Road-Network-Based Fast Geolocalization
Yongfei Li, Dongfang Yang, Shicheng Wang, Hao He, Jiaxing Hu, and Huaping Liu, Senior Member, IEEE

Abstract—In this article, a road-network-based geolocalization method is proposed. We match roads in the onboard images to the reference road vector map, and realize successful localization over areas as large as a whole city. The road network matching problem is treated as a point cloud registration problem under the homography transformation and solved under the hypothesize-and-test framework. To tackle the point cloud registration problem, a global projective-invariant feature is proposed, which consists of two road intersections augmented with their tangents. In addition, we propose the necessary conditions for the features to match. This can reduce the candidate matching features, thus accelerating the search to a great extent. These matching candidates are first “filtered” with the model consistency check in parameter space and then tested with similarity metrics to identify the correct transformation. The experiments show that our method can localize an aerial image over an area larger than 1000 km² within several seconds on a single CPU. Our code can be found at: https://github.com/FlyAlCode/RCLGeolocalization-2.0.

Index Terms—Aerial image, geolocalization, homography transformation, road intersection.

I. INTRODUCTION

AERIAL image geolocalization, which aims to estimate a transformation that aligns the aerial image to a known geographical coordinate system (GCS), is particularly essential for many remote-sensing tasks like map creating.

The standard for creating georeferenced aerial images is using an accurate GPS and IMU and/or measuring ground control points manually [1], which is usually expensive and time consuming. Considering that there are large numbers of features on the ground, such as roads, buildings, and so on, which are rather robust across time and are labeled on the existed geographic information system (GIS), localization with such information is feasible. With the use of deep learning in pixelwise segmentation [2]–[4], road segmentation has been considered relatively mature. So we employ the road network as the reference and develop a road-network-based geolocalization method.

Manuscript received October 14, 2019; revised May 15, 2020; accepted July 4, 2020. Date of publication August 18, 2020; date of current version June 24, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61673017 and Grant 61403398; and in part by the Natural Science Foundation of Shaanxi Province under Grant 2017JM6077 and Grant 201805040YD18CG24. (Corresponding author: Dongfang Yang.)

Yongfei Li is with the Department of Automatic Control, Xi’an Research Institute of Hi-tech, Xi’an 710025, China (e-mail: lyfe314@163.com).

Dongfang Yang, Shicheng Wang, and Hao He are with the Department of Automatic Control, Xi’an Research Institute of Hi-tech, Xi’an 710025, China (e-mail: yangdf301@163.com).

Jiaxing Hu was with the School of Electrical Engineering, Southeast University, Nanjing 211189, China. He is now with Nanyang Technological University, Singapore 639798.

Huaping Liu is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.

Digital Object Identifier 10.1109/TGRS.2020.3011034

There are many advantages for geolocalization using road network: first, reference road networks are readily available from many kinds of GIS, such as road maps from OpenStreetMap; second, the presentation of road maps is highly compact, so only a small amount of storage is needed to store the reference database when compared with geolocalization with aerial images; third, road networks are stable and they do not change much over time; finally, with the application of deep learning in road segmentation, relatively accurate road detection results have been achieved. Although its application prospect is appealing, this task is quite demanding. Despite the high precision of the state-of-the-art road extraction methods, much noise still exists, which often causes great changes in road topology structure. And the reference road vector map may not be up-to-date, resulting in some differences between roads in the query aerial images and reference road network map. Furthermore, variations in scale, orientation, and road width pose great challenges to register the road map estimated from the image captured by the onboard camera to reference road network. Under a homography transformation, which is the most general case, the length of the line segment, and the angle are not invariant, causing few features can be used to perform the matching. Because of this, many existing works [5]–[8] treat the transformation between the aerial image and the reference road network map as a similarity transformation, and extra attitude and altitude measurements from other sensors on the UAVs must be used to transform the aerial image to the reference coordination. However, this may fail when large drifts occur in attitude and/or altitude estimation. Finally, registration over a large area is very time consuming, and a wise search strategy must be developed to enable the practical application.

To address all these challenges, we treat the road-network-based geolocalization problem as the 2-D point cloud registration under a homography transformation. Different from registration under a Euclidean transformation or similarity transformation, most of the local geometric features, such as angle and length of line, change with the transformation. But the order of contact, intersection, and tangency stay unchanged [9]. So in this article, road intersections are selected as the basic geometric features. Through road intersections contain little information when treated as individuals, and thus less discriminative, the relative positions between them are often unique. Inspired by this fact, we propose a projective-invariant global feature which works with pairs of intersections augmented with their tangents. We call the pairs of intersections “2-road-intersections-tuple,” which is treated as a whole basic feature unit. It is shown that the homography

1https://www.openstreetmap.org
transformation that aligns the detected road map and the reference road map can be determined in closed form from a single pair of corresponding intersections tuples. And the rules to find the corresponding tuple candidates are deduced to allow fast search. After that, all the putative tuple correspondences are tested in a hypothesis-and-test framework for global registration.

The rest of this article is organized as follows. Section II presents a literature review; the detailed implementation of the proposed method is illustrated in Section III; Section IV contains experiments and analyses; following it are the conclusions and future research directions.

II. RELATED WORK

Generally speaking, vision-based geolocalization can be divided into two categories, according to the reference information used. One is the aerial-images-based methods, and the other is GIS-based methods.

