Vehicle-to-Vehicle Collaborative Graph-based Proprioceptive Localization

Hsin-Min Cheng, Chieh Chou, and Dezhen Song, *Senior Member, IEEE*

***Abstract*—Proprioceptive localization (PL) refers to robot or vehicle egocentric localization methods that do not rely on the perception and recognition of external landmarks. These methods depend on a prior map and proprioceptive sensors such as inertial measurement units and/or wheel encoders. PL is intended to be a low-cost and fallback solution when everything else fails due to bad weather or poor environmental conditions. With the development of communication technology, vehicle-to-vehicle (V2V) communication enables information exchange between vehicles. It becomes possible to leverage V2V communication to develop a multiple vehicle/robot collaborative localization scheme. Named as collaborative graph-based proprioceptive localization (C-GBPL), we extract heading-length sequence from the trajectory as features. When rendezvousing with other vehicles, the ego vehicle aggregates the features from others and forms a merged query graph. We match the query graph with a pre-processed heading-length graph (HLG) abstracted from a prior map to localize the vehicle under a graph-to-graph matching approach. We have implemented our algorithm and tested it in both simulated and physical experiments. The C-GBPL algorithm significantly outperforms its single-vehicle counterpart in localization speed and robustness to trajectory and map degeneracy.**

***Index Terms*—Localization, Autonomous Vehicle Navigation, Multi-Robot Systems**

I. Introduction

LOCALIZATION is important for a vehicle or a robot navigating in an urban area. Common localization methods employ exteroceptive sensors, such as a camera, a laser range finder, or a global position system (GPS), which may be hindered under extreme weather or poor lighting conditions. In [1], [2], we propose proprioceptive localization (PL) as a fallback solution that is naturally immune to these extreme environmental conditions because the PL method only depends on a prior map and proprioceptive sensors such as inertial measurement units (IMUs) and/or wheel encoders. PL methods localize vehicles by extracting features from the estimated vehicle trajectories using the proprioceptive sensors and matching the features with a prior map.

However, the existing approaches suffer from strong dependence on trajectory types and slow convergence. Combining the PL method with modern vehicle-to-vehicle (V2V) communication which allows real time information exchange between vehicles, we design a new collaborative graph-based proprioceptive localization (C-GBPL) method (see Fig. 1). We extract trajectory features which are straight segments of trajectories and generate a merged query graph by combining inputs from neighboring vehicles. The localization problem becomes a graph-to-graph matching problem. Our algorithm outputs potential vehicle locations based on the the maximization of belief functions which often quickly converges to actual location over time.

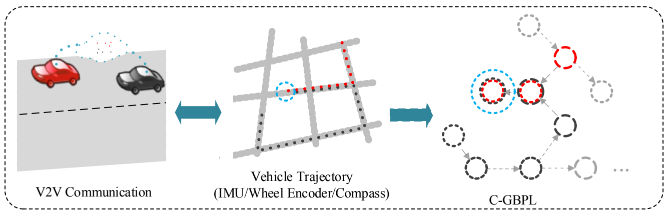


Fig. 1. An illustration of C-GBPL method using a two-vehicle rendezvous case. The ego vehicle is colored in red and the other vehicle is in black. At rendezvous, we aggregate vehicle trajectories to form a merged feature graph to facilitate localization.

Building our existing work on the single vehicle PL [2], our new collaborative framework alleviates the trajectory-dependence issue by exploiting the shared information. To facilitate the matching, we also pre-process the prior map which extends our heading-length graph (HLG) by adding super-vertices based on three different vehicle rendezvous types. The new algorithm that builds on new graph structure speeds up the matching computation. Furthermore we explicitly prove that it can accelerate the convergence of belief functions for vehicle location. We have implemented the C-GBPL algorithm and tested it against the existing approach [2]. The C-GBPL algorithm significantly outperforms its single-vehicle counterpart in localization speed and is less sensitive to trajectory and map limitations.

II. Related Work

Our C-GBPL method is related to localization using sensor fusion, map-based localization, and multi-robot localization.

Localization methods can be grouped into two categories: exteroceptive sensors and proprioceptive sensors according to sensor modalities. Exteroceptive sensors perceive external signals and recognize landmarks in the environment to estimate location. Mainstream exteroceptive sensors include cameras [3]–[5], laser range finders [6], and GPS receivers [7], [8]. These methods are considerably susceptible to adversary environmental conditions such as low visibility, extreme weather

Manuscript received: October 15, 2020; Revised December 22, 2020; Accepted January 19, 2021.

This paper was recommended for publication by Editor Civera, Javier upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by ational Science Foundation under NRI-1925037, and by General Motors and SAE. (*Corresponding author: Dezhen Song.*)

H. Cheng, C. Chou, and D. Song are with CSE Department, Texas A&M University, College Station, TX 77843, USA. Emails: `hmcheng@tamu.edu`, `jc491986@gmail.com`, and `dzsong@cs.tamu.edu`.

Digital Object Identifier (DOI): see top of this page.

conditions, or electromagnetic inference. On the contrary, proprioceptive sensors, such as IMUs [9] and wheel encoders [10] do not rely on external signals and are inherently immune to external conditions. However, they are more susceptible to error drift. Recent approaches combining a camera or a laser range finder with an IMU [11] which design is an exteroceptive-proprioceptive sensor fusion approach, greatly improves system robustness and has become popular in applications. However, these approaches still heavily rely on their exteroceptive sensors and cannot function properly under the aforementioned adversary environmental conditions.

Our PL method is a map-based localization [3], [12]–[15]. Thrun et al. [16] classify map representation into two categories: the location-based and the feature-based. The location-based maps are represented with specific locations of objects. For example, those existing geographic maps consisted of coordinate of locations such as Google Maps™ [17] and OpenStreetMaps™ (OSM) [18]. Geographic maps often based on GPS measurements. Researchers develop map matching techniques such as point-to-point, point-to-curve, curve-to-curve matching, or advanced extensions [19]. The feature-based map consists of features of interest at its location. An example is ORB features [20] for visual simultaneous localization and mapping. Our PL methods extract heading-length features from proprioceptive sensors and prior maps to convert a location-based map matching to a feature-based map matching which improves localization robustness to sensor drift and also speeds up computation in the process.

In the area of multi-robot research, decentralized estimation of robot poses has gained considerable attention [21]–[24]. The multi-robot systems outperform single-robot systems in many aspects, such as improving localization efficiently, reducing computational cost, increasing accuracy and fault tolerance, and accelerating map exploration and coverage. Our work is related to multi-robot localization in particular [21], [25]–[28]. The loosely coupled cooperative localization methods [27], [28] require relative measurement between robots which need exteroceptive perceptions between robots. In contrary, our method utilizes proprioceptive sensors only. To tackle the decentralized multi-robot localization problem, researchers propose [25] and use [24] the concept of checkpoints representing delayed synchronization of observation after exchanges of information between robots. Also, the concept is extended to the decentralized information transfer scheme [29] based on communication constraints. Our problem is similar in the way that we can benefit from exchanged motion information from other robots, but it is different because we do not rely on landmarks or exteroceptive sensing to acquire relative positions with other robots.

