Analyzing the impact of semantic LoD3 building models on image-based vehicle localization

Antonia Bieringer[1], Olaf Wysocki[1], Sebastian Tuttas[3], Ludwig Hoegner[4], Christoph Holst[1,2]

[1] Professorship of Photogrammetry and Remote Sensing, [2] Chair of Engineering Geodesy,
TUM School of Engineering and Design, Technical University of Munich, 80333 Munich, Germany
- (antonia.bieringer, olaf.wysocki, christoph.holst)@tum.de
[3] 3D Mapping Solutions GmbH, 83607 Holzkirchen, Germany - sebastian.tuttas@3d-mapping.de
[4] Hochschule München University of Applied Sciences, 80335 Munich, Germany - ludwig.hoegner@hm.edu

Keywords: LoD2, LoD3, semantic 3D city models, car localization, map-based localization, image-based positioning

Abstract

Numerous navigation applications rely on data from global navigation satellite systems (GNSS), even though their accuracy is compromised in urban areas, posing a significant challenge, particularly for precise autonomous car localization. Extensive research has focused on enhancing localization accuracy by integrating various sensor types to address this issue. This paper introduces a novel approach for car localization, leveraging image features that correspond with highly detailed semantic 3D building models. The core concept involves augmenting positioning accuracy by incorporating prior geometric and semantic knowledge into calculations. The work assesses outcomes using Level of Detail 2 (LoD2) and Level of Detail 3 (LoD3) models, analyzing whether facade-enriched models yield superior accuracy. This comprehensive analysis encompasses diverse methods, including off-the-shelf feature matching and deep learning, facilitating thorough discussion. Our experiments corroborate that LoD3 enables detecting up to 69% more features than using LoD2 models. We believe that this study will contribute to the research of enhancing positioning accuracy in GNSS-denied urban canyons. It also shows a practical application of under-explored LoD3 building models on map-based car positioning.

1. Introduction

Navigating in complex urban environments can pose challenges for positioning systems, primarily when vehicles depend on GNSS. As vehicles move through signal-obstructed urban canyons, the effectiveness of positioning systems can diminish. Consequently, vehicles shall turn to alternative cues to determine their location in such scenarios.

Various methods have been developed to address this issue, combining different sensors to enhance location accuracy. While GNSS enables global positioning, supporting it with local cues is an intuitive choice. For example, directly enhancing GNSS signals by using 3D city models (Dreier et al., 2021). More frequently, however, the researchers have focused on utilizing cameras for enhanced positioning owing to their abundance on cars and optical features.

In order to localize the vehicle with optical cameras, features have to be found in the images. There are various techniques for feature finding and matching, e.g., SIFT (Lowe, 2004), SURF (Bay et al., 2008), or ORB (Rublee et al., 2011). Over time, researchers compared those methods against each other (Bansal et al., 2021) and improved them. However, to globally position the images, a reference 3D map with the correct scale and position shall be used.

Such 3D map can be extracted directly from governmental and proprietary geoportals (Lucks et al., 2021), where semantic 3D building models are available. These models are used in different Level of Details (LoD). LoD1 models are limited to geometrical information that includes the vertices and edges of the buildings; they are cuboid polyhedral models with flat roofs. LoD2 models are more detailed and include the generalized structure of the roof. In comparison, LoD3 is enhanced by facade details such as windows and doors (Biljecki et al., 2016).

A few papers have already investigated the usage of LoD1 and LoD2 models for vehicle localization. One of the examples of utilizing LoD1 and LoD2 models is the paper of Vogel et al., who introduce a localizing approach in the inner-city outdoor environments using 3D city models (Vogel et al., 2018). To this end, they modify an iterated extended Kalman filter to be able to integrate additional information into it, e.g., available maps. A more reliable and precise georeferencing is achieved since geometric circumstances are considered (Vogel et al., 2018). Their idea is refined in later papers of (Moftizadeh et al., 2021, Lucks et al., 2021). However, the usage of LoD3 models facade features for localization has yet to be investigated.

Our contributions are as follows:

- Map-supported vehicle localization by using high fidelity LoD3 models and images.
- Harmonizing two distinct modalities: Visual information encapsulated within optical images and the structure of semantic 3D models' non-textured surfaces.
- Comparing the benefits of LoD3 over LoD2 building models for camera localization.

2. Related Work

We devote our work to analyzing the impact of image-based vehicle positioning using semantics of 3D maps. Therefore, in this Section, we present the state-of-the-art in semantic 3D city models and image- and map-based positioning.

2.1 Semantic 3D city models

At city, regional, and national scales, semantic 3D city models offer comprehensive representations of structures, taxonomies,

and aggregations. Semantic 3D city models are frequently described by the internationally adopted CityGML standard, established by the Open Geospatial Consortium (OGC) (Gröger et al., 2012); This data model standard adopts two encodings using either Geography Markup Language (GML) or CityJSON (Kutzner et al., 2020, Ledoux et al., 2019). The CityGML standard enables the modeling of urban objects with 3D geometry, appearance, topology, and semantics across four different Levels of Detail (LoD).

