Automatic Detection and 3D Modeling of City Furniture Objects using LiDAR and Imagery Mobile Mapping Data

Hiba Doi[1], Anass Yarroudh[2], Imane Jeddoub[2], Rafika Hajji[1], Roland Billen[2]

[1] College of Geomatic Sciences and Surveying Engineering, Agronomy and Veterinary Institute Hassan II,
Rabat 10101, Morocco (doihiba,r.hajji)@iav.ac.ma
[2] GeoScITY, Spheres Research Unit, University of Liège, 4000 Liège, Belgium (ayarroudh,i.jeddoub,rbillen)@uliege.be

Keywords: City Furniture, Mobile Mapping Systems, Object detection, Semantic segmentation, Camera-LiDAR fusion, CityJSON.

Abstract

City furniture objects hold valuable information about urban traffic and city dynamics, making their integration into 3D city models essential for enhancing these models. This study implements two methodologies for detecting, classifying, and positioning City Furniture objects, as well as one approach for their automatic 3D modeling. The first approach uses Mobile Mapping System (MMS) imagery with YOLO for object detection and classification, coupled with the Line of Bearing (LoB) method for extracting XYZ coordinates and height. A spatial operation was conducted to determine object orientation. The second approach employs camera-LiDAR fusion, integrating KPConv for semantic segmentation and connected components for instance segmentation. Classification is performed using two complementary approaches: using Fast Global Registration (FGR) on point clouds, for lamppost types, and image-based, projecting point cloud instances to classify traffic lights and signs. This fusion approach leverages image-based classification models and point cloud accuracy, achieving an RMSE of 0.32 against ground truth data. The point cloud approach shows promise but requires refinement to improve noise sensitivity in FGR. This study presents a comprehensive workflow from detection to Level of Detail 4 (LOD4) modeling, combining KPConv and multi-source data to enhance feature detection and classification for city furniture.

1. Introduction

Identifying and determining the location and shape of city furniture objects plays a crucial role due to their diverse applications, including sign maintenance and inventory management de la Escalera et al. (2003). The objects' features, such as their locations and their type, are significant factors Timofte et al. (2014). The Mobile Mapping System (MMS) provides accurate spatial referencing and aligns effectively navigation data with images Amano et al. (2006). While many scholars focus on leveraging point cloud data for the detection of city pole-like objects Nurunnabi et al. (2023); Zou et al. (2021), the deployment of imagery remains relatively less investigated. Images present two main advantages: the lower cost of acquisition compared to LiDAR systems and the maturity of semantic analysis and object detection methods for street object localization, in contrast to the semantic segmentation of point clouds. However, to enhance results, some methodologies integrate LiDAR data alongside imagery Mori et al. (2018); Zhou et al. (2022).

Current research has not fully addressed the complexity of detecting city furniture objects. Most studies focus primarily on pole-like structures and do not emphasize accurate positioning, often employing methods that compromise precision. Moreover, a comprehensive workflow covering all aspects from detection to modeling of city furniture objects has not yet been developed.

In our study, we focused on detecting city furniture objects using two approaches: one relying solely on images and another combining both point cloud data and images. Both approaches follow the main steps: detection, positioning and modeling using different data types. Additionally, each approach performance and the quality of results are evaluated.

2. Related Work

The detection, classification, and localization of city furniture in urban settings using a combination of imagery and point cloud data have been the subject of extensive research in recent years. With the integration of novel deep-learning approaches, this field has seen significant advancements in both efficiency and accuracy.

2.1 Object Detection and Labeling

Grounding DINO, a pre-trained vision-language model for object detection, has been instrumental in enhancing automated labeling of urban data. Liu et al. (2023) demonstrated the effectiveness of Grounding DINO for automatically generating precise labels in diverse contexts, which greatly facilitates subsequent model training by reducing the manual annotation workload.

The YOLO framework Redmon et al. (2016) has been extensively used for real-time object detection in urban settings, thanks to its speed and accuracy. Comparatively, Faster R-CNN (Ren et al., 2015) offers higher precision for small objects but at the cost of reduced inference speed, making YOLO a more suitable option for our real-time requirements. The combination of Grounding DINO for labeling and YOLO for detection significantly improved our workflow by leveraging accurate, automated annotations for training a robust detection model.

2.2 Position Estimation and Automation

For determining the spatial positions of detected objects, the line-of-bearing (LOB) approach was effectively utilized by Li et al. (2022) for estimating the geolocation of urban elements using multi-perspective imagery. Their work highlights the utility of LOB for precise localization without the need for

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-2-W8-2024-125-2024 | © Author(s) 2024. CC BY 4.0 License.

extensive ground control points. Compared to methods like Structure from Motion (SfM) as used by Snavely et al. (2010), which require overlapping imagery for 3D reconstruction, the LOB approach provides a computationally efficient solution, particularly in scenarios where image data is limited. In our work, this methodology was extended ti calculate the Height of the object and their elevation.

2.3 Point Cloud Segmentation

The use of point cloud data for semantic segmentation has also gained prominence due to its ability to provide accurate 3D information. Thomas et al. (2019) introduced KPConv, a point convolution approach specifically designed for effective semantic segmentation of irregular 3D point clouds. Earlier approaches, such as Qi et al. (2017a,b), pioneered point cloud processing but struggled with capturing local geometric features, an issue that KPConv effectively addressed. KPConv's ability to model local dependencies made it particularly useful for segmenting city furniture objects from dense point clouds.

2.4 Integration of Imagery and Point Cloud Approaches

The integration of both imagery-based detection and point cloud segmentation provides a more robust framework for city furniture modeling. Mori et al. (2018); Zhou et al. (2022) highlighted that combining data from imagery and LiDAR sensors improves the detection accuracy of small, occluded objects. This finding contrasts with purely LiDAR-based methods, such as those by Nurunnabi et al. (2023); Li et al. (2019), which, although effective for large-scale mapping, involved a large number of steps using machine learning algorithms.

