^{n} \mathbf{g}^{-1}\big([\mathbf{y}_i]\big).$$

which is equivalent to

$$\mathbf{g}(x) \in \bigcap\nolimits_{i=1}^{n} [\mathbf{y}_i].$$

It can be further rewritten as

$$\mathbf{g}(x) \in [\mathbf{y}_i],\ i = 1, \cdots, n.$$

Hence, the following equation holds,

$$x \in \mathbf{g}^{-1}\big([\mathbf{y}_i]\big),\ i = 1, \cdots, n,$$

which indicates

$$x \in \mathbf{g}^{-1}\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big)$$

Therefore, **Corollary** 1 is proved.

Appendix B. Proof of Corollary 2

Proof: For brevity, let us denote

$$[\mathbf{g}^{-1}]\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big) \triangleq \big[\mathbf{z}\big], \tag{B.1}$$

where $\mathbf{z} = ([\underline{z}^1, \overline{z}^1], [\underline{z}^2, \overline{z}^2], \cdots, [\underline{z}^d, \overline{z}^d])^T$ is a box.

According to **Corollary** 1, the following equation holds,

$$\begin{aligned} \mathbf{g}^{-1}\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big) &= \bigcap\nolimits_{i=1}^{n}\mathbf{g}^{-1}\big([\mathbf{y}_i]\big) \\ &\subseteq [\mathbf{g}^{-1}]\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big). \end{aligned} \tag{B.2}$$

In addition, the following equation stands,

$$\bigcap\nolimits_{i=1}^{n}\mathbf{g}^{-1}\big([\mathbf{y}_i]\big) \subseteq \bigcap\nolimits_{i=1}^{n}[\mathbf{g}^{-1}]\big([\mathbf{y}_i]\big), \tag{B.3}$$

which is in accordance with inclusion function attributes.

Now proving **Corollary** 2 equals to prove $\forall \mathbf{z}$, if

$$\mathbf{z} \in [\mathbf{g}^{-1}]\big(\bigcap\nolimits_{i=1}^{n}[\mathbf{y}_i]\big) \setminus \bigcap\nolimits_{i=1}^{n}\mathbf{g}^{-1}\big([\mathbf{y}_i]\big) \tag{B.4}$$

holds, then the following equation stands,

$$\mathbf{z} \in \bigcap\nolimits_{i=1}^{n}[\mathbf{g}^{-1}]\big([\mathbf{y}_i]\big). \tag{B.5}$$

Suppose that $\exists \mathbf{z}_i = ([\underline{z}_i^1, \overline{x}_i^1], [\underline{x}_i^2, \overline{x}_i^2], \cdots, [\underline{x}_i^d, \overline{x}_i^d])^T$ with $i \in \mathbb{N}^+$, equation (B.4) holds but (B.5) does not, which means that there exist at least one dimension $j \in \{1, \cdots, d\}$, such that

$$\underline{x}_i^j \geqslant \underline{x}_i \quad \text{or} \quad \overline{x}_i^j \leqslant \overline{x}_i. \tag{B.6}$$

Without loss of generality, let's suppose that $\underline{x}_i^j \geqslant \underline{x}_i$ stands. Then a new box

$$\begin{aligned}\mathbf{o} &= ([\underline{o}_1, \overline{o}_1], [\underline{o}_2, \overline{o}_2], \cdots, [\underline{o}_N, \overline{o}_N]) \\ &= \mathbf{x} \setminus \mathbf{x}_i\end{aligned} \tag{B.7}$$

can be obtained, which satisfies

$$\mathbf{o} \bigcap \mathbf{g}^{-1}\big(\bigcap_{i=1}^{n} [\mathbf{y}_i]\big) \neq \emptyset. \tag{B.8}$$

This contradicts (B.3). Therefore, **Corollary** 2 stands.

Acknowledgment

Experiments were carried out in the framework of the Equipex ROBOTEX (ANR-10- EQPX-44-01). We are grateful also to the support by the National Natural Science Foundation of China (Grant No. 61703387), UK EPSRC for funding this work through EP/T013265/1 project NSF-EPSRC:ShiRAS - Towards Safe and Reliable Autonomy in Sensor Driven Systems and the USA National Science Foundation under Grant NSF ECCS 1903466.

References

Abdallah, F., Gning, A., Bonnifait, P., 2008. Box particle filtering for nonlinear state estimation using interval analysis. Automatica 44 (3), 807–815.

Alefeld, G., Mayer, G., 2000. Interval analysis: theory and applications. Journal of Computational and Applied Mathematics 121 (1-2), 421–464.

Brovelli, M.A., Minghini, M., Molinari, E.M., Zamboni, G., 2016. Positional accuracy assessment of the openstreetmap buildings layer through automatic homologous pairs detection: The method and a case study. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 41, 615–620.

Drevelle, V., Bonnifait, P., 2013. Localization confidence domains via set inversion on short-term trajectory. IEEE Transactions on Robotics 29 (5), 1244–1256.

Freitas, A.D., Mihaylova, L., Gning, A., Schikora, M., Ulmke, M., Angelova, D., Koch, W., 2018. A box particle filter method for tracking multiple extended objects. IEEE Transactions on Aerospace and Electronic Systems 55 (4), 1640–1655.

Gil, A., Juliá, M., Reinoso, Ó., 2015. Occupancy grid based graph-SLAM using the distance transform, SURF features and SGD. Engineering Applications of Artificial Intelligence 40, 1–10.