Precollected aerial images have been deployed for georeferencing. Lindsten et al. [10] encoded the images with classification of the scene. Both UAV images and reference Google Map images are segmented into superpixels and then classified as grass, asphalt, and house. Class histograms are constructed by selecting circular regions around a point, resulting in a rotation-invariant descriptor. However, without using the rotation information, the method can only estimate the absolute translation. Moreover, the matching accuracy is poor in large homogeneous regions. Shan et al. [11] used Google Map images for both global localization and continuous navigation. The UAV position is initialized via correlation, after which the optical flow is used to predict its position in subsequent frames, and the onboard images are registered to Google Map with HOG features finally.

GIS-based geolocalization methods are usually more robust than images-based methods because they use semantic information which is much more stable than low-level features, such as intensity and keypoint. Li et al. [12] and Wang et al. [13] considered the road network matching as a graph matching problem. Both the reference road networks and the query road networks are first represented as attributed relational graphs. Then the image-to-map registration problem can be modeled as the subgraph isomorphism problem, which can be solved using methods like the relaxation labeling algorithm. However, such methods often fail because of the incorrect graph representation of road networks caused by noisy road segmentation results. Dumble and Gibbens [6], Jung et al. [7], and Wu and Hu [8] utilized road intersections for vision-aided navigation. All of them suppose the transformation between the query road networks and the reference ones can be represented with a similarity transformation. Under such a hypothesis, some geometric attributes, such as the relative lengths between road intersections and the angles between branches of road intersections, are invariant to the transformation and thus can be used as features for road intersections matching. However such an assumption rarely holds in reality, and additional information, such as the attitude of the UAV, must be provided to align the on-board image to the ground. Máttyus and Fraundorfer [14] geolocalized the aerial image sequences to reference road maps using geometric hash method. They extract road fragments by car tracking and match them to the reference road maps with the geometric hash method. To the best of our knowledge, this is the first method which can geolocalize an aerial image over an area as large as a whole city within reasonable time. Even so, the method is not efficient enough, because it needs to search in different scales of query image step by step.

In summary, all these methods do not address the geolocalization problem under a 2-D projective transformation, which allows large viewpoint differences between the query images and the reference road vector maps in practical application. To the best of our knowledge, this is the first work to geolocalize the aerial images to the road map under the projective transformation. Our main contributions are: 1) a global projective invariant feature named 2-road-intersections-tuple is specially designed, and the necessary conditions to find matching tuple are studied and 2) a fast geolocalization method is proposed, which can perform a successful localization over an area as large as a whole city within seconds, or even within tens of milliseconds when special road intersections appear in the query image.

III. ROAD-BASED GEOLOCALIZATION

In this section, we will introduce the road-based geolocalization method in detail. We start by describing the road intersections detection method. Following that, the projective-invariant 2-road-intersections-tuple feature is introduced and the necessary conditions for a 2-road-intersections-tuple feature to be a correspondence of another is deduced. Finally, the 2-road-intersections-tuple feature-based geolocalization method will be presented fully.

A. Road Intersections Detection and Tangents Estimation

Our road intersections detection is based on road segmentation. Many algorithms have been developed for road segmentation. Any method that can detect road effectively can be used in our localization method. In this article, we use the method proposed by He et al. [4].

As shown in Fig. 1, the morphological filtering is conducted on the road segmentation result image to generate a clean binary road image after road segmentation. And then, the binary road image is skeletonized using the method proposed in [15]. After that, all the points with more than two neighbor road points are treated as road intersections. In practice, there often exist more than one such kind of points
Fig. 2. Road intersection branches segmentation and tangents estimation. There are three branches for intersection $P$, and three lines are fitted using the points on the three branches respectively. Since the fitted lines $l_1$ and $l_3$ of branches $B_1$ and $B_3$ are approximately collinear, they are treated as the same tangent. Thus, there are two tangents for intersection $P$. The branches are drawn in green, while the tangents are in red.

near a road intersection, so we merge all points within one certain radius to form a road intersection. At last, we extract interconnecting pixels on the road skeletons to find all the branches for the road intersection.

As shown in Fig. 2, we estimate the tangents for the road intersection $P$ using the point sets in the extracted road intersection branches $P_{Bi}$ = $\{p_{ij}\}$. For each branch point set $P_{Bi}$, a line fitting is conducted to get tangent $q_i$. Because the road intersection detection method above cannot estimate the intersection center accurately, we perform a fine tuning for the intersection center using all the branch lines of it

$$ q_i^T x = 0, \quad i = 1, 2, \ldots, n $$

(1)

where $q_i$, $i = 1, 2, \ldots, n$ are the tangents of the intersection, and the $x$ is the homogeneous coordinate of the center of the intersection.

Considering that the tangent lines fitted by different branch point sets may be collinear and should be treated as one tangent line, we merge tangent lines whose angles are below a certain threshold

$$ \langle q_i, q_j \rangle < \delta $$

(2)

where $\delta$ is the angle threshold, $\langle q_i, q_j \rangle$ is the angle between tangent line $q_i$ and $q_j$, and can be calculated as

$$ \langle q_i, q_j \rangle = \arccos(\bar{q}_i \cdot \bar{q}_j), \quad i, j = 1, 2, \ldots, n $$

(3)

where $\bar{q}_i$, $\bar{q}_j$ are the corresponding unit tangent vectors of $q_i$, $q_j$ respectively.

B. 2-Road-Intersections-Tuple Feature and Its Matching Rules

As shown in Fig. 3, the 2-road-intersections-tuple consists of a pair of road intersections $P_1, P_2$, together with their tangents $Q_k = [q_k^i]_i = 1, \ldots, N_k, N_k \geq 2, k = 1, 2$. The homography transformation can be determined with four matching lines using methods like the direct linear transformation (DLT) algorithm [9]. A pair of corresponding 2-road-intersections-tuples provide at least four matching tangents, which is enough to estimate the homography transformation.

