2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

Crossview Mapping with Graph-based Geolocalization on City-Scale Street Maps

Zhichao Ye, Chong Bao, Xinyang Liu, Hujun Bao, Zhaopeng Cui, Guofeng Zhang*
State Key Lab of CAD&CG, Zhejiang University

***Abstract*— 3D environment mapping has been actively studied recently with the development of autonomous driving and augmented reality. Although many image-based methods are proposed due to their convenience and flexibility compared to other complex sensors, few works focus on fixing the inherent scale ambiguity of image-based methods and registering the reconstructed structure to the real-world 3D map, which is very important for autonomous driving. This paper presents a low-cost mapping solution that is able to refine and align the monocular reconstructed point cloud given a public street map. Specifically, we first find the association between the street map and the reconstructed point cloud structure by a novel graph-based geolocalization method. Then, optimized with the corresponding relationship, the map accuracy is significantly improved. The rich environment information can also be associated with the point cloud by the geographical location. Experiments show that our geolocalization algorithm can locate the scene on a gigantic city-scale map (173.46 km^2) in two minutes and support 3D map reconstruction with absolute scale and rich environmental information from Internet videos.**

I. Introduction

Map reconstruction is a fundamental task in computer vision and has a wide range of applications in autonomous driving, robotics, and the digital twin. In the past decades, researchers have made significant progress in reconstructing large-scale scenes, including structure-from-motion (SfM) [1], [2], [3], [4], [5] with image-only observations, and the sensor-fusion approaches [6], [7], [8], [9] which leverages LiDAR, IMU or even GNSS to boost the mapping quality. Image-based mapping methods (such as SfM techniques), which take as input only video sequences or photo collections for scene reconstruction, is easy to use and can even carry out large-scale reconstruction on Internet data. However, the reconstruction accuracy is sacrificed and lacks the absolute mapping to the real world. On the other hand, the multi-sensor fusion methods can effectively improve the mapping accuracy by observing multiple sensors but bear the expensive equipment cost and complex manual acquisition (such as sensor calibration and synchronization). A follow-up question is: can we reconstruct a large-scale map with low cost while still maintaining the absolute mapping to the real world (i.e. recovering the scale of the scene and binding the geographic location to the real world)?

Fortunately, with the development of Google Maps and the OpenStreetMap (OSM) project, we can readily obtain the road information of the city. The street map provided by these projects commonly contains the geographic locations in the form of longitude and latitude and richly annotated labels, including road names, building names, and functionality of the buildings. If we can leverage this road information, there is a chance to reconstruct the large-scale scene with image-only observations while keeping the absolute scale and having the identical mapping from the reconstructed map to the real world. In this paper, we propose a low-cost and easy-to-use scene reconstruction pipeline, which matches road edges distilled from point clouds to the corresponding road structures of the street map, and utilizes the geographic locations to enhance the mapping procedure. As show in Fig. 1, the proposed method can locate point clouds on a city-scale street map. At the same time, the point cloud can be further optimized by geographic positioning information and fit the satellite image better.

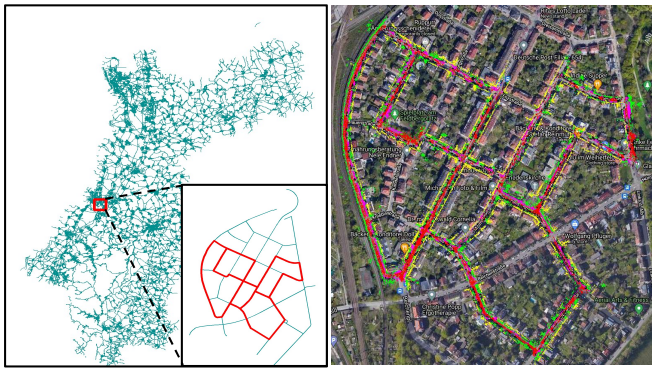

Fig. 1. **Result of our mapping method on a city-scale street map**. The left view is the street map of Regierungsbezirk Karlsruhe, which has 43,909 km of roads and 607,495 junctions. A small red rectangle marks the geolocalization result, and the details are illustrated in the right-bottom corner with matching roads highlighted in red. The right view denotes the optimized point cloud, which is projected onto the aerial image.

In order to make the whole mapping process easy to use, a powerful geolocalization module is necessary (e.g., locate only by the name of the city where the scene is located). Previous works [10], [11] tend to locate the camera position with a particle filter. In the beginning, these methods usually estimate camera poses with low confidence caused by insufficient observations and then gradually converges as the odometer or semantic information feed continuously. However, when processing in large-scale scenes, the calculation time is unacceptable due to the excessive number of camera candidate states. Unlike these works, we tackle the problem by finding correspondence between the reconstructed road and the street map with a graph-based **geolocalization mod-**

*Corresponding authors: Guofeng Zhang (zhangguofeng@zju.edu.cn)
The authors except Zhaopeng Cui are also affiliated with ZJU-SenseTime Joint Lab of 3D Vision. This work was partially supported by NSF of China (No. 61932003).

