A Monte Carlo particle filter formulation for mapless-based localization

André Przewodowski*
carlos.andre.filho@usp.br

Fernando Santos Osório*
fosorio@usp.br

***Abstract*— In this paper, we extend the Monte Carlo Localization formulation for a more efficient global localization using coarse digital maps (for instance, the OpenStreetMap maps). The proposed formulation uses the map constraints in order to reduce the state dimension, which is ideal for a Monte Carlo-based particle filter. Also, we propose including to the data association process the matching of the traffic signals' information to the road properties, so that their exact position do not need to be previously mapped for updating the filter. In the proposed approach, no low-level point cloud mapping was required and neither the use of LIDAR data. The experiments were conducted using a dataset collected by the CARINA II intelligent vehicle and the results suggest that the method is adequate for a localization pipeline. The dataset is available online and the code is available on GitHub.**

I. Introduction

Localization is an important task for autonomous navigation and is mostly performed using Global Navigation Satellite System (GNSS) devices, High-definition (HD) maps or both. While the former is prone to signal shadowing due to dense vegetation or buildings in the surroundings, the latter requires mapping the environment precisely before the vehicle can actually drive autonomously on it. Also, the HD maps tend to be built using expensive sensor suite, composed by one or many LIDAR (light detection and ranging) devices and semantic layers of information and maps are required to be updated constantly.

On the other hand, the mapless[1] approaches are an alternative that consider using less precise and geometrically detailed maps. In this sense, the OpenStreetMap (OSM)[2] is a very interesting resource, since it contains high-level maps that are useful for human-level navigation and updating its data is easier than updating HD maps. The problem, however, lies in performing meaningful data association between the low-level data of the sensors and the map representation. Even more: features are sparse when compared to the point clouds provided by the HD map approaches.

Therefore, in this paper, we propose a novel Monte Carlo Localization (MCL) formulation for performing localization in coarse digital maps, as the OSM, using a compact state representation. In this paper, we apply the proposed formulation using the road map layer provided at the OSM website as the map. The implementation of this method is available at `https://github.com/cabraile/Mapless-MCL-ROS`, where the ROS nodes, demonstrations and links for the collected dataset are provided as well.

During the experiments, the output of the proposed method was compared to the groundtruth both alone and fused in a localization pipeline. The average Euclidean distance to the groundtruth was approximately $6.1m$ using the alone approach and $5.6m$ using the localization pipeline. Notice that no assistance of the GPS was required for the proposed localization pipeline.

The contributions of this project are:

- A new, minimalist, formulation for Monte Carlo localization;
- No prior mapping step with the intelligent vehicle was required;
- The inclusion of the higher-level data-association, which steps beyond the current semantic approaches;
- The ROS packages for global localization using the proposed method.

This manuscript is organized as follows:

- In Section II we provide a brief survey on mapless localization;
- In Section III we detail the proposed method;
- The setup and results of the conducted experiments are detailed in Section IV;
- We cover the main topics presented in this paper and provide conclusions in Section V.

II. Related Work

Mapless-based localization is a broad topic and involves many types of techniques from local to global localization. By mapless, in this paper we refer to approaches that do not require the detailed 3D maps for localization or, mostly, any type of prior SLAM-based mapping of the environment.

Methods that belong to this scope, as a consequence, tend to model data association different from the usual low-level, keypoint-based, approaches. As an example, in [2] they proposed performing digital map-based localization matching text signs in the surroundings with the digital map text database. In their approach, Global Positioning System (GPS) is used for reducing the scope of texts to be detected - which in return constrains the positions of the agent. Their approach does not consider the sequence of data, which contrasts to the proposed method. Our proposed method considers the sequence of data, odometry and data associations for estimating the global position of the agent.

* The authors are with the *Laboratório de Robótica Móvel* (LRM) of the *Instituto de Ciências Matemáticas e de Computação* (ICMC), at the *Universidade de São Paulo* (USP), *São Carlos (SP)*, Brazil.