There are usually thousands of road intersections in the road network of a whole city, and thus millions of 2-road-intersections-tuples. It is vitally important to find the matching candidates of a 2-road-intersections-tuple to enable fast geolocalization. We achieve this by encoding the 2-road-intersections-tuple with two kinds of features: the topology feature and the geometry feature.

1) Topology Feature: Considering that the number of branches and the number of tangents of a certain intersection are invariant under the homography transformation, the topology feature which can be presented as (4), is homography-invariant

$$ d_t = \{N^1_b, N^2_b, N^1_q, N^2_q\} $$

(4)

where $N^1_b, N^2_b$ are the numbers of branches for the first and the second intersections in the 2-road-intersections-tuple, while $N^1_q, N^2_q$ are the numbers of tangents for them.

2) Geometry Feature: The geometry feature is used to described the relative positions between the tangents of a 2-road-intersections-tuple, which is represented as the cross ratios of tangents. Four concurrent lines have a cross ratio [9], which can be computed as

$$ cr(l_1, l_2, l_3, l_4) = \frac{\sin \alpha_{12} \sin \alpha_{24}}{\sin \alpha_{13} \sin \alpha_{24}} $$

(5)

where $\alpha_{ij}$ is the angle between the $l_i$ and $l_j$ line.

As shown in Fig. 3, let $l$ be the connection between road intersections $P_1, P_2$. And all lines passing through intersection $P_i (i = 1, 2)$ can be presented as $M_i = Q_i \cup l (i = 1, 2)$. Let $N_{M_i}$ be the number of elements in $M_i$. When $N_{M_i}$ is no less than 4, there exist at least four lines passing through the intersection, and thus cross ratios can be estimated using these lines. Supposing there are $N_{M_i}$ lines at the intersection $P_i$, $C_{N_{M_i}}^4$ cross ratios can be determined. These cross ratios encode the relative position information for the 2-road-intersections-tuple, and we call them “cross ratio descriptor” in this article

$$ d_t = [c^1_{i_1}, c^2_{i_2}, c^3_{i_3}, c^4_{i_4}]^T, \quad N_i = C_{N_{M_i}}^4 \quad (i = 1, 2) $$

(6)

where $c^j_{i}$ is the $j$th cross ratio for the $k$th intersection.

Matching Rules for the 2-Road-Intersections-Tuple: Even for a matching 2-road-intersections-tuple pair, the correspondence between their tangents is not unique. Let $P_1' - P_2'$ be the corresponding 2-road-intersections-tuple for $P_1 - P_2$, and then the connections of the two intersections $l$ and $l'$ are surely to be corresponding. Therefore, there exist only two kinds of correspondences for tangent set $Q_i$ and $Q_i'(i = 1, 2)$:
all the tangents are matched to each other clockwise, or one clockwise while the other counterclockwise, as is illustrated in Fig. 4. Thus, the rules for two 2-road-intersections-tuples to be corresponding are as follows.

1) \( d_i = d'_i \), where \( d_i \) and \( d'_i \) are the topology descriptors

2) 

\[
\|d^{++}_c - d'^{++}_c\|_2 < \delta, \quad \text{or} \quad \|d^{-+}_c - d'^{-+}_c\|_2 < \delta \\
\text{or} \quad \|d^{+-}_c - d'^{-+}_c\|_2 < \delta, \quad \text{or} \quad \|d^{--}_c - d'^{-+}_c\|_2 < \delta
\]

(7)

where + means the cross ratios are computed when the tangents are sorted clockwise, meanwhile - means cross ratios are computed when the tangents are sorted counterclockwise, \( d_i \) and \( d'_i \) are cross ratio descriptor of the query tuple and the reference tuple, respectively, and \( \delta \) is the threshold of cross ratio descriptor matching.

C. 2-Road-Intersections-Tuple-Based Geolocalization

At this point, given two corresponding 2-road-intersections-tuples, we are able to determine the homography transformation \( H \). In addition, we propose a way to describe the 2-road-intersections-tuple with the homography-invariant topology descriptor and cross ratio descriptor, and derive the rules for a 2-road-intersections-tuple to be a potential match of the other. The current challenge is in quickly establishing the correspondences such that a geolocalization is obtained. The solution to this problem is introduced in this section.

As shown in Fig. 5, our method can be divided into two stages: the offline search tree construction stage and the online search stage. The search tree construction starts with the extraction of all the road intersections in the reference road map using the method described in III-A. After that, all 2-combinations of road intersections with a distance less than maximum observable range are extracted to form 2-road-intersections-tuples. Then, all these 2-road-intersections-tuples are classified into different groups according to their topology descriptors, and a KD-tree is created using all the cross ratio descriptors for each group. In practice, there exist only a few possible values for the number of branches \( N_b \) and the number of tangents \( N_t \) (typically \( N_b = 3, 4, 5 \), and \( N_t = 2, 3, 4 \)), so the possible values of topology descriptor are also quite finite. This means only dozens of KD-trees are needed. Finally, the number of 2-road-intersections-tuples for creating the corresponding KD-trees is counted, which is used to identify how “special” such kind of 2-road-intersections-tuples are and is used to determine the sampling priority of 2-road-intersections-tuple in the query road map in the online search stage. The smaller the number is, the more “special” such kind of 2-road-intersections-tuples are and the higher sampling priority they receive.