2022 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9681-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRA46639.2022.9811743

ule (see Fig. 2). Practically, we first convert the road structure of the point cloud and the street map provided by OSM into two graphs, with the road segments as nodes and connectivity as edges. In this way, the road correspondence search is transformed into a graph vertex matching problem, which can be solved by a method similar to subgraph isomorphism. Since the number of road segments is much smaller than the number of cameras and we further combine similar segments, the number of states involved in the calculation is reduced by several orders of magnitude. Moreover, we can start the matching process from any nodes with significant discrimination (i.e., a representative crossroad) and reduce the matching candidates rapidly. Experiments show that we can finish the matching of a 120km$\times$180km scene within only two minutes on KITTI00, which demonstrates the efficiency of the proposed matching method.

After figuring out the correspondence from the reconstructed map to the street map, we leverage the geographic locations to further optimize the reconstruction with a **optimization module** (see Fig. 2). Specifically, we transform geographic locations and the point cloud to a unified coordinate space and adjust the reconstruction through a two-stage optimization (i.e. pose-graph and full bundle adjustment). It is also noteworthy that the geographic locations from the street map inherently provides loop closures that are difficult to discover by visual cues. For example, when the camera walks through the same path from two opposite directions, the visual descriptors cannot be matched due to the variations of a perspective view, and the same regions would be unexpectedly reconstructed into two parts. In contrast, the geographic location provided by the street map is always globally consistent. When two parts of the point cloud are mapped to the same road segment, the potential loop can be detected, which is hard for the traditional loop closure method due to the lack of visual correlation. Therefore, we develop a novel loop-closing mechanism that exploits the structural information of the scene and close the loop in a plane-wise manner instead of a point-wise manner.

To sum up, our major contributions are as follows:

- We propose a low-cost mapping solution to refine and register the point cloud with geographic observations of a street map. The proposed mapping solution makes it possible to recover good 3D scene structure from rich monocular Internet videos.
- We propose an efficient graph-based geolocalization method to find the location of a 3D point cloud in a city-scale street map. Experiments show that we need only two minutes to locate on a city-scale(173.46 km^2) street map.
- We proposed a loop closure method based on the association of planar structures, which can merge and refine the point cloud without relying on feature matches.

II. Related Work

A. Reconstruction

Structure-from-motion (SfM) is a technology to determine the three-dimensional structure from images. The closed-form solution is firstly derived in [12]. It implements a virtual tour system that incrementally reconstructs scenes with exhaustive pair matching and frequent calls for bundle adjustment(BA)[13]. As an attempt to run in large Internet photo collection, [14], [1] exploited the skeleton graph of reconstruction and [15], [2] highlighted the iconic image representing the main structure of the scene. A lot of works[3], [16], [17], [18] had introduced several strategies for robust estimation framework in efficiency and accuracy. The robust open source SfM systems, OpenMVG[19] and COLMAP[4], provide reliable reconstruction solutions. In recent years, global SfM[20], [21], [22], [23], [24], [25] and hybrid SfM[26], [27], [28] have significantly improved the reconstruction. However, the SfM suffers the ambiguity of scale because the observability of camera pose is limited in the monocular sequence or Internet photo collection. A lot of works[29], [30], [31] combine GPS information into SfM to revise the drift in scale. However, this requires additional equipment to obtain GPS observations, not accessible on Internet data.

B. Crossview GeoLocalization

Crossview geolocalization has been received significant attention recently. Geolocation based on a single image usually relies on extracting high viewpoint invariance features. Some researchers[32], [33], [34] directly learning the embedding of cross-view images and performed image retrieval to search most similar images in a database. These methods rely on a vast database established in advance and are challenging to work in unknown scenarios. Other researchers[35], [36], [37] focus on the urban scene and use the scene structure to bridge the ground images and aerial images. Since the information of the single image is limited, geolocalization on sequential images can reach more high-precision results with hints of motion and structure. [10] uses a particle filter to geolocalize a ground image on a small scale. [11] uses a similar method to locate the 3D LiDAR data on an aerial image that semantic information is extracted to filter. The crossview semantic-based mapping[38] result has good accuracy, but accurate semantic segmentation of satellite images and additional sensors are required for this method. [39] depends on motion estimation from SLAM to localize in a street map with a probabilistic model. However, the localization speed is not ideal for a large street map due to too many initial candidate states of cameras. Therefore, we propose a road-based geolocalization method that is extracted from our point cloud.