3. Imagery-based approach

In this part of the paper, we propose an automatic detection, positioning, and modeling method for CityFurniture using Mobile mapping System (MMS) Imagery. Which mainly includes three parts:CityFurniture detection models, LOB-based detection Models.

3.1 Data Collection and Object Detection

As illustrated in Figure 1, our workflow begins with data collection, capturing 360 images along urban streets, with each image accompanied by its corresponding camera position and orientation. The next step involves detecting our key objects of interest, specifically traffic signs, traffic lights, lampposts, and bus stops. To accomplish this, we utilize YOLOv8 as our primary detection model, chosen for its maturity and speed, outperforming other models. Successful training of this model requires an extensive set of labeled images. To generate these, we employ Grounding DINO Liu et al. (2023), an open-set object detector, to produce a sufficient quantity of annotated images. These labels were manually reviewed and refined, allowing YOLOv8[1] models to effectively identify various city furniture elements, including traffic signs, traffic lights, lampposts, and bus stops.
The models show high performance across all categories, with bus stops achieving the best detection metrics. Traffic signs, lampposts, and traffic lights are also detected, with consistently good IoU, precision, recall, mAP50, and mAP50-95 values.

For certain classes with high variability, such as traffic signs, we implement a cascaded detection approach, as shown in figure 3. We start by applying inference on all images in our dataset using a model designed to detect traffic signs, as shown in (step 2). Subsequently, cropping is performed to isolate each traffic sign individually (step 3). We then run a classification model specifically trained on Belgian traffic signs on the cropped images (step 4). Finally, we remap the subclass labels to the original images to preserve the exact positions of the traffic signs (step 5). This is essential for obtaining the pixel coordinates of the objects in the original images, which are needed for the localization step.
As shown in figure 2, we created a dataset for traffic signs following the Belgian Traffic Sign Codification. This dataset includes 63 classes, each containing approximately 50 images. The dataset was generated by running a general YOLOv8 detection model, previously trained, to identify traffic signs. We then cropped the resulting bounding boxes and manually curated the dataset to ensure accuracy and completeness.

3.2 Image Segmentation

To accurately find the pixel coordinates of a detected object, we need to locate its exact position within the bounding box, as shown in Figure 4. For objects like traffic lights and lampposts, we use Segment-Anything-Model (SAM) (Kirillov et al. (2023)) to extract the mask of the object from the bounding box. After that, we identify the lowest and the highest points of the mask. This information is then used in the positioning step to calculate the precise location, elevation and height of the object.

3.3 Positioning and Features Extraction

For the positioning step, we developed an algorithm that uses a spherical camera orientation system and photogrammetry equation to calculate the bearing between the camera and the object. This process was first proposed by Li et al. (2022). The first step in solving the positioning issue involves understanding epipolar geometry and spherical panorama concepts.

The conversion from pixel coordinates (x, y) to spherical coordinates (ϕ, λ), as shown in Figure 5, is given by:

$$\phi = \left(x - \frac{w}{2}\right) \cdot \frac{2\pi}{w} \tag{1}$$

$$\lambda = \left(y - \frac{h}{2}\right) \cdot \frac{\pi}{h} \tag{2}$$

The conversion from spherical coordinates (ϕ, λ) to Cartesian coordinates (x, y, z) is given by:

$$x = r\cos(\lambda)\cos(\phi) \tag{3}$$

$$y = r\cos(\lambda)\sin(\phi) \tag{4}$$

$$z = r\sin(\lambda) \tag{5}$$

The transformation from image space coordinates to world coordinates is given by:

$$\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = s \cdot R \cdot \begin{bmatrix} x_c \\ y_c \\ -z_c \end{bmatrix} + \begin{bmatrix} x_{\text{cam}} \\ y_{\text{cam}} \\ z_{\text{cam}} \end{bmatrix} \tag{6}$$

$$R = Rz(yaw) \cdot Ry(pitch) \cdot Rx(roll)$$

where s is the depth coefficient, R is the rotation matrix, (x_c, y_c, z_c) are the image space coordinates, and $(x_{\text{cam}}, y_{\text{cam}}, z_{\text{cam}})$ are the camera position coordinates in the world frame.

[1] You Only Look Once

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-2-W8-2024-125-2024 | © Author(s) 2024. CC BY 4.0 License.