Our online search scheme can be divided into three main components: hypothesis generation, hypothesis verification, and the adaptive stopping mechanism. The scheme starts by extracting all the road intersections in the query road map, and then all 2-combinations of them are picked to form 2-road-intersections-tuples. Then, the 2-road-intersections-tuple with the highest sampling priority is picked, and all its 2-road-intersections-tuple matching candidates in the reference road map is obtained by querying the KD-tree with the same topology descriptor. With each of the matching candidates, one homography transformation hypothesis that aligns the query road map to the reference road map can be computed. Inspired by [16], these hypotheses are not verified directly with our similarity metrics to finally decide whether to accept the models. Instead, we “filter” these models with the model consistency test in the parameter space so as to reject the vast majority of the hypotheses before the similarity verification. Once a hypothesis yielding the similarity higher then a given threshold is found, it is reported as the correct hypothesis, and the search scheme ends. If all the candidates have been processed and the stopping criteria are not met, the algorithm repeats this process for the one with the highest sampling priority in the remaining 2-road-intersections-tuples of the query road map. The whole geolocalization algorithm is shown in Algorithm 1.

We highlight the model consistency check in the parameter space for its effectiveness in improving the efficiency of our algorithm. The improvement comes from two aspects: on the one hand, the proposed 2-road-intersections-tuple is some kind of “weak” feature, which means there usually exist thousands of candidates for a given 2-road-intersections-tuple; on the other hand, the model consistency check, which can run in constant time, is much cheaper when compared with the road similarity verification.

1) Model Consistency Check in Parameter Space: Considering that correct hypotheses are tightly clustered together in the latent parameter domain [16], one estimated model is correct only when it is close enough to a certain previously estimated model. We test a hypothesis model by check whether there exists a previously estimated model close enough to it. The process is called the model consistency check in parameter space.

Since a metric is needed to measure the distance between two homography transformation, we present the homography
Fig. 5. Overview of our geolocalization method.

**Algorithm 1** 2-Road-Intersections-Tuple-Based Geolocalization Algorithm

**Input:** $P_q$ - 2-road-intersections-tuple set in query road map; $P_r$ - 2-road-intersections-tuple set in reference road map; $\lambda$ - confidence of successful geolocalization; $p_m$ - possibility that there exists the correct matching tuple for a 2-road-intersections-tuple in the query road map; $\delta$ - similarity threshold of correct geolocalization.

**Output:** $H^*$ - best homography transformation.

1: $\text{sim}^* \leftarrow 0$
2: for $i = 1 \rightarrow n(\lambda, p_m)$ do
3: $s_i^q \in P_q$ - Draw a 2-road-intersections-tuple in query road map.
4: $M_i^r$ - Find all the matching 2-road-intersections-tuple candidates in the reference road map with the matching rules in III-B.
5: for all $s_i^{j,i} \in M_i^r$ do
6: $H_{ij}$ - Estimation the homography matrix with $s_i^q$ and $s_i^{j,i}$.
7: $\text{collision\_found} \leftarrow$ Model consistency check in the parameter space with Alg. 2.
8: if $\text{collision\_found} == \text{True}$ then
9: $\text{sim}^{ij} \leftarrow$ Compute the similarity.
10: if $\text{sim}^{ij} > \text{sim}^*$ then
11: $H^*, \text{sim}^* \leftarrow H_{ij}, \text{sim}^{ij}$.
12: end if
13: if $\text{sim}^* > \delta$ then
14: Return.
15: end if
16: end if
17: end for
18: end for

transformation with the 4pt parameterization, with which the distance between two homography transformation $h_1, h_2$ is

$$\|h_1 - h_2\| = \max_{i=1,2,4} (\|x_i^1 - x_i^2\|, \|y_i^1 - y_i^2\|)$$

where, $(x_1^k, x_2^k, \ldots, x_4^k, y_2^k)$ are the 4pt parameterization of $h_k$.

Now, the problem comes to a range search problem to find the previously estimated transformations that are at a distance of up to a certain tolerance $t$. Similar to [16], we use the random grid algorithm [18], which can solve the problem in constant time.

Different from [16], which hashes the whole model parameter $h$, we just hash the projection point of origin in query road map in consideration that our search area is so large that it is not memory sufficient to hash the whole model parameter with sufficient accuracy. Besides, the two models are close enough only when the projection point of origin in query road map under the two models are close enough, which means it is sufficient to reject models far away from each other. The detailed hashing process is given in Algorithm 2.

2) **Model Evaluating:** The similarity between the query road map and the reference road map is measured with three metrics: the road intersections inlier ratio, query road points inlier ratio, and the reference road points inlier ratio which are defined similar with the Hausdorff distance [19].

An intersection in the query road map is treated as an inlier under a certain homography transformation when there exists one intersection with the same topology descriptor in the reference road map that is close enough to the projection center of the given intersection. So the road intersections inlier ratio can be calculated as

$$\text{sim}_{ri} = \frac{N \left( \left\{ x \mid \min_{x' \in Q} \|h(H, x) - x'\|_2 < \delta_{ri}, d_i(x) = d_i(x') \right\} \right)}{N_{sum}}$$

where $h$ is the projection function, $H$ is the estimated homography matrix, $Q^i$ is the road intersection set of the reference road map, $d_i$ is the topology descriptor of the intersection,
Algorithm 2 Model Consistency Check in Parameter Space

Input: $h$ - the current estimated homography model;
$L$ - number of hashing table; $c$ - random grid cell size;
$\delta_{dh}$ - model collision distance threshold.
1: for $i = 1 \rightarrow L$
2: Initialize an empty hash table $H_i$.
3: Generate random offset $t_i$.
4: end for
5: for $i = 1 \rightarrow L$
6: $x_h \leftarrow$ Compute projection of origin under model $h$.
7: $r_{h} \leftarrow$ Compute the hash index of $x_h$ with hash($\frac{h}{c}$).
8: $S_h = \{ h' | t_h=r_h \} \leftarrow$ Find all models that occupy the cell with the hash index $r_h$.
9: for all $h' \in S_h$
10: $d_h = ||h-h'|| \leftarrow$ Compute the distance between $h$ and $h'$.
11: if $d_h < \delta_{dh}$ then
12: Report a collision.
13: end if
14: end for
15: Store model $h$ in cell $H_i[t_h]$.
16: end for

$\delta_{h}$ is the inlier distance threshold, and $N_{\text{sum}}$ is the total number of road intersections in the query road map.