Gning, A., Ristic, B., Mihaylova, L., 2012. Bernoulli particle/box-particle filters for detection and tracking in the presence of triple measurement uncertainty. IEEE Transactions on Signal Processing 60 (5), 2138–2151.

Gning, A., Ristic, B., Mihaylova, L., Abdallah, F., 2013. An introduction to box particle filtering [lecture notes]. IEEE Signal Processing Magazine 30 (4), 166–171.

Holỳ, B., 2018. Registration of lines in 2D LIDAR scans via functions of angles. Engineering Applications of Artificial Intelligence 67, 436–442.

Ilog, S., 1999. Revising hull and box consistency. In: Proceedings of the International Conference on Logic Programming (ICLP), MIT Press, pp. 230–245.

Jaulin, L., 2009a. A nonlinear set membership approach for the localization and map building of underwater robots. IEEE Transactions on Robotics 25 (1), 88–98.

Jaulin, L., 2009b. Robust set-membership state estimation; application to underwater robotics. Automatica 45 (1), 202–206.

Jaulin, L., Desrochers, B., 2014. Introduction to the algebra of separators with application to path planning. Engineering Applications of Artificial Intelligence 33, 141–147.

Jaulin, L., Kieffer, M., Didrit, O., Walter, E., 2001. Interval Analysis. In: Applied Interval Analysis. Springer, pp. 11–43.

Javanmardi, E., Gu, Y., Javanmardi, M., Kamijo, S., 2019. Autonomous vehicle self-localization based on abstract map and multi-channel lidar in urban area. IATSS research 43 (1), 1–13.

Joshi, A., James, M.R., 2015. Generation of accurate lane-level maps from coarse prior maps and lidar. IEEE Intelligent Transportation Systems Magazine 7 (1), 19–29.

Kuutti, S., Fallah, S., Katsaros, K., Dianati, M., Mccullough, F., Mouzakitis, A., 2018. A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications. IEEE Internet of Things Journal 5 (2), 829–846.

Li, J., Zhao, J., Kang, Y., He, X., Ye, C., Sun, L., 2019. DL-SLAM: Direct 2.5 D LiDAR SLAM for autonomous driving. In: Proceedings of the IEEE Intelligent Vehicles Symposium (IV), IEEE, pp. 1205–1210.

Luo, J., Qin, S., 2018. A fast algorithm of simultaneous localization and mapping for mobile robot based on ball particle filter. IEEE Access 6, 20412–20429.

Merlinge, N., Dahia, K., Piet-Lahanier, H., Brusey, J., Horri, N., 2019. A box regularized particle filter for state estimation with severely ambiguous and non-linear measurements. Automatica 104, 102–110.

Pendleton, S.D., Andersen, H., Du, X., Shen, X., Meghjani, M., Eng, Y.H., Rus, D., Ang, M.H., 2017. Perception, planning, control, and coordination for autonomous vehicles. Machines 5 (1), 6–60.

Reid, T.G., Houts, S.E., Cammarata, R., Mills, G., Agarwal, S., Vora, A., Pandey, G., 2019. Localization requirements for autonomous vehicles. arXiv preprint arXiv:1906.01061.

Rohou, S., Jaulin, L., Mihaylova, L., Le Bars, F., Veres, S.M., 2018. Reliable non-linear state estimation involving time uncertainties. Automatica 93, 379–388.

Senaratne, H., Mobasheri, A., Ali, A.L., Capineri, C., Haklay, M., 2017. A review of volunteered geographic information quality assessment methods. International Journal of Geographical Information Science 31 (1), 139–167.

Suger, B., Burgard, W., 2017. Global outer-urban navigation with openstreetmap. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 1417–1422.

Tamas, L., Goron, L.C., 2014. 3D semantic interpretation for robot perception inside office environments. Engineering Applications of Artificial Intelligence 32, 76–87.

Teslić, L., Škrjanc, I., Klančar, G., 2011. EKF-based localization of a wheeled mobile robot in structured environments. Journal of Intelligent & Robotic Systems 62 (2), 187–203.

Vargas-Munoz, J.E., Srivastava, S., Tuia, D., Falcão, A.X., 2021. Openstreetmap: Challenges and opportunities in machine learning and remote sensing. IEEE Geoscience and Remote Sensing Magazine 9 (1), 184–199.

Wang, P., Chen, Z., Zhang, Q., Sun, J., 2016. A loop closure improvement method of gmapping for low cost and resolution laser scanner. IFAC-PapersOnLine 49 (12), 168–173.

Wang, P., Xu, P., Bonnifait, P., Jiang, J., 2018. Box particle filtering for SLAM with bounded errors. In: Proceedings of the International Conference on Control, Automation, Robotics and Vision (ICCARV), IEEE, pp. 1032–1038.

Wang, P., Zhang, Q., Chen, Z., 2015. A grey probability measure set based mobile robot position estimation algorithm. International Journal of Control, Automation and Systems 13 (4), 978–985.

Zhang, J., Singh, S., 2014. LOAM: Lidar odometry and mapping in real-time. In: Proceedings of Robotics: Science and Systems Conference (RSS), vol. 2, pp. 109–118.

Zheng, Y., Izzat, I.H., 2018. Exploring openstreetmap capability for road perception. In: Proceedings of the IEEE Intelligent Vehicles Symposium (IV), IEEE, pp. 1438–1443.