[1] Despite the name, by *mapless* we refer to approaches that do not require detailed and high precision maps as in [1]

[2] At `https://www.openstreetmap.org`

Closer to the approach proposed in this paper, in [3], they perform localization using the road maps from the OSM and visual odometry only. The model of the state representation used in this paper is adapted from their model. However, their approach performs association in the whole map scope, which implies on a longer time for convergence before providing the first estimation. In our approach we propose using the agent's planned trajectory for reducing the scope of localization and matching the high-level information extracted from the images to the map features - which increase the pose confidence. Another difference is that they use a Gaussian Mixture Model for estimating the state, while the method proposed in this paper uses an adaptation of the Monte Carlo localization.

An end-to-end Deep Learning approach for autonomous navigation was proposed in [4], in which localization is one of the outcomes of their neural network. In their approach, GNSS data, three camera images and a local image of the roads provided by the digital map are provided as input to the network. In our approach, GNSS data is not required, even though can be used for improving estimations. Furthermore, we do not use Deep Learning for the localization task: Deep Learning was used for the detection of traffic lights and stop signs, which can be replaced by other types of methods if necessary.

A more recent end-to-end Deep Learning approach was proposed in [1] where, given high-level goals and a voxelized LIDAR's pointcloud of the surroundings, the vehicle was able to estimate its trajectory. Our proposal goes towards the usage of higher-level goals but is planned for being modular, which contrasts to end-to-end approaches.

Even though the methods mentioned so far avoid using low-level features for data association, in [5] and [6], they make use of the 2D geometry of the environment provided by a digital map for state estimation. In [5], they use a 3D digital map, which is a 2D digital map with height information of the building footprints, which was provided by the Japan Geospatial Information authority. They use a particle filter approach for state estimation fusing a GNSS, an Inertial Navigation System (INS) and the depth image computed using stereo cameras for lane-level localization. In [6] , they use LIDAR data to match 2D geometric edges from objects of a digital map, as buildings and street furniture. In their approach, they also use the Monte Carlo Localization (MCL) for position estimation. However, their approach require updated and detailed mapping of the objects and buildings.

In [7] we presented the state representation used in this paper and a Gaussian approximation of the state distribution for estimating the state. The method proposed in our previous work was limited, considering that it relied on very sparse data for data association and that the probability distribution used for state estimation could explore more of the map's structure. Both issues were addressed by the proposed method in this paper.

III. METHOD

Monte Carlo Localization (MCL) is a particle filter method in which the particles are sampled from the motion model and are weighted proportional to the measurement model. These weighted particles are resampled and the remaining particles represent the distribution of the estimation.

Traditionally, MCL was designed for estimating 2D pose directly given a 2D point map. However, in this paper, we propose representing the map and the state different to the classic MCL approaches. This difference in the state model requires manipulating the proposal and target distributions in order to validate theoretically the proposed concepts. These concepts are detailed later in this section.

A. Map Representation

Towards the goal of higher-level localization, we used the road network downloaded from the OpenStreetMap map. From the road network, information as the speed limit, direction and intersections are available. In our last research paper [7] we explored the speed limits and landmarks as features for data association. However, while the former is a very sparse, the latter required recent images from the landmark surroundings.

Therefore, in this paper we propose using two scene elements as *evidence* of road intersections: stop signs and traffic lights. Notice that, while these elements do not need to be explicitly mapped, their presence in the field-of-view of the vehicle is a strong evidence of the existence of intersections, which should impact on the probability distribution of the agent.

Therefore, this type of map representation is compact, requires relatively low effort for updating the map and allows efficient state estimation.