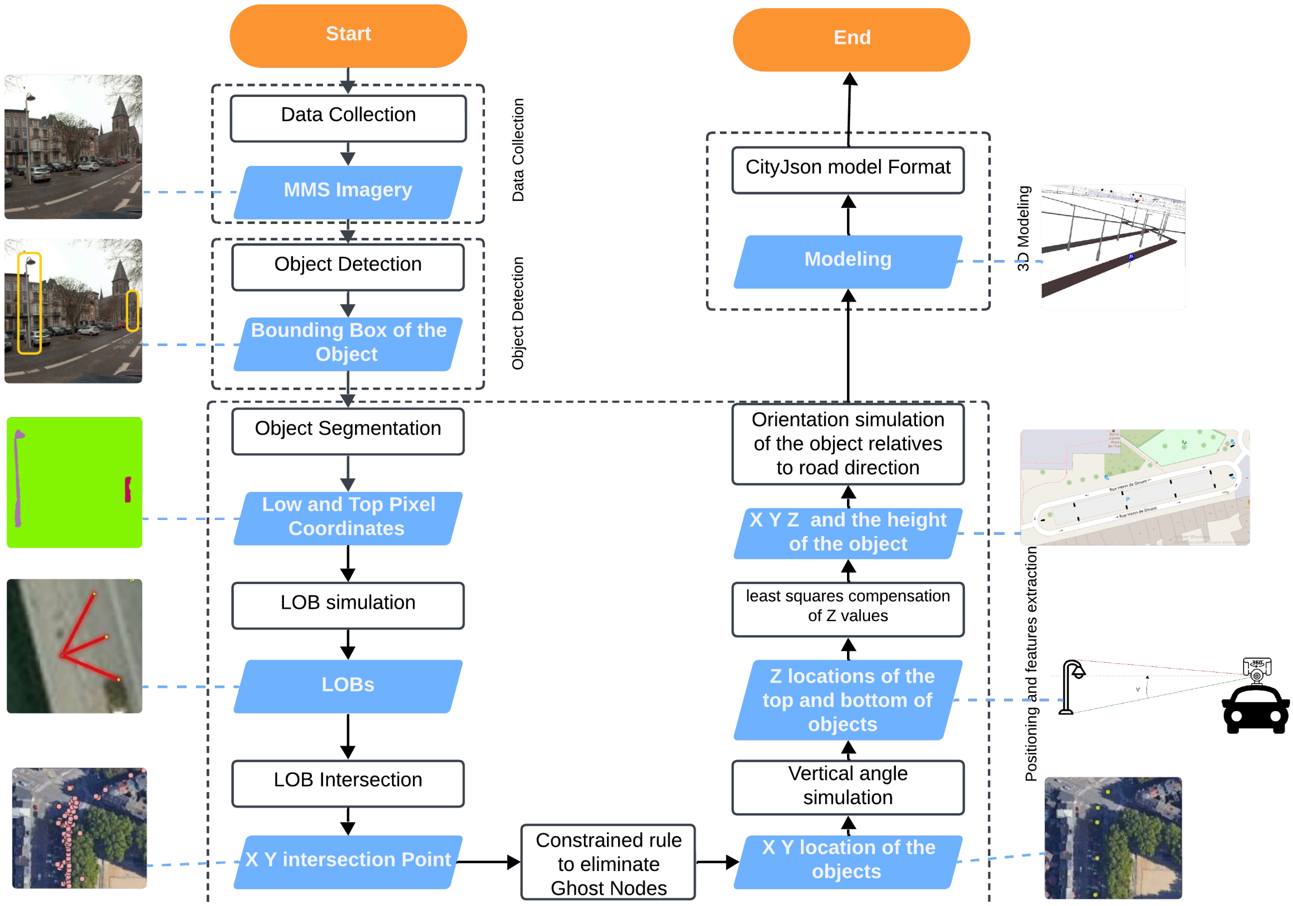
Start
Data Collection
MMS Imagery
Data Collection
Object Detection
Bounding Box of the Object
Object Detection
Object Segmentation
Low and Top Pixel Coordinates
LOB simulation
LOBs
LOB Intersection
X Y intersection Point
Constrained rule to eliminate Ghost Nodes
X Y location of the objects
Vertical angle simulation
Z locations of the top and bottom of objects
least squares compensation of Z values
X Y Z and the height of the object
Orientation simulation of the object relatives to road direction
Positioning and features extraction
Modeling
CityJson model Format
3D Modeling
End

Figure 1. Imagery based methodology

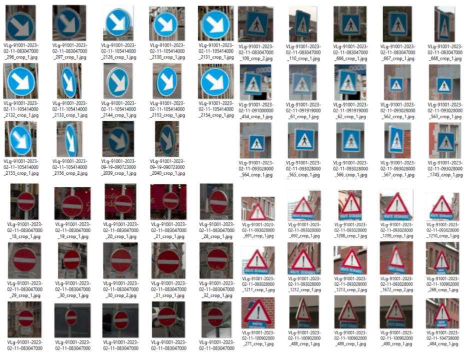

Figure 2. Traffic Sign Classification Model dataset

3.4 Line of bearing and vertical angle calculation

- Bearing

Because the s depth factor is unknown for us in a single image the soly information that we want is the bearing and vertical angle as in surveying problem . The bearing , b, corresponding to the direction camera-object can be expressed by Equation (7):

$$bearing = \arctan\left(\frac{y_c - y_{\text{cam}}}{x_c - x_{\text{cam}}}\right) \qquad (7)$$

- Vertical Angle

$$\text{V} = \arctan\left(\frac{(-z_c) - z_{\text{cam}}}{\sqrt{(x_c - x_{\text{cam}})^2 + (y_c - y_{\text{cam}})^2}}\right)$$

Line of Bearing/azimuth is represented by l, as shown in Equation (3):

$$l = (x_{\text{cam}}, y_{\text{cam}}, z_{\text{cam}}, bearing)$$

To address the positioning step, we drew inspiration from the LOB constrained method which essentially calculates the bearing line between the camera and the object, followed by an elimination algorithm to retain only the actual object, as illustrated in Figure 6 described in Li et al. (2022). However, we developed our own code, adding a component that utilizes the calculated vertical angle to determine the elevation and height of the object, based on the known positions of the top and bottom pixels.

3.5 Orientation Calculation

Orientation is a tricky characteristic to determine accurately from images alone. Therefore, we have decided to adopt a different approach. Given that the orientation of most urban objects is highly predictable (e.g., lampposts are typically perpendicular to the road, traffic signs face the same or opposite direction as the road), we can generalize this principle to other city furniture such as traffic lights, buses, and bus stops.