A road point is defined as an inlier when the distance of its projection point to the closest road point in the reference road map is less than a certain threshold. The road point inlier ration can be computed as

$$sim_{rq} = \frac{N\left( \left\{ x' \mid x' \in Q' \land ||h(H, x') - x'||_2 < \delta_{rq} \right\} \right)}{N_{\text{sum}}}$$

(10)

where $Q'$ is the reference road map point set, $\delta_{rq}$ is the inlier distance threshold, and $N_{\text{sum}}$ is the total number of the road points in the query road map.

Analogously, the reference road points inlier ratio is defined as

$$sim_{rr} = \frac{N\left( \left\{ x' \mid x' \in Q' \land ||H^{-1}(x') - x'||_2 < \delta_{rr}, x' \in Q' \right\} \right)}{N_{\text{sum}}}$$

(11)

where the $Q'$ is the set of road points in the projection area of the query road map in the reference road map, $\delta_{rr}$ is the distance threshold, and $N_{\text{sum}}$ is the total number of the road points in $Q'$. The Voronoi surface [20] of the query road map and the reference road map are calculated in advance to accelerate the computation of the $sim_{rq}$ and $sim_{rr}$. The overall similarity is the minimum of the three metrics

$$sim = \min(\text{sim}_{rr}, \text{sim}_{rq}, \text{sim}_{rr}).$$

3) Termination Condition: Since we filter bad models with the model consistency check in the parameter space, the correct model can be found when at least two correct matching 2-road-intersections-tuple pairs are sampled. Suppose the possibility that there exists the right matching tuple for a given tuple in the query road map is $p_m$, the possibility of failing to find the correct model after sampling $n$ tuples in the query road map is

$$p = (1 - p_m)^n + n p_m (1 - p_m)^{n-1}.$$  (13)

The minimum sample number of 2-road-intersections-tuples needed in the query road map to ensure that the possibility to find the correct model is no less than $\lambda$ is

$$n^* = \arg \left( 1 - (1 - p_m)^n + n p_m (1 - p_m)^{n-1} < 1 - \lambda \right).$$  (14)

IV. EXPERIMENTAL RESULTS

In this section, we perform experiments on both synthetic and real aerial images to evaluate our algorithm.

A. Experiments on Synthetic Aerial Image Data Sets

We conducted experiments to study the influence of camera pose, the ground sample distance (GSD) on our algorithm. And also we tested the generalization of our algorithm by performing our method on as many as 20 different scenes. Since a large amount of aerial images taken under different camera poses, GSDs, and scenes are needed, which are difficult to obtain with real flight, we generated the aerial images with our simulated flight procedure, which can be found at https://github.com/FlyAICode/FlightSim.

1) Data Set: In the simulated flight procedure, a color aerial orthoimageray was attached to the ground. And a UAV with a camera mounted to flew over the area. The homography transformation between the image plane and the ground was estimated with the pose of the camera and the camera intrinsic parameters as

$$H = K \begin{bmatrix} r_1 & r_2 & t \end{bmatrix}$$

(15)

where $K$ is the camera intrinsic parameters matrix, $r_1$ and $r_2$ are the first and second columns of the rotation matrix, respectively, and $t$ is the translation vector.

The synthetic aerial images were generated by projecting the pixels in the color aerial orthoimageray onto the camera image plane.

Our color aerial orthoimagerys were downloaded from Massachusetts Document Repository. In these imagerys, which cover the whole Massachusetts, Massachusetts are divided into 38 areas, and each area is stored in one separate MrSid file with a maximum resolution of 0.3 m/pixel. And the imagerys are aligned to the epsg:2033 GCS.

We generated three data sets, named “multiposes data set,” “multi-GSDs data set,” and “multiareas data set” in this article. To generate the first two data sets, we chose the 13th area and the 17th area, because the roads in the 13th area are sparse while roads in the 17th area are much denser. In the “multiposes data set,” we chose 100 camera positions randomly within each area. And in each position, we fixed the camera yaw but changed the pitch from 0$^\circ$ to 45$^\circ$ with a

2https://docs.digital.mass.gov/
3https://en.wikipedia.org/wiki/MrSid
4https://epsg.io/2033
step of 5°. In the “multi-GSDs data set,” we fixed both the camera position and camera pose but changed the focal length of the camera. In this way, we obtained aerial images with different GSDs. In each place, we generated six different aerial images whose GSDs were 0.25, 0.5, 0.75, 1.0, 1.5, and 2.0 m/pixel, respectively. In the “multiareas data set,” we chose the first 20 areas which cover an area of 368–1109 km² to generate the aerial images. In each area, we generated 100 aerial images with randomly selected camera poses and camera positions. The GSDs of obtained aerial images ranged from 0.5 to 2.0 m/pixel, the yaw ranged from 0° to 360°, and the pitch and roll ranged from 0° to 20°.