B. State Representation

While the road networks provide the mentioned benefits, the vehicle requires trajectories to be provided for navigating. These trajectories can be provided in many ways, among them: the world coordinates over which they need to navigate; or the sequence of lanes used for navigation. In this project, we assume that the trajectories are represented by subsets of coordinates, from which each subset indicates its total length and a identifier that references its corresponding road the road map. In this project, the trajectory, also referenced by *route*, is responsible for constraining the state of the agent and, therefore, is a fundamental part for the state representation.

Let us represent the state of the agent at the current time step t being $S_t = (L_t, R)$, where: L represents the random variable of the cumulative forward offset in the route, which is incremented using odometry data; and R corresponds to the random variable of the trajectory that the agent follows.

Therefore, if we know the route $R = r$ and the cumulative offset $L = l$ of the agent along the route, retrieving the agent's position in the world coordinates is trivial: since each way provides the two spatial nodes, p_i and p_f (in latitude and

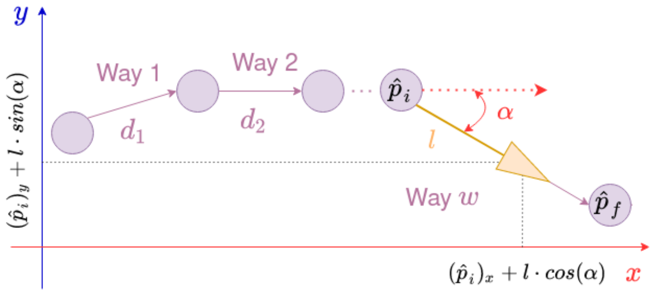

Fig. 1: Conversion from the vehicle state (l, r) (represented by the orange triangle) to the (x, y) coordinate system.

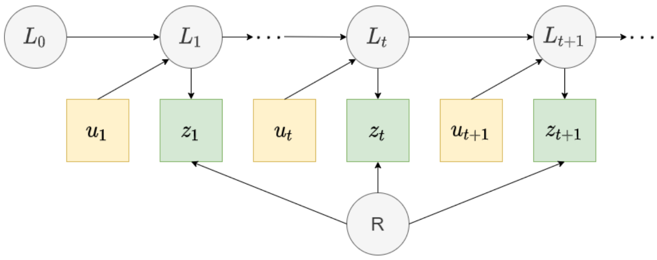

Fig. 2: Bayes network for the derivations used in this project. The gray circles represent the hidden (latent) variables. The squares represent evidence variables, in other words, variables that can be measured. The yellow squares represent the odometry source u, the green squares represent the measurement z.

longitude coordinates), we convert then to Cartesian coordinates, namely $\hat{p}_i = ((\hat{p}_i)_x, (\hat{p}_i)_y)$ and $\hat{p}_f = ((\hat{p}_f)_x, (\hat{p}_f)_y)$, and compute the line segment equation (based on the method presented by [3]):

$$\begin{cases} x = (\hat{p}_i)_x + d_{diff} \cdot cos(\alpha) \\ y = (\hat{p}_i)_y + d_{diff} \cdot sin(\alpha) \end{cases}, \tag{1}$$

where $\alpha = arctan2((\hat{p}_f)_y - (\hat{p}_i)_y, (\hat{p}_f)_x - (\hat{p}_i)_x)$ is the angle of the line segment with respect to the easting direction and d_{diff} is the distance from the agent to the node $\hat{p}_i$. d_{diff} is computed subtracting from l the cumulative route length $\sum_{j=1}^{w} d_j$ up to the way where the agent is.

This conversion from (l, r) to (x, y) is illustrated in Figure 1.

C. State Assumptions

For the proposed method, we use the Markov assumptions for state estimation in which, for the motion model, the current offset L_t requires knowledge only of the previous offset L_{t-1} and the control action u_t that led L_{t-1} to L_t. Also, the measurement model of z_t is dependent only on the knowledge of L_{t-1} and the route R. These assumptions are encoded in the Bayes network illustrated in Figure 2 and are used for the model of the Monte Carlo Localization proposed in this paper.