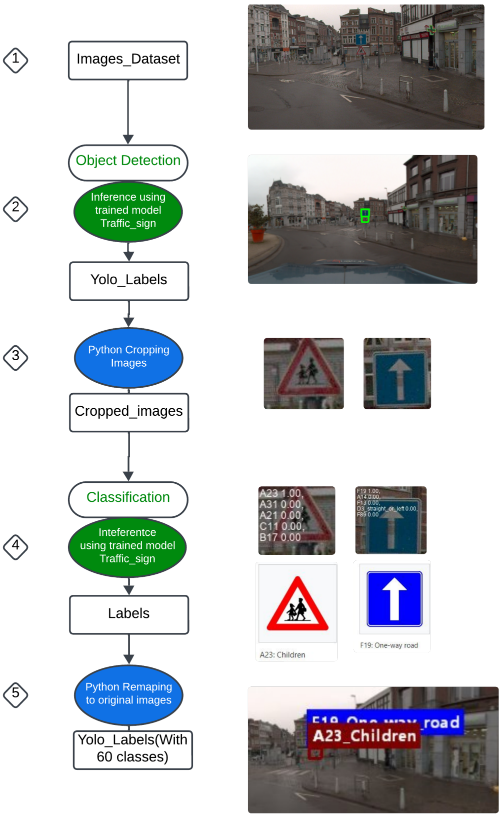
1
Images_Dataset
Object Detection
2
Inference using trained model Traffic_sign
Yolo_Labels
3
Python Cropping Images
Cropped_images
Classification
4
Inteferentce using trained model Traffic_sign
Labels
5
Python Remaping to original images
Yolo_Labels(With 60 classes)
A23 1.00, A31 0.00, A21 0.00, C11 0.00, B17 0.00
A23: Children
F19: One-way road
F19 One-way road
A23_Children

Figure 3. Cascaded Detection approach

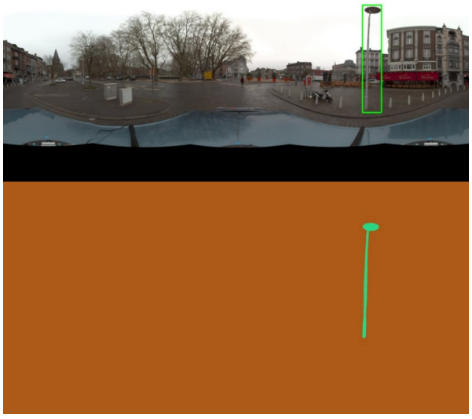

Figure 4. Resulting output from SAM based on the bounding box from the object detection

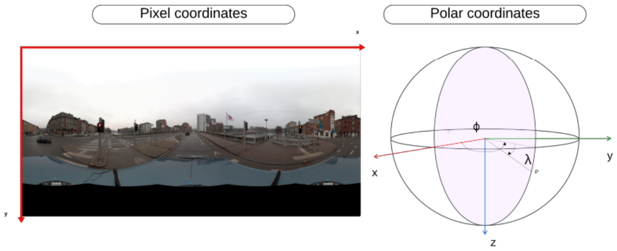

Figure 5. Transformation from pixel coordinates to polar coordinates

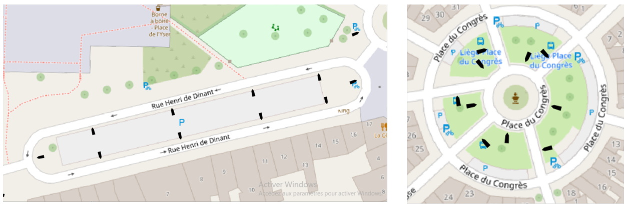

Figure 7. Results of the orientation calculation.

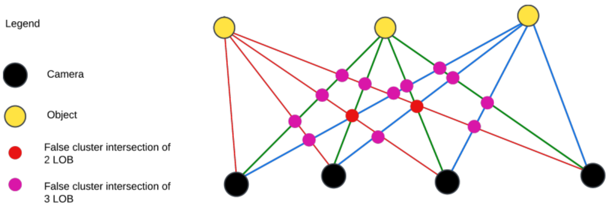
Legend
Camera
Object
False cluster intersection of 2 LOB
False cluster intersection of 3 LOB

Figure 6. Ghost Node

4. LiDAR-Camera fusion approach

4.1 Semantic Segmentation

The LiDAR-camera fusion approach, presented in Figure 8, starts with the semantic segmentation of 3D point clouds using a trained KPConv model on Toronto3D dataset Tan et al. (2020); **?**. The pole class, which in Toronto3D incorporates all vertical city furniture objects including traffic signs and lights, is successfully segmented with an Intersection-over-Union (IoU) of 78.4%.

4.2 Instance Segmentation

The second step in the process is instance segmentation, where we use Label Connected Components (LCC) algorithm to separate individual objects of the pole class, assigning each a unique identifier.

4.3 Classification Approach

Afterward, we proceed with a classification step constrained by height to differentiate between lampposts and other city furniture before performing further analysis.

For the lampposts, we apply a point cloud-based method using Fast Global Registration (FGR) and Iterative Closest Point (ICP) algorithms to calculate Fitness and Root Mean Square Error (RMSE) metrics with a pre-defined city furniture objects database. First, we identify the unique object models present in the dataset. Then, we perform FGR for coarse alignment, followed by refinement with ICP between each instance and the existing references, classifying the objects based on the highest Fitness value.

For other city furniture shorter than lampposts, we employ an image-based methodology. First, we run a projection algorithm to convert 3D bounding boxes into 2D images, projecting onto the four closest images as shown in the step named **reprojection to 2D images** in Figure 8. We then train a YOLOv8 model for the classification task focusing on traffic signs and lights as explained earlier in the image-based approach.

5. 3D Modeling

For the 3D modeling process, we provide a workflow to generate a CityJSON model from the extracted points, incorporating the position and 3D geometry template. The orientation, calculated during the feature extraction step, was then applied to modify the transformation matrix accordingly.