2) Baseline: Our method is compared with the state-of-the-art method in [14]. They do not make their code public, so we implement their method on our own. There are some differences between our implementation and that reported in their paper. First, we use a road detector instead of a car tracker to extract all the roads. This will lead to a road detection result with higher completeness but lower correctness. Second, since we have extracted the complete road networks, the road intersection can be detected and used to create the basis. In our implementation, only the road intersection is selected as the start point $p_{s1}$ to construct the basis in both database construction stage and shortlist retrieval stage. This eliminates the need to shift the road segments to search for the right position, thus accelerating the search to a large extent. Third, to compare the two methods fairly, the code runs in a single thread instead of multiple threads. Finally, some parameters, such as the scale search step, are not reported in this article, and we have to choose these parameters on our own. Except for these parameters, all the parameters are kept the same as those in this article. Our implementation for the method can be found at https://github.com/FlyAlCode/PLBGHashing.

3) Metric: We evaluated the performance of both methods with three metrics: precision, recall, and run time. A geolocalization is regarded as a correct one when the distance between the estimated homography transformation and the ground truth is less than a certain threshold. In all our experiments on synthetic aerial image data sets, the threshold was set as 400. We found that the query road map overlapped well visually with the reference road map when the estimated model was within such distance of the ground truth. The PLBGHashing delivers a shortlist of locations. We test the first ten locations with the highest voting scores, and the localization is regarded as the correct one as long as one of the ten locations is close enough to the ground truth. Our method returns only one location.

a) Geolocalization under different camera poses: We studied the influence of camera pose on both the PLBGHashing method and ours on the “multiposes data set.” In this experiment, the parameters of the two methods were selected empirically and kept fixed. Some parameters of our algorithm were set as: numbers of road branch points used to estimate the tangent $N = 0.02 \nu$, where $\nu$ is the width of the image; random grid cell size $c = 50$; homography model distance threshold $\delta_{hn} = 200$; number of hashing table $L = 2$; possibility that there exists a correct matching 2-road-intersections-tuple in the reference road map for a tuple in the query road map $p_{m} = 0.2$; confidence of success $p = 0.99$; distance threshold of the intersection inlier ratio $\delta_{i} = 0.033 \nu$; distance threshold of the query road points inlier ratio $\delta_{q} = 40$; distance threshold of the reference road points inlier ratio $\delta_{r} = 0.033 \nu$; similarity threshold of a correct geolocalization $\delta_{sim} = 0.8$. Some parameters of the PLBGHashing were set as: hashing cell size $c = 15$; reference map tile size $c_{tile} = 5000$ with 50% overlap; minimum search scale $s_{min} = 0.33$; maximum search scale $s_{max} = 3.0$; scale search step $s_{step} = 0.05$. The precisions and recalls for both methods are shown in Fig. 6.

As can be seen, the precision and recall of the PLBGHashing decrease sharply with the increase in camera pitch. The PLBGHashing fails even when the camera obliques slightly, such as a pitch of 5°. This suggests that the PLBGHashing cannot deal with the general homography transformation case since they model the transformation between the onboard image and the reference road map as the similarity transformation, which means the PLBGHashing only works when the image plane is parallel to the ground strictly or when the accurate camera poses are provided to project the onboard images onto the ground. Since our method models the transformation between the onboard image and ground as the homography transformation, our method can deal with the case when the aerial images are taken under severe oblique views to some degree. In all cases, the precisions of our method are higher than 90%, and the recalls of our method are much higher than the PLBGHashing even under the similarity transformation when the camera pitch is 0°. The recalls of our method also decrease with the camera pitch when the pitch is larger than 5°. There are two reasons for this: first, as is shown in Fig. 7, the appearance of the aerial images under large oblique views are quite different from those used in training, leading to great challenges to the road segmentation algorithm; second, under large oblique views, the width of the road cannot be ignored and the tangents of the road branches estimated with road skeleton are noisy. Some successful geolocalization cases are shown in Fig. 7.

We also report the run time of both methods in Fig. 8. It can be seen that our method runs much faster (about ten times
Fig. 7. Examples of successful geolocalization on the “multiposes data set.” The aerial images are shown in the first column and the geolocalization results are shown in the second column by projecting the aerial images onto the reference road map (blue). We added a padding of 200 m for each image. The blanks in the aerial images are the sky. The three aerial images were taken under the same position, yaw and roll, and with (Top) 0°, (Middle) 20°, and (Bottom) 40° pitch, respectively. Pay attention to the severe deformation of roads caused by the large oblique view.

faster) than the PLBGHashing, especially in the 17th area where the roads are dense. The main reasons leading to the improvement of the efficiency are twofold: on the one hand, we use local road geometric feature, the road intersections, to generate model hypotheses, which reduces the search space to a large degree; on the other hand, we filter the generated hypotheses with our model consistency check in parameter space before the expensive model similarity validation. The
difference of geolocalization time between different aerial images in the same area is larger compared with that of the PLBGHashing method. This is because the numbers of candidate matching 2-road-intersections-tuples of each tuple in the query road map vary a lot. When there exist ‘special’ tuples in the query road map with few matching tuples, our method finds the right model much faster.

