MapAnything: Evaluating Monocular Metric Depth Models for 3D Urban Asset Localization

Miriam L. Carnot[1], Jonas Kunze[1], Erik Q. Fastermann[2], Eric Peukert[1], André Ludwig[2,3], Bogdan Franczyk[2,4]

[1] ScaDS.AI, University of Leipzig, Germany
(carnot, peukert)@informatik.uni-leipzig.de, jonas_benedikt.kunze@uni-leipzig.de
[2] University of Leipzig, Germany
cx19uziz@studserv.uni-leipzig.de, (ludwig, franczyk)@wifa.uni-leipzig.de
[3] Kühne Logistics University, Hamburg, Germany
[4] Wrocław University of Economics and Business, Poland

Keywords: Geo-Localization, Urban Digital Twins, Street-View Imagery, Road Infrastructure Assets

Abstract

City administrations increasingly rely on comprehensive databases and digital twins of city assets, such as traffic signs and trees, as well as incidents such as graffiti or road damage, to maintain an effective overview of urban conditions. Digitization has increased the demand for continuously updated spatial datasets, yet current data acquisition and maintenance processes still involve considerable manual effort, posing significant scalability challenges. This paper introduces MapAnything, a systematic evaluation pipeline that automates the spatial mapping of urban objects and incidents from a single monocular image. By leveraging advanced Metric Depth Estimation models, MapAnything accurately calculates object geocoordinates, converting 2D image data into valuable 3D spatial information. The methodology integrates the estimated camera-to-object distance with geometric principles and known camera specifications. We present a detailed validation of the framework, comparing its distance-estimation accuracy against high-precision LiDAR point clouds in complex urban environments. Our evaluation provides a granular analysis of spatial performance across various distance intervals and semantic areas, such as roads and vegetation. Finally, we demonstrate the framework's practical efficacy through specific use cases, including mapping traffic signs and road pavement damage, and provide recommendations for its integration into automated urban inventory systems.

1. Introduction

Every city has countless infrastructure assets financed by public funds that help ensure people can live together efficiently, safely, and comfortably. These include all traffic safety installations, such as traffic lights, signs, and street lamps, as well as recreational facilities such as benches, planting, bus stops, and bicycle stands. City governments maintain spatial databases and digital twins for many of these assets, which record each asset's location, type, and, where applicable, condition. There are two fundamental challenges here: 1. the initial creation of the data record, and 2. keeping the database up to date. Many objects can be damaged or even be stolen over time. Thus, it is important to carry out regular checks. The opposite can also be true, for example, when new traffic signs are installed on a construction site. It may also be of interest to map incidents such as graffiti or road pavement damage. By equipping vehicles such as trams, taxis, and garbage trucks with cameras, the necessary images can be captured to provide a continuously updated, cost-effective spatial overview.

Current solutions for mapping objects have their pitfalls. Some use either additional aerial imagery or point clouds from LiDAR sensors (Wegner et al., 2016, Branson et al., 2018, Boller et al., 2019), which are not always available, as such recordings are expensive and therefore cannot be taken regularly. Other works merely use the camera's coordinates at the time of recording (Novack et al., 2020), which is insufficient for accurate mapping. Yet other studies include knowledge of the object's size in the calculation (Campbell et al., 2019). This approach assumes that the objects searched for have the same size and that this size is known. Finally, some approaches analyze several consecutive images and then locate objects using triangular relationships (Hebbalaguppe et al., 2017, Cheng et al., 2018, Krylov et al., 2018, Krylov and Dahyot, 2019). This requirement that an object must be visible in several images is not always given. To address this, we introduce a highly scalable, low-cost method for 3D data acquisition. By developing a universal framework that extracts 3D spatial data from single 2D images, we eliminate the need for prior object knowledge or multiple overlapping viewpoints. This makes our approach particularly valuable for leveraging crowd-sourced imagery, which often lacks high-quality, corresponding LiDAR point clouds or consistent multi-view coverage.

To address this, our study is guided by the following research question: To what extent can state-of-the-art monocular metric depth estimation models be reliably utilized for the 3D geo-localization of urban assets, and how do they perform across varying distances and semantic backgrounds?