D. Monte Carlo Localization Formulation

In the Monte Carlo Localization, the goal is to estimate the distribution density of the trajectory from the start time 0 to time t given the control actions $u_{1:t}$ and the measurements $z_{1:t}$, $Bel(S_{0:t}) = P(S_{0:t}|u_{1:t}, z_{1:t})$. In MCL, estimating the belief $Bel(S_{0:t})$ is performed using the Sampling Importance Resampling (SIR) principles.

Sampling is performed over the motion model applied on the previous belief as the **proposal** distribution, also known as the predicted state, $P(S_{0:t+1}|u_{1:t+1}, z_{1:t})$, which can be reformulated to

$$\begin{aligned} &P(S_{0:t+1}|u_{1:t+1}, z_{1:t}) \\ &= P(S_{t+1}|S_{0:t}, u_{1:t+1}, z_{1:t})P(S_{0:t}|u_{1:t+1}, z_{1:t}) \\ &= P(L_{t+1}, R|L_{0:t}, R, u_{1:t+1}, z_{1:t})P(S_{0:t}|u_{1:t+1}, z_{1:t}) \\ &= P(L_{t+1}|L_{0:t}, R, u_{1:t+1}, z_{1:t})P(S_{0:t}|u_{1:t+1}, z_{1:t}). \end{aligned}$$

We can apply the conditional independence properties encoded in the Bayes Network proposed resulting in:

$$\begin{aligned} &P(S_{0:t+1}|u_{1:t+1}, z_{1:t}) \overset{\text{Assumptions}}{=} \\ &P(L_{t+1}|L_t, u_{t+1})P(S_{0:t}|u_{1:t}, z_{1:t}), \end{aligned}$$

which is the product between the motion model and the previous belief

$$P(S_{0:t+1}|u_{1:t+1}, z_{1:t}) = P(L_{t+1}|L_t, u_{t+1})Bel(S_{0:t}). \tag{2}$$

In practice this means that, given the set of particles that describe the belief $Bel(S_{0:t})$, for each particle k, we can sample its new position from $l_{t+1}^{[k]} \sim P(l|l_t^{[k]}, u_{t+1})$, while its route remains unchanged.

Even though the N particles are sampled from $P(S_{0:t+1}|u_{1:t+1}, z_{1:t})$, the goal is to estimate $Bel(S_{0:t+1})$ (**target** distribution). The weights to be computed are defined as $w^{[k]}$ = target $\div$ proposal. Therefore, for a particle k:

$$\begin{aligned} w_{t+1}^{[k]} &= \frac{p(s_{0:t+1}^{[k]}|u_{1:t+1}, z_{1:t+1})}{p(s_{0:t+1}^{[k]}|u_{1:t+1}, z_{1:t})} \\ &\overset{\text{Bayes Rule}}{=} \eta p(z_{t+1}|s_{0:t+1}^{[k]}, u_{1:t+1}, z_{1:t}) \frac{p(s_{0:t+1}^{[k]}|u_{1:t+1}, z_{1:t})}{p(s_{0:t+1}^{[k]}|u_{1:t+1}, z_{1:t})} \\ &= \eta p(z_{t+1}|s_{0:t+1}^{[k]}, u_{1:t+1}, z_{1:t}) \end{aligned}$$

where η is constant with respect to s for all particles. Therefore, applying the Bayes network assumptions results in

$$w_{t+1}^{[k]} = \eta p(z_{t+1}|s_{t+1}^{[k]}). \tag{3}$$

After computing their weights, all the particles pass through a resampling step proportional to their weights. In this paper, we use the low-variance sampler.

Notice that, even though the belief $Bel(S_{0:t})$ is the distribution of the state for all the trajectory of the agent, keeping only the particles sampled at time t is enough for represent the belief.

E. Data Association

One of the advantages of using digital maps as the OSM is that higher-level information of each way is available, as their speed limit, direction (which are one-way or not) and road intersections. This allows using high-level information detected on images for filtering the possible locations of the agent without known the objects' exact position.