Firstly translating the 3D model from the initial format to CityJSON, the model required modifications to be used as a ‘Geometry Template. Figure 9 below illustrates the sequence of operations performed to prepare this basic CityJSON model. The first task involved merging the geometry while ensuring the correct application of textures to different surfaces. The second task was applying scale homogenization, rotating along

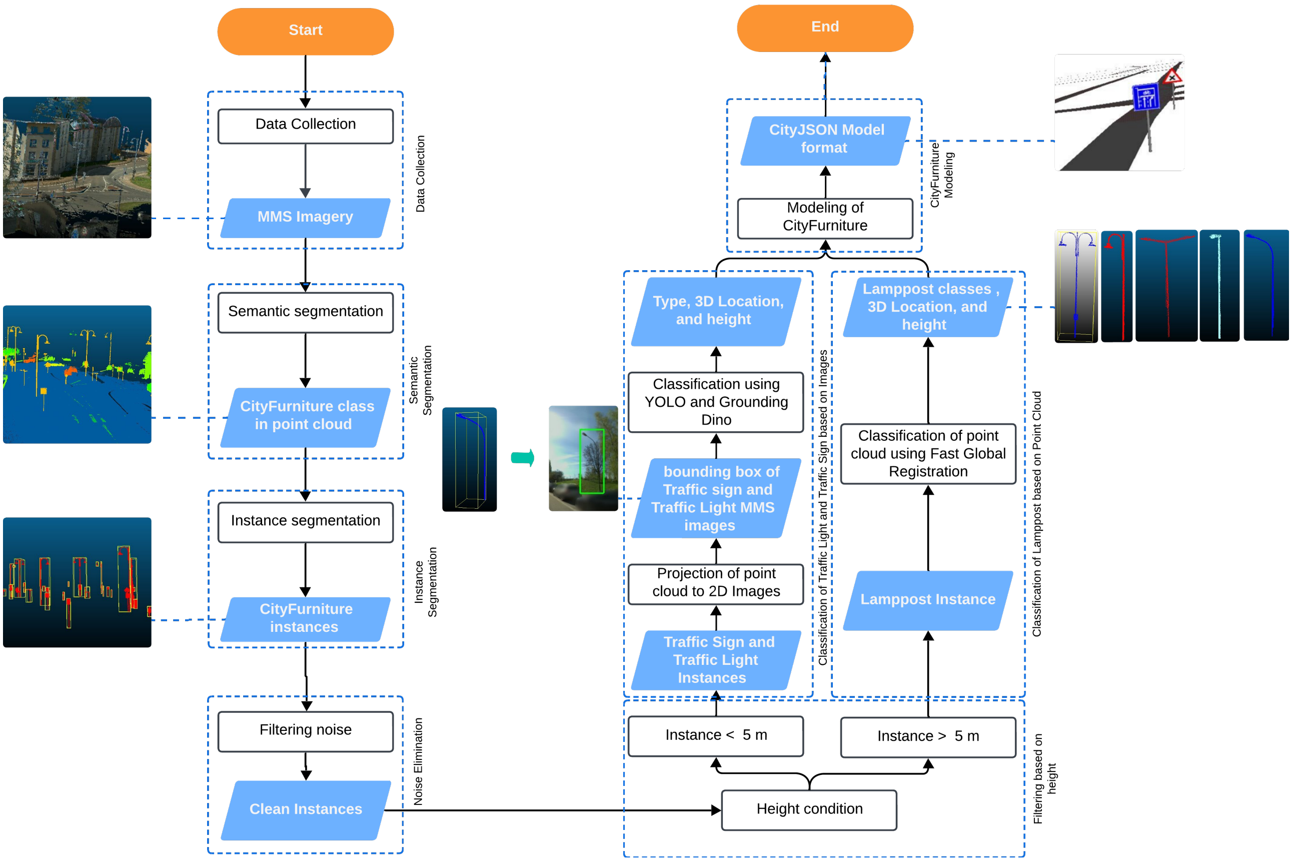
Start
Data Collection
MMS Imagery
Data Collection
Semantic segmentation
CityFurniture class in point cloud
Semantic Segmentation
Instance segmentation
CityFurniture instances
Instance Segmentation
Filtering noise
Clean Instances
Noise Elimination
Height condition
Instance < 5 m
Instance > 5 m
Filtering based on height
Traffic Sign and Traffic Light Instances
Projection of point cloud to 2D Images
bounding box of Traffic sign and Traffic Light MMS images
Classification using YOLO and Grounding Dino
Type, 3D Location, and height
Classification of Traffic Light and Traffic Sign based on Images
Lamppost Instance
Classification of point cloud using Fast Global Registration
Lamppost classes , 3D Location, and height
Classification of Lamppost based on Point Cloud
Modeling of CityFurniture
CityJSON Model format
CityFurniture Modeling
End

Figure 8. Combined Methodology

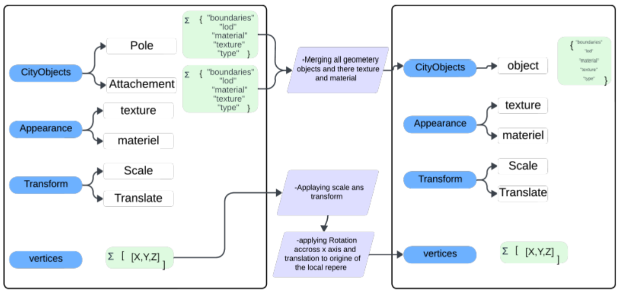
CityObjects
Pole
Attachement
Σ { "boundaries" "lod" "material" "texture" "type" }
Σ { "boundaries" "lod" "material" "texture" "type" }
Appearance
texture
materiel
Transform
Scale
Translate
vertices
Σ [[X,Y,Z]]
-Merging all geometery objects and there texture and material
-Applaying scale ans transform
-applying Rotation accross x axis and translation to origine of the local repere
CityObjects
object
{ "boundaries" "lod" "material" "texture" "type" }
Appearance
texture
materiel
Transform
Scale
Translate
vertices
Σ [[X,Y,Z]]

Figure 9. Merge geometry to one geometry object

the X-axis, and translating the model to the origin. All these adjustments were executed through an algorithm.