III. Method

As illustrated by Fig. 2, we take as input a point cloud and a street map picked up from OpenStreetMap [40]. The overall pipeline consists of two main modules: 1) the geolocalization module which represents the point cloud and the street map into two graphs and localization by searching the vertex association between graphs (III-A), 2) the optimization module that adjust the reconstruction (the point cloud and camera

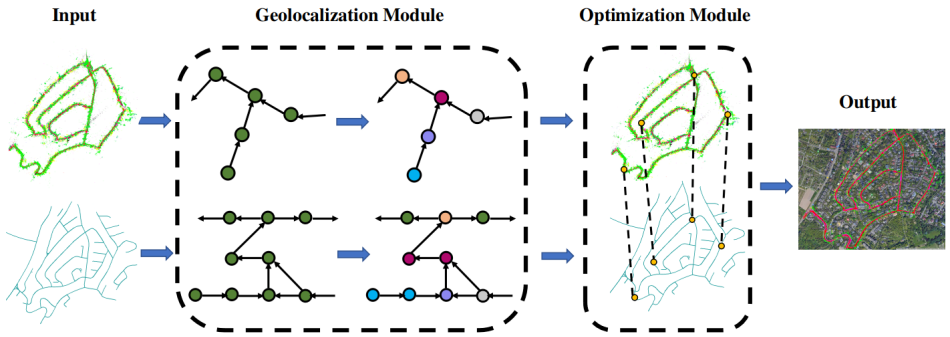

Fig. 2. **The pipeline of our mapping method.** The input is a point cloud and a street map. In the **geolocalization module**, the road structures are converted into graphs respectively, and then the corresponding relationship (marked by the same color) is searched. The **optimization module** refines according to the provided geographical location and finally generates a high-quality point cloud with real scale.

poses) to match the real-world geographic location according to the node association (III-B).

A. Geolocalization Module

Extraction of road structure. The goal of the geolocalization module is to associate the real-world geographic location from the street map to the reconstructed point cloud. In practice, we only use the road structure (i.e. geometry and topology of roads) of the input point cloud and the street map. Correspondingly, we need to extract the road structure from the point cloud. We first annotate the 'road' regions of the images through semantic segmentation with Deeplab2 [41] and label the point cloud with the 2D-3D association. Then we divide the point cloud into several local maps and calculate the road ground plane from the 'road' points on each local map. The road edges are projected onto these road ground planes and become a series of continuously broken line segments through a line fitting algorithm. Then, the direction and length of the road centerline are estimated by the road edges. At last, the road structure of the point cloud is represented by several connected broken line segments consistent with the street map. Please note that this road extraction is replaceable, and other similar algorithms are also applicable, but we found this simple process is sufficient for our task.

Converting road structure to road segment graphs. Although the extracted road structure is consistent with the representation of the road map, it is difficult to match them directly because there is an unknown *Sim(3)* transformation between the two coordinate systems. Furthermore, there may be errors and scale drift in the reconstruction, which further increases the difficulty of localization. We observed that the direction change of adjacent roads or the curvature of curved roads could be expressed by an *angle*. For a similarity transformation, this *angle* is an invariant measurement, and the connectivity of extracted road structure is also maintained in the street maps. Besides, the above geometry and topology of roads can be well represented by the label graph. Inspired by subgraph isomorphism, we propose a novel graph-based matching algorithm. We define a labelled directed graph g as a triple (V, E, L), where V is a set of vertices representing a one-way road segment, E is a set of directed edges which defines the connectivity of road segments, and L is a label

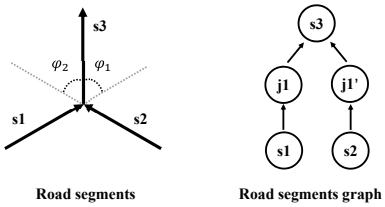

Fig. 3. **The graph representation of a road structure**: The left view is a three-way fork in the street map. φ_1, φ_2 denotes the *angle* of road direction change from s_1, s_2 to s_3 . The right view is the graph representation where two additional connection nodes j_1,j_1' are inserted and labelled by φ_1, φ_2 respectively.

function which maps a vertex to a label. For simplicity, all the two-way roads will be split into two individual one-way roads, with additional connection vertexes inserting between these two adjacencies and assigning different orientations. For each connected V (one-way road segment), we label with the *angle* of the road direction difference. Fig. 3 is an example showing how road segments is converted to a graph.