Since the exact position of the objects are not known or, yet, since they serve as a proxy to another feature - as traffic lights and stop lights suggest the existence of road intersections - the probability distribution of the measurement model is not performed the same way as in the approaches that use low-level features. Instead, our model considers regions of detections in the trajectory, as we proposed in [7].

The proposed measurement model considers the sensitivity rate (S) and the false positive rate (F) of the detection model: for particles that are within segments of the trajectory length where the feature is detectable, the probability assigned is $S \div (S+F)$, while particles that are out of the regions where the feature is detectable are assigned with $F \div (S+F)$.

This model allows assigning weights for the sampled particles when the features are detected.

F. Putting it all together

Initialization. Given a map, when the trajectory is provided, particles are generated at its origin.

Prediction. For each odometry or velocity measurement received u_{t+1}, the new position l_{t+1} of the particles are sampled according to $P(l_{t+1}^{[k]}|u_{t+1}, l_t^{[k]})$ from the previous set of particles (which represent $Bel(S_{0:t})$). In practice, this step changes the particles' position.

Update. Given the most recent measurement z_{t+1} - which, for instance, can be a traffic light or stop sign - compute for each particle k its unnormalized weights using $p(z_{t+1}|s_{t+1}^{[k]})$, normalize so that the sum of weights is one, and then apply the low-variance sampling algorithm.

Position estimation. Finally, retrieving the estimated position, when the estimator is localized, can be performed computing the mean of the particles represented in world coordinates (using Equation 1).

IV. Experiments

For the evaluation of the proposed method, a long, continuous, route without data interruption was required. Also, the goal was to use the preliminary experiments' results as a proof-of-concept. Therefore the experiments conducted in this project used data collected using the CARINA II intelligent vehicle (Figure 3), so that it was easier to assess the divergences in the map.

A. Setup

The sensors used for collecting the data used in this project were:

- A Bumblebee XB3 stereo camera;
- A Septentrio AsteRx2eH GPS with RTK correction signals;
- A Xsens MTi-100 IMU (Inertial Measurement Unit);
- Velodyne HDL-32E 3D laser scanner.

Fig. 3: The CARINA II intelligent vehicle. Source: the *Laboratório de Robótica Móvel* webpage, at `http://lrm.icmc.usp.br`.

Processor	RAM Memory	Graphics
Intel Core i7-8750	16GB	NVIDIA GeForce GTX 1050 Ti

TABLE I: Setup of the computer used for the conducted experiments.

The data provided from the stereo camera were used for the visual odometry used during the prediction step; and the ground truth was provided fusing the GPS data with visual and inertial odometry. The point cloud data collected using the LIDAR could also be used as an odometry source or for detection improvement. However, this sensors' data was not used in the experiments.

The data were collected during a cloudy day in downtown São Carlos (Brazil) in an 8 minutes drive. For evaluating the method, the last $1.5Km$ of the driven trajectory were used. In the Figure 4, the path driven and the selected trajectory are illustrated.

The computer used for conducting the experiments is a Dell XPS 15 9570 laptop, from which the setup is displayed in Table I.

B. Tools

The data were collected and stored in the Robotics Operating System (ROS) [8] *rosbag* format, which allows easier data handling and standardization. Also, ROS provides many open-source tools off-the-shelf, among them the visualization, state estimation filters and image processing tools. As the proposed method requires a source of odometry, the ROS package of the stereo odometry module from the RTabMap library was used [9]. For the object detection, the ROS implementation of the Darknet Yolo[3] was used. We selected the YoloV3 model trained on the COCO dataset, which conveniently detects traffic lights and stop signs. Finally,

[3]From: `https://github.com/leggedrobotics/darknet_ros`.

Fig. 4: The path driven for collecting the data (line in white) and the selected trajectory for evaluation (line in orange).