b) Geolocalization under different GSDs: In this section, we studied the performance of our method under different GSDs with experiments on the “multi-GSDs.” The parameters are set slightly different from these in the experiment on the “multiposes data set.” In the experiment the parameters of the random grid algorithm changed with the GSD: $c = 100, \delta_{dh} = 400$ for 0.25 m/pixel, $c = 75, \delta_{dh} = 300$ for 0.5 m/pixel and $c = 50, \delta_{dh} = 200$ for others. Since the accuracy of estimated tangents is lower with smaller GSD, a higher collision detection threshold is needed to allow correct models to pass the test. The recalls and precisions are shown in Fig. 9. As can be seen, our method works best under the GSDs of 1.0 and 1.5 m/pixel. The recall decreases with the increase of GSD when the GSD is smaller than 1.0 m/pixel, which is mainly caused by the inaccurate estimation of the tangents. When the GSD is small, the width of the road is dozens of pixels, which cannot be ignored in the estimation of tangents. When the GSD is too large, the road map cannot capture some detailed geometric features of roads, causing a decrease of recall.

c) Geolocalization in multiple scenes: We tested the generalization of our method by conducting geolocalization experiments on the “multiareas data set.” The parameters of our algorithm kept the same with these in the experiments on the “multiposes data set.” Since the PLBGHashing does not work under the homography transformation, we only report the results of our method. The precisions and recalls are shown in Fig. 10, and the run time is shown in Fig. 11. In all areas, the precisions are higher than 95%, which proves that in most cases the roads in the aerial images are unique to determine their positions and our similarity metrics are effective to distinguish correct geolocalization from these incorrect.

The recalls vary in different areas. As is discussed in IV-A3.a and IV-A3.b, both the camera pose and the GSD have some influence on the recall. And also the reference road map has a great impact on the recall. When there are many homologous local roads, it is more difficult for our algorithm to find the
right place. The run time also varies with reference road maps. Usually, the denser the roads are, the more time is needed. Besides, the intersections in the query road map also have a big influence on search time, because the distribution of different types of intersections is usually uneven in the reference road map. When there exist “special” 2-road-intersections-tuples in the query image, the search runs very fast (within 1 s). This is because there exist few candidate tuples for a “special” tuple.

d) Discussion: As is shown in the experiments above, the camera pose, the GSD, and the reference road map have an impact on the performance of our algorithm to some extent. Our geolocalization method is not quite suitable to deal with aerial images taken under severe oblique views since in such cases both the road segmentation results and the estimation of tangents are imprecision. When the GSD of the aerial images is less than 0.5 m/pixel, our method does not work that well. However, this can be improved greatly by resizing the aerial images to the GSD close to 1.0–1.5 m/pixel if we know the approximate GSD of the aerial images, which can be estimated with the camera intrinsic parameter and altitude of the camera. Besides, there needs to be enough roads in the aerial image to determine its location uniquely. It is hard to recognize the location of the aerial image where there are few roads with no intersection using only roads matching method. In our experiments, the correct locations were found before no more than 40 2-road-intersections-tuples in the query road map were sampled. This means our method works well when more than six road intersections are in the aerial images.

B. Experiment on Real Aerial Image Data Set

We have also performed experiments to test our method on real aerial image sequences over China. We captured 12 image sequences with a camera on the Phantom 4 Pro V2.0. Both the internal camera parameters and the camera distortion parameters were precalibrated and kept fixed. The original resolution of the acquired video was $3840 \times 2160$ and was resized into $960 \times 540$. The GSD of the resized image was about 1.0 m/pixel. The altitude over the ground was approximately 1000 m. Since roads in the area were sparse and the coverage of a single frame was not large enough to contain enough roads, we stitched the frames in the image sequence.

![Fig. 12. Some successful geolocalization on real aerial images. The aerial image mosaics are projected onto the reference road raster map.](image-url)
into larger aerial images. For a long image sequence where the camera moves about 4 km, we divided the sequence into parts and stitched each part into a single aerial image, and for a shorter image sequence, we stitched it into a single aerial image directly. Finally, we got 22 aerial images totally, and each covered an area of approximately 1 km × 2 km. Since the area was rural and the OpenStreetMap did not contain detailed road data in such place, we downloaded our reference road map from the Gaode map website5 with QGIS, which was saved in the raster map format with the resolution of 2 m/pixel. The whole reference road map covered an area of 54,985 km × 37,966 km.

Geolocalization on the data set is much more challenging. The roads in these aerial images are quite different from each other, making it difficult to detect roads correctly and completely. Besides, the reference road map is not up-to-date, causing some differences between the reference road map and the real one. We verified whether the geolocalization was successful by projecting the mosaic images on the reference road map with the estimated homography matrix and checking whether they overlap with the reference road map well visually. Our method succeeded to find the correct locations of 15 out of 22 aerial images. The total searching time varied from 4.5 to 19.3 s. And some successful geolocalization results are shown in Fig. 12. The unsuccessful geolocalization is mainly caused by the inconsistency between the reference road map and the detected roads, which will lead to low similarity even when the estimated homography transformation is correct and so is rejected wrongly.

V. CONCLUSION

In this article, we address the problem of geolocalizing aerial images with roads in a large area. We use only road vector maps as the reference which is easily accessible and highly compact. We have demonstrated the effectiveness of our method in experiments on synthetic and real aerial images. In our current work, the performance of our algorithm depends on the result of road segmentation. In future work, we will try methods that regress road tangents directly from the aerial image with deep-learning-based method. We believe this will improve the performance of our method further.

REFERENCES

[1] K. Kraus, Photogrammetry: Geometry From Images Laser Scans. Berlin, Germany: Walter de Gruyter, 2011.

[2] S. Sun, J. Pang, J. Shi, S. Yi, and W. Ouyang, “Fishnet: A versatile backbone for image, region, and pixel level prediction,” in Proc. Adv. Neural Inf. Process. Syst., S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. New York, NY, USA: Curran Associates, Inc., 2018, pp. 754–764. [Online]. Available: http://papers.nips.cc/paper/7356-fishnet-a-versatile-backbone-for-image-region-and-pixel-level-prediction.pdf

[3] Z. Miao, W. Shi, H. Zhang, and X. Wang, “Road centerline extraction from high-resolution imagery based on shape features and multivariate adaptive regression splines,” IEEE Geosci. Remote Sens. Lett., vol. 10, no. 3, pp. 583–587, May 2013.

[4] H. He, D. Yang, S. Wang, S. Wang, and Y. Li, “Road extraction by using atrous spatial pyramid pooling integrated encoder-decoder network and structural similarity loss,” Remote Sens., vol. 11, no. 9, p. 1015, Apr. 2019.

[5] D. Costea and M. Leordeanu, “Aerial image geolocalization from recognition and matching of roads and intersections,” 2016, arXiv:1605.08323. [Online]. Available: http://arxiv.org/abs/1605.08323

[6] S. J. Dumble and P. W. Gibbens, “Airborne vision-aided navigation using road intersection features,” J. Intell. Robotic Syst., vol. 78, no. 2, pp. 185–204, May 2015.

[7] J. Jung, J. Yun, C.-K. Ryoo, and K. Choi, “Vision based navigation using road-intersection image,” in Proc. 11th Int. Conf. Control, Autom. Syst., Oct. 2011, pp. 964–968.

[8] L. Wu and Y. Hu, “Vision-aided navigation for aircrafts based on road junction detection,” in Proc. IEEE Int. Conf. Intell. Comput. Intell. Syst., Nov. 2009, pp. 164–169.

[9] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[10] F. Lindsten, J. Callmer, H. Ohlsson, D. Törnqvist, T. B. Schön, and F. Gustafsson, “Geo-referencing for UAV navigation using environmental classification,” in Proc. IEEE Int. Conf. Robot. Autom., May 2010, pp. 1420–1425.

[11] M. Shan, F. Wang, F. Lin, Z. Gao, Y. Z. Tang, and B. M. Chen, “Google map aided visual navigation for UAVs in GPS-denied environment,” in Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO), Dec. 2015, pp. 114–119.

[12] S. Z. Li, J. Kittler, and M. Petrou, “Matching and recognition of road networks from aerial images,” in Proc. Eur. Conf. Comput. Vis., Berlin, Germany, 1992, pp. 857–861.

[13] C. Wang, A. Stefanidis, and P. Agouris, “Relaxation matching for georegistration of aerial and satellite imagery,” in Proc. IEEE Int. Conf. Image Process., Sep. 2007, p. 449.

[14] G. Müttys and F. Fraundorfer, “Aerial image sequence geolocalization with road traffic as invariant feature,” Image Vis. Comput., vol. 52, pp. 218–229, Aug. 2016.

[15] T. Y. Zhang and C. Y. Suen, “A fast parallel algorithm for thinning digital patterns,” Commun. ACM, vol. 27, no. 3, pp. 236–239, Mar. 1984.

[16] S. Korman and R. Litman, “Latent RANSAC,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 6693–6702.

[17] S. Baker, A. Datta, and T. Kanade, “Parameterizing homographies,” Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-06-11, 2006.

[18] D. Aiger, E. Kokipoulou, and E. Rivlin, “Random grids: Fast approximate nearest neighbors and range searching for image search,” in Proc. IEEE Int. Conf. Comput. Vis., Dec. 2013, pp. 3471–3478.

[19] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, “Comparing images using the Hausdorff distance,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 15, no. 9, pp. 850–863, Sep. 1993.

[20] D. P. Huttenlocher, K. Kedem, and M. Sharir, “The upper envelope of Voronoi surfaces and its applications,” Discrete Comput. Geom., vol. 9, no. 3, pp. 267–291, Mar. 1993.

Yongfei Li received the B.S. and M.S. degrees in automatic control from the Xi’an Research Institute of Hi-tech, Xi’an, China, in 2014 and 2016, respectively, where he is pursuing the Ph.D. degree. His research interests include SLAM, vision-based localization, and navigation.

Dongfang Yang received the B.S., M.S., and Ph.D. degrees in automatic control from the Xi’an Research Institute of Hi-tech, Xi’an, China, in 2006, 2009, and 2013, respectively. He is an Associate Professor with the Xi’an Research Institute of Hi-tech. His research interests include autonomous aerial vehicles, modern navigation technology, and nonlinear filtering.
Shicheng Wang received the B.S., M.S., and Ph.D. degrees in automatic control from the Xi’an Research Institute of Hi-tech, Xi’an, China, in 1985, 1988, and 1992, respectively. He is a Professor with the Xi’an Research Institute of Hi-tech. His research interests include flight vehicle guidance and simulation, vision-based navigation, and integrated navigation methods.

Hao He received the B.S. and M.S. degrees in automatic control from the Xi’an Research Institute of Hi-tech, Xi’an, China, in 2013 and 2015, respectively, where he is pursuing the Ph.D. degree. His research interest includes remote sensing image processing focusing on learning.

Jiaxing Hu received the B.S. degree in electrical engineering from Southeast University, Nanjing, China, in 2020. He is pursuing the M.S. degree with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include computer control and automation, machine vision, and neural systems.

Huaping Liu (Senior Member, IEEE) received the B.S. degree in mechanical engineering from the University of Shanghai for Science and Technology, Shanghai, China, in 1997, the M.S. degree in electrical engineering from Tongji University, Shanghai, in 2000, and the Ph.D. degree in computer science and technology from Tsinghua University, Beijing, China, in 2004. He is an Associate Professor with Tsinghua University. His research interest includes robot multimodal perception, control, and reinforcement learning.