Graph-based matching. We convert the street map and road structure of the point cloud into two directed labelled graphs $G_{sfm} = (V_1, E_1, L_1)$ and $G_{osm} = (V_2, E_2, L_2)$. Now, the task can be formulated to the determination the correspondence between the nodes of G_{sfm} with nodes of G_{osm}. If the road segmentation is the same in G_{sfm} and G_{osm}, the correspondence can be represented by an injective function that makes the task equivalent to a subgraph isomorphism problem in which there are mature solutions. However, in real scenarios, the road segmentation on the street map is completed by crowdsourcing and does not follow stringent rules. In this case, the correspondence is relaxed as a many-to-many relationship. The problem is NP-hard and even worse than subgraph isomorphism. Therefore, we merge adjacent road segments with similar direction changes in G_{sfm} that makes the target correspondence become a one-to-many relationship (i.e. one vertex in G_{sfm} corresponding a chain in G_{osm}).

Before introducing the overall matching algorithm, we first describe how to get a matched path set P for a node u, with the related pseudo-code shown in Algorithm 1. The inputs are the degree $L(u)$, a starting vertex v_s, and data graph g where v_s is the a vertex in g. The output P consists of the ordered vertex lists, such as $(v_{i0}, ...v_{ik})$ which represents a path in g. The algorithm is realized by a simple breadth-first

Algorithm 1 Find matched chain

Input: query degree $L(u)$
Input: starting vertex v_s
Input: data graph g
Output: a path set P

$P = \emptyset$
let Q be a queue
Q.enqueue($(v_s, L(u),)$)
while Q is not empty **do**
  $(v, d, v_{i_0}...v_{i_k}) := Q$.dequeue()
  $d' = d - L(v)$
  if $d' < th_1$ **then**
    $P = P \cup \{v_{i_0}...v_{i_k}v\}$
  end if
  if $d' > th_2$ **then**
    continue
  end if
  for do$(v, v') \in E(g)$
    Q.enqueue($(v', d', v_{i_0}...v_{i_k}v)$)
  end for
end while

Algorithm 2 Graph-based matching method

Input: query graph q
Input: data graph g
Output: a mapping C

$V_m = \emptyset$, $E_m = \emptyset$
while $|V_m| < |V(q)|$ **do**
  $u =$ NEXTQUERYVERTEX(V, M)
  $V_s = \{v|\forall(u_0, u) \in E_m, (..., v) \in C(u_0)\}$
  for $v_s \in V_s$ **do**
    $P =$ FindMatchedChain($L(u), v_s, g$)
    $C(u) = C(u) \cup P$
  end for
  $V_m = V_m \cup \{u\}$
  $E_m = E_m \cup \{(u, u_1)|(u, u_1) \in E(q)\}$
  for $(u_0, u) \in E_m$ **do**
    $V_{prev} = \{v|(...v) \in C(u_0)\}$
    $C(u) = \{(v...)|\exists(v...) \in C(u), v \in V_{prev}\}$
  end for
end while

search (BFS). We start from one node and then calculate the difference between the degree of the path and $L(u)$ in each iteration. If the difference of the road direction change is small enough, the path will be added into P as a potential matched element. If the difference of the road direction change is so significant that for the path to match vertex u, subsequent matches for this path will be terminated, and we will skip to the next iteration directly. In our implementation, th_1 is set to 0.2 and th_2 is set to 0.5.

The pseudo-code of our matching algorithm is shown in Algorithm 2. The inputs are query graph q and data graph g, and output C consists of matched pair (u, P) where u is the a vertex in q and P is a path set in g. In each step of the matching process, we search the corresponding path for a vertex in q. Unlike the previous method matching by order of image sequence, we adopt a matching order strategy that begins from vertexes with extensive discrimination(i.e. a representative crossroad) that significantly restricts the search scope into several small areas. For example, a straight line may have many correspondences in the data graph, but the amount of three-way forks with specific angles is relatively small. On the other hand, if $(u', u) \in E(q)$, the start vertex of the path corresponding to u must link with an end vertex of the path corresponding u', that is, the mapping C is edge-preserving. With this constraint, we limit the starting vertex in each search step. Moreover, after each calculation of $C(u)$, we will traverse each matched edge and filter out the paths that do not meet the edge-preserving constraints. After the matching is completed, we will further filter the results through the relative scale of path length to reduce the ambiguity.