For the Generation of the Final CityJSON model of the City Furniture Object, we used the CityJSON Geometry Templates Mapper, developed within GeoScITY Lab. The basic command required the following inputs: a shapefile containing the detected objects from the positioning step, and an OBJ file to serve as a template for generating the final CityJSON geometry. The shapefile includes attributes such as orientation,height (corresponding to the Z-coordinate), and classification (CITYFURNITURE). The local rotation along the Z-axis is used to set the orientation of the objects. The output is a CityJSON file with the geometries of the objects, incorporating their positions, orientations, and attributes as defined by the input shapefile and template OBJ file. The entire process is detailed in Figure 10, which outlines the CLI command algorithm. We introduced several modifications, such as replacing the SHP file with a GeoParquet file and using a pre-prepared CityJSON file containing the template, instead of directly utilizing the OBJ file.

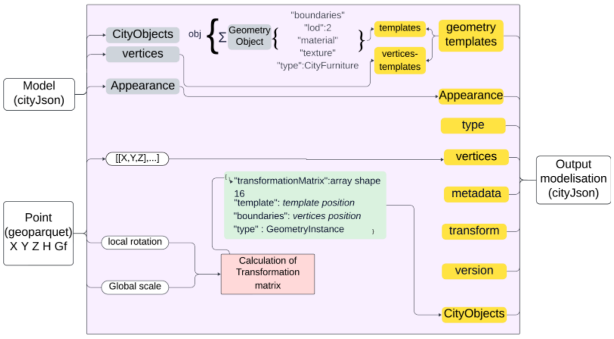
Model (cityJson)
CityObjects
vertices
Appearance
obj
Σ Geometry Object
"boundaries" "lod":2 "material" "texture" "type":CityFurniture
templates
vertices-templates
geometry templates
Appearance
type
[[X,Y,Z],...]
vertices
Point (geoparquet) X Y Z H Gf
local rotation
Global scale
Calculation of Transformation matrix
"transformationMatrix":array shape 16
"template": template position
"boundaries": vertices position
"type" : GeometryInstance
metadata
transform
version
CityObjects
Output modelisation (cityJson)

Figure 10. CLI command detailed structure

6. Experimental Results and Discussions

6.1 Data Collection and Research Area

In this Paper, we use two different Dataset, The first dataset, as shown in Figure 11, we used is collected by the MMS of the **DrivenBy** company in Liège City in Belgium. The system is equipped with a Ladybug panoramic camera, GNSS, and IMU. The image stream data output by the Ladybug panoramic camera is read and spliced to form a panoramic image with a 360° viewing angle, stored in a general picture format with an 8192 × 4096-pixel resolution. Along with the trajectory information (latitude, longitude, and elevation), the data includes camera orientation details (roll, pitch, and yaw).

https://doi.org/10.5194/isprs-archives-XLVIII-2-W8-2024-125-2024

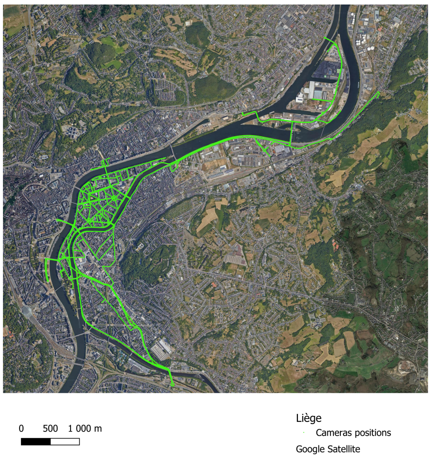

Figure 11. Liège MMS Trajectories

The second dataset we use, as shown in Figure 12, is collected by the MMS of the **GlobeZenith** Company in Arlon City In Belgium.The system is equipped with a Panorama 360° camera, GNSS, Leica TRK700 Evo with two LIDAR scanners Z+F 9020 and IMU. The image stream data output by the camera is read and spliced to form a panoramic image with a 360° viewing angle, stored in a general picture format with an 7040×3520-pixel resolution. Along with the trajectory information (X,Y and elevation) in ESPG 31370, the data includes camera orientation details (phi, omega, kappa).

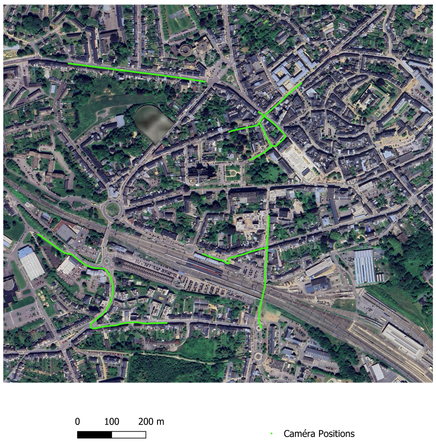

Figure 12. Arlon MMS Trajectories

6.2 Object Detection and segmentation

The success of the Object Detection Task relies on the performance of the metrics used to evaluate our trained models. In this case, we trained four models in the initial step for a general object detection task.

Specifically, we trained four models for detecting four types of objects: lampposts, traffic lights, traffic signs, and bus stops, as shown in Table 1, using labeled data prepared by us from Liège and Arlon Dataset, as shown in Table 2. However, for classes with multiple subclasses, such as traffic signs and lampposts, we employed a cascade detection and classification approach to effectively differentiate between the various subclasses.

Table 1. Performance metrics for direct object detection models.