PL methods are gaining attention in vehicle localization [1], [2], [12], [30], [31], all of which are map-aided localization using proprioceptive sensors. Wahlstrom et al. [30] employ a minimal setup using vehicle speed and speed limit information map. Yu et al. [31] develop an extended Kalman filter (EKF)-based dead reckoning approach based on odometer and gyroscope readings and a map is used to correct errors. However, an initial position from GPS is required for both methods. Our localization solution does not require a known initial position. In [12], the velocity from the wheel encoder and steering angle are used for odometry and a particle filter based map matching scheme helps estimating vehicle positions.

Our group studies localization using proprioceptive sensors under different setups [1], [2]. This paper extends our prior work [2] for a single vehicle to a multiple-vehicle PL method.

III. Problem Formulation and System Design

All vehicles have a prior map of the city. Each vehicle is equipped with proprioceptive sensors including an IMU and an on-board diagnostics (OBD) scanner which provides velocity feedback (similar to a wheel encoder). To compensate for direction drift, each vehicle has a digital compass. To formulate our collaborative localization problem, we have the following assumptions:

a.0 The ego vehicle is able to navigate in the environment and make turns at appropriate locations. All vehicles are nonholonomic.

a.1 The prior road map contains straight segments in most part of its streets and streets are not strict grids with equal side lengths.

a.2 IMU and compass are co-located, pre-calibrated, and fixed inside the vehicle. Their readings are synchronized and time-stamped.

a.3 Vehicles communicate with each other in close range and we can assume that they are on the same street or same intersection.

As part of the input of the problem, a prior road map consisting of a set of roads with GPS waypoints is required. Common notations are defined as follows,

- $\mathcal{M}_p$ represents the prior road map which is a set of GPS positions.
- $\mathbf{z} = \{\mathbf{a}, \boldsymbol{\omega}, \phi, \mathbf{v}\}$ denote *in situ* sensory readings of vehicle, where $\mathbf{a}$ denotes accelerometer readings of the IMU, $\boldsymbol{\omega}$ denotes gyroscope readings of the IMU, ϕ denotes compass readings, and $\mathbf{v}$ denotes velocity readings.

The C-GBPL problem is defined as follows.

Problem 1: When ego vehicle l rendezvous with vehicle l', localize vehicles collaboratively given sensory reading $\mathbf{z}$, $\mathbf{z}'$, and $\mathcal{M}_p$.

It is worth mentioning that this problem formulation only concerns a two-vehicle rendezvous case. However, it is the atomic case of multiple vehicles rendezvous case because an n-vehicle rendezvous case can be easily decomposed into a sequence of $n-1$ two-vehicle rendezvous cases.

IV. Algorithm

Since our algorithm builds on GBPL algorithm, we begin with a brief review of GBPL algorithm [2] in Section IV-A. Then we will extended it into C-GBPL in Section IV-B.

A. GBPL Review

As a single vehicle localization method, GBPL employs the proprioceptive sensors to estimate vehicle trajectory and match it with a prior map. GBPL is a feature-based map matching method instead of raw trajectory matching because 1) there is a significant drift issue in the dead reckoning process and 2) the vehicle trajectory may not match the GPS waypoints on the map where a street may contain multiple lanes and trajectories may differ due to lane selection or different traffic patterns. GBPL has three main building blocks: *heading length graph (HLG) construction, query generation from sensory data, and vehicle localization*. GBPL can be viewed as a feature-based map matching with each feature to be a straight segment of a road with heading and length as its feature descriptors. *HLG construction* reduces the prior map $\mathcal{M}_p$, which is a set of GPS waypoints, to a discrete feature structure HLG $\mathcal{M}_h$ to facilitate the feature matching. *Query generation* extracts features from dead reckoning trajectory based on the proprioceptive sensors. The *vehicle localization* performs the feature-based Bayesian map matching and vehicle tracking afterwards.

1) HLG construction: HLG is a feature representation of the prior map. In HLG, a vertex $v_i \in \mathcal{V}_h$ represents a straight and continuous road segment with no intersections. An edge $e_{i,i'} \in \mathcal{E}_h$ characterizes the orientation change between the connected two vertices v_i and $v_{i'}$. $\mathcal{M}_h$ have two types of edges: road intersections and curve segments; and two types of vertices: long straight segment vertices and short transitional segment vertices. The long straight segment vertices are used for heading-length matching later. The short transitional segment vertices are often formed between curve segments or curved roads entering intersections. Each vertex contains the following information

$$v_i = \{\mathbf{X}_i, \theta_i, d_i, b_i\}, \tag{1}$$

where $\mathbf{X}_i = [\mathbf{x}_{i,s}^{\mathsf{T}}, \cdots, \mathbf{x}_{i,e}^{\mathsf{T}}]^{\mathsf{T}}$ contain all 2D positions of waypoints on the road segment, orientation $\theta_i \in (-\pi, \pi]$ is the angle between the geographic north, $d_i := ||\mathbf{x}_{i,s} - \mathbf{x}_{i,e}||$ is road segment length, and b_i is the binary variable indicate if the vertex is a long road segment. Only long road segments ($b_i = 1$) are used in localization which defines vertex subset $\mathcal{V}_{h,l} \subset \mathcal{V}_h$ corresponding to long straight segments.

2) Query Generation: When the vehicle is driving down the road, we can estimate the trajectory from sensory readings with an EKF-based approach and generate a query heading-length sequence. In the state representation, the state vector $\mathbf{X}_{s,j}$ at time j of the EKF is: $\mathbf{X}_{s,j} := [\mathbf{p}_j^I, \mathbf{v}_j^I, \Theta_j^I]^{\mathsf{T}}$, where $\mathbf{p}^I \in \mathbb{R}^3$, velocity $\mathbf{v}^I \in \mathbb{R}^3$, and the X-Y-Z Euler angles Θ^I in fixed inertial frame $\{I\}$. The EKF-based dead reckoning provides a vehicle trajectory but is inevitably drift-prone. Instead of using it for directly matching to the prior map, we extract heading and length of the straight segments for our feature-based matching. The resulting query heading-length sequence is denoted by

$$Q := \{\Theta_q, D_q\},$$

where $\Theta_q = \{\Theta_{q,k} | k = 1, \cdots, n\}$, $D_q = \{d_{q,k} | k = 1, \cdots, n\}$, and $\Theta_{q,k}$ and $d_{q,k}$ are the observations of heading and length from EKF for a straight segment k.