Our developed framework relies on these models to estimate the distance from the camera to each pixel in a single image. We examine four recently published state-of-the-art models: DepthAnything (Yang et al., 2024), DepthPro (Bochkovskii et al., 2024), Metric3D (Yin et al., 2023, Hu et al., 2024), and UniDepth (Piccinelli et

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W1-2026-57-2026

al., 2024). To test their suitability for our practical urban application, we compare the generated depth images with projections of 3D LiDAR point clouds from the same scene, evaluating errors across semantic classes (such as roads, vegetation, and buildings) and distance ranges. With the estimated depth and the camera's position and orientation, the object's geo-coordinates are then extracted using angular deviations from the center point and triangulation. Finally, we evaluate two practical use cases: traffic signs and road pavement damage. In both cases, we assess localization accuracy, i.e., the distance between the predicted and true coordinates. For the traffic signs, we also determine how many of the annotated signs were found and compare the predictions to the city council's database.

The main contributions of this paper include:

- a systematic evaluation of a highly scalable, low-cost pipeline for extracting the geo-coordinates of urban assets from single 2D street-view images,
- a benchmarking of four SOTA monocular metric depth estimation models, assessing their distance-estimation accuracy against high-precision LiDAR point clouds across various urban semantic areas,
- a demonstration of practical utility through two real-world case studies (mapping traffic signs and road pavement damages), evaluating the reliability of the predicted geo-coordinates for updating digital twins.

Code, annotations, and test data are available under: `https://github.com/miriamcarnot/MapAnything`.

2. Related Work

This Section discusses approaches to estimating the geo-location of urban objects and provides information on Monocular Depth Estimation, a core part of the MapAnything framework.

2.1 Mapping Occurrences or Objects

For the most part, data collection for urban objects in geographic information systems (GIS) and city models relies on high-precision, resource-intensive methods. City administrations typically use Mobile Mapping Systems (MMS) with LiDAR sensors, aerial photogrammetry, or manual terrestrial surveys to create and update precise digital twins of the city. While these approaches provide high-accuracy spatial data, they require expensive equipment, labor-intensive manual post-processing, and dedicated mapping campaigns. This significantly limits the frequency of data collection, making the continuous maintenance of an up-to-date city inventory a challenge both financially and in terms of scalability. To address these limitations, several studies have investigated the geo-referencing of objects using Street View imagery for urban applications. The most basic method is to use the camera's geo-coordinates at the time of recording. For example, Novack et al. (Novack et al., 2020) used this approach for geo-referencing graffiti on building façades. Another possible solution is to include aerial images that provide additional information. Boller et al. (Boller et al., 2019) use high-quality orthophotos to develop a mapping method for panorama images. They present a formula for calculating the geo-coordinates that also uses the distance between the object and the camera, but they do not reveal its origin. Their evaluation is purely qualitative. Another study that used street-level and aerial imagery tested their method on trees (Wegner et al., 2016). In a follow-up study, they also classified tree species and tracked and classified changes (Branson et al., 2018).

A third valid approach is combining the information from multiple images. These approaches assume that objects can be seen across multiple images and are therefore not suitable for all types of data collection. One example of this line of research is the work by Hebbalaguppe et al. (Hebbalaguppe et al., 2017). They detect telecommunication inventory in GSV image pairs of consecutive locations to perform triangulation and estimate coordinates. One research group also used a depth estimation model to retrieve the distance to the object. They estimate depth and then fuse distance information across multiple images using Markov Random Fields for Triangulation. In their first work (Krylov et al., 2018), they detect traffic lights and telegraph poles in Google Street-View (GSV) images while they look into crowd-sourced image data (Mapillary) in their second study (Krylov and Dahyot, 2019). If a large number of images are available, one may also use photogrammetry. Cheng et al. (Cheng et al., 2018) retrieve many crowd-sourced images of one building and use them to generate a point cloud. The location and extension of the building can then be determined from the created point cloud.

Some mapping approaches are object-specific, such as the one described by Campbell et al. (Campbell et al., 2019). They geo-reference traffic signs in GSV images using information about their dimensions. They determine the distance between the sign and the camera by calculating the ratio of the sign's real-world size to its image-space size. Then they calculate the angular offsets relative to the object in the image to determine its coordinates. Another study by Vishnani et al. (Vishnani et al., 2020) uses a simple regression model and the Haversine formula to map manholes detected in GSV imagery. However, it is not further explained or evaluated. In our work, we aim to determine the geographic coordinates of an urban object without any additional records or information about it, though we acknowledge that relying solely on single-image data introduces inherent limitations in absolute accuracy compared to multi-modal approaches.

Similar localization challenges arise in autonomous driving and robotics, yet their primary objective is to detect 3D objects from 2D images (Qin et al., 2018, Li et al., 2023b). Because their focus is on navigation and obstacle avoidance, these networks are designed to extract precise 3D bounding boxes and 6-DoF poses for dynamic objects (e.g., vehicles). While many of these models incorporate depth estimation, they prioritize extracting precise 3D bounding boxes and object orientation. In contrast, finding the geo-coordinates of static urban objects does not require finding the bounding box and its orientation. Furthermore, to achieve high reliability, the most robust methods in these domains typically rely on multi-modal or multi-view data rather than a single image, utilizing LiDAR (Zhang et al., 2024), Radar (Lin et al., 2024), RGB-D cameras (Qi et al., 2018), or multiple images

taken simultaneously (Li et al., 2023a) or in sequence.