	Bus Stop	**Lamppost**	**Traffic Light**	**Traffic Sign**
IOU	0.850	0.810	0.780	0.830
Precision	0.974	0.853	0.885	0.880
Recall	0.909	0.838	0.896	0.875
mAP50	0.942	0.909	0.946	0.927
mAP50-95	0.685	0.648	0.737	0.552

Table 2. Number of images used to trained to trained the model

	Bus Stop	**Lamppost**	**Traffic Light**	**Traffic Sign**
Number of labels	421	1365	1823	300
Number of classes	1	3	1	1

For object classes like **traffic signs** and **lampposts**, which have numerous subclasses, we applied a cascade detection and classification approach. This method allows us to differentiate between 50 types of traffic signs and 9 types of lampposts, ensuring more accurate subclass identification within these categories.

The Table 3 presents the metrics of the classification model trained.

Table 3. Top-1 and Top-5 Accuracy Metrics for the Traffic Sign and Lamppost Classification Model.

City Furniture type	num of classes	Accuracy_Top_1	Accuracy_Top_5
Traffic sign	61	0.99054	0.99369
Lamppost	9	0.985	1

After applying inference using the trained object detection and classification models, the table 4 presents the number of images in which objects have been detected.

Table 4. Inference result

	Liège	Arlon
Number of images	23836	734
Lammpost single	12046	272
Traffic Light	961	246
Traffic Sign	6165	283
Bus Stop	638	_
Lammpost double	_	31

6.3 Positioning results

The XY positions of the extracted objects were compared to a baseline data extracted from the PICC dataset, which is the 3D digital cartographic reference for Wallonia, and MMS point cloud data. Table 5 presents the calculated positioning deviation. A total of 579 points were analyzed, resulting in a mean error of 0.27 meters, which is the average positional discrepancy between the datasets. The RMSE of 0.32 meters quantifies the overall variation in positional differences. Figure 13 illustrates the detection of lampposts.

Metric	Value
Mean error	0.27 m
RMSE	0.32 m

Table 5. Positioning error.

Figure 13 presents the result of lamppost detection in Liège Dataset. It shows the identified lampposts, with the top-right image serving as an example, where 100% of the existing lampposts have been successfully detected

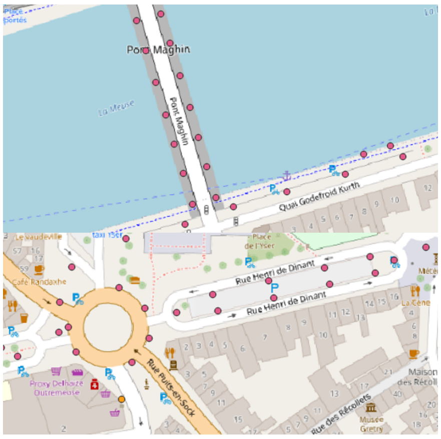

Figure 13. Lampposts positions.

The Figures 14, and 15 illustrate examples of the results from the positioning step. The traffic lights shown in the figures are accurately detected at the intersection, with all existing lights successfully identified in the Liège dataset. Additionally, the prohibited traffic sign along the street in Outremeuse, Liège, is also correctly identified.

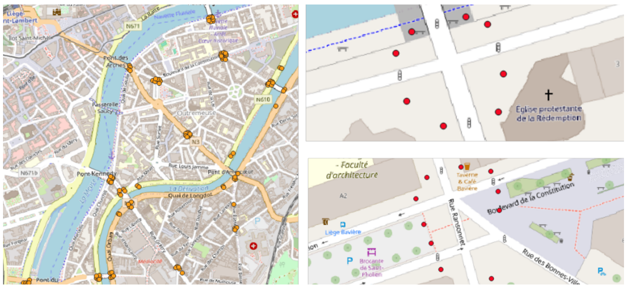
Église protestante de la Rédemption
Faculté d'architecture
Boulevard de la Constitution

Figure 14. Traffic Light Positioning

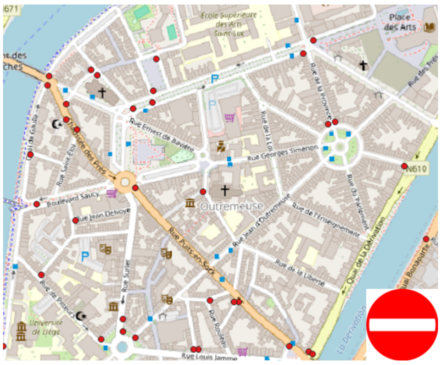

Figure 15. Result of C1 detection from the MMS imagery

6.4 3D modeling results

The 3D modeling process takes as input the position, the orientation and the height of each detected object. The orientation and height attributes are calculated in the feature extraction step. Figures 7 show the results of the orientation calculation process and Figure 16,17 and 18 present the final 3D models contain the various types of detected models lampposts , bus stop ans traffic sign .

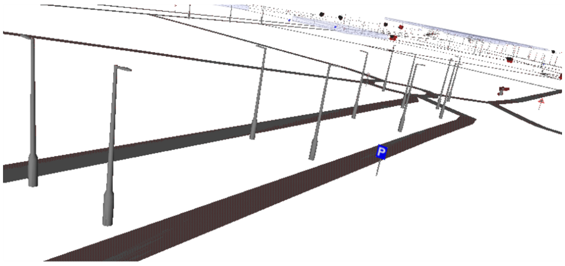

Figure 16. Final CityJSON model.