Building descriptions are pivotal to semantic 3D city models due to the foundational role of urban dwellings in cities (Biljecki et al., 2015). LoD1 and LoD2 building models, currently prevalent, boast around 220 million models available in countries such as Germany, Japan, the Netherlands, Switzerland, the United States, and Poland[1]. While LoD1 building models are represented by prismatic 3D models of height-extruded building footprints, LoD2 models additionally represent complex roof structures. The widespread adoption of these models is attributed to robust 3D reconstruction algorithms and the availability of building footprints combined with aerial observations (Roschlaub and Batscheider, 2016, Haala and Kada, 2010).

However, LoD1 and LoD2 models lack a more detailed description of facade elements. Addressing this gap are LoD3 building models featuring descriptive facades with objects such as windows, doors, balconies, and underpasses (Wysocki et al., 2022). The current practice shows that these highly-detailed objects are manually modeled (Uggla et al., 2023, Chaidas et al., 2021). Yet, a great deal of research has been devoted to enabling automatic LoD3 reconstruction, which has sparked advancements in the LoD3 data adoption [1]. The recently developed LoD3 reconstruction methods promise to increase the availability of such models (Wysocki et al., 2023, Hoegner and Gleixner, 2022, Huang et al., 2020).

2.2 Image-Based Vehicle Positioning

One way of reconstructing the 3D structure of a scene and estimating the camera poses from a collection of 2D images is Structure from Motion (SfM) (Schönberger and Frahm, 2016). This technique works by leveraging the relationships between features in different images and the scene's geometry to recover the scene's 3D structure and camera pose.

Another approach of estimating the vehicle's position by using a series of camera images is Visual Odometry (VO). It is advantageous over other methods due to its insensitivity to soil mechanics and lower drift rates (Howard, 2008). Additionally, it performs well in places where a satellite connection is limited, e.g., indoors or on narrow streets. However, the trajectories estimated by visual odometry tend to drift along with increasing distance from the start point. Consequently, researchers investigated to improve the concept by expanding VO ideas to another image-based localization method called SLAM (Gao and Zhang, 2019).

Simultaneous Localization and Mapping (SLAM) reconstructs reality from observation data, e.g., images and points clouds. Compared to SfM and VO, it computes the trajectory in real time and maps the vehicle's environment (Gao and Zhang, 2019). The map is updated while driving (Rehder and Albrecht, 2015), and consequently, the current point is automatically corrected by previously calculated results. A prerequisite for the map-supported SLAM are surroundings that are easy to detect and can be used to scale and correctly map the trajectory (Hungar et al., 2020). As Lucks et al. show, the prior 3D building map can enhance the SLAM-based calculated trajectory (Lucks et al., 2021).

3. Methodology

In Figure 1, the workflow of the developed method is shown. First, virtual pictures of the building models are created. The resulting images are then used to find features in the corresponding optical images. For this purpose, the ORB detector (Rublee et al., 2011) is used. The generated 2D-image coordinates are correlated with a 3D coordinate by benefiting from the LoD model semantics and geometry. In the end, these coordinates are needed to estimate the camera position for every single image with spatial resection. This workflow is designed for LoD1, LoD2 and LoD3 building models. The developed method's code is published in the GitHub repository[2].

3.1 Creating Virtual Images of the LoD Models

For the first step, the virtual images are created by using ray casting. To work with the ray casting algorithm, a polyhedral building model is tranformed into a mesh model using triangulation since all results are based on the hit of a triangle by virtual rays. The algorithm creates those virtual rays and sends them through a virtual camera, preferably in the direction of the model. Every time a ray hits a point, the corresponding triangle and the barycentric coordinates of the exact hit point in

[1] https://github.com/OloOcki/awesome-citygml

[2] https://github.com/tum-pf/LoD3ForLocalization

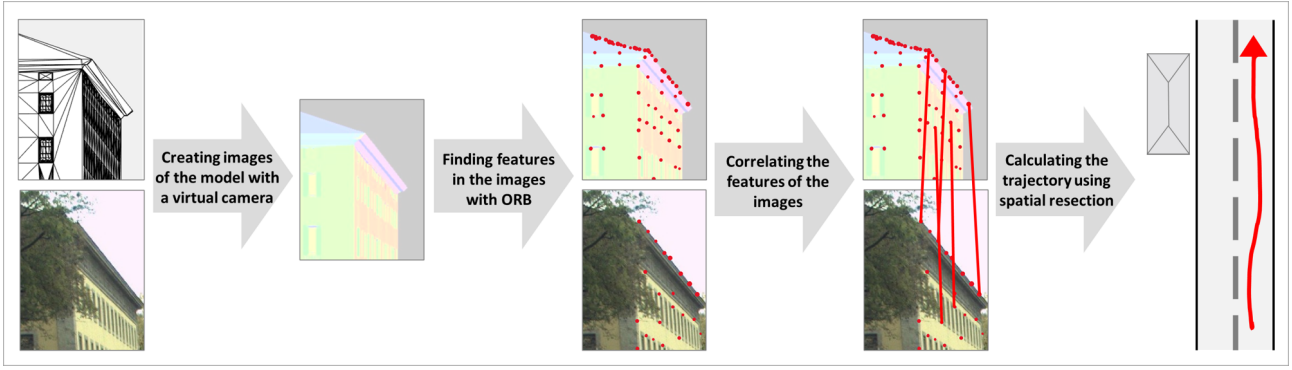

Figure 1. Developed workflow: The images of the real world are matched with virtual images of the LoD models. The combination of both will be used to calculate the trajectory.

the triangle are saved. Therefore, the 3D coordinates in the real world can be calculated by only knowing the 2D coordinates of the virtual image. The calculation is described in Section 3.3.