3) Vehicle localization: This is two-step process: i) perform global graph match that finds the query-HLG match and ii) track the vehicle location to provide continuous localization result after a global match is identified. Since the second step is the trivial, we focus on global graph match only. Given $Q = \{\Theta_q, D_q\}$, let us denote a candidate heading-length vertex sequence in $\mathcal{M}_h$ by $M := \{\Theta, D\} = \{\{\theta_k, d_k\} | k = 1, \cdots, n\}$ correspondingly. As a convention in this paper, for random vector $\star$, $\mu_\star$ represents its mean vector. The belief that Q and M match is the following conditional probability

$$P(\mu_Q = \mu_M | Q, M) \propto \prod_{k=1}^{n} f_T(t(\theta_{q,k}, \theta_k)) f(z(d_{q,k}, d_k)) \tag{2}$$

due to independent sensor noises where $f_T(t(\theta_{q,k}, \theta_k))$ is the probability density function (PDF) of Student's t-distribution and $f(\cdot)$ is the PDF of standard normal distribution. As the length of the matching sequence grows, the belief function converges for the correct matching and a global location is identified when thresholding condition is satisfied and only one solution remains. We search for the best matching by generating different candidate sequences on the map using breadth-first search on the HLG.

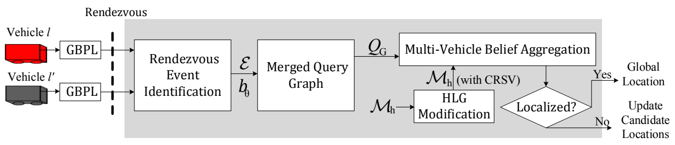


Fig. 2. C-GBPL system diagram.

B. C-GBPL

In a single vehicle case, we simply match query sequence with a candidate sequence constructed from the HLG. This changes when vehicles can talk to each other. The *rendezvous events* describe moments when a vehicle moves into another vehicle's communication range (assumption a.3). At the moment of rendezvous, one vehicle can pass its query sequence to the other. The two query sequences combine into a query graph which will be matched against HLG. The matching problem evolves into a graph-to-graph matching problem.

Fig. 2 show the system diagram of C-GBPL. Before rendezvous, each vehicle runs GBPL algorithm in [2]. In C-GPBL, we propose the following building blocks to compose and solve the graph-to-graph matching problem to simultaneously improve localization efficiency and reduce computational cost. The first block is *HLG modification* where we modify HLG with super vertex groups to capture potential rendezvous locations to facilitate the graph-to-graph matching process. With the modified HLG, the remaining three blocks are *rendezvous event identification*, *merged query graph*, and *multi-vehicle belief aggregation*. We will detail each block in separate subsections.

1) HLG Modification: At rendezvous, the vehicle pair must be within communication range of each other to exchange information (Assumption a.3). This only occurs with a limited set of possibilities and can be utilized to facilitate graph-matching because we can trim the searching space on subset of vertices in $\mathcal{M}_h$ by focusing on the possible rendezvous locations. We capture all possible rendezvous locations by augmenting the pre-processed HLG with an additional layer which are candidate rendezvous super vertex (CRSV) groups (see Fig. 3) which only have three types.

- Type 0: *same vertex*. This type is embedded in the original HLG and does not require additional processing. The number of Type 0 CRSV is $O(\mathcal{V}_{h,l})$.

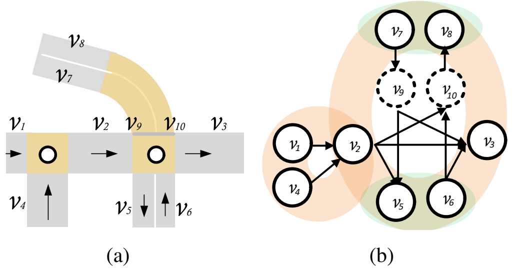


Fig. 3. HLG modification. (a) A prior road map overlay with edges (in light yellow color) and vertices (in gray color). (b) An illustration of two CRSV types: Type 1 coupled vertices (in green color) and Type 2 intersection sharing vertices (in orange color).

- Type 1: *coupled vertices.* We define coupled vertices by pair of vertices with opposite directions on same road segment. In Fig. 3(b), we show examples of coupled vertices in green color. The number of Type 1 CRSV is also $O(\mathcal{V}_{h,l})$.
- Type 2: *intersection sharing vertices.* We define intersection sharing vertices as a group of vertices sharing the same road intersection. For vertices in a group, the corresponding road segments share the same intersection. In some cases that the road intersections are connected with a curved road segment (e.g. an edge in $\mathcal{M}_h$), the nearest vertices are included instead. We show an example in Fig. 3(b) where v_7 and v_8 are considered as intersection sharing vertices. The number of Type 2 CRSV is $O(\mathcal{V}_{h,l})$ which happens when a map consists with square grids and bi-directional roads.

2) Rendezvous Event Identification: On the vehicle side, we also need to identify corresponding rendezvous types using on-board sensors.

Recall that the ego-vehicle index is denoted by l and the other vehicle index by l', where $l, l' \in L$. We develop algorithm from ego-vehicle l view and as it is established vice versa for the other vehicle l' where prime symbol $'$ indicates the other vehicle. We label the vehicle status by 'V' or 'E' based on whether it is on a vertex or an edge. For vehicles (l, l'), this results in four kinds of rendezvous events denoted by $\mathcal{E}_{E,E}$, $\mathcal{E}_{E,V}$, $\mathcal{E}_{V,E}$, and $\mathcal{E}_{V,V}$. To further reduce the four kinds into the three types in the CRSV groups, we check whether vehicles (l, l') have the same headings at rendezvous. Note whenever vehicle changes heading, it does not have stable heading. Thus we specify vehicle rendezvous heading by its *latest stable* heading. We set binary variable to $b_\theta = 1$ if vehicle (l, l') have same headings and $b_\theta = 0$ otherwise. See Fig. 4 for examples. The three types are identified as follows,

- Type 0: When $b_\theta = 1$, vehicles travel through same vertex in all four rendezvous events as shown in top four figures in Fig. 4. Vehicle trajectories are linked by hidden super vertices (Type 0) of HLG.
- Type 1: When $b_\theta = 0$, $\mathcal{E}_{V,V}$ and vehicles have opposite orientations, vehicles travel through vertices corresponding to same road segment with opposite directions. Their trajectories are linked by coupled vertices (Type 1) of HLG. This case is shown in the right bottom of Fig. 4.
- Type 2: When $b_\theta = 0$ and $\{\mathcal{E}_{E,E}, \mathcal{E}_{E,V}, \mathcal{E}_{V,E}\}$ or $\mathcal{E}_{V,V}$ and vehicles have different orientations other than opposite, vehicles rendezvous at intersection from different directions which are connected by intersection sharing vertices (Type 2) of HLG.