B. Optimization Module

Through the proposed cross-view geolocalization method, we can obtain the correspondences between OSM data and the roads extracted from the point cloud. Using Geographical observation from this correspondence, the point cloud can be well optimized. We first convert all OSM observations to a 3D coordinate system by the Mercator projection and transform the point cloud to the same coordinate system by a similarity transformation solved by the Umeyama algorithm [42]. A general approach is applying pose graph optimization to constrain the camera poses to the correspondence geographic location measurements. Each camera pose is represented by a similar transformation *Sim(3)*:

$$S_i = \begin{bmatrix} s_i R_i & t_i \\ 0 & 1 \end{bmatrix}. \tag{1}$$

The relative motion between camera poses is maintained by:

$$E(S_i, S_j) = ||log_{sim3}(\Delta S_{ij} S_i S_j^{-1})||^2, \tag{2}$$

with ΔS_{ij} denoted relative similarity transformation measurement. The loss function can be described as:

$$L_{geo} = \sum_{i=1}^{n_{geo}} ||t_{k_i} - g_i||^2 + \sum_{i,j}^{(i,j)\in P_c} E(S_i, S_j). \tag{3}$$

Here, the camera position t_{k_i} corresponds to a geographic location g_i, and P_c contains all the image pairs which share common tracks.

However, we can only get the geographic location observation of the road instead of camera poses. There is an offset between the camera position and the road. Furthermore, this problem will be more severe on the video collected by handheld or UAV, so it is not good to add constraints directly to the camera position. In contrast, we use some anchor points $\{p_i|i = 1...n\}$ as bridges to associate camera pose and observation information, where the anchor points are the intersections of roads that have been extracted from the point cloud.

Then, only latitude and longitude can be obtained from OSM data, while the altitude information is inaccurate or

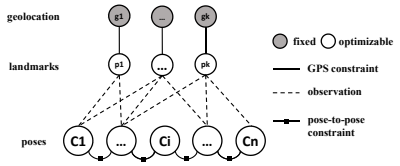

Fig. 4. The factors of pose graph optimization.

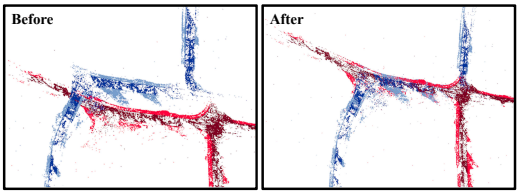

Fig. 5. **The results of the proposed loop closure method on KITTI 02**. The proposed loop closure method can fuse two separated local maps(colored by blue and red) well.

missing. Directly assuming all roads to be on the same plane is sub-optimal because they do not conform to the actual road conditions, and Completely ignoring the height limitation will also make the problem ill-posed. Therefore, we implement to reduce the weight along the height direction (y-axis), and the loss function can be rewritten as:

$$F(S_i, p_j) = ||\pi(R_i p_j + t_i) - s_i x_{ij}||^2, \tag{4}$$

$$L_{geo} = \sum_{i=1}^{n_{geo}} ||\lambda(p_i - g_i)||^2 + \sum_{i,j}^{(i,j)\in P_r} F(S_i, p_j) + \sum_{i,j}^{(i,j)\in P_c} E(S_i, S_j). \tag{5}$$

Here, λ is a diagonal matrix $diag(1, 1, 0.01)$ to adjust the weight of height direction, P_r contains the correspondence of cameras and points. Fig. 4 shows the connection of factors during optimization.

The correspondence between the street map and the point cloud also identify which part of the point cloud is on the same road. This makes it possible to detect the loop even if lack of visual correlation. Fig. 5 shows two local maps that cannot be merged by visual-based loop closure because the perspective views are very different in the two trajectory segments, which makes the feature descriptors fail to be matched. With the help of the graph matching results, we can easily find potential loops. However, how to merge these two local maps is still a problem. We proposed a novel loop closing method through the global measurement. The local maps of the ends of the broken loop are denoted as M_1 and M_2. First, we extract several planes from M_1 and M_2 and form a plane set $\mathcal{L}$. Each element l_i of $\mathcal{L}$ consists of (θ_i, P^i) that θ_i is the plane parameters and P^i is points on the plane. Since M_1 and M_2 reflect the same road segment, the correspondence (a_i, b_i) can be found that make l_{a_i} and l_{b_i} the same plane. For each correspondence, we can construct a new plane element $(\theta_{a_i}, P^{a_i} \cup P^{b_i})$ and form a new plane set $\mathcal{L}*$. The coplanar constraints can be formulated as

$$L_{co} = \sum_{i}^{(\theta_i, P^i)\in\mathcal{L}*} \sum_{j}^{p_j^i \in P^i} ||f(\theta_i, p_j^i)||^2, \tag{6}$$

here $f(\theta, p)$ is the distance from point p to plane θ. By minimizing L_{co}, we can align the points on the plane, so as to close the loop successfully which is shown in Fig. 5.

TABLE I

TIMING RESULTS[S] OF GEOLOCALIZATION ON THE KITTI DATASET.