Moreover, the virtual camera's pose is required (see Figure 2). As an approximation solution, the position is taken from the GNSS data of the current corresponding optical image. On top, the camera's height above the GNSS antenna must be considered, leading to GNSS values adapted to the camera position. The camera is pointing to the adapted GNSS position of the consecutive image. Afterward, the orientation of the camera requires to be adjusted: The angles for roll, pitch, and yaw are required to approximate the point close to the GNSS position in consideration of the camera tilt. A visual example of pitch can be seen in Figure 2. The approximation is calculated as follows:

$$dy[m] = r_{GNSS} \cdot dy[^\circ] \cdot \frac{\pi}{180^\circ} \tag{1}$$

An overview of the above-described virtual setup can be seen in Figure 2.

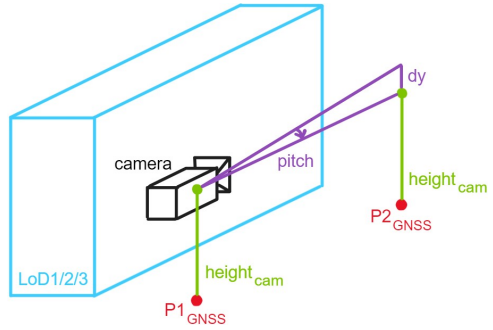

Figure 2. Virtual setup for ray casting.

After applying the ray casting algorithm, the results are available in a tensor. There are five results: the distance, the geometry IDs, the primitive IDs, the primitive normals, and the barycentric coordinates. For an example of LoD2 combined with one LoD3 building, see Figure 3:

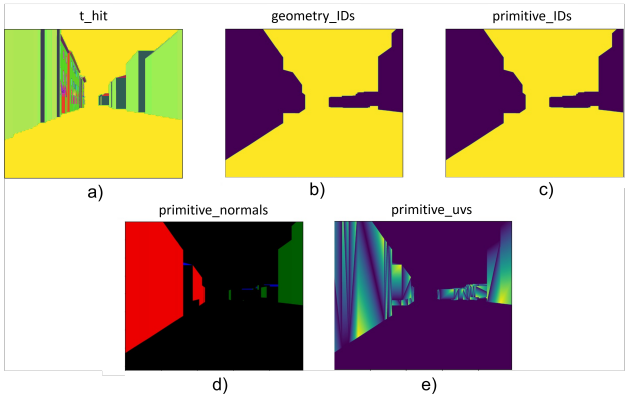

Figure 3. Ray casting results for LoD3 a) hit distance b) geometry IDs c) primitive IDs d) primitive normals e) barycentric coordinates.

From these results, the primitive-normals-data is saved as an image. As shown in Figure 3d, some walls of the buildings have the same color as the background. To make every plane of the building visible, the values of the primitive normals are adjusted so that only the absolute values are considered since the exact value is unimportant for the virtual images. This adjustment is necessary so that every side of the building has a unique color. Otherwise, some buildings' planes will not be considered during feature matching. Note that the LoD3 building in the virtual image in Figure 3 is more detailed due to the additional information on the model, especially regarding windows and roof eaves.

3.2 Feature Matching

An overview of the approach is visualized in Figure 4. The feature matching is performed on images with reduced details. Therefore, the images become more abstract and only focus on important information. Then, the images are matched against themselves: the virtual and the optical image.

The next step is done for the image of the real world, as well as for the one of the virtual camera. First, another abstract, virtual feature image is created and implemented as a matrix with the same values for every pixel. If features were found in matching the images against themselves, those pixels are colored differently. This is realized by assigning a different value to this pixel in the matrix. The image, therefore, only contains information about the essential points of the image before.

Afterward, these newly created feature images are matched with an ORB-feature detector again.

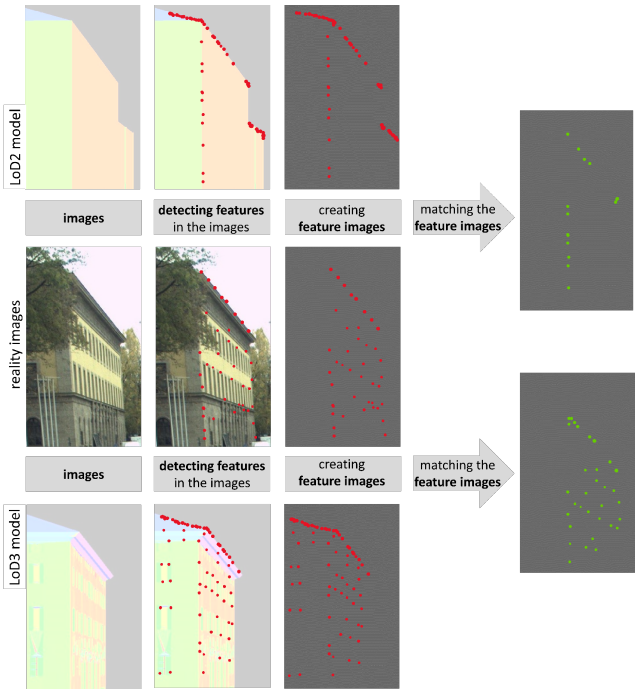

Figure 4. Workflow of the generation of the feature images.

3.3 Correlating the Features with a 3D-Coordinate

During ray casting, every hit of the mesh by a ray is saved. In this case, the corresponding triangle IDs are stored. Then, the corresponding triangles in the mesh are found, which enables the extraction of which vertices belong to a triangle that was hit. From the IDs of those vertices, the 3D coordinates from the vertex positions are extracted in the mesh. Finally, the exact point needs to be weighted by the vertices of its triangle because only the relative positions from the vertices are known (Xia et

al., 2022). Those relative positions are named u, v, s. The results only return two barycentric coordinates for each hit point since the sum of the three coordinates equals one:

$$1 = u + v + s \tag{2}$$

This means that s can be directly derived from u and v. The weighting to receive the final coordinates works as follows:

$$P = u * P_{vertex_1} + v * P_{vertex_2} + s * P_{vertex_3} \tag{3}$$

3.4 Calculating the Trajectory

The vehicle's trajectory is simultaneously calculated by driving and reading the generated images. To this end, we leverage the spatial resection concept. Spatial resection is a set of statistical formulas to compute the accuracy of observations by finding a model that fits them best. Consequently, the accuracy of the observations and related values can be calculated afterward. In this case, the camera position is estimated while the observations are optimized at the same time. The position calculation is performed with the help of the photogrammetrical co-linearity equations, which can be seen in Equation 4 and Equation 5.

$$x = \hat{x_0} + z\frac{(\hat{r_{11}}(X - \hat{X_0}) + \hat{r_{21}}(Y - \hat{Y_0}) + \hat{r_{31}}(Z - \hat{Z_0})}{(\hat{r_{13}}(X - \hat{X_0}) + \hat{r_{23}}(Y - \hat{Y_0}) + \hat{r_{33}}(Z - \hat{Z_0})} \tag{4}$$

$$y = \hat{y_0} + z\frac{(\hat{r_{12}}(X - \hat{X_0}) + \hat{r_{22}}(Y - \hat{Y_0}) + \hat{r_{32}}(Z - \hat{Z_0})}{(\hat{r_{13}}(X - \hat{X_0}) + \hat{r_{23}}(Y - \hat{Y_0}) + \hat{r_{33}}(Z - \hat{Z_0})} \tag{5}$$

For the calculation, six values are to be determined, representing the camera's position (X_0, Y_0, Z_0) and orientation (ω, ϕ, κ). Consequently, for the solution, at least six corresponding points are required. More corresponding points are preferred to get even better accuracy.

The spatial resection is performed as follows. A direct linear transformation generates the approximate camera orientation values. Input parameters are at least six corresponding points. The approximate camera position is taken from the corresponding GNSS point. The approximate values are required because the spatial resection needs starting points to optimize. Otherwise, the algorithm might run into a local minimum and keeps iterating in an infinite loop.

In the optimization loop, the derivations of the A-matrix are computed. This is necessary because the refinements of the observations for the best-fitting model are calculated with the help of this matrix. The A-matrix has the size of [n x m], while n is the number of observations and m is the number of values to be determined.

The values that are to be determined are estimated in every iteration by calculating the following:

$$\delta\hat{x} = \frac{(A^T * P_{bb} * A)}{(A^T * P_{bb} * w)} \tag{6}$$

Where A is the already described A-matrix. P_{bb} is a matrix that contains weights. Those weights decide how important an observation is, e.g., if the measurement is probably more precise than others because of the geometrical position. The w-vector contains the current errors between the approximated $\delta\hat{x}$ and the actual observations. The optimized observation values are computed by:

$$\delta\hat{v} = A * \delta\hat{x} - w \tag{7}$$

4. Experiments

The developed method is evaluated in three different test areas, which are visualized in Figure 5.

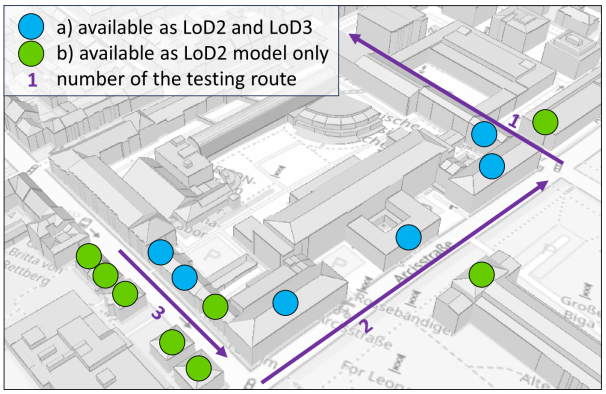

Figure 5. Visualization of the test area divided into three sub-areas (purple) with different availability of building models: a) both in LoD2 and LoD3 (blue), and b) solely in LoD2 (green).