Settings	Average Distance	Distance Standard Deviation
Proposed (no fusion)	$6.4m$	$4.7m$
Proposed (fused)	$5.9m$	$4.0m$

TABLE II: The Euclidean distance from the estimations to the groundtruth.

we used the extended Kalman Filter (EKF) state estimation nodes from the *Robot Localization* package [10] both for computing the groundtruth and for fusing the output of the proposed method.

We emphasize that other high-level features or objector detectors could be used and that we also compared the method to the groundtruth without fusing it to the local estimations.

C. Results

The quantitative results of the experiments are provided in Table II, while the estimated trajectories for each of the methods is illustrated in Figure 5. The number of particles used for the proposed estimation was $N = 100$.

D. Analysis

While the primary goal of the proposed method was to perform global position correction, fusing it with a source of local correction using an EKF did not bring major improvements to the estimation, which suggests that this method can be potentially improved so that it can replace a localization pipeline.

V. Discussion

In this paper, we proposed: (i) a new formulation for using the Monte Carlo Localization in a mapless context and; (ii) new higher level features and their association using a coarse digital map.

The proposed approach uses the trajectory as a constraint for the particles, which are weighed using the association between the detected in the image elements to the map abstraction elements, in this case we associated traffic lights and stop signs to the evidence of road intersection existence.

While the proposed method was able to provide estimations that, in average, were around $6m$ away from the groundtruth, there was no prior mapping step *in situ* with the intelligent vehicle.

In future works we expect to integrate orientation for filtering and as a part of the output; to explore other types of features that can be used for associating the proposed map representation; and to improve the measurement model of the proposed features.

Acknowledgment

The authors would like to thank Raiane Rodrigues and the members from the *Laboratório of Robótica Móvel* (LRM) Luis Alberto Rosero and Iago Pachêco Gomes for their aid in conducting the experiments and data collection.

This study was financed in part by the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil* (CAPES) – Finance Code 001.

References

[1] S. Casas, A. Sadat, and R. Urtasun, “MP3: A Unified Model to Map, Perceive, Predict and Plan,” *CoRR*, vol. abs/2101.0, 2021. [Online]. Available: https://arxiv.org/abs/2101.06806

[2] N. Radwan, G. D. Tipaldi, L. Spinello, and W. Burgard, “Do you see the bakery? Leveraging geo-referenced texts for global localization in public maps,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, pp. 4837–4842, 2016.

[3] M. A. Brubaker, A. Geiger, and R. Urtasun, “Map-based probabilistic visual self-localization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 652–665, 2016.

(a) No fusion.

(b) Fused using EKF.

Fig. 5: The trajectories computed for each of the settings. The colored diamonds represent the Euclidean distance from each estimation to the groundtruth (in meters).

[4] A. Amini, G. Rosman, S. Karaman, and D. Rus, “Variational end-to-end navigation and localization,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 8958–8964, 2019.

[5] P. Bhattacharyya, Y. Gu, J. Bao, X. Liu, and S. Kamijo, “3D Scene Understanding at Urban Intersection Using Stereo Vision and Digital Map,” in *IEEE Vehicular Technology Conference*, vol. 2017-June. Institute of Electrical and Electronics Engineers Inc., 11 2017.

[6] T. Quack, F. J. Heßeler, and D. Abel, “Fast real-time localization with sparse digital maps for connected automated vehicles in urban areas,” in *IFAC-PapersOnLine*, vol. 52, no. 5. Elsevier B.V., 1 2019, pp. 366–371.

[7] A. Przewodowski and F. S. Osorio, “A Gaussian approximation of the posterior for digital map-based localization using a particle filter,” 2021.

[8] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “ROS: an open-source Robot Operating System,” in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[9] M. Labbé and F. Michaud, “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation,” *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[10] T. Moore and D. Stouch, “A Generalized Extended Kalman Filter Implementation for the Robot Operating System,” in *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer, 2014.