	00	**01**	**02**	**03**	**05**
LOST[39](S)	253.10	494.73	280.18	122.01	389.54
LOST[39](M)	16955.97	8918.42	12544.87	22660.44	20453.53
ours(S)	0.02	0.07	0.01	0.01	0.02
ours(M)	7.09	1.20	2.09	1.65	4.41
ours(L)	76.75	16.23	24.96	~~16.16~~	25.59
Comp.(M)	247.34	501.00	281.24	119.51	387.84

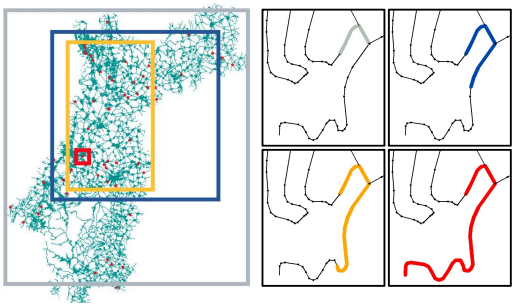

Fig. 6. **The intermediate results of geolocalization.** Left: The candidate matching region (represented by a box containing all candidate paths) decreases with the increase of iterations 6 (grey),8 (blue),10 (yellow),12 (red). Right: Road structures correspond to different iteration times.

IV. EXPERIMENTS

We evaluate our algorithm with five sequences on the KITTI visual odometry dataset. As for the input data, we use seven street maps downloaded from the OSM project [40]. In detail, five small maps correspond to five sequences, and each contains only the scene where the sequence is located. One medium map covers 18 km^2 and contains all scenes of five sequences. One large map contains 43,909 kilometres of streets covering the whole city (Regierungsbezirk Karlsruhe). The input point cloud is reconstructed by COLMAP [4]. According to the geolocalization module and optimization module, the experiment is divided into two parts. First, we compare the running time of the proposed geolocalization method with the open-source work LOST [39] with different scales of street maps. Then, we evaluate the accuracy improvement of the optimization method and qualitatively evaluate the mapping result of the Internet video. The experiments are conducted on a desktop PC with an Intel i7-9700K 3.6GHz CPU, 64GB of memory, and an Nvidia RTX-2070-8G graphics card. To inspect the robustness under common settings, we carry out all the experiments on monocular sequences.

Geolocalization. As shown in Table I, we test five sequences on maps of three scales, i.e. smalls (S), medium (M), and large (L). Following LOST [39], we omit sequence 04 because its trajectory is close to a straight line and thus cannot be matched due to insufficient structural information. For fair comparisons, all the algorithms run on a single thread. Compared with LOST, our geolocalization module has an order of magnitude advantage in computing time in all scale maps, making city-scale localization possible. For the large map, the running time of our method is 76.75s in sequence 00, while LOST is hard to run on such a vast map. Sequence 03 is very ambiguous, so two candidate results appear on the large-scale map (L), as shown in Fig. 7. In order to better reflect the positioning process of our algorithm, we illustrate

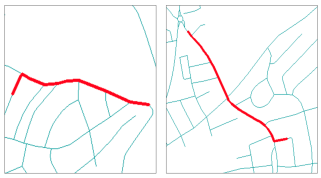

Fig. 7. **The ambiguity of road structure.** Left: The correct roads correspond to sequence 03 in the large street map. Right: An error roads correspond caused by almost the same road structure.

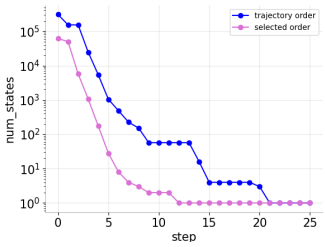

Fig. 8. Comparison of different selection strategies in the proposed geolocalization method.

the changes of the candidate matching region of sequence 02 in the search process in Fig. 6.

However, our method and LOST are not two opposing solutions, as we obtain the location of roads in a global manner and LOST locates camera positions in a sequential manner. Therefore, we also attempt to evaluate a composite strategy (denoted as Comp.) by using our method to find a small area corresponding to the point cloud from the original street map and then applying LOST to obtain the camera position in the small area. As Table I shows, this composite strategy is much more efficient than directly using LOST for geolocation. It is worth noting that LOST can only locate the camera trajectory which reflects the real scale (e.g. poses from Stereo-SLAM or VIO), since it relies on the absolute length information.

In addition, to more intuitively observe the influence of candidate vertex selection strategies on the algorithm, we also draw the decreasing trend of the number of states with the search step under different matching orders in Fig. 8. Our matching order strategy is significantly better than matching in trajectory order, has fewer states at the beginning, and converges faster.