Notably, there were no instances in the test area where LoD3 models were present on both sides of the street, which mimics a frequently present real-world situation. The missing LoD3 models were supplemented from the LoD2 dataset as a solution. Therefore, the same models from the LoD2 dataset were used for experiments involving solely LoD2 models. This hybrid approach ensures a balanced distribution of potential reference points and promotes an effective configuration. Owing to the experimental setup, we were able to directly evaluate the influence of the LoD3 over to the usage of only LoD2 models.

The test site was located in Munich, Germany, and represented a Central European architecture comprising a mixed residential area and a university campus. The images were acquired by a mobile mapping vehicle of 3D Mapping Solutions GmbH (3D Mapping Solutions GmbH, 2024). The LoD2 models were downloaded through the governmental open-geodata portal of Bavaria, Germany (Bayerische Vermessungsverwaltung, 2023), whereas the LoD3 models were developed within the interdisciplinary tun2twin project (Barbosa et al., 2023).

4.1 Different Approaches for Feature Matching

The feature matching was performed on five different pillars. Each approach is described in the following subsections.

4.1.1 Corresponding Image-Pairs of Different Types

This approach aims to find features from image pairs that are not the same type. This means that one virtual image was matched with one image of reality. Both images were taken from the same GNSS point. The results show only a median of 9-18 suitable feature matches for each image pair.

4.1.2 Segmented Images with Deep Learning

Matching real images with virtual images presents a challenge due to disruptive elements not being shared between the two image types. To mitigate this issue, the approach involves segmenting the real images. The used algorithm was built on the SegFormer network (Xie et al., 2021), using the MMSegmentation framework (OpenMMLab, 2023). This algorithm generated an output image where distinct elements (e.g., buildings) were each assigned a unique color.

To match these semantically segmented images, a method is devised wherein the segmented images act as masks for the real images to isolate the building areas from the real images. Consequently, only the building-related parts of the real image are retained and matched with their virtual counterparts. This strategic approach effectively minimizes any potential disruptions from extraneous objects that could potentially result in erroneous matches.

4.1.3 Sobel-Filter and Canny Edge Detector

An established approaches for feature matching focuse solely on features along object edges, in this case, including structures such as window edges, e.g., Sobel filters and Canny edge detector (Singh and Singh, 2015).

In instances where it was desirable to exclusively match buildings, applying a bounding box to actual images proved effective. These filters were harmonized with the segmented images detailed in Section 4.1.2 to ensure precision. This strategic combination ensured that only relevant objects for vehicle localization were considered.

5. Results and Discussion

The experiments were evaluated by comparing the number of found features and the standard deviation of the camera position. For every table, the corresponding gain was computed as a percentage increase or decrease between LoD2 and LoD3.

5.1 Number of found features

The different approaches led to different results regarding the number of found features, which represented those that can be successfully matched between two images. The selection pertains to the median of all tested images within each area. The best result for each experiment is written in bold, comparing LoD2 and LoD3. An example of the found features in the images is visualized in Figure 6.

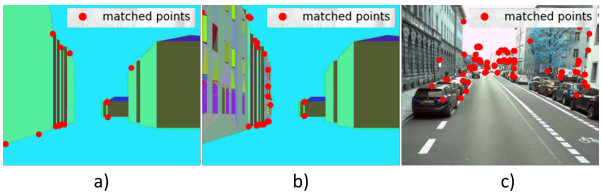

Figure 6. Matched features a) LoD2 b) LoD3 c) optical image.

A comprehensive comparison for each model, namely LoD2 and LoD3, is presented across different test areas in Tables 1, 2, and 3.

As our experiments corroborate, the LoD3 models, on average, result in the same or more found features between the images.

Table 1. Number of Found Features (Median) and Gain in the Test Area 1.

	LoD2	LoD3	gained
Corresponding images	17	**18**	6%
Feature images	20	**50**	60%
Sobel-filter	**14**	13	-8%
Canny edge detection	21	21	0%
Mask	18	18	0%
Mask and Sobel-filter	**21**	11	-91%
Mask and Canny edge detection	**34**	22	-55%

Table 2. Number of Found Features (Median) and Gain in the Test Area 2.

	LoD2	LoD3	gained
Corresponding images	**12**	9	-33%
Feature images	20	**65**	69%
Sobel-filter	10	10	0%
Canny edge detection	8	**14**	43%
Mask	**15**	12	-25%
Mask and Sobel-filter	13	**14**	7%
Mask and Canny edge detection	21	**27**	22%

Table 3. Number of Found Features (Median) and Gain in the Test Area 3.

	LoD2	LoD3	gained
Corresponding images	15	**18**	17%
Feature images	20	**47**	57%
Sobel-filter	13	**15**	13%
Canny edge detection	14	**20**	30%
Mask	13	**20**	35%
Mask and Sobel-filter	15	**17**	12%
Mask and Canny edge detection	**26**	21	-24%

We deem the feature images algorithm as the most reliable feature extraction between semantic 3D building models and optical images, as it scored more features on average in the three test areas (60%, 69%, and 58%, respectively).