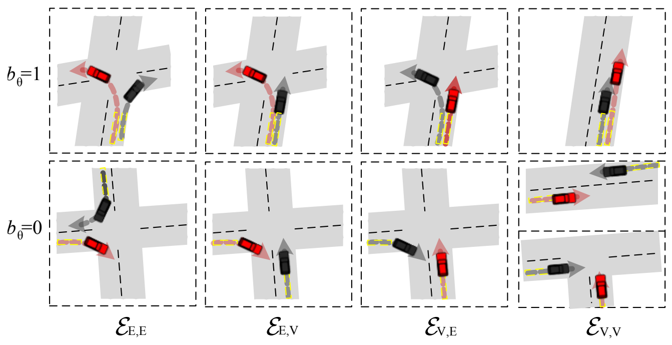


Fig. 4. Rendezvous events with same/different heading (b_θ) which we use to identify vehicle relative locations. The yellow rectangular boxes mark out the latest vehicle stable headings used to determine b_θ.

3) Merged Query Graph: With different rendezvous event types identified, we can combine individual query sequences and form a merged query graph. We denote the merged query graph by Q_G which consists of nodes and edges in query sequences/graph Q and Q' for vehicles l and l', respectively. They are connected together using the type info.

4) Multi-Vehicle Belief Aggregation: The main part of global localization is the matching of the merged query graph Q_G to the corresponding part on the modified HLG. This requires us to establish a belief function to evaluate Q_G and a candidate matching graph M_G on the map.

The type associated with Q_G helps us identify the corresponding M_G which consist of nodes in M and M', where M and M' are candidate sequences for vehicles l and l', respectively.

At rendezvous, the matching belief from vehicle l' is aggregated with matching belief from ego-vehicle l. We extend the single vehicle belief function in (2) to a multi-vehicle belief.

Lemma 1: The multi-vehicle belief function is the conditional probability for the joint belief function which is obtained by multiplying individual components below.

$$\begin{aligned} &P(\mu_Q = \mu_M, \mu_{Q'} = \mu_{M'} | Q, M, Q', M') \\ &\quad = P(\mu_Q = \mu_M | Q, M) P(\mu_{Q'} = \mu_{M'} | Q', M') \\ &\quad \propto \prod_{k=1}^{n} f_T(t(\theta_{q,k}, \theta_k)) f(z(d_{q,k}, d_k)) \times \\ &\qquad \prod_{k'=1}^{n'} f_T(t(\theta_{q,k'}, \theta_{k'})) f(z(d_{q,k'}, d_{k'})), \end{aligned} \tag{3}$$

where n and n' are number of observations in Q and Q', respectively, and k and k' are index variables.

Proof: Note that $\mu_Q = \mu_M$ and $\mu_{Q'} = \mu_{M'}$ are independent given Q, M, Q' and M'. We decompose (3) into two terms

$$\begin{aligned} &P(\mu_Q = \mu_M, \mu_{Q'} = \mu_{M'} | Q, M, Q', M') \\ &= P(\mu_Q = \mu_M | Q, M, Q', M') P(\mu_{Q'} = \mu_{M'} | Q, M, Q', M') \end{aligned} \tag{4}$$

Since $\mu_Q = \mu_M$ is independent of Q', M' given Q, M,

$$P(\mu_Q = \mu_M | Q, M, Q', M') = P(\mu_Q = \mu_M | Q, M), \tag{5}$$

which is the belief that Q and M match in (2). Similarly,

$$P(\mu_{Q'} = \mu_{M'} | Q, M, Q', M') = P(\mu_{Q'} = \mu_{M'} | Q', M'). \quad (6)$$

Plugging (5), (6) using the form derived in (2) into (4), we obtain the lemma. ■

Fig. 5 illustrates the ego vehicle belief and aggregated multi-vehicle belief after rendezvous with the other vehicle.

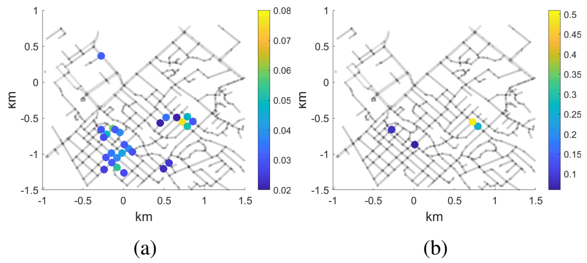


Fig. 5. An example of multi-vehicle belief aggregation (best viewed in color). The probability value of each location is colored according to color bar. (a) The candidate locations with corresponding probabilities of ego vehicle. (b) After rendezvous with the other vehicle, the candidate locations and probabilities are updated. There is a significant drop in number of candidate locations after rendezvous.

5) Algorithm Framework: Lemma 1 provides us with a method to localize the vehicle by thresholding the belief function over candidate solution M_G. It can be done by applying a breadth-first search strategy starting with matching CRSV types. We summarize the C-GBPL framework in Algorithm 1. We denote the set of Type 1 CRSV by $\mathcal{V}_{\mathrm{T}_1}$ where $\mathcal{V}_{\mathrm{T}_1} = \{V_{\mathrm{T}_{1,m_1}} | m_1 = 1, \cdots, n_{\mathrm{T}_1}\}$ and each $V_{\mathrm{T}_{1,m_1}}$ contain vertices in the same group. n_{T_1} is the cardinality of $\mathcal{V}_{\mathrm{T}_1}$. Similarly, We define $\mathcal{V}_{\mathrm{T}_2} = \{V_{\mathrm{T}_{2,m_2}} | m_2 = 1, \cdots, n_{\mathrm{T}_2}\}$. Recall that $\mathcal{V}_{h,l}$ is vertex set of HLG. For $v_i \in \mathcal{V}_{h,l}$, we augment information of CRSV type and element index for searching purpose. Note that v_i may belong to more than one CRSV types.

Not all pairs of candidate sequences M and M' satisfy the CRSV type constraint. Let us use Fig. 6 as an example. In the left figure, we show trajectories of two vehicles and their rendezvous. The right figure shows the graph matching between M and M' in HLG $\mathcal{M}_h$. In this example, the Q_G is classified as Type 2 and thus M_G belongs to Type 2 CRSV group. We show three candidates (M'_1, M'_2, M'_3) for ego vehicle. Only M'_1 satisfies the CRSV and feature sequence matching constraints and both M'_2 and M'_3 are trimmed.

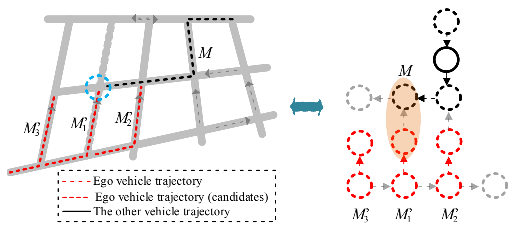

M
M'_3
M'_1
M'_2
Ego vehicle trajectory
Ego vehicle trajectory (candidates)
The other vehicle trajectory
M
M'_3
M'_1
M'_2

Fig. 6. An example of graph-based candidate trimming. Left figure: we show trajectories of two vehicles and their rendezvous. Right figure: HLG $\mathcal{M}_h$ and candidate heading-length vertex sequence M and M'. We show three candidates (M'_1, M'_2, M'_3) for ego vehicle and trim candidates are not correct CRSV type, both M'_2 and M'_3 are trimmed.

To reduce possible prior correlations, we use the following two rules. First, when the ego vehicle rendezvouses the other vehicle, we do not update its belief if they have same candidate vertices. Second, we only update the ego vehicle belief once for the same rendezvous with the other vehicle. According to assumption a.3, this implies the next rendezvous happens when either vehicles has new observations.

In Lemma 1, the product format of belief function in (2) still holds for multi-vehicle case as described in (5) and (6). The only difference is the number of nodes involved in the product. Computing the multi-vehicle beliefs in (3) can be computation intensive if we do not effectively reuse the prior computation. We store the prior computation from each individual vehicle and exchange the information between vehicles. For each vehicle, we define the candidate vertex information set $\mathcal{C}_k$ where $k = 1, \cdots, n$ is the length of the query sequence. The candidate vertex set is denoted by $\mathcal{C}_k = \{\{v_i, \mathcal{V}_{M,i}, p_i\} | i = 1, \cdots, n_{\mathcal{C}_k}\}$, where each element in $\mathcal{C}_k$ record the candidate vertex v_i (the starting vertex of the trajectory/path), $\mathcal{V}_{M,i}$ is the set of vertex path, and the matching probability p_i in (2) and $n_{\mathcal{C}_k}$ is the cardinality of $\mathcal{C}_k$. By thresholding the belief function over candidate solutions, we might still get many candidate solutions because the hypothesis is conservative in rejection as in GBPL [2]. The Otsu method [32] can be applied to further trim candidate trajectories with lower probabilities. If more than one solution survives, it indicates that more observations are needed to localize the vehicle.

We now analyze the complexity of C-GBPL algorithm. The upper bound of candidate vertex cardinality is $\mathcal{V}_{h,l}$ and thus it takes $O(|\mathcal{V}_{h,l}|)$ to traverse $\mathcal{C}_n$ and $\mathcal{C}'_n$. For Type 0 cases, we can find set intersection of $\mathcal{V}_i$ and $\mathcal{V}'_i$ in $O(|\mathcal{V}_{h,l}| \log(|\mathcal{V}_{h,l}|))$ by sorting and binary search. Similarly, for Type 1 or Type 2 cases, we can find (v_i, v'_i) pairs with the same group index in $O(n_{\mathrm{T}_1} \log(n_{\mathrm{T}_1}))$ or $O(n_{\mathrm{T}_2} \log(n_{\mathrm{T}_2}))$. In the worst case scenario, both n_{T_1} and n_{T_2} are $O(|\mathcal{V}_{h,l}|)$. Then line 7, 10, 13 are the same in complexity of $O(|\mathcal{V}_{h,l}| \log(|\mathcal{V}_{h,l}|))$. The classification of probabilities into two groups is $O(|\mathcal{V}_{h,l}|)$ using Hoare's selection algorithm. We summarize the computational complexity of Algorithm 1 in Lemma 2.

Lemma 2: The computation complexity of the C-GBPL is $O(|\mathcal{V}_{h,l}| \log(|\mathcal{V}_{h,l}|))$.

C. Localization Analysis

Intuitively, combining inputs from the other vehicle helps the ego-vehicle to reduce ambiguity in map matching and hence leads to faster convergence in the localization process. Let us show this by analyzing the conditional probability of localization given the matching sequence changes.

Let us define three binary events: $A_k = 1$ if $\mu_{d_{q,k}} = \mu_{d_k}$, $B_k = 1$ if $\mu_{\theta_{q,k}} = \mu_{\theta_k}$, and $C_k = 1$ if vertex k in M_h is the actual location. Same as [2], we employ hypothesis testing to reject unlikely matching by setting the significance level α, where α is a small probability. According to the definition, $P(A_k|C_k) = (1-\alpha)$ and and $P(B_k|C_k) = (1-\alpha)$ are the conditional probabilities that a correct matched sequence survive the test for pair $\mu_{d_{q,k}} = \mu_{d_k}$ and $\mu_{\theta_{q,k}} = \mu_{\theta_k}$ correspondingly. For convenience, for vehicle l we define joint events: $\mathcal{A} = A_1 \cdots A_n$, $\mathcal{B} = B_1 \cdots B_n$, and $\mathcal{C} = C_1 \cdots C_n$. The joint event $\mathcal{C}$ is equivalent to say that $M := \{\Theta, D\}$ represents the true trajectory, whereas we know joint event $\mathcal{AB}$ from sequence matching. Similarly, for vehicle l' we define joint events: $\mathcal{A}'$, $\mathcal{B}'$, and $\mathcal{C}'$. We define a binary event E if vehicle l rendezvous with l'. Also, there

Algorithm 1: C-GBPL

```
Input: $\mathcal{M}_h$, $\{\mathcal{C}_n, Q\}$ and $\{\mathcal{C}'_n, Q'\}$
Output: vehicle $l$ location or updated $\{\mathcal{C}_n, Q\}$
1  $\mathcal{C}_{res} \leftarrow \emptyset$, $p_{res} \leftarrow \emptyset$                                   $O(1)$
2  Identify rendezvous event and Recognize CRSV type using $Q$ and $Q'$   $O(1)$
3  Form merged query graph $Q_G$                                         $O(1)$
4  $\mathcal{V}_i \leftarrow$ latest vertices in $\mathcal{C}_n$, $\mathcal{V}'_i \leftarrow$ latest vertices in $\mathcal{C}'_n$       $O(|\mathcal{V}_{h,l}|)$
5  switch CRSV Type do
6      case 'Type 0' do
7          $\mathcal{V}_i \leftarrow$ intersect of $\mathcal{V}_i$, $\mathcal{V}'_i$          $O(|\mathcal{V}_{h,l}|\log(|\mathcal{V}_{h,l}|))$
8      case 'Type 1' do
9          $\mathcal{V}_i \leftarrow \forall v_i \in \mathcal{V}_i$ with label $\mathtt{T}_1$, $\mathcal{V}'_i \leftarrow \forall v'_i \in \mathcal{V}'_i$ with label $\mathtt{T}_1$   $O(|\mathcal{V}_{h,l}|)$
10         Find $(v_i, v'_i)$ pair with same element index;      $O(n_{\mathtt{T}_1}\log(n_{\mathtt{T}_1}))$
11     case 'Type 2' do
12         $\mathcal{V}_i \leftarrow \forall v_i \in \mathcal{V}_i$ with label $\mathtt{T}_2$, $\mathcal{V}'_i \leftarrow \forall v'_i \in \mathcal{V}'_i$ with label $\mathtt{T}_2$   $O(|\mathcal{V}_{h,l}|)$
13         Find $(v_i, v'_i)$ pair with same element index;      $O(n_{\mathtt{T}_2}\log((n_{\mathtt{T}_2}))$
14 for $v_i \in \mathcal{V}_i$ do                                         $O(|\mathcal{V}_{h,l}|)$
15     Access $(v_i, v'_i)$ pair information                             $O(1)$
16     $p \leftarrow p_i \cdot p'_i$                                         $O(1)$
17     $p_{res} \leftarrow p_{res} \cup p$                                     $O(1)$
18     $\mathcal{C}_{res} \leftarrow \mathcal{C}_{res} \cup \mathcal{C}_{n,i}$                          $O(1)$
19 Classify $p_{res}$ using Otsu's method;                              $O(|\mathcal{V}_{h,l}|)$
20 Remove elements in $\mathcal{C}_{res}$ with lower probabilities;            $O(1)$
21 if $|\mathcal{C}_{res}| == 1$ then
22     Return vehicle location;                                          $O(1)$
23 else
24     $\mathcal{C}_n \leftarrow \mathcal{C}_{res}$                                         $O(1)$
25     Return $\{\mathcal{C}_n, Q\}$;                                         $O(1)$
```

are n observations for vehicle l and n' observations for vehicle l' at the rendezvous.

In the analysis, we denote $n_v = |\mathcal{V}_{h,l}|$ as the cardinality of $\mathcal{V}_{h,l}$ and n_b as the expected number of neighbors for each vertex. n_b depends on how many streets an intersection has. We describe map/trajectory property in a rudimentary way by assuming k_d levels of distinguishable discrete headings in $[0, 2\pi)$ and k_l levels of distinguishable discrete road lengths. Each vertex takes a heading value and length value with equal probabilities of $1/k_d$ and $1/k_l$ correspondingly. Generally speaking, we know $n_v \gg k_d \geq n_b$ and $n_v \gg k_l \geq n_b$ for most maps. We denote n_c the total observations of vehicle l combining of vehicle l' trajectory given event E. We have the following lemma.

Lemma 3: The joint conditional probability that M is the true matching sequence given Q matches M, M' is the true matching sequence given Q' matches M' with rendezvous is,

$$P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E) = \left[(1-\alpha)^2 k_d k_l\right]^{n+n'} \frac{1}{n_b^{n_c-1}} \frac{1}{n_v}. \quad (7)$$

Proof: Applying the Bayesian equation, we have

$$P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E) = \frac{P(\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E|\mathcal{C}, \mathcal{C}')P(\mathcal{C}, \mathcal{C}')}{P(\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E)} \quad (8)$$

Since $\mathcal{A}$, $\mathcal{B}$, $\mathcal{A}'$, $\mathcal{B}'$ are conditional independent to E given $\mathcal{C}$ and $\mathcal{C}'$, we have $P(\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E|\mathcal{C}, \mathcal{C}') = P(\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}'|\mathcal{C}, \mathcal{C}')P(E|\mathcal{C}, \mathcal{C}')$. Also, $\mathcal{A}$ and $\mathcal{B}$ are conditional independent to $\mathcal{A}'$ and $\mathcal{B}'$ given $\mathcal{C}$ and $\mathcal{C}'$ since each vehicle perform GBPL independently. Thus $P(\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}'|\mathcal{C}, \mathcal{C}') = P(\mathcal{A}, \mathcal{B}|\mathcal{C})P(\mathcal{A}', \mathcal{B}'|\mathcal{C}')$. We rewrite (8) by

$$\begin{aligned} &P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E) \\ &= \frac{P(\mathcal{A}, \mathcal{B}|\mathcal{C})P(\mathcal{A}', \mathcal{B}'|\mathcal{C}')P(E|\mathcal{C}, \mathcal{C}')P(\mathcal{C}, \mathcal{C}')}{P(\mathcal{A})P(\mathcal{B})P(\mathcal{A}')P(\mathcal{B}')P(E)} \\ &= \frac{P(\mathcal{A}, \mathcal{B}|\mathcal{C})}{P(\mathcal{A})P(\mathcal{B})} \frac{P(\mathcal{A}', \mathcal{B}'|\mathcal{C}')}{P(\mathcal{A}')P(\mathcal{B}')} P(\mathcal{C}, \mathcal{C}'|E) \end{aligned} \quad (9)$$

It is shown in [2] that the first two terms are

$$\frac{P(\mathcal{A}, \mathcal{B}|\mathcal{C})}{P(\mathcal{A})P(\mathcal{B})} = (1-\alpha)^{2n} \frac{1}{(k_d k_l)^n}, \quad (10)$$

$$\frac{P(\mathcal{A}', \mathcal{B}'|\mathcal{C}')}{P(\mathcal{A}')P(\mathcal{B}')} = (1-\alpha)^{2n'} \frac{1}{(k_d k_l)^{n'}}. \quad (11)$$

Joint conditional probability $P(\mathcal{C}, \mathcal{C}'|E)$ can be seen as two vehicle trajectories stitch at rendezvous and form a combined trajectory. We know $P(C_1) = 1/n_v$ given there are n_v possible solutions, and $P(C_2|C_1) = 1/n_b$ because there are n_b neighbors of C_1. And the combined trajectory has n_c unique observations where

$$\max(n, n') \leq n_c \leq (n + n'). \quad (12)$$

By induction,

$$P(\mathcal{C}, \mathcal{C}'|E) = \frac{1}{n_b^{n_c-1}} \frac{1}{n_v}. \quad (13)$$

Plugging (10)-(13) into (9), we obtain the lemma. ■

$P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E)$ reflects how fast the conditional probability increase as query graph grows when combining inputs from two vehicles. Its counterpart in single vehicle localization is $P(C|\mathcal{A}, \mathcal{B})$ from [2]. Comparing the two, we have the following theorem.

Theorem 1: The C-GBPL algorithm converges to actual localization is at least as fast as GBPL, because

$$P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E) \geq P(C|\mathcal{A}, \mathcal{B}) \quad (14)$$

Proof: For single vehicle, it is shown in [2] that the conditional probability that $M = \{\Theta, D\}$ is the true matching sequence given $Q = \{\Theta_q, D_q\}$ matches M is,

$$P(C|\mathcal{A}, \mathcal{B}) = [(1-\alpha)^2 k_d k_l]^n \frac{1}{n_b^{n-1}} \frac{1}{n_v}. \quad (15)$$

Comparing (7) with (15), we have

$$\frac{P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E)}{P(C|\mathcal{A}, \mathcal{B})} = \left[(1-\alpha)^2 (k_d k_l)\right]^{n'} n_b^{n_c-n} \geq 1, \quad (16)$$

Because $k_l > \frac{1}{1-\alpha}$ and $k_d > \frac{1}{1-\alpha}$ are generally true, $n_b > 1$, and $n_c - n \geq 0$ according to (12). ■

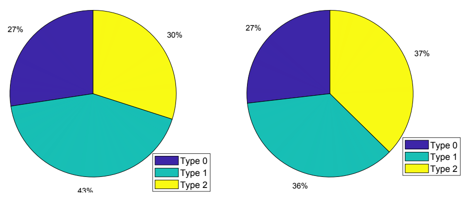
The minimum value of n_c happens when vehicle trajectories are the same. The maximum value of n_c happens when vehicles have non-overlapping trajectories. When $n' = 0$, we have $n_c = n$ and thus (16) has a ratio of 1. In such case, there is no gain on localization efficiency. However, for most cases, $P(\mathcal{C}, \mathcal{C}'|\mathcal{A}, \mathcal{B}, \mathcal{A}', \mathcal{B}', E)$ is much bigger than $P(C|\mathcal{A}, \mathcal{B})$ which results in significant increase in localization speed.

V. Experiments

We have implemented the proposed C-GBPL method using MATLAB and validated the algorithm in both simulation and physical experiments. The experiments are based on a map of College Station, Texas, U.S. which is 3.24 km^2 in area with 52.7 km roads. The map is from OSM and termed as CSMap. We pre-process it to the modified HLG with 1102 nodes. For comparison purpose, we evaluate its performance against GBPL [2], which is the single vehicle localization counterpart.

A. Simulation

1) Data Generation: We track the localization performance of the 25 seed vehicles as vehicles of interests and gradually increase other vehicles on the street from a total of 25 to 1000 vehicles on the map. It simulates sparse to moderately dense traffic. At 1000 vehicles in CSMap, it means that the mean car-space is around 53 meters. All simulation results are the statistics of the 25 seed vehicles.



(a) (b)

Fig. 7. Pie chart of rendezvous events. (a) Trajectory type: random walk. (b) Trajectory type: same-street.

To generate vehicles trajectories, each vehicle starts at random vertices at the same time. Vehicles take two different driving strategies: *random-walk* and *same-street*. In the random-walk strategy, the vehicle takes random turns at intersections. For the same-street strategy, the vehicle keeps driving on the same street. In reality, a vehicle's driving behavior is somewhere between the two extreme types. Fig. 7(a) and Fig. 7(b) show rendezvous event distribution for random walk and same-street for 1000 simulated vehicles, respectively. It is interesting that random walk generate more type 1 events (i.e. vehicles meet at the same street with different direction) while same street generates more type 2 events (vehicles meet at intersections).

2) Localization Comparison: To compare the two algorithms, we first compute the failure rate. A localization failure occurs when the algorithm fails to converge to a unique location. It may happen due vehicle trajectory or map itself. For example, if a city map consists of perfect square grid everywhere, our algorithm will fail. Fig. 8 shows the failure percentage for same-street strategy. We omit the failure percentage of random walk strategy because in this setting all vehicles can be localized. It is expected because random walk generates more unique query graphs. From Fig. 8, it is clear that C-GBPL utilizes the information from other vehicles and hence reduces failure rate to zero as traffic increases.

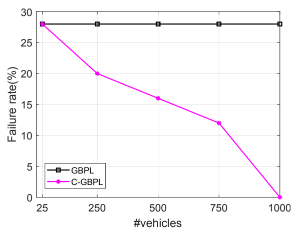
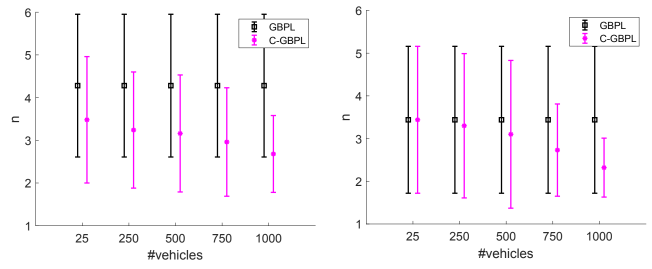


Fig. 8. Failure rate percentage of trajectory type: same-street.

When GBPL and C-GBPL algorithms successfully localize the vehicle, it is also important to compare how fast the localization process takes. we use n which is the number of straight segments in the query to measure *localization speed*, because it tells how many inputs it takes to localize the vehicle. For a given n, the algorithm may provide multiple solutions if there are many similar routes in the map. If the number of solutions is one, then the vehicle is uniquely localized. Fig. 9 illustrate n's mean and $\pm\sigma$ range for both algorithms under the two driving strategies where σ is the standard deviation of n. Again, C-GBPL outperforms GBPL with a smaller n. The mean value n for C-GBPL decreases as traffic increases while the mean value for n for GBPL remains unchanged.



(a) (b)

Fig. 9. Localization speed comparison. Each marker position is the mean, and each vertical segment is its corresponding $1\pm\sigma$ range. (a) Trajectory type: random walk. (b) Trajectory type: same-street.

B. Physical Experiments

We compare GBPL and C-GBPL in physical experiment and test on different rendezvous scenarios. We collect three sequences which correspond to the trajectories of three vehicles. We record IMU readings at 400Hz and compass readings at 50Hz using a Google Pixel™ phone. Also, we access vehicle speed readings at 46.6Hz in average use an panda OBD-II Interface which provide velocity feedback from wheel encoder. To test C-GBPL, we vary rendezvous time of these three vehicles and test on six rendezvous scenarios as shown in Fig. 10. We use #sols (number of solutions) for a given n to measure algorithm efficiency. For a given n, if the algorithm has fewer #sols than the other algorithm, the better in efficiency.

Tab. I shows the comparison between GBPL and C-GBPL in both localization efficiency and speed in terms of #*sols* and n accordingly. In summary, in all tests C-GBPL has better efficiency and with a speedup factor of 1.6x on average.

TABLE I
LOCALIZATION SPEED AND EFFICIENCY.

			GBPL		C-GBPL	
Fig. 10 (Case)	Vehicle (Ego, the other)	CRSV (Type)	Speed (n)	Efficiency (#sols)	Speed (n)	Efficiency (#sols)
(a)	(car3, car1)	Type 2	5	122	**1**	**1**
(b)	(car2, car1)	Type 0	6	5	**5**	**1**
(c)	(car2, car1)	Type 2	6	3	**4**	**1**
(d)	(car1, car2)	Type 2	5	9	**3**	**1**
(e)	(car1, car2)	Type 0	5	9	**4**	**1**
(f)	(car2, car1)	Type 0	6	3	**4**	**1**

VI. Conclusion and Future Work

To assist vehicles in extreme weather conditions, we developed the C-GBPL method that did not rely on the perception and recognition of external landmarks to localize robots/vehicles in urban environments. The method was a multiple robot/vehicle collaborative localization scheme using V2V communication which combines features from rendezvous vehicles to accelerate the mapping process. We identified different rendezvous events to form the merged query graph. We performed graph-to-graph matching by aggregating

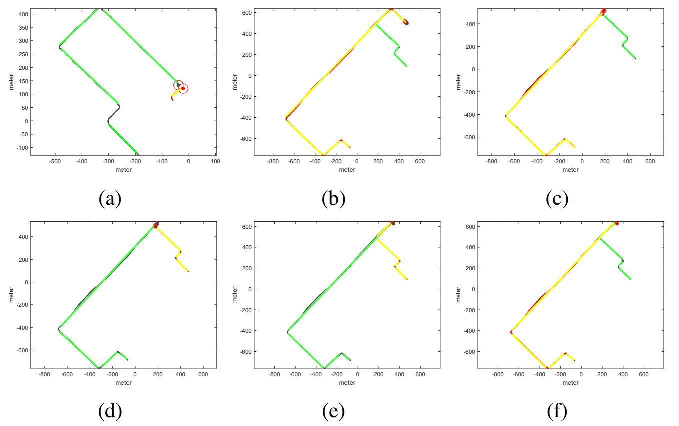


Fig. 10. Rendezvous of three vehicles under different scenarios. We show the ego vehicle trajectory in red and mark straight segments in translucent yellow. The other vehicle trajectory is shown in gray and straight segments in translucent green.

vehicle prior beliefs and trim candidate vertex. We proved that the collaborative localization strategy is faster than its single vehicle counterpart in general cases. The algorithm was tested in both simulation and physical experiments and showed superior performance over the single vehicle counterpart.

In the future, we will work on alternative vehicle trajectory strategies to analyze collaborative localization success rate. We will also consider using maps with altitude information and develop new algorithms. We will apply the algorithm online with multiple vehicles. We will consider extracting relative constellation information and sharing it among the vehicles. We will report new results as they emerge.

Acknowledgment

The authors would like to thank B. Li, S. Yeh, A. Kingery, A. Angert, D. Wang, and S. Xie for their contributions to the NetBot Lab at Texas A&M University.

References

[1] H. Cheng, D. Song, A. Angert, B. Li, and J. Yi, "Proprioceptive localization assisted by magnetoreception: A minimalist intermittent heading-based approach," *IEEE Robotics and Automation Letters*, 2018.

[2] H. Cheng and D. Song, "Graph-based proprioceptive localization using a discrete heading-length feature sequence matching approach," *IEEE Transactions on Robotics (accepted, to appear)*, 2020. [Online]. Available: http://arxiv.org/abs/2005.13704

[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, April 2016.

[4] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[5] Y. Lu and D. Song, "Visual navigation using heterogeneous landmarks and unsupervised geometric constraints," in *IEEE Transactions on Robotics (T-RO)*, vol. 31, no. 3, June 2015, pp. 736–749.

[6] J. Levinson and S. Thrun, "Robust vehicle localization in urban environments using probabilistic maps," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4372–4378.

[7] T. Hunter, P. Abbeel, and A. Bayen, "The path inference filter: model-based low-latency map matching of probe vehicle data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 507–529, 2014.

[8] Y. Cui and S. S. Ge, "Autonomous vehicle positioning with gps in urban canyon environments," *IEEE transactions on robotics and automation*, vol. 19, no. 1, pp. 15–25, 2003.

[9] H. Aly, A. Basalamah, and M. Youssef, "Accurate and energy-efficient gps-less outdoor localization," *ACM Trans. Spatial Algorithms Syst.*, vol. 3, no. 2, pp. 4:1–4:31, Jul. 2017.

[10] C. Chou, A. Kingery, D. Wang, H. Li, and D. Song, "Encoder-camera-ground penetrating radar tri-sensor mapping for surface and subsurface transportation infrastructure inspection," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1452–1457.

[11] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual–inertial odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[12] P. Merriaux, Y. Dupuis, P. Vasseur, and X. Savatier, "Fast and robust vehicle positioning on graph-based representation of drivable maps," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2787–2793.

[13] P. Ruchti, B. Steder, M. Ruhnke, and W. Burgard, "Localization on openstreetmap data using a 3d laser scanner," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5260–5265.

[14] R. Jiang, S. Yang, S. S. Ge, H. Wang, and T. H. Lee, "Geometric map-assisted localization for mobile robots based on uniform-gaussian distribution," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 789–795, 2017.

[15] Y. Jin and Z. Xiang, "Robust localization via turning point filtering with road map," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*. IEEE, 2016, pp. 992–997.

[16] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[17] Google Maps contributors, https://www.google.com/maps/ , 2017.

[18] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org , 2017.

[19] M. Quddus and S. Washington, "Shortest path and vehicle trajectory aided map-matching for low frequency gps data," *Transportation Research Part C: Emerging Technologies*, vol. 55, pp. 328–339, 2015.

[20] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2564–2571.

[21] S. I. Roumeliotis and G. A. Bekey, "Distributed multirobot localization," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, Oct 2002.

[22] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Autonomous robots*, vol. 8, no. 3, pp. 325–344, 2000.

[23] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, 2013.

[24] C. Kim, D. Song, Y. Xu, J. Yi, and X. Wu, "Cooperative search of multiple unknown transient radio sources using multiple paired mobile robots," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1161–1173, Oct. 2014.

[25] K. Y. Leung, T. D. Barfoot, and H. H. Liu, "Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 62–77, 2009.

[26] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, "Recursive decentralized collaborative localization for sparsely communicating robots," in *Proceedings of Robotics: Science and Systems*, AnnArbor, Michigan, June 2016.

[27] J. Zhu and S. S. Kia, "Cooperative localization under limited connectivity," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1523–1530, 2019.

[28] H. Li and F. Nashashibi, "Cooperative multi-vehicle localization using split covariance intersection filter," *IEEE Intelligent Transportation Systems Magazine*, vol. 5, no. 2, pp. 33–44, 2013.

[29] E. D. Nerurkar and S. I. Roumeliotis, "Asynchronous multi-centralized cooperative localization," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 4352–4359.

[30] J. Wahlström, I. Skog, J. G. P. Rodrigues, P. Händel, and A. Aguiar, "Map-aided dead-reckoning using only measurements of speed," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 3, pp. 244–253, Sep. 2016.

[31] B. Yu, L. Dong, D. Xue, H. Zhu, X. Geng, R. Huang, and J. Wang, "A hybrid dead reckoning error correction scheme based on extended kalman filter and map matching for vehicle self-localization," *Journal of Intelligent Transportation Systems*, vol. 23, no. 1, pp. 84–98, 2019.

[32] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.