Matching Buildings Between 2.5D Maps and Dash Cam Images for Building Identification

Yuki Awano
NTT DATA Corporation
Tokyo, Japan
Yuki.Awano@nttdata.com

Takuya Nishimura
NTT DATA Corporation
Tokyo, Japan
Takuya.Nishimura@nttdata.com

ABSTRACT

Technology that matches buildings between dash cam images and digital maps (GPS) assists in the identification of buildings. The identification enables to collect city information as well as the textures for 3D building models. However, the matching technology has two challenges. First, GPS locations can be highly inaccurate in cities that have tall buildings, which means the GPS sometimes needs to be relocated to capture the correct building features for matching. Second, it can be tricky to set the position and orientation of the camera by making correspondence of certain features of buildings in images and maps. In this work, we propose a method of matching buildings in dash cam images and 2.5D maps that uses the height information of the buildings equivalent to the LoD1 in the CityGML format. For the first challenge, we relocated GPS locations by using a map-matching method. For the second challenge, we adjust positions and orientations by matching the building edges in the images and maps after extracting the edges with a deep-learning-based detection model. Tests using real-world datasets demonstrated that our proposed method matched the buildings better than the baseline.

CCS CONCEPTS

• **Information systems** → **Spatial Analysis and Integration**.

KEYWORDS

2.5D map, building identification, edge detection model

ACM Reference Format:
Yuki Awano and Takuya Nishimura. 2021. Matching Buildings Between 2.5D Maps and Dash Cam Images for Building Identification. In *29th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '21), November 2–5, 2021, Beijing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3474717.3483966

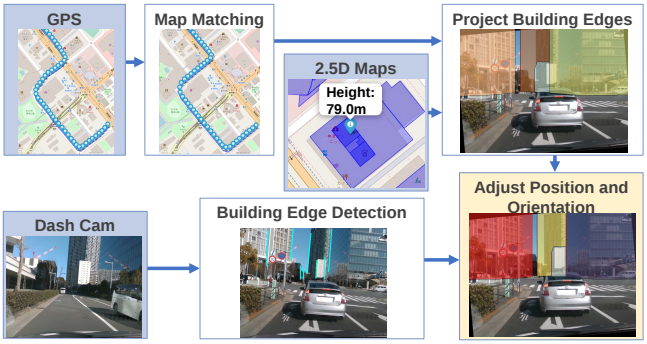

Figure 1: Overview of proposed method. Camera position and orientation are adjusted by matching buildings between dash cam images and 2.5D maps. Map information in this paper is from OpenStreet Map[6]

.

1 INTRODUCTION

Many urban development applications require dense datasets collected from a wide city area. For example, surface textures are required for modeling 3D buildings, and human traffic in a city is analyzed to define popular paths for planning store locations. It is difficult to collect data for such analyses using a limited number of fixed cameras and people registered to perform the tracking. Therefore, data needs to be collected from moving cameras (e.g., dash cams) that film wide areas. Identifying and matching the buildings in the dash cam videos with the buildings in the maps enables the collection of texture information and analysis of the relationship between human traffic information and the map.

In this paper, we discuss matching buildings in 2.5D maps, which include the height information of structures and dash cam images. Matching in this context means identifying the pixel-by-pixel correspondence between buildings in an image and the same buildings in a map (Figure 1) to estimate the texture information of every part of the buildings. The 2.5D maps used in this research are defined in the form of Level of Detail 1 (LOD1) in CityGML[5], a format for managing urban information. Taking building height into account improves the matching accuracy in urban areas where the buildings are densely packed.

There are two challenges when it comes to matching 2.5D map information with dash cam images: handling large camera position errors and corresponding building features in images and maps to adjust the position and orientation of a camera. Regarding the first challenge, while GPS is the most popular location estimation method, it is known to generate noise at around 10 to 20 meters. This noise can be even stronger in a city with tall buildings. Regarding the second challenge, most previous studies have used building edges as a feature to match the buildings. They project building edges, extracted from the building footprint of a map, onto images and estimates the corresponding edges by image recognition for the adjustment. However, many of these methods use simple image

SIGSPATIAL '21, November 2–5, 2021, Beijing, China

ACM ISBN 978-1-4503-8664-7/21/11.
https://doi.org/10.1145/3474717.3483966

processing techniques such as image gradients[2, 4, 16], which tend to suffer from misdetection when the building textures are complex. Therefore, we propose a method that uses map matching for the GPS error relocation, deep learning for the edge extraction, and a rule-based method for the matching process.

2 RELATED WORK

First, we overview methods for correcting the position of dash cams. Map matching is used to correct the GPS trajectory [8]. This method utilizes the fact that the GPS route always goes along roadways.Another possible approach is to use the simultaneous localization and mapping (SLAM) [10] technique to estimate the movement trajectory of a camera from a captured image and then transform it into a real-world coordinate system using the GPS trajectory [13]. We utilize map matching for the position correction in our method. SLAM position estimates are highly accurate when the SLAM technique is successfully executed, we opted not to use it in this study because its operation may be unstable depending on the amount of sunshine and the surrounding geological objects at the time of shooting.

Next, we discuss methods that match buildings between images and maps. In most cases, matching is done with the detected edges or another part related to the edges. Bansal et al.[4] used the contour of the sky above buildings as the matching feature, but it is difficult to obtain the exact corners because there are other areas with large horizontal image gradients, e.g., from building textures and obstacles. Yuan et al. [16] projected the edges of buildings on a 2D map from the camera position onto the image and searched for the most likely combination of projections in an arbitrary range. However, this is not a general-purpose method because it requires most of the corners of the buildings in the image to be visible without being covered by other geographic features (e.g., poles that extend far vertically). As mentioned earlier, matching methods that use simple image processing cannot easily handle real-world complexity. Armagan et al. [1] utilized deep-learning-based semantic segmentation to build edge extraction. However, since this method extracts building edges as regions, it cannot directly make the edges calculated from the images and the maps correspond; subsequent correction must be performed by a supervised deep learning model, which requires a lot of training data. There are some methods that do not use edges, such as one by Liu et al. [9] that utilizes SLAM to create a 3D point cloud from a video and matches it with a 2.5D map to minimize the differences between the two. However, such methods that use 3D shapes cannot be used when there are obstacles such as pedestrians and automobiles in cities.

3 PROPOSED METHOD

Figure 1 shows an overview of our method. First, a map matching method is applied to narrow down the search area to the road. We then detect building edges as building features from dash cam images. The detected edges are used to adjust the position and orientation with reference to the GPS and map information. Finally, the position is adjusted by making the building edges between images and maps correspond while considering building height.

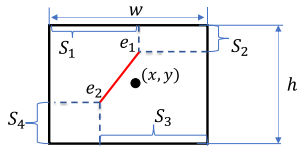

Figure 2: Illustration of proposed output variables of regression prediction net (see Figure 4). Output variables are $(x, y, w, h, \frac{S_1}{w}, \frac{S_2}{h}, \frac{S_3}{w}, \frac{S_4}{h})$. (x, y, w, h) are central coordinate, width and height of the boundary box. $e_1 = (\frac{S_1}{w}, \frac{S_2}{h})$, $e_2 = (\frac{S_3}{w}, \frac{S_4}{h})$ are relative positions of edge vertices in the boundary box.

3.1 MAP MATCHING

We used the fast map matching (FMM) method [15] to match the GPS trajectory to the most probable road. Although FMM can correct a GPS trajectory, it relocates to the closest position of the most probable road, which is not close enough to the correct position to match the buildings. Therefore, for further adjustment, we propose an edge detection model in Section 3.2 and use the edges to adjust the position and orientation in Section 3.3.

3.2 EDGE DETECTION MODEL

In this section, we introduce a deep-learning-based edge detection model. A CNN based object detection model is utilized as the base. There are several object detection models [11, 12] we could use. We added feature variables to its regression output layer to detect the edges in a bounding box. We chose EfficientDet [14] (see Figure 4) as the base model because it is a 1-stage model that is generally faster than 2-stage models (e.g., Faster-RCNN) and more accurate than models of similar size. In the regression net's output layer, we added variables e_1, e_2 that are relative positions of the edge vertices in its boundary box (see Figure 2). We derived edge coordinates from the variables, the bounding box size, and the coordinates in the post process. A class prediction net was utilized to infer the class of the edges. GPS location errors occur mainly on the horizontal plane, so we opted to classify vertical edges (see Figure 4): outside of the building (red line) and inside of the building (green line).

3.3 EDGE-BASED ADJUSTMENT

We adjusted the camera position and orientation by searching for the position that can minimize the distance between the edges detected in images (Section 3.2) and projected from maps. The building edges in a map were extracted from the corner of building footprints. We searched for the position where the edges of the images and the maps matched the most. None of the edges match each other exactly due to detection errors and map errors; so we searched for the actual camera position that would minimize the distance between pairs with the most closely matched edges. We used the Hungarian method [7] to search for the pair of edges that minimized the average distance between the pairs. The search area was limited to the roadway and the orientation direction, which are dominant errors when projecting buildings from the maps. There are three conditions related to building height. First, the detected edges can be lower than those of the maps because of obstacles such

as trees, but cannot be higher. Second, edges that do not overlap in the height direction are not considered to match. Third, edges should be close in height. As we assume that the majority of pairs will be close in height, distances between each pair are weighted by their height difference. The object function and conditions are summarized below.

Object function

$$L = \sum_{i}^{N} w(l_e^i, l_{map}^i) d(l_e^i, l_{map}^i) \tag{1}$$

Conditions

$$H - y_+(l_{map}) + \alpha > H - y_+(l_e) \tag{2}$$

$$y_+(l_{map}) < y_-(l_e), \quad y_-(l_{map}) > y_+(l_e) \tag{3}$$

$$w(l_e, l_{map}) = \left(\frac{(H - y_+(l_{map}))}{H - max(y_+(l_e), y_+(l_{map}))}\right)^{0.5} \tag{4}$$

l_e is the detected edge, l_{map} is the edge projected from the maps, H is the height of the image, α is the tolerance of the height, y_+ is the upper vertex of the edge, y_- is the lower vertex of the edge, and function d is the distance between a pair of edges. Note that these are treated in the image coordinate systems whose origin is in the upper left corner of the image.

4 EXPERIMENT

4.1 DATASET

We took footage in the Toyosu neighborhood of Tokyo with a dash cam to evaluate the building matching accuracy. The dash cam was sized 640 × 480 and faced out from the front of the vehicle. Images taken at intervals of 20m were used. We took four routes and collected a total of 221 images for the evaluation. Buildings in the test images were annotated with their building ID in the map. We tilted images 5 degrees anticlockwise since the dash cam was tilted at that angle. We collected 2.5D map information from AW3D[3]. As we want to treat this data as an LoD1 map, we altered the information so that the height of each building was equal to the tallest height of its element. We empirically set the FMM parameters to values that are suitable for our GPS module. We used OpenStreetMap's road network[6] for the matching. To train the edge detection model, we utilized 206 street view images filmed in Tokyo using the same dash cam and annotated the buildings.

4.2 DATA OBSERVATION

Figure 3 shows the GPS location and images of the route with building projection from maps based on GPS location. There were several errors located off the road and nearly 50 meters off the road at the crossroad. Even though the GPS error was not large, it is still difficult to correctly establish the edge correspondence when the buildings are densely packed. For example, buildings are projected to the relatively correct position at the bottom left in Figure 3,but it is difficult to create correct correspondences using only the edge positions because there are so many buildings in such a small area. Since the heights of buildings vary, it should be possible to improve the accuracy of correspondence creation by taking this into account.

Figure 3: GPS location and images of route with building projection from maps based on GPS location. Red lines indicate routes where the car with the dash cam droves. Blue pins indicate raw GPS data.

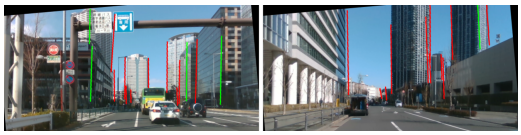

Figure 4: Example results of edge detection model. Footprint corner outside of building (red) and footprint corner inside of building (green) are detected.

4.3 RESULTS

First we utilized FMM to relocate the GPS routes to the roadway. Most of the routes were located close to the correct one, but we observed one major error on one route. The starting point of this route was located on the other road, hotel driveway, which was consistently closer to the GPS trajectory. This tends to happen at the beginning or the end of tracking, as the system does not know the beginning and end positions and it could place them anywhere.

Figure 4 shows the results of the detected edges. Most of the edges were detected correctly. The frequently mis-detected ones belonged to buildings that were reflected on other buildings' windows.

For the edge-based adjustment, we searched the position on the roadway direction in the range of [-10m, 10m] with a step size of 2.5m. The orientation was searched by [-5degree, 5degree] with a step size of 1degree. The estimated position might not be reliable when there are pairs of edges whose distance is large or when the average of the distance of all pairs is large. For this reason, we set a threshold for the adjustment, namely, 80 pixels for the distance of one pair and 40 pixels for the average of the distance of all pairs. Compared to the outside edges of the building (red line in figure 4), inside edges (green line) were relatively few that they were strongly affected by mis-detected edges. For this reason, we ignored the inside edges of the building and only used the outside edges for the adjustment.

We compared our method, GPS location fixed by the FMM and edge-based adjustment considering building height (DNN+ HEIGHT), to projecting the buildings from the GPS location (baseline), having the location fixed by the FMM (FMM), and having the location fixed by the FMM and the edge-based adjustment without considering building height (DNN). We also compared a model that detects edges by image gradient for the edge-based adjustment (IMAGE-GRADIENT). This model detects edges where the mean intensity

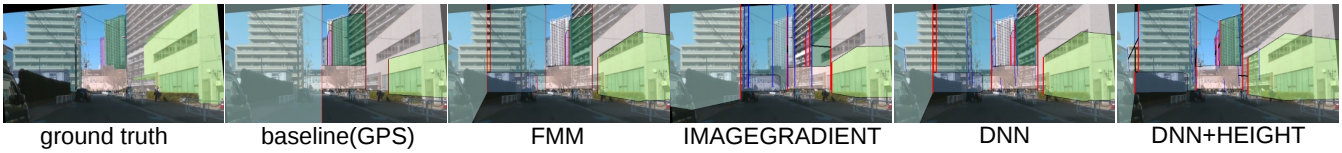

Figure 5: Example results of building projection of each model. Each building is represented by distinct colors.

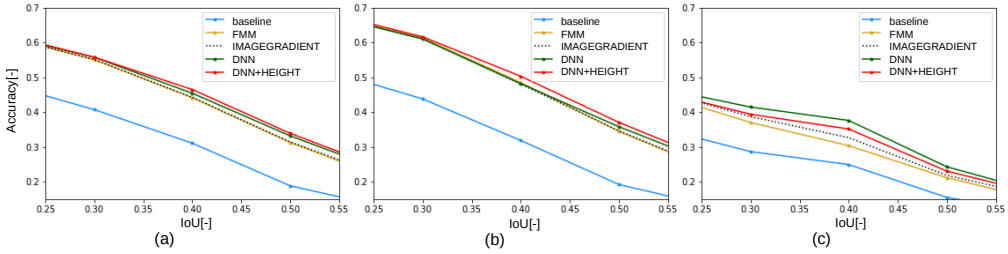

Figure 6: Average of matching accuracy. (a):all section, (b):straight section, (c):curved section

gradient of the horizontal direction is over a certain threshold. Figure 5 shows a example result of each method. We can see here that considering the building height leads to a better accuracy in the area with dense buildings.

Next, to quantitatively assess the accuracy of our method, we calculated the area of overlap between the projected buildings and the ground truth, i.e. the Intersection over Union (IoU). We then calculated the ratio of buildings that exceeded a certain IoU (=0.3, 0.4, 0.5) in each image. Note that this comparison with the ground truth considers both the building segmentation and the building ID, as our method is intended for building identification. This evaluattion was done with buildings larger than 80×60 (i.e., 12.5%×12.5%) in the image, as we are mostly interested in buildings that appear large enough to observe in the image. Figure 6(a) shows the average matching accuracy in all sections. Our method (DNN+HEIGHT) was the most accurate method at every IoU threshold in this case. Figure 6(b) shows the average matching accuracy in straight sections. Only our method reached 50% accuracy at IoU=0.4. The matching accuracy significantly decreased at curved sections, as shown in Figure 6(c). Since the scenery changes drastically at curved sections, it is necessary to search with finer steps in both the position and the orientation. One reason the matching accuracy did not reach a high value at a higher IoU threshold (>0.5) is that the buildings were also projected onto the objects located in front of them. To limit the projection to only the visible building region, matching the projection with instance segmentation may be a solution.

5 CONCLUSION

To match the buildings between dash cam images and maps, we proposed a method that performs robust edge-aware matching for complex textures and deals with positional noise by map matching and edge-based correction. We demonstrated how difficult edge-aware building matching is in dense urban areas by analyzing real data. We also showed that height should be considered even after correcting the map matching when buildings are close together.

REFERENCES

[1] Anil Armagan, Martin Hirzer, Peter M Roth, and Vincent Lepetit. 2017. Learning to align semantic segmentation and 2.5 d maps for geolocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3425–3432.
[2] Clemens Arth, Christian Pirchheim, Jonathan Ventura, Dieter Schmalstieg, and Vincent Lepetit. 2015. Instant outdoor localization and slam initialization from 2.5 d maps. *IEEE Transactions on Visualization & Computer Graphics* 21, 11 (2015), 1309–1318.
[3] AW3D. [n.d.]. Retrieved 2021 from https://www.aw3d.jp/en/
[4] Mayank Bansal and Kostas Daniilidis. 2014. Geometric urban geo-localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3978–3985.
[5] CityGML. [n.d.]. Retrieved 2021 from https://www.ogc.org/standards/citygml
[6] OpenStreetMap contributors. 2017. Planet dump retrieved from https://planet.osm.org. https://www.openstreetmap.org.
[7] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2, 1-2 (1955), 83–97.
[8] Yang Li, Qixing Huang, Michael Kerber, Lin Zhang, and Leonidas Guibas. 2013. Large-scale joint map matching of GPS traces. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 214–223.
[9] Ruyu Liu, Jianhua Zhang, Shengyong Chen, and Clemens Arth. 2019. Towards SLAM-based outdoor localization using poor GPS and 2.5 D building models. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 1–7.
[10] Raul Mur-Artal and Juan D Tardós. 2017. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics* 33, 5 (2017), 1255–1262.
[11] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).
[12] Mingxing Tan, Ruoming Pang, and Quoc V Le. 2020. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10781–10790.
[13] Shinji Umeyama. 1991. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 13, 04 (1991), 376–380.
[14] Yongchao Xu, Mingtao Fu, Qimeng Wang, Yukang Wang, Kai Chen, Gui-Song Xia, and Xiang Bai. 2020. Gliding vertex on the horizontal bounding box for multi-oriented object detection. *IEEE transactions on pattern analysis and machine intelligence* 43, 4 (2020), 1452–1459.
[15] Can Yang and Gyozo Gidofalvi. 2018. Fast map matching, an algorithm integrating hidden Markov model with precomputation. *International Journal of Geographical Information Science* 32, 3 (2018), 547–570.
[16] Jiangye Yuan and Anil M. Cheriyadat. 2016. Combining Maps and Street Level Images for Building Height and Facade Estimation. *UrbanGIS '16 Proceedings of the 2nd ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics*. arXiv:1601.07630 [cs.CV]