In the test area 1 with prevalent LoD2 models, the algorithm matched features for LoD2 until the point where the whole virtual image was facing the model-absent underpass. Then, no feature matching was possible again and the algorithm failed. This happens due to the frequently absent underpasses in LoD2, where instead a vertical wall is rendered to the ground level, as seen in Figure 7.

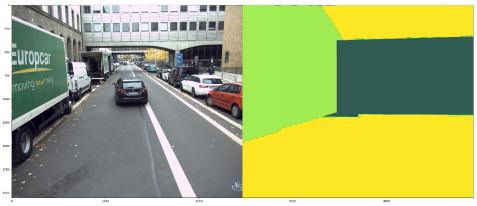

Figure 7. Visualization of an absent underpass in LoD2.

Especially in test area 2, more features lead to a more reliable estimation of the trajectory point: up to 69% more features (see Table 2).

As described in Section 4, test area 3 exhibited the optimal situation for localizing the vehicle with the help of high-detail polyhedral models. This can be seen in Table 3, where 57% more features were found, and seven of the tested methods achieved an increase of found features. We also observe that the higher the buildings, the more details can be found for feature matching. Furthermore, narrow streets are advantageous because the

buildings are closer to the camera and can be captured in more detail. In contrast, GNSS data is most accurate in urban environments when surrounded by only low buildings; narrow streets lead to positionining deviations. Consequently, the localization with images and LoD3 models can be advantageous over using GNSS in urban areas. Both methods are like counterparts: where one method fails, the other has advantages.

5.2 Positioning accuracy

To analyze the advantage can be achieved by LoD3 models, Tables 4, 5, 6 show the accuracy of the estimated camera position in comparison.

Table 4. Standard Deviation of the Camera Position (Median) and Gain in Test Area 1.

		LoD2	LoD3	gained
Corresponding images	σ_X	35.81m	**15.26m**	57%
	σ_Y	**36.39m**	46.10m	-27%
	σ_Z	44.25m	**21.11m**	52%
Feature images	σ_X	**81.72m**	111.87m	-37%
	σ_Y	**50.40m**	152.39m	-202%
	σ_Z	**74.92m**	94.45m	-26%
Sobel-filter	σ_X	66.13m	**51.23m**	23%
	σ_Y	67.34m	**58.56m**	13%
	σ_Z	65.85m	**31.64m**	52%
Canny edge detection	σ_X	48.71m	**35.82m**	26%
	σ_Y	50.27m	**47.18m**	6%
	σ_Z	52.06m	**27.31m**	48%
Mask	σ_X	**19.93m**	56.26m	-182%
	σ_Y	**32.32m**	44.57m	-38%
	σ_Z	**29.15m**	48.74m	-67%
Mask and sobel-filter	σ_X	**55.96m**	65.78m	-18%
	σ_Y	74.12m	**23.43m**	68%
	σ_Z	59.12m	**42.22m**	29%
Mask and canny edge detection	σ_X	59.45m	**38.27m**	36%
	σ_Y	92.93m	**55.16m**	41%
	σ_Z	50.31m	**42.67m**	15%

Table 5. Standard Deviation of the Camera Position (Median) and Gain in Test Area 2.

		LoD2	LoD3	gained
Corresponding images	σ_X	48.02m	**17.49m**	64%
	σ_Y	44.22m	**41.67m**	6%
	σ_Z	58.98m	**49.33m**	16%
Feature images	σ_X	**130.94m**	168.78m	-29%
	σ_Y	191.27m	**169.88m**	11%
	σ_Z	194.87m	**158.22m**	19%
Sobel-filter	σ_X	93.47m	**21.96m**	77%
	σ_Y	76.02m	**26.54m**	65%
	σ_Z	51.87m	**15.89m**	69%
Canny edge detection	σ_X	**43.62m**	48.33m	-11%
	σ_Y	**24.39m**	52.87m	-117%
	σ_Z	38.84m	**33.06m**	15%
Mask	σ_X	**9.62m**	13.77m	-43%
	σ_Y	**13.59m**	25.69m	-89%
	σ_Z	14.63m	**10.12m**	31%
Mask and sobel-filter	σ_X	**67.45m**	94.36m	-40%
	σ_Y	91.99m	**85.56m**	7%
	σ_Z	96.01m	**80.71m**	16%
Mask and canny edge detection	σ_X	**37.74m**	42.08m	-11%
	σ_Y	**40.91m**	58.02m	-42%
	σ_Z	40.69m	**27.19m**	33%

Regarding test area 1, the accuracy of the feature images algorithm drops to [-37%, -202%, -26%]. This shows, that the algorithm is not yet working perfectly for every situation. The configuration with the underpass leads also in LoD3 to problems in the accuracies. Still, other methods show the advantages of LoD3 also in this case. The tackling of the problems is described in Section 5.3.

Table 6. Standard Deviation of the Camera Position (Median) and Gain in Test Area 3.

		LoD2	LoD3	gained
Corresponding images	σ_X	71.37m	**40.05m**	44%
	σ_Y	76.23m	**66.23m**	13%
	σ_Z	80.26m	**77.21m**	4%
Feature images	σ_X	171.01m	**21.14m**	88%
	σ_Y	127.70m	**38.17m**	70%
	σ_Z	104.11m	**21.44m**	79%
Sobel-filter	σ_X	59.78m	**40.71m**	32%
	σ_Y	**43.54m**	65.81m	-51%
	σ_Z	42.67m	**35.07m**	18%
Canny edge detection	σ_X	24.98m	**9.35m**	63%
	σ_Y	33.31m	**11.62m**	65%
	σ_Z	19.12m	**6.07m**	68%
Mask	σ_X	38.90m	**20.10m**	48%
	σ_Y	**33.51m**	60.34m	-80%
	σ_Z	40.14m	**27.97m**	30%
Mask and sobel-filter	σ_X	**21.05m**	33.62m	-60%
	σ_Y	**23.25m**	47.88m	-106%
	σ_Z	**19.21m**	44.98m	-134%
Mask and canny edge detection	σ_X	159.73m	**7.83m**	95%
	σ_Y	168.19m	**13.24m**	92%
	σ_Z	180.55m	**5.19m**	97%

The developed algorithm works less accurately in urban areas when buildings are available on only one side of the street, see Tables 2 and 5 (test area 2), especially for the Feature images algorithm.

As seen in Tables 1, 2 and 3, LoD3 reaches a higher number of features. More found features provide a better distribution in the image and, therefore, result in a better standard deviation of the camera point, e.g., up to [88%, 70%, 79%] more accurate in test area 3 (see Table 3). Considering the standard deviation of the camera position for the evaluation, LoD3 is favorable in test area 3.

5.3 Limitations

The developed algorithm shows the advantages of using LoD3 over LoD2 models. Still, the deviations of the estimated point from the corresponding GNSS point are large. To use those models for positioning, the feature matching has to be optimized so that matched features refer to exactly one single point in the real world. This is not the case for all matched features so far.

In general, the deviation of the calculated camera position performs slightly better in LoD3 compared to LoD2. A reason for the deviation from the GNSS point is an imprecise calculation of the 3D coordinates. The exact coordinates are weighted by the barycentric coordinates in the triangle. Consequently, each triangle vertex belongs to exactly one of the barycentric coordinates u, v, s. This depends on the sequence in which the triangle coordinates are saved in the mesh. The order of the vertices is a requirement currently investigated for LoD2 and LoD3 models. The topology for such models has to be consistent in every model, e.g., in the right-hand rule. This is not yet common for all developed LoD3 models since the automatic generation remains in its infancy and is prone to manual, randomly induced errors. These errors lead to an imprecise calculation of the 3D coordinates and, therefore, to a deviation of the calculated camera position from the GNSS point. The method developed in this paper works assuming that the models are correct and the vertices are saved in the right-hand rule.

6. Conclusion

In this paper, we propose an approach to compare the performance of LoD2 against LoD3 models for map- and image-based vehicle positioning. Using a feature image method, we harmonize semantic 3D city models with optical images. Our experiments corroborate that although we had different representations, optical images vs. quasi-optical images of the LoD models, our method performs well in feature matching. The presented method performs best when buildings cover both sides of the street. We also observe that the higher the building models, the more details can be found for feature matching. Additionally, narrow streets are advantageous because the buildings are closer to the camera and can be captured in more detail. We are convinced our image and model-based navigation algorithm can inspire upcoming work. In the future, we plan to improve upon measurements, e.g., by using SLAM. With this, we hope to decrease the large deviations in our calculations from the GNSS points. Additionally, we will work on reducing false feature matches to achieve higher accuracy. We also plan to do more experiments on more datasets and aim to expand the LoD3 datasets.

Acknowledgment

The authors would like to thank 3D Mapping Solutions GmbH for providing the images to develop and test the algorithm. This work was supported by the Bavarian State Ministry for Economic Affairs, Regional Development and Energy within the framework of the IuK Bayern project *MoFa3D - Mobile Erfassung von Fassaden mittels 3D Punktwolken*, Grant No. IUK643/001. Moreover, the work was conducted within the framework of the Leonhard Obermeyer Center at the Technical University of Munich (TUM). We gratefully acknowledge the tum2twin team at the TUM for the valuable insights and for providing the CityGML datasets.

References

3D Mapping Solutions GmbH, 2024. https://www.3d-mapping.de/en/about-us/our-divisions/data-acquisition/unser-vermessungssystem/. Accessed: 2024-01-31.

Bansal, M., Kumar, M., Kumar, M., 2021. 2D object recognition: a comparative analysis of SIFT, SURF and ORB feature descriptors. *Multimedia Tools and Applications*, 18839—18857.

Barbosa, J., Schwab, B., Wysocki, O., Heeramaglore, M., Huang, X., 2023. https:// github.com/tum-gis/tum2twin. Accessed: 2023-08-10.

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., 2008. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 346–359.

Bayerische Vermessungsverwaltung, 2023. https://geodaten.bayern.de/opengeodata. Accessed: 2023-08-10.

Biljecki, F., Ledoux, H., Stoter, J., 2016. An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, 59, 25-37.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., Çöltekin, A., 2015. Applications of 3D city models: State of the art review. *ISPRS International Journal of Geo-Information*, 4(4), 2842–2889.

Chaidas, K., Tataris, G., Soulakellis, N., 2021. Seismic Damage Semantics on Post-Earthquake LOD3 Building Models Generated by UAS. *ISPRS International Journal of Geo-Information*, 10(5).

Dreier, A., Zimmermann, F., Klingbeil, L., Holst, C., Kuhlmann, H., 2021. Strategien zur Selektion von Satelliten in kinematischen GNSS-Anwendungen auf Basis von 3D-Umgebungsmodellen. *allgemeine vermessungsnachrichten (AVN)*, 128(1), 13–22.

Gao, X., Zhang, T., 2019. *Introduction to Visual SLAM: From Theory to Practice*. Springer Nature Singapore Pte Ltd., Singapore.

Gröger, G., Kolbe, T. H., Nagel, C., Häfele, K.-H., 2012. OGC City Geography Markup Language CityGML Encoding Standard. Open Geospatial Consortium: Wayland, MA, USA, 2012.

Haala, N., Kada, M., 2010. An update on automatic 3D building reconstruction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(6), 570 - 580.

Hoegner, L., Gleixner, G., 2022. Automatic extraction of facades and windows from MLS Point clouds using voxelspace and visibility analysisautomatic extraction of facades and windows from MLS Point clouds using voxelspace and visibility analysis. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2022, 387–394.

Howard, A., 2008. Real-time stereo visual odometry for autonomous ground vehicles. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 1–7.

Huang, H., Michelini, M., Schmitz, M., Roth, L., Mayer, H., 2020. LOD3 building reconstruction from multi-source images. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2020, 427–434.

Hungar, C., Jürgens, S., Wilbers, D., Köster, F., 2020. Map-based localization with factor graphs for automated driving using non-semantic lidar features. *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 1–6.

Kutzner, T., Chaturvedi, K., Kolbe, T. H., 2020. CityGML 3.0: New Functions Open Up New Applications. *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88(1), 43–61.

Ledoux, H., Arroyo Ohori, K., Kumar, K., Dukai, B., Labetski, A., Vitalis, S., 2019. CityJSON: A compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4(1), 1–12.

Lowe, D. G., 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 91–110.

Lucks, L., Klingbeil, L., Plümer, L., Dehbi, Y., 2021. Improving trajectory estimation using 3D city models and kinematic point clouds. *Transactions in GIS*.

Moftizadeh, R., Vogel, S., Neumann, I., Bureick, J., Alkhatib, H., 2021. Information-Based Georeferencing of an Unmanned Aerial Vehicle by Dual State Kalman Filter with Implicit Measurement Equations. *Remote Sensing*, 13(16), 3205.

OpenMMLab, 2023. https://github.com/open-mmlab/mmsegmentation. Accessed: 2023-07-20.

Rehder, E., Albrecht, A., 2015. Submap-based slam for road markings. *2015 IEEE intelligent vehicles symposium (IV)*, IEEE, 1393–1398.

Roschlaub, R., Batscheider, J., 2016. An INSPIRE-conform 3D building model of Bavaria using cadastre information, LiDAR and image matching. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B4, 747–754.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An efficient alternative to SIFT or SURF. *2011 International Conference on Computer Vision*, 2564–2571.

Schönberger, J., Frahm, J.-M., 2016. Structure-from-Motion Revisited. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 4104–4113.

Singh, S., Singh, R., 2015. Comparison of various edge detection techniques. *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, IEEE, 393–396.

Uggla, M., Olsson, P., Abdi, B., Axelsson, B., Calvert, M., Christensen, U., Gardevärn, D., Hirsch, G., Jeansson, E., Kadric, Z., Lord, J., Loreman, A., Persson, A., Setterby, O., Sjöberger, M., Stewart, P., Rudenå, A., Ahlström, A., Bauner, M., Hartman, K., Pantazatou, K., Liu, W., Fan, H., Kong, G., Li, H., Harrie, L., 2023. Future Swedish 3D City Models - Specifications, Test Data, and Evaluation. *ISPRS International Journal of Geo-Information*, 12(2).

Vogel, S., Alkhatib, H., Neumann, I., 2018. Iterated extended kalman filter with implicit measurement equation and nonlinear constraints for information-based georeferencing. *IEEE 21st International Conference on Information Fusion (FUSION)*, IEEE, 1209–1216.

Wysocki, O., Hoegner, L., Stilla, U., 2022. Refinement of semantic 3D building models by reconstructing underpasses from MLS point clouds. *International Journal of Applied Earth Observation and Geoinformation*, 111, 102841.

Wysocki, O., Xia, Y., Wysocki, M., Grilli, E., Hoegner, L., Cremers, D., Stilla, U., 2023. Scan2LoD3: Reconstructing semantic 3D building models at LoD3 using ray casting and Bayesian networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 6547–6557.

Xia, Y., Yu, C., He, C., 2022. An Exploratory Distributed Localization Algorithm Based on 3D Barycentric Coordinates. *IEEE Transactions on Signal and Information Processing over Networks*, 8, 702-712.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., Luo, P., 2021. SegFormer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34, 12077–12090.