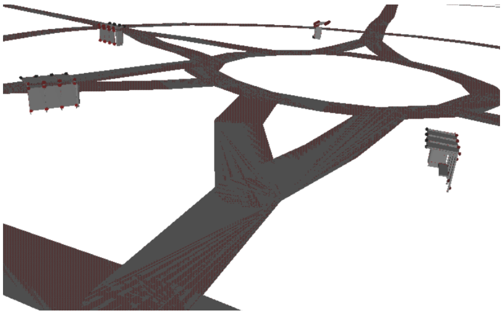

Figure 17. Bus Stop Detection

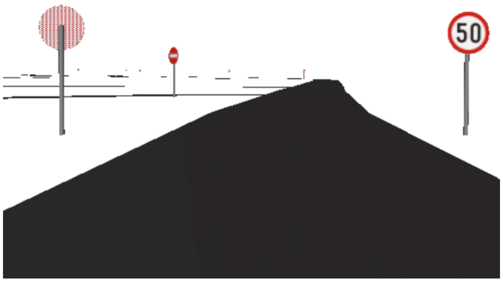

Figure 18. C43: Driving over the indicated speed forbidden

7. Discussion

This study explored two methodologies for detecting, and classifying CityFurniture Object: The imagery-based approach and the camera-LiDAR fusion approach. And one approach for automatically modeling city furniture objects Imagery-based methodology leveraged YOLOv8 models for high-precision object detection, supplemented by a cascade classification system to manage complex subclassifications. Photogrammetry and stereoscopic geometry techniques were used for accurate object localization, while the Segment Anything Model (SAM) further refined object boundaries, enhancing precision. The camera-LiDAR fusion methodology employed KPConv for semantic segmentation, Label connected component for instance segmentation, Fast Global Registration (FGR) for Lamppost Classification providing a robust means of identifying city furniture, and Back projection to 360 Imagery for traffic sign and traffic. However, FGR's sensitivity to noise and difficulty handling outliers limited its effectiveness in noisy environments, requiring future improvements. A key feature of this work was the use of the CityJSON format, which facilitated efficient automation of the feature extraction process via geometry templates and CLI code. This enabled accurate extraction of key attributes such as height, orientation, and type, improving 3D modeling. Overall, both methodologies demonstrate strong potential for improving urban object detection. This study provides an innovative and highly effective framework for city furniture modeling, establishing a strong foundation for future developments in urban analytics and intelligent systems.

8. Conclusion

As conclusion, this study focused on the automatic detection, localization and modeling of city furniture objects. Additionally it compared two different approaches: imagery-based approach and LiDAR-camera fusion approach, based on three key metrics: accuracy, detection rate, and classification quality.

Accuracy: Theoretically, the LiDAR-camera fusion approach provides higher positioning accuracy since it directly determines the position of objects. In contrast, the imagery-based approach achieves good accuracy, particularly in XY coordinates.

Detection rate: We observe that the imagery-based approach has a higher detection rate, primarily because objects appear in multiple images, increasing the likelihood of detection. This redundancy makes it difficult for an object to be missed. On the other hand, the LiDAR-camera fusion approach is more sensitive to occlusions and filtering processes; a single observation can be easily overlooked, making this approach more prone to omission and requiring more careful processing.

Classification quality: In both approaches, using images to classify city furniture objects directly or after projection from 3D point cloud, traffic signs and traffic lights were accurately detected. However, using solely the point cloud approach to classify lampposts by FGR/ICP registration is critical. Without proper filtering, the algorithm struggles with noise, which could hinder the classification quality.

References

Amano, Y., Hashizume, K. I. T. O. Y. A. T., s hi, K., Junichi-Takiguchi, J., kamamelc ocojp Takashi Fujishima, Tanaka, Y., 2006. Development of a vehicle-mounted road surface 3d measurement system development of a vehicle-mounted road surface 3d measurement system jun-ichi takiguchi mitsubishi electric corporation kamakura works 325. about mms.

de la Escalera, A., Armingol, J., Mata, M., 2003. Traffic sign recognition and analysis for intelligent vehicles.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., Girshick, R., 2023. Segment Anything. http://arxiv.org/abs/2304.02643.

Li, G., Lu, X., Lin, B., Zhou, L., Lv, G., 2022. Automatic Positioning of Street Objects Based on Self-Adaptive Constrained Line of Bearing from Street-View Images. *ISPRS International Journal of Geo-Information*, 11.

Li, Y., Wang, W., Li, X., Xie, L., Wang, Y., Guo, R., Xiu, W., Tang, S., 2019. Pole-like street furniture segmentation and classification in mobile LiDAR data by integrating multiple shape-descriptor constraints. *Remote Sensing*, 11.

Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J. et al., 2023. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*.

Mori, Y., Kohira, K., Masuda, H., 2018. Classification of pole-like objects using point clouds and images captured by mobile mapping systems. 42, International Society for Photogrammetry and Remote Sensing, 731–738.

Nurunnabi, A., Sadahiro, Y., Teferle, F. N., Laefer, D. F., Li, J., 2023. Detection and segmentation of pole-like objects in mobile laser scanning point clouds. 48, International Society for Photogrammetry and Remote Sensing, 27–34.

Qi, C. R., Su, L., Mo, H., Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 652–660.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems (NeurIPS)*, 5105–5114.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. 2016-December, IEEE Computer Society, 779–788.

Snavely, N., Simon, I., Goesele, M., Szeliski, R., Seitz, S., 2010. Scene Reconstruction and Visualization From Community Photo Collections. *Proceedings of the IEEE*, 98, 1370 - 1390.

Tan, W., Qin, N., Ma, L., Li, Y., Du, J., Cai, G., Yang, K., Li, J., 2020. Toronto-3D: A large-scale mobile lidar dataset for semantic segmentation of urban roadways. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 202–203.

Thomas, F., Büttner, L., Geiger, A., Brox, T., 2019. Kpconv: Flexible and deformable convolution for point clouds. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 4508–4517.

Timofte, R., Zimmermann, K., Gool, L. V., 2014. Multi-view traffic sign detection, recognition, and 3D localisation. *Machine Vision and Applications*, 25, 633-647.

Zhou, Y., Han, X., Peng, M., Li, H., Yang, B., Dong, Z., Yang, B., 2022. Street-view imagery guided street furniture inventory from mobile laser scanning point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 189, 63-77.

Zou, Y., Weinacker, H., Koch, B., 2021. Towards urban scene semantic segmentation with deep learning from lidar point clouds: A case study in baden-württemberg, germany. *Remote Sensing*, 13.