2.2 Monocular Depth Estimation

Depth estimation aims to infer the third dimension of a recorded scene, estimating the distance of each pixel from the camera. Monocular depth estimation performs this task using a single image, without stereo or multi-view information (Masoumian et al., 2022). The task can be categorized into relative and metric depth estimation. Relative depth estimation focuses on capturing depth relationships between objects in an image, ensuring correct ordering rather than absolute scale. Metric depth estimation, in contrast, aims to predict absolute distances between objects and the camera, requiring scale awareness through additional priors or estimations (Zhu et al., 2024). Early monocular depth estimation models relied on supervised learning, using single-dataset training with consistent camera parameters and recording conditions (e.g., focal length, field of view, and lighting) (Eigen et al., 2014, Garg et al., 2016, Godard et al., 2016). These models often lacked generalization across diverse environments. Later, models such as MiDaS (Ranftl et al., 2020) introduced multi-dataset training, significantly improving cross-dataset performance. Recently, zero-shot monocular depth models have demonstrated robust generalization across diverse images, handling varying domains without fine-tuning (Bhat et al., 2023, Bochkovskii et al., 2024). Within the last few years, several new models beating previous results were introduced to estimate metric depth: DepthAnything (Yang et al., 2024), Depth Pro (Bochkovskii et al., 2024), UniDepth (Piccinelli et al., 2024), and Metric3D (Yin et al., 2023)(Hu et al., 2024), among others. These models use additional constraints, such as learned geometric priors or scale normalization, to achieve better predictions of absolute depth values. So far, these models have only been evaluated on known benchmarks, not yet for concrete applicability to street-view images in practical urban use cases.

3. Methodology

In this Section, we present the three key steps to map any urban object or occurrence with a single street-view image:

1. *Detect/Segment the desired object* in the image.
2. Apply the *MapAnything Framework*:
   (a) Estimate the metric depth for every pixel.
   (b) Calculate each object's coordinate based on the predicted depth at this position and its angular offset from the image's center.
3. *Remove duplicates* of objects that were recognized in multiple images and/or match with database entries.

3.1 Detect/Segment the Object in the Image

To map an object, the system needs to know where in the image it appears. These image areas can be annotated manually or predicted by a model. There are freely available models for detecting or segmenting various objects on platforms such as HuggingFace. Of course, it is also possible to train a model from scratch or fine-tune it on specific data. Among the valuable datasets containing large quantities of annotated urban images are Mapillary Vistas (Neuhold et al., 2017), Cityscapes (Cordts et al., 2015), and A2D2 (Geyer et al., 2020). For our experimental evaluation (Section 4), we use a U-Net trained on the A2D2 dataset for traffic-sign segmentation and a Segformer trained on the Cityscapes dataset for general semantic segmentation.

3.2 The MapAnything Framework

To calculate the geographic coordinates of an object, two key pieces of information are required: the camera's geo-coordinates at the time of recording and the recording direction, i.e., the deviation from north (yaw). All other camera information is not strictly necessary, as it can be estimated. If unknown, the camera's focal length can be estimated directly from the image utilizing integrated modules within contemporary depth models, such as DepthPro. And the camera's pitch can be estimated based on visual features in the image (such as the horizon or buildings). In our approach, we use a depth estimation model to predict a dense metric depth map, assigning a depth value (in meters) to every pixel. We then compute the overall distance to the target object using either the depth at the center pixel of its 2D bounding box or a weighted average of the depths within its segmentation mask. For larger objects or occurrences, a coordinate range can be determined between the outermost points, e.g., for a widespread graffiti painting. Four recent Monocular Metric Depth Estimation models and their variations will be evaluated in Section 4.

Applying the standard pinhole camera model, we set up the following equations. First, we normalize the pixel coordinates using the focal lengths f_x and f_y:

$$x_{norm} = \frac{x - W/2}{f_x}, \quad y_{norm} = \frac{y - H/2}{f_y} \tag{1}$$

Next, we determine the angular offsets α_x and α_y relative to the camera's optical axis, effectively converting the image-plane coordinates into real-world angles:

$$\alpha_x = \tan^{-1}(x_{norm}), \quad \alpha_y = \tan^{-1}(y_{norm}) \tag{2}$$

We adