IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Geo-localization based on Dynamically Weighted Factor-graph

Miguel Ángel Muñoz-Bañón, Alejandro Olivas, Edison Velasco-Sánchez,
Francisco A. Candelas, and Fernando Torres

***Abstract*— Feature-based geo-localization relies on associating features extracted from aerial imagery with those detected by the vehicle's sensors. This requires that the type of landmarks must be observable from both sources. This lack of variety of feature types generates poor representations that lead to outliers and deviations produced by ambiguities and lack of detections, respectively. To mitigate these drawbacks, in this paper, we present a dynamically weighted factor graph model for the vehicle's trajectory estimation. The weight adjustment in this implementation depends on information quantification in the detections performed using a LiDAR sensor. Also, a prior (GNSS-based) error estimation is included in the model. Then, when the representation becomes ambiguous or sparse, the weights are dynamically adjusted to rely on the corrected prior trajectory, mitigating outliers and deviations in this way. We compare our method against state-of-the-art geo-localization ones in a challenging and ambiguous environment, where we also cause detection losses. We demonstrate mitigation of the mentioned drawbacks where the other methods fail.**

I. INTRODUCTION

Autonomous navigation is a significant research topic because it can automate complex tasks using mobile robots, Unmanned Ground Vehicles (UGV), or self-driving cars. Navigating through an environment autonomously relies strongly on the localization module. The more extended approach for this purpose is *Simultaneous Localization And Mapping* (SLAM) [1], where the vehicle navigates building a model of the environment (the map) while simultaneously using it for self-localization. Alternatively, to simplify the localization, the mapping process could be avoided using an environment representation previously created by dedicated mapping vehicles [2]. However, creating a map is usually expensive, especially for oversized areas. Moreover, it requires several loop closures for consistency, but despite this, a mapping process often accumulates minor errors that lead to global inconsistencies.

Over the last few years, the so-called geo-localization or geo-referencing has increased in importance in the literature. For the localization, this approach uses an environment representation obtained from aerial imagery. This avoids the expensive mapping process and the need for loop closures and provides implicit global consistency. We can distinguish two different strategies to perform geo-localization: *end-to-end learned* [3] and *handcrafted-feature-based* [4].

The *end-to-end learned* strategy uses the raw aerial image as an environmental representation while perceiving it with local sensors such as LiDAR [5], RADAR [3], or cameras [6]. Then, it uses end-to-end learned models to extract dense features from both data sources to infer the vehicle's pose in the geo-referenced aerial image. In [7], [8], [9], the authors use a wide aerial image as a representation and infer the pose by crossing the sensor's information directly extracting the dense features from the whole image. In contrast, in [5], [10], [11], the authors presented a pipeline that gets a crop around the prior pose to perform the end-to-end strategy.

The *handcrafted-feature-based* geo-localization uses aerial imagery to extract handcrafted sparse features, while the same type of features should be detected from the sensors' data. Then, after data association, the vehicle's pose is estimated. This strategy implies a requirement: the type of feature used must be observable from both aerial and onboard vehicle sensors. In [12], [13], [14], [15], [16], building walls are used as features, while in [4], [17], [18], the authors choose lane marking as landmarks that satisfy the observability requirement. In other works [19], [20], [21], the authors match the vehicle's trajectory with the lanes map, which is commonly named in the literature as map matching [22]. However, we prefer to categorize it as a *handcrafted* where the feature is the trajectory. The *handcrafted-feature-based* strategy allows the measure of information in the detections before the pose inference. For example, in [4], the authors estimate prior confidence in the data and use it to self-tune the data association method depending on that confidence. For this work, we introduce the ground boundaries as a new feature type observable from both sources.

The mentioned observability requirement carries fewer features, leading to a sparse representation. This issue and the geo-referenced nature of the problem generate some drawbacks in the *handcrafted-feature-based* strategy: (i) A sparse map implies that some areas are ambiguous for the data association in the front direction of navigation, which can produce a considerable number of outliers. (ii) The poor variety of features introduces the risk of lack of detection in some navigation areas. (iii) Geo-localization needs a geo-referenced prior, such as GNSS (Global Navigation Satellite Systems). Those systems are usually precise but inaccurate, introducing offsets that vary smoothly through time and space, especially when there is a multipath problem [23]. In a previous work [4], we addressed the drawback (i). But with this approach, if we find a situation of type (ii),

This work has been supported by the regional Valencian Community Government and the European Union through the project PROMETEO/2021/075, as well as by the Spanish government through the grants PRE2019-088069 and PRE2022-101680 and the project PID2021-122685OB-I00.

Authors are with the Group of Automation, Robotics and Computer Vision (AUROVA), University of Alicante, San Vicente del Raspeig S/N, Alicante, Spain. miguelangel.munoz@ua.es

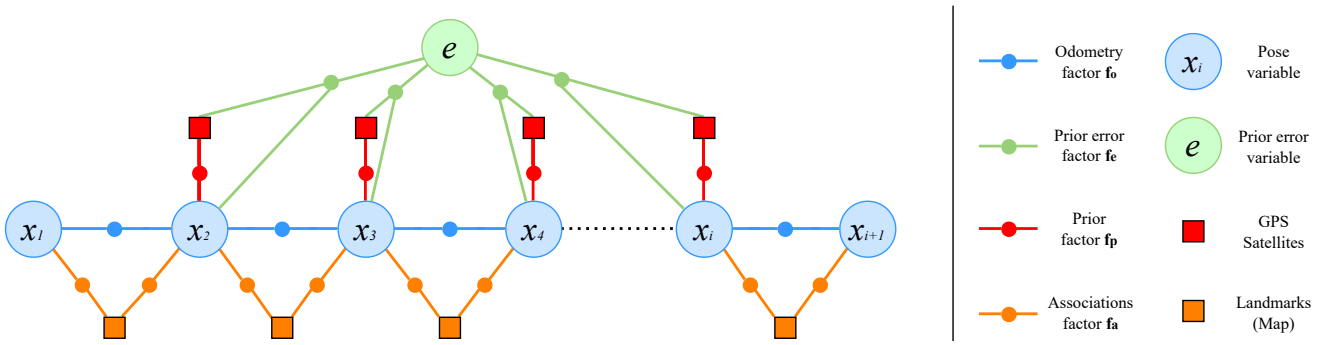

Fig. 1. **Proposed factor graph**: Each factor represents the difference between the observations and the model predictions. Section II explains the formalization of the (weighted) factors to obtain the residuals. The empty circles represent the variables that should be estimated by factor residual minimization. Finally, we denote by squares the elements that produce exteroceptive observations, i.e., GPS satellites and landmarks read from the map.

the localization converges to a prior trajectory that usually has an undesired offset, as we mentioned in (iii). Notably, the cited *handcrafted-feature-based* geo-localization works usually don't pay special attention to avoid these drawbacks, so we consider it interesting to focus our research here.

This paper presents a *handcrafted-feature-based* geo-localization that mitigates the exposed drawbacks through a dynamically weighted factor graph implementation in the vehicle's trajectory estimation. The weight adjustment in this model depends on information quantification in the detections performed using a LiDAR sensor. Furthermore, a prior (GNSS-based) error estimation is included in the model to avoid the drawback (iii). In this way, if, for example, we drive through an area ambiguous for data association, the weight is adjusted to trust more on the corrected prior trajectory, mitigating the drawback (i). In the case of the lack of detections (ii), again, the system will rely more on corrected prior.

In summary, our contributions are the following:

- A weighted factor graph that dynamically adjusts its weights depending on the information quantified from the detections. That produces a mitigation of the drawbacks (i), (ii), and (iii) previously mentioned.
- An information quantification strategy developed upon a previous one [4], [24] that quantifies the information based on the associated map points instead of the raw detections. This quantification is the primary measure to adjust factor weights.
- A prior (GNSS-based) error estimation included in the model. With this corrected prior, we can hold the localization for low informative detections.

II. WEIGHTED FACTOR-GRAPH

In this section, we formalize the proposed factor-graph model (Fig. 1), where each factor is dynamically weighted depending on the information in the data (Section III-A). In this way, the more confident residuals will contribute mainly to the loss function in the optimization process, giving less importance to those who can generate unwanted minimums, for example, when outliers occur or when there is a lack of landmarks.

The factors explained in Sections II-A, II-B, and II-C are the most commonly implemented and the ones that generate the residuals to estimate the trajectory state variable $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_N)$, where each pose is $\mathbf{x}_i \doteq (\mathbf{R}_i, \mathbf{t}_i)$, $\mathbf{t}_i \in \mathbb{R}^2$ is the translation, and $\mathbf{R}_i \in SO(2)$ is the rotation matrix.

In contrast, in this work, we propose an additional factor explained in Section II-D that generates the residuals to estimate the error $\mathbf{e} = (e_x, e_y)$ in the prior signal. The estimation of this variable allows corrections to the GPS observations, thus contributing to maintaining localization in outlier and lack regimes and avoiding undesired GPS errors in the loss function.

It is worth noting that this section explains the high-level formalization of the model (the *back-end*). In contrast, in the next section (Section III-A), we present the low-level (the *front-end*) with more details, e.g., data association implementation, type of landmark, detections, etc.

A. Odometry factor

We assume we have an odometry system estimated from the LiDAR, cameras, IMU, and/or encoders. Then, given the relative transformations from consecutive frames i and $i' = i - 1$ from the odometry trajectory $\hat{\mathbf{X}} = (\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_N)$, and the poses in the estimated $\mathbf{X}$, we can define the odometry factors as follows:

$$f_i^o = \omega_i^o \left\| \mathbf{R}_i^\mathsf{T} \left(\mathbf{t}_{i'} - \mathbf{t}_i \right) - \hat{\mathbf{t}}_{i,i'} \right\|_2^2 + \omega_i^o \left\| \mathbf{R}_i^\mathsf{T} \mathbf{R}_{i'} - \hat{\mathbf{R}}_{i,i'} \right\|_F^2 . \quad (1)$$

As we can see in (1), each norm is weighed by ω_i^o. The value of that weight is dynamically obtained in each iteration depending on the quantification of the data information explained in Section III. When the data is considered poorly informative, this weight will acquire strength. The subscript F in the second term indicates the Frobenius norm.

B. Prior factor

For geo-localization, it is essential to have a geo-referenced prior localization based on GNSS, concretely, in our case, GPS. Such systems usually provide position information $\mathbf{t}_j^g = (x_j^g, y_j^g)$ as j-th observation. Given the time stamp availability, it is easy to obtain the association

between GPS j-th observation and i-th odometry estimation. Then, we define the prior residuals as:

$$f_j^p = \omega_j^p \left\| \mathbf{t}_i - \left(\mathbf{t}_j^g - \mathbf{e} \right) \right\|_2^2, \tag{2}$$

where $\mathbf{e}$ is the estimated error that corrects the GPS observation, and the weight ω_j^p is calculated similarly to ω_i^o (Section III). With odometry and prior residuals, obtaining a so-called prior trajectory is possible. This trajectory provides satisfactory results in differential terms. But, due to the inaccuracy problems of GPS systems, the path usually has an offset that varies smoothly over time and space. We aim to correct this with the $\mathbf{e}$ estimation. However, to estimate $\mathbf{e}$, it is necessary to use a geo-referenced map to associate its information with local detections, obtaining a trusted localization, as explained in the following subsection. After fine $\mathbf{e}$ estimation, it is reasonable that when the data is considered poorly informative, ω_j^p and ω_i^o will acquire strength, and the final estimation will maintain their localization trusting on that corrected prior trajectory.

C. Data associations factor

As mentioned above, we have a geo-referenced world's representation, defined as a set of landmark $\mathcal{L}$. Then, for each i-th frame in $\hat{\mathbf{X}}$, we observe the landmarks of the environment using the vehicle's sensors. From now on, we name these observations as detections $\mathcal{D}_i$. Using $\mathcal{L}$ and $\mathcal{D}_i$, we must perform a data association process, where its result is a set of pairs $((\mathbf{d}_{i_1}, \mathbf{l}_{i_1}), ..., (\mathbf{d}_{i_K}, \mathbf{l}_{i_K}))$. Given these associations, we can define the residuals between landmarks and detections as follows:

$$f_i^a = \omega_i^a \sum_{k=1}^{K} \left\| (\mathbf{R}_i \mathbf{d}_{i_k} + \mathbf{t}_i) - \mathbf{l}_{i_k} \right\|_2^2. \tag{3}$$

As shown in (3), the residual depends on the pose that transforms the detection from the local sensor to the map coordinates frame. In this case, in contrast to the odometry and prior residuals, the weight ω_i^a is strongest when the data is more informative.

D. Prior error factor

In Section II-B, we mentioned that the prior trajectory presents variable offset produced by GPS inaccuracies. Then, when we measure less-informative data, and consequently, the prior path gets strength in the optimization, the final localization could carry the mentioned inaccuracies. To avoid this effect, we estimate the prior error $\mathbf{e}$ to correct the GPS observation in (2). The factor for the error estimation is the following:

$$f_j^e = \sum_{j'=j-w}^{j'=j} \omega_{j'}^e \left\| \mathbf{e} - \left(\mathbf{t}_{j'}^g - \bar{\mathbf{t}}_{i'} \right) \right\|_2^2. \tag{4}$$

GPS error varies smoothly over time, so we estimate that variable using factors from limited past poses, e.g., from $j' = j - w$ to $j' = j$, being i' the i-th position associated with j'. The notation of $\bar{\mathbf{t}}_{i'}$ means that is the position of the state estimated, but in this case, it is used as observation instead of as a variable.

E. Optimization

The sum of all exposed factors is the cost function. Thus, the optimal state $\mathbf{X}^*, \mathbf{e}^*$ is such that it minimizes the said cost:

$$\mathbf{X}^*, \mathbf{e}^* = \arg\min_{\mathbf{X}, \mathbf{e}} \left(\sum_{i}^{N} \left(f_i^o + f_i^a \right) + \sum_{j}^{M} \left(f_j^p + f_j^e \right) \right), \tag{5}$$

where M is the number of GPS observations, and N is the number of odometry observations (that coincide with estimations). The weights explained in this section directly affect the cost function form, allowing us to avoid the problems exposed in Section I: (i) when the data information is insufficient with ambiguities risk, the system can prevent outliers by holding the trajectory taking strength on the corrected prior trajectory. (ii) the same occurs when we have a lack of detections. (iii) we avoid little deviations in the final estimation when correcting GPS inaccuracies.

The dynamic adjustment of these weights depends directly on the information in the data. The following section describes the information calculation and the consequent weight adjustment.

III. DYNAMIC WEIGHT ADJUSTMENT

While in the previous section, we focused our explanation on the factor graph model, this section aims to expose the data information quantification and the adjustment of the weights for that model. In Section III-A, we describe how to obtain the data information quantification. For that, we need to talk previously about the landmarks and detection obtention and the data association process. In section III-B, we specify how to adjust the weights as a function of the data information.

A. Data information quantification

In geo-localization, the type of landmark chosen for localization must be observable from aerial imagery and local sensors. The literature usually includes lane markings, vertical structures such as building walls, and even the vehicle's trajectory. In this work, we introduce another feature that satisfies the mentioned requirement: ground boundaries. This type of landmark is suitable for roads, city streets, and pedestrian areas such as university campuses.

In the following points, we describe how to obtain those detections $\mathcal{D}_i$ from a LiDAR sensor, get the map as a set of landmarks $\mathcal{L}$, and quantify the information after the data association.

1) Detections: A LiDAR sensor usually provides a point cloud with 3D environment information and the reflectivity for each point. With this information, we mount a front-view representation in RGB image format. In this image, the GB channels contain reflectivity information, while in the R

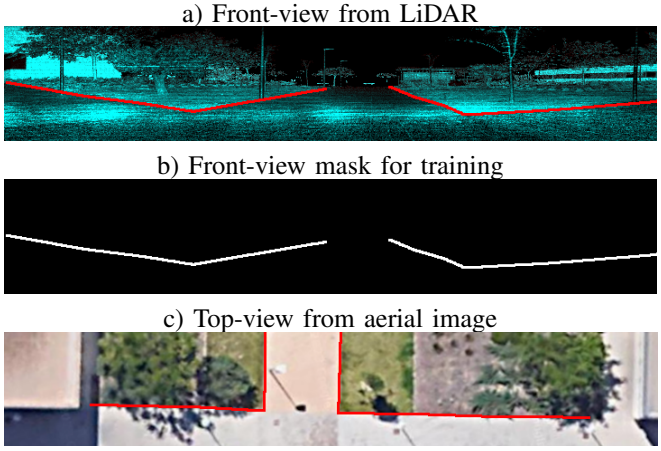

Fig. 2. **Example of $\mathcal{D}_i$ and $\mathcal{L}_i$**: a) Front-view from LiDAR information where the red line marks the ground boundaries ($\mathcal{D}_i$). b) Front-view mask extracted from (a). c) Top-view from the aerial image where the red line represents the exact boundaries shown in (a) and (b) ($\mathcal{L}_i$).

channel, we include the range image. In Fig. 2 a), we show an example of a LiDAR scan as a front-view representation.

Then, we use the Convolutional Neural Network (CNN) Unet++ [25] with backbone resnet18 [26] for ground boundary detections. To train the model, we generated our own dataset of 825 images and labeled them by hand, obtaining the ground truth masks (Fig. 2 b)). We divided our dataset into 80% for training and 20% for testing with non-overlapping. Regarding generalization, the mentioned dataset was recorded in the Scientific Park of the University of Alicante, and we observed satisfactory results on the university campus, which is a different environment with more vegetation and a different pavement. In contrast, we needed to label new images in our experiments in KITTI, where the environment is totally different[1]. After training, during the detection, each u-th pixel with value 1 has its correspondence as a 3D point projected in 2D, being a detection $\mathbf{d}_{i_u} \in \mathcal{D}_i$, where $\mathbf{d}_{i_u} = (x^d_{i_u}, y^d_{i_u})$.

In a previous work [4], we used the polylines that describe detections to quantify the information using the angle between adjacent segments in the polyline. In this case, we observe that the boundaries in detections $\mathcal{D}_i$ are not arranged sequentially. We could process the data to obtain the polylines, which implies an undesired complex process in computational time terms. Thus, we opted to quantify the data using the structuration of the landmarks $\mathcal{L}$ associated with detections instead of directly using detections.

2) Landmarks: The landmarks $\mathcal{L}$ that form the map could also be detected with neural networks. Still, we use a handcrafted map generated by applications such as OpenStreetMaps [27] to avoid post-processing. This implies that the map comprises a polyline set arranged in a friendly way to quantify the data using differential angles between adjacent segments in the polylines [24]. In Fig. 2 c), we show labeled ground boundaries in an aerial image. These boundaries are the same labeled in front-view representations (Fig. 2 a),b)).

Before i-th data association, we have detections set $\mathcal{D}_i$; then we must crop $\mathcal{L}$ around $\mathbf{x}_i$, obtaining $\mathcal{L}_i$. The v-th landmark $\mathbf{l}_{i_v} \in \mathcal{L}_i$ is defined as $\mathbf{l}_{i_v} = (x^l_{i_v}, y^l_{i_v}, \alpha_{i_v})$, where α_{i_v} is the differential angle between adjacent polyline segments.

Quantifying the information using $\mathcal{L}_i$ is unsuitable because it could contain landmarks not observed by the sensors, adding undesired information to quantification. In the next point, we describe quantifying the information after the data association process.

3) Raw quantification from data association: Given $\mathcal{D}_i$ and $\mathcal{L}_i$, we perform the data association process. In this work, we use an ICP (*Iterative Closest Point*) due to its efficient implementation in PCL (*Point Cloud Library*). First, we co-register $\mathcal{D}_i$ with $\mathcal{L}_i$, obtaining a new transformed set $\mathcal{D}'_i$. Finally, we find the closest point in $\mathcal{L}_i$ below a certain distance threshold for each $\mathbf{d}'_{i_u} \in \mathcal{D}'_i$. This process generates a set of associated pairs $((\mathbf{d}_{i_1}, \mathbf{l}_{i_1}), ..., (\mathbf{d}_{i_K}, \mathbf{l}_{i_K}))$ used in (3). Then, to quantify the information of the data rawly, we sum the values of the delta angle in the associated landmarks as:

$$s_i = \sum_{k=1}^{K} \alpha_{i_k}. \tag{6}$$

This raw quantification is used to adjust the weights, as explained in the next section.

It is worth noting that this quantification based on polyline map representation is suitable for the rest of the feature types used in the literature, i.e., lane markings, building walls, or trajectories.

B. Weights as a function of data information

As shown in (6), the raw information quantification is an accumulative value, where the minimum is $s_i^{min} = 0$, and its maximum depends on the environment, where our experiments observe a maximum $s_i^{max} \approx 60$. To obtain the s_i^{max} value in a different area, we must drive only the part with more landmark density and calculate the maximum s value. This result is intractable to weight the factor graph as we proposed in Section II. For this reason, to adjust the data association weight, we use a sigmoid function to restrict the quantification in a range $[0; 1]$:

$$\omega_i^a(s_i) = \omega_j^e(s_i) = \frac{1}{1 + e^{-\Phi_i(s_i)}}. \tag{7}$$

As we can see in (7), given j and i synchronization, we can adjust the prior error weight $\omega_j^e(s_i)$ as that $\omega_i^a(s_i)$. The form of the sigmoid depends on the function $\Phi_i(s_i)$, and we propose two different definitions compared in the evaluation section.

First, as an option (a), if we consider the range of information quantification as $s_i = [0; s_i^{max}]$ and if we take

[1]The generalization of the complete localization method depends on that detection module but is independent of the contribution of this work. Roughly speaking, the detection module is like a black box in our approach. If we require more generalization, we would need to research a more sophisticated black box, which is out of the scope of this paper.

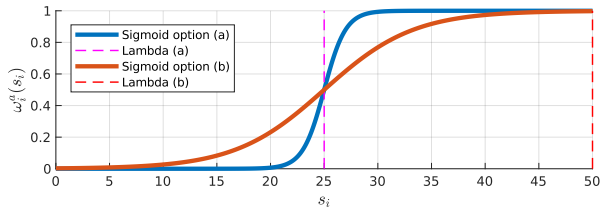

Fig. 3. Example of $\Phi_i^{(a)}(s_i)$ and $\Phi_i^{(b)}(s_i)$ application in a sigmoid function. The dotted lines indicate the values for $\lambda^{(a)}$ and $\lambda^{(b)}$.

into account that a sigmoid function changes around zero in the range of $[-6; 6]$, we can displace the zero in s_i so that $\Phi_i(s_i)$ remains as:

$$\Phi_i^{(a)}(s_i) = \sum_{k=1}^{K} \alpha_{i_k} - \lambda^{(a)}, \tag{8}$$

where $\lambda^{(a)}$ is a configurable parameter. In Fig. 3, we show an example of this sigmoid configuration, where the pink dotted line marks the value of $\lambda^{(a)}$. This configuration provides a function that we can see as a smoothed step function where the parameter $\lambda^{(a)}$ tunes how restrictive the system is against information in the data.

Second, as an option (b), we propose a smoothest function. In this case, we transform the sigmoid range $[-6; 6]$, where its size is h, to an information quantification range $\left[0; \lambda^{(b)}\right]$, where $\lambda^{(b)}$ is close to s_i^{max}:

$$\Phi_i^{(b)}(s_i) = \frac{h}{\lambda^{(b)}} \sum_{k=1}^{K} \alpha_{i_k} - \frac{h}{2}. \tag{9}$$

In Fig. 3, we show an example of this second option, where the red dotted line marks the value of $\lambda^{(b)}$.

Finally, we define the expressions to adjust the prior trajectory weights. First, for the odometry weight as:

$$\omega_i^o(s_i) = (K_i + 1)\left(2 - \omega_i^a(s_i)\right). \tag{10}$$

And second, its variant for the prior as:

$$\omega_j^p(s_i) = \frac{(K_i + 1)\left(2 - \omega_i^a(s_i)\right)}{\left(\sigma_j^{x,y} + 1\right)}, \tag{11}$$

where K_i is the number of associations, this first term scales the prior trajectory weights to provide the same strength as the associations' residuals. We can see in (10) that the second term provides strength when the data information is poor. In (11), $\sigma_j^{x,y}$ is the variance in x, y plane for the GPS observations. Thus, noisily observations reduce the strength of the weight. The $+1$ regularizations in the three terms are to avoid zeros.

IV. EVALUATION

This article argues that our contributions mitigate some undesired effects in geo-localization approaches. To demonstrate it, we focus our evaluation on that way. Before that, we evaluate whole trajectories in a general way, comparing

Fig. 4. Aerial image of the University of Alicante Scientific Park, where the evaluation was performed through circuits in Fig. 5.

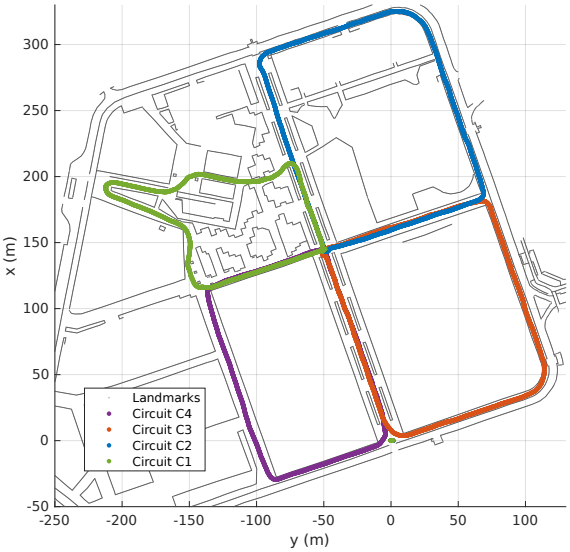

Fig. 5. The ground truth of the four circuits drove around the UA Scientific Park. The landmarks were obtained from the aerial image in Fig. 4.

different configurations of our method and three state-of-the-art *handcrafted-feature-based* methods (Section IV-B). Due to our approach being in this category, we consider it as the best baseline for comparison. From there, we discuss the mitigation of the mentioned problems: (i) Section IV-C shows how our approach mitigates the outliers produced by ambiguities. (ii) Section IV-D evaluates the mitigation of lack of detections. (iii) Section IV-E discusses how our GPS error estimation can improve the results in a whole trajectory. Finally, we provide a comparison in KITTI for two methods *end-to-end learned*.

A. Setup

The evaluation was performed in the University of Alicante (UA) Scientific Park (Fig. 4). This pedestrian area is

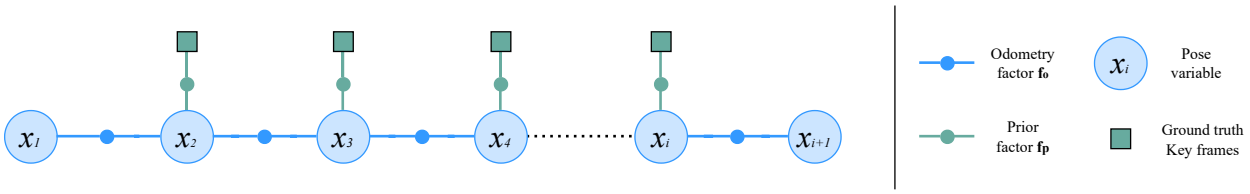

Fig. 6. **Factor graph for ground truth generation**: We use low-bias odometry as in Fig. 1, while we use handcrafted positions as trusted keyframes generating residuals by prior factors. We obtain a whole ground truth trajectory by optimizing this model in offline mode.

TABLE I

STATE-OF-THE-ARTS METHODS COMPARED FOR THE EVALUATION.

Methods	**Data source**	**Map source**	**Feature type**	**Model used**	**Evaluated in**	**GPS**
Frosi *et al.* [12]	LiDAR	OpenStreetMap	Buildings	Factor graph	KITTI	No
Own implementation [12]	LiDAR	OpenStreetMap	Ground boundaries	Factor graph	KITTI/Own	Yes
Cho *et al.* [13]	LiDAR	OpenStreetMap	Buildings	None	KITTI	No
Own implementation [13]	LiDAR	OpenStreetMap	Ground boundaries	Pose graph	KITTI/Own	Yes
Muñoz-Bañón *et al.* [4]	LiDAR and Cameras	OpenStreetMap	Lane markings	Factor graph	Their Own	Yes
Own implementation [4]	LiDAR	OpenStreetMap	Ground boundaries	Factor graph	Own	Yes
Ours	LiDAR	OpenStreetMap	Ground boundaries	Factor graph	KITTI/Own	Yes

TABLE II

WHOLE TRAJECTORIES EVALUATION BY ABSOLUTE TRAJECTORY ERROR (ATE) IN TRANSLATION AND ROTATION.

Session	**ATE**	**Ours** $\Phi^{(a)}$ **+ e**	**Ours** $\Phi^{(a)}$ **- e**	**Ours** $\Phi^{(b)}$ **+ e**	**Ours** $\Phi^{(b)}$ **- e**	**Prior**	**Muñoz-Bañón** *et al.* [4]	**Frosi** *et al.* [12]	**Cho** *et al.* [13]
C1	*trans.* (m)	0.147	0.493	0.165	0.552	2.032	**0.098**	0.280	0.416
	rot. (deg)	1.138	1.415	1.144	1.372	1.592	**1.002**	1.358	1.453
C2	*trans.* (m)	**0.225**	1.102	0.272	1.167	1.790	0.312	8.967	-
	rot. (deg)	0.876	1.053	**0.858**	1.140	0.916	0.989	4.896	-
C3	*trans.* (m)	**0.113**	0.904	0.184	0.933	1.211	0.152	0.949	1.244
	rot. (deg)	0.789	0.935	0.829	0.894	**0.734**	0.887	1.107	1.766
C4	*trans.* (m)	**0.096**	0.604	0.107	0.689	1.304	0.113	0.748	0.690
	rot. (deg)	**0.644**	0.725	0.653	0.744	0.764	0.892	0.885	0.856
C1'	*trans.* (m)	**0.230**	0.726	0.255	0.806	-	0.671	0.798	0.757
	rot. (deg)	**1.033**	1.772	1.202	1.679	-	1.283	1.750	1.913
C4'	*trans.* (m)	**0.113**	0.994	0.288	0.774	-	0.774	1.257	1.115
	rot. (deg)	**0.895**	1.140	0.937	1.137	-	1.104	0.996	1.093

where we drove through the four circuits shown in Fig. 5: C1, C2, C3, and C4. These circuits present some areas where the data is ambiguous for the data association being areas with outliers risk, especially C2, and C3. These parts are then adequate to evaluate (i). Moreover, to assess the effect (ii), we repeat two trajectories but eliminate the detections in some navigation parts. We name these repetitions C1' and C4'. Then, we consider that we have six paths for evaluation.

Regarding the mentioned ambiguity risk, we observe different challenge levels in these circuits. C1 has less risk due to passing areas with more corners, as we can see in Fig. 5. C4 presents some straight regions, but others are informative. Finally, we consider C2 and C3 the more challenging because passes through large straight areas, especially C2.

We drove these circuits using our own developed UGV platform BLUE (*roBot for Localization in Unstructured Environments*) [23], [28], which mounts a LiDAR Ouster OS1-128 for environment perception.

To obtain ground truth, we manually align the detections for some trajectory frames with the map, producing, in this way, some ground truth poses. We name these corrected poses as ground truth keyframes. Then, as we have low-bias LiDAR odometry [29], we use it to interpolate the ground truth poses to whole trajectories. We perform this interpolation by optimizing the entire circuit using the odometry factors defined in (1) and the prior factors described in (2), but in the last case, using the ground truth positions. In Fig. 6, we show the graph model for the ground truth generation.

In Table I, we show the specifications of the state-of-the-art methods used for comparison. It is worth noting that, for a fair comparison, we implemented in C++ the methods cited as described in their papers but adapted to our implementation. e.g., by using GPS and ground boundary features. For [12], we included the GPS factor in their own factor graph model. In the case of [13], we generated a pose graph including the poses calculated using the descriptor presented in [13], the odometry, and the GPS factors. This GPS augmentation doesn't contradict the contributions of the papers because in [12], the authors comment that GPS is an "optional" signal in their method, while in [13], the authors don't aim to replace GPS; they seek to replace LiDAR maps. Table I specifies the differences indicating our implementation. We use the Absolute Trajectory Error (ATE) metric for these comparisons.

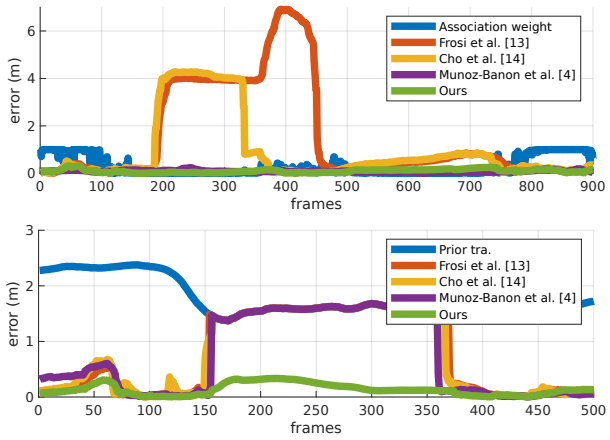

Fig. 7. A comparison of **ATE evolution per frame** for each compared method, where we evaluate the behavior for *top: outlier mitigation* and for *bottom: mitigation of detection losses*.

B. Whole trajectories evaluation

Table II shows the results for the whole trajectory evaluation. The first four columns are the combinations of different configurations of our approach by combining the two proposed Φ functions with including or not the prior error estimation $\mathbf{e}$. We also show the prior trajectory and the three state-of-the-art methods compared results. The blue-marked values mean the best outcome for each circuit.

With the combination of our approach, we observe that the $\Phi^{(a)}$ + $\mathbf{e}$ implementation produces slightly better results than $\Phi^{(b)}$ + $\mathbf{e}$. In those cases, we mitigate the effect (i) even in the more challenging circuits C2 and C3. In both cases, we see that the error in C1' and C4' is held, mitigating the effect (ii). When we eliminate $\mathbf{e}$ estimation, the errors increase due to an ambiguous part of the circuit converging to a non-corrected prior trajectory.

Regarding the method Muñoz-Bañón *et al.* [4], we can see that the effect (i) is mitigated for the circuits C1 to C4. However, when detections are lost, the method converges to non-corrected prior, increasing the error by the effect of drawback (ii).

Methods Frosi *et al.* [12] and Cho *et al.* [13] cannot mitigate either (i) and (ii), and the errors become too large in the most challenging circuits C2 and C3. Cho *et al.* [13] did not finalize the most challenging circuit, C2, because it became lost. Method Frosi *et al.* [12] can complete that circuit but gets lost in some parts. Even when GPS augments [13], [12], they can be lost because, without weight adjustment, the LiDAR has more observations and produces more residuals in the optimization process. For the same reason, the ambiguity effect can produce results that overcome the prior error.

Looking into the results shown in Table II, we can infer how problems (i) and (ii) affect each method, but in the following section, we depict some concrete examples.

The Unet++ is a high-speed network, and we process images with low resolution (OS1-128 LiDAR resolution, 128x2048). Then, the experiments were performed in real-time, where our loop spent $57ms$ for the whole process, less than the $100ms$ required for real-time, which is the period of the LiDAR sensor. Muñoz-Bañon *et al.* and Frosi *et al.* spent around $90ms$, while Cho *et al.* occupied $170ms$, which involves processing 1 of each two scans for real-time implementation. The experiments have been performed on an i7-7700HQ CPU with 16 GB of RAM in C++. The network was implemented in PyTorch using a GPU GTX 1050 Ti.

C. Outlier mitigation

To evaluate outlier mitigation in more detail, we crop a trajectory through an area with outlier risk, i.e., a place where the ω_i^a has values close to zero. Then we evaluate the error against the ground truth per each frame for the compared methods.

In Fig. 7 (*top*), we can see the results of that process, where we show the value of ω_i^a in blue. When this value is near zero, the data is non-informative, and there is an outlier risk. The methods Frosi *et al.* [12], and Cho *et al.* [13] increase their errors, while Muñoz-Bañón *et al.* [4] and our proposed approach both mitigate the drawback (i).

D. Mitigation of detection losses

As mentioned in the setup section, we caused a lack of detections in parts of circuits C1 and C4, being then these circuits as C1' and C4'. Then, to look into the trajectories in detail to evaluate the effect (ii) mitigation, as in the previous section, we crop a path through an area where we stopped the detection process. Finally, we evaluate the error against the ground truth per each frame for the compared methods.

In Fig. 7 (*bottom*), we can see the results of that process, stopping the detections between the frames 150 and 350 approx. The prior trajectory error is shown in blue. We can see how when the detections are blocked, the errors in all compared methods become the same value as the prior error. In contrast, our approach can maintain stable error by mitigating the drawback (ii).

E. Effects of GPS error estimation influences

As a drawback (iii), we argue that inaccuracies in the GNSS-based prior trajectory introduce errors in the final estimation through the prior factor (2). Evaluating how $\mathbf{e}$ estimation can improve the final pose inference in areas where there is no lack, and no ambiguities is complicated because in our approach, when there is no $\mathbf{e}$ estimation, the ATE increases because of ambiguities areas. Plot areas with no risk provide not enough information. As a possible way to get some clue, we propose looking into circuit C1 results because it is the one where less ambiguities risk. In this case, we can see that our method is worst when $\mathbf{e}$ is not estimated.

F. Comparison with end-to-end learned methods

In the previous sections, we evaluate the mitigation of the typical drawbacks led by *handcrafted-feature-based* methods. However, evaluating our approach compared with *end-to-end learned* techniques is interesting. In this way, apart from demonstrating the mitigation of the discussed weaknesses, we show that our method is state-of-the-art for all

geo-localization strategies. It is worth noting that the [10], [5] approaches don't use GPS, and we don't augment it because in their methods, as in most *end-to-end learned*, the authors present their strategies as GPS replacements.

We implemented our approach in the KITTI Odometry Benchmark using the provided odometry and LiDAR information (range and reflectivity). We labeled the road boundaries for 938 images from the different Odometry Benchmark scenes. Then, we divided our dataset into 80% for training and 20% for testing with non-overlapping, obtaining 72% of the IoU metric as individual frame performance. Table III shows the result for sequences 00, 07, 09, and 10. We chose the sequences to have diverse environments and roads. The empty values indicate that the author's paper doesn't provide results for such a sequence.

TABLE III

WHOLE TRAJECTORIES EVALUATION (ATE) IN KITTI.

Method	**Sec. 00**	**Sec. 07**	**Sec. 09**	**Sec. 10**
Fervers *et al.* [10] (m)	-	0.85	-	0.96
Li *et al.* [5] (m)	-	**0.44**	1.16	0.93
Ours $\Phi^{(a)}$ + $\mathbf{e}$ (m)	**0.35**	0.47	**0.22**	**0.31**

V. CONCLUSIONS

This paper presented a geo-localization approach based on a weighted factor graph that dynamically adjusts its values depending on the information measured in the data. Moreover, the GNSS-based prior error estimation is included in the model. This strategy mitigates typical drawbacks in the *handcrafted-feature-based* geo-localization approaches: (i) The outlier raised from ambiguous representation. (ii) The deviations produced for sparse representations. (iii) The errors introduced by the GNSS-based prior. We demonstrate those mitigations experimentally by improving recent state-of-the-art methods in this way.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] J.-H. Pauls, K. Petek, F. Poggenhans, and C. Stiller, "Monocular localization in hd maps by combining semantic segmentation and distance transform," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4595–4601.

[3] T. Y. Tang, D. De Martini, D. Barnes, and P. Newman, "Rsl-net: Localising in satellite images from a radar on the ground," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1087–1094, 2020.

[4] M. Á. Muñoz-Bañón, J.-H. Pauls, H. Hu, C. Stiller, F. A. Candelas, and F. Torres, "Robust self-tuning data association for geo-referencing using lane markings," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 339–12 346, 2022.

[5] L. Li, Y. Ma, K. Tang, X. Zhao, C. Chen, J. Huang, J. Mei, and Y. Liu, "Geo-localization with transformer-based 2d-3d match network," *IEEE Robotics and Automation Letters*, 2023.

[6] F. Fervers, S. Bullinger, C. Bodensteiner, M. Arens, and R. Stiefelhagen, "Uncertainty-aware vision-based metric cross-view geolocalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 621–21 631.

[7] S. Zhu, M. Shah, and C. Chen, "Transgeo: Transformer is all you need for cross-view image geo-localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1162–1171.

[8] S. Hu and G. H. Lee, "Image-based geo-localization using satellite imagery," *International Journal of Computer Vision*, vol. 128, no. 5, pp. 1205–1219, 2020.

[9] L. M. Downes, D.-K. Kim, T. J. Steiner, and J. P. How, "City-wide street-to-satellite image geolocalization of a mobile ground agent," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 102–11 108.

[10] F. Fervers, S. Bullinger, C. Bodensteiner, M. Arens, and R. Stiefelhagen, "Continuous self-localization on aerial images using visual and lidar sensors," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7028–7035.

[11] T. Y. Tang, D. De Martini, and P. Newman, "Point-based metric and topological localisation between lidar and overhead imagery," *Autonomous Robots*, pp. 1–21, 2023.

[12] M. Frosi, V. Gobbi, and M. Matteucci, "Osm-slam: Aiding slam with openstreetmaps priors," *Frontiers in Robotics and AI*, vol. 10, p. 1064934, 2023.

[13] Y. Cho, G. Kim, S. Lee, and J.-H. Ryu, "Openstreetmap-based lidar global localization in urban environment without a prior lidar map," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4999–5006, 2022.

[14] J. Kim and J. Kim, "Fusing lidar data and aerial imagery with perspective correction for precise localization in urban canyons," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5298–5303.

[15] H. Roh, J. Jeong, and A. Kim, "Aerial image based heading correction for large scale slam in an urban canyon," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2232–2239, 2017.

[16] F. Yan, O. Vysotska, and C. Stachniss, "Global localization on openstreetmap using 4-bit semantic descriptors," in *2019 European conference on mobile robots (ECMR)*. IEEE, 2019, pp. 1–7.

[17] H. Hu, M. Sons, and C. Stiller, "Accurate global trajectory alignment using poles and road markings," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 1186–1191.

[18] M. Javanmardi, E. Javanmardi, Y. Gu, and S. Kamijo, "Towards high-definition 3d urban mapping: Road feature-based registration of mobile mapping systems and aerial imagery," *Remote Sensing*, vol. 9, no. 10, p. 975, 2017.

[19] M. M. Atia and S. L. Waslander, "Map-aided adaptive gnss/imu sensor fusion scheme for robust urban navigation," *Measurement*, vol. 131, pp. 615–627, 2019.

[20] B. Suger and W. Burgard, "Global outer-urban navigation with openstreetmap," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1417–1422.

[21] S. Singh, J. Singh, and S. S. Sehra, "Genetic-inspired map matching algorithm for real-time gps trajectories," *Arabian Journal for Science and Engineering*, vol. 45, no. 4, pp. 2587–2603, 2020.

[22] Z. Huang, S. Qiao, N. Han, C.-a. Yuan, X. Song, and Y. Xiao, "Survey on vehicle map matching techniques," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 1, pp. 55–71, 2021.

[23] I. del Pino, M. A. Munoz-Banon, S. Cova-Rocamora, M. A. Contreras, F. A. Candelas, and F. Torres, "Deeper in blue: Development of a robot for localization in unstructured environments," *Journal of Intelligent & Robotic Systems*, vol. 98, pp. 207–225, 2020.

[24] M. Á. Muñoz-Bañón, J.-H. Pauls, H. Hu, and C. Stiller, "Da-lmr: A robust lane marking representation for data association," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2193–2199.

[25] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: Redesigning skip connections to exploit multiscale features in image segmentation," *IEEE transactions on medical imaging*, vol. 39, no. 6, pp. 1856–1867, 2019.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive computing*, vol. 7, no. 4, pp. 12–18, 2008.

[28] M. Á. Muñoz-Bañón, E. Velasco-Sánchez, F. A. Candelas, and F. Torres, "Openstreetmap-based autonomous navigation with lidar naive-valley-path obstacle avoidance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 428–24 438, 2022.

[29] E. P. Velasco-Sánchez, M. Á. Muñoz-Bañón, F. A. Candelas, S. T. Puente, and F. Torres, "Lilo: Lightweight and low-bias lidar odometry method based on spherical range image filtering," *arXiv preprint arXiv:2311.07291*, 2023.