Mapping Accuracy We also compare the mapping accuracy between our reconstruction method and other algorithms. In the implementation of the proposed method, Ceres Solver is used as the optimizer. The results are illustrated in Table II. Except for the original point cloud generated by COLMAP scaled to the actual scale, all the comparisons are at a fixed scale. For almost all the sequences, the proposed method improves the accuracy significantly and provides the absolute scale. Besides, Miller et al. [11] uses the LiDAR, IMU and satellite image for reconstruction, while we achieve higher accuracy with only monocular images and simple street information. We also compare a version (denoted as ours*) that assumes that all geographical observations are on the same plane and set λ as Identity matrix in Eq. (5) to verify the effectiveness of relaxing height constraints. In order to analyze the possible error sources, we calculate the pose errors of different axes and find that the error in the height

TABLE II

COMPARISON OF TRANSLATION RMSE(M).

	00	**01**	**02**	**03**	**05**
Miller et al.[11]	2.0	X	9.1	X	X
COLMAP [4]	4.31	6.51	28.08	1.86	4.23
ours*	1.86	4.18	7.03	1.88	2.13
ours	1.66	4.24	3.90	1.86	2.02

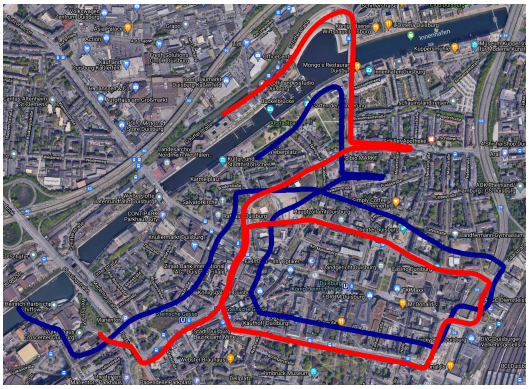

Fig. 9. **The mapping results of the Internet video**. We evaluate our mapping solution on the Internet video captured by a car driving in Duisburg. The original (blue) and optimized trajectories (red) are projected onto satellite images for comparison.

direction is the primary source due to the lack of height information in the OSM. On the other hand, we also project the OSM data to the satellite image. There is a slight error between the geographic nodes of OSM data and the actual location in longitude and latitude. This error may affect the mapping accuracy at a fine-grained level, but the overall reconstruction is still good.

We also experiment with Internet data to prove the robustness of our method as well as its great potential in dealing with rich Internet information. We download an Internet video from YouTube which is captured by a car driving in Duisburg. As shown in Fig. 9, we project the reconstruction result onto the satellite image. Since we have carried out geolocalization, the initial trajectory can be projected onto satellite images. The initial camera trajectory (blue) and our optimized trajectory (red) are visualized to compare the accuracy before and after optimization. We can see that the original trajectory completely deviates from the road due to scale offset and reconstruction error, while the optimized trajectory delicately fits the correct roads.

V. CONCLUSION

This paper presents a low-cost and easy-to-use mapping method, which is able to use the road structure of the street map to improve the mapping accuracy via a novel geolocalization algorithm. The comprehensive evaluation shows that the geolocalization time of the proposed method in a huge city-scale street map only takes a few minutes, and the reconstruction accuracy can be improved, which has excellent potential for Internet data. The proposed method has good performance when the road structure is discriminative, but it may be affected by similar road structures or graph matching errors. One interesting future work is to explore how to use other structures or semantic information to solve the ambiguity of street structure.

REFERENCES

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, "Building rome in a day," *Communications of the ACM*, vol. 54, no. 10, pp. 105–112, 2011.

[2] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik *et al.*, "Building rome on a cloudless day," in *European conference on computer vision*. Springer, 2010, pp. 368–381.

[3] C. Wu, "Towards linear-time incremental structure from motion," in *2013 International Conference on 3D Vision-3DV 2013*. IEEE, 2013, pp. 127–134.

[4] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.

[5] S. Zhu, R. Zhang, L. Zhou, T. Shen, T. Fang, P. Tan, and L. Quan, "Very large-scale global sfm by distributed motion averaging," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4568–4577.

[6] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.

[7] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765.

[8] P. Li, T. Qin, B. Hu, F. Zhu, and S. Shen, "Monocular visual-inertial state estimation for mobile augmented reality," in *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2017, pp. 11–21.

[9] P. Wang, R. Yang, B. Cao, W. Xu, and Y. Lin, "Dels-3d: Deep localization and segmentation with a 3d semantic map," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5860–5869.

[10] K. Y. K. Leung, C. M. Clark, and J. P. Huissoon, "Localization in urban environments by matching ground level video images with an aerial image," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 551–556.

[11] I. D. Miller, A. Cowley, R. Konkimalla, S. S. Shivakumar, T. Nguyen, T. Smith, C. J. Taylor, and V. Kumar, "Any way you look at it: Semantic crossview localization and mapping with lidar," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2397–2404, 2021.

[12] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," in *ACM siggraph 2006 papers*, 2006, pp. 835–846.

[13] M. Lourakis and A. Argyros, "The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm," Technical Report 340, Institute of Computer Science-FORTH, Heraklion, Crete ..., Tech. Rep., 2004.

[14] N. Snavely, S. M. Seitz, and R. Szeliski, "Skeletal graphs for efficient structure from motion," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008. [Online]. Available: https://doi.org/10.1109/CVPR.2008.4587678

[15] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm, "Modeling and recognition of landmark image collections using iconic scene graphs," in *European conference on computer vision*. Springer, 2008, pp. 427–440.

[16] K. Ni, D. Steedly, and F. Dellaert, "Out-of-core bundle adjustment for large-scale 3d reconstruction," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[17] G. Zhang, H. Liu, Z. Dong, J. Jia, T.-T. Wong, and H. Bao, "Efficient non-consecutive feature tracking for robust structure-from-motion," *IEEE Transactions on Image Processing*, vol. 25, no. 12, pp. 5957–5970, 2016.

[18] Z. Ye, G. Zhang, and H. Bao, "Efficient covisibility-based image matching for large-scale sfm," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8616–8622.

[19] P. Moulon, P. Monasse, R. Perrot, and R. Marlet, "Openmvg: Open multiple view geometry," in *International Workshop on Reproducible Research in Pattern Recognition*. Springer, 2016, pp. 60–74.

[20] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri, "Global motion estimation from point matches," in *2012 Second international conference on 3D imaging, modeling, processing, visualization & transmission*. IEEE, 2012, pp. 81–88.

[21] M. Brand, M. Antone, and S. Teller, "Spectral solution of large-scale extrinsic camera calibration as a graph embedding problem," in *European Conference on Computer Vision*. Springer, 2004, pp. 262–273.

[22] V. M. Govindu, "Combining two-view constraints for motion estimation," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 2. IEEE, 2001, pp. II–II.

[23] O. Ozyesil and A. Singer, "Robust camera location estimation by convex programming," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2674–2683.

[24] K. Wilson and N. Snavely, "Robust global translations with 1dsfm," in *European Conference on Computer Vision*. Springer, 2014, pp. 61–75.

[25] Y. Chen, J. Zhao, and L. Kneip, "Hybrid rotation averaging: A fast and robust rotation averaging approach," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 358–10 367.

[26] B. Bhowmick, S. Patra, A. Chatterjee, V. M. Govindu, and S. Banerjee, "Divide and conquer: Efficient large-scale structure from motion using graph partitioning," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 273–287.

[27] C. Sweeney, V. Fragoso, T. Höllerer, and M. Turk, "Large scale sfm with the distributed camera model," in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 230–238.

[28] S. Zhu, T. Shen, L. Zhou, R. Zhang, J. Wang, T. Fang, and L. Quan, "Parallel structure from motion from local increment to global averaging," *arXiv preprint arXiv:1702.08601*, 2017.

[29] M. Lhuillier, "Incremental fusion of structure-from-motion and gps using constrained bundle adjustments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 12, pp. 2489–2495, 2012.

[30] R. Carceroni, A. Kumar, and K. Daniilidis, "Structure from motion with known camera positions," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 477–484.

[31] L. Svärm and M. Oskarsson, "Structure from motion estimation with positional cues," in *Scandinavian Conference on Image Analysis*. Springer, 2013, pp. 522–532.

[32] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5007–5015.

[33] Y. Shi, L. Liu, X. Yu, and H. Li, "Spatial-aware feature aggregation for image based cross-view geo-localization," *Advances in Neural Information Processing Systems*, vol. 32, pp. 10 090–10 100, 2019.

[34] S. Zhu, T. Yang, and C. Chen, "Revisiting street-to-aerial view image geo-localization and orientation estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 756–765.

[35] M. Bansal, H. S. Sawhney, H. Cheng, and K. Daniilidis, "Geo-localization of street views with aerial image databases," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 1125–1128.

[36] Y. Tian, C. Chen, and M. Shah, "Cross-view image matching for geo-localization in urban environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3608–3616.

[37] T. Vojir, I. Budvytis, and R. Cipolla, "Efficient large-scale semantic visual localization in 2d maps," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

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

[39] 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, 2015.

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

[41] M. Weber, H. Wang, S. Qiao, J. Xie, M. D. Collins, Y. Zhu, L. Yuan, D. Kim, Q. Yu, D. Cremers, L. Leal-Taixe, A. L. Yuille, F. Schroff, H. Adam, and L.-C. Chen, "DeepLab2: A TensorFlow Library for Deep Labeling," *arXiv: 2106.09748*, 2021.

[42] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 13, no. 04, pp. 376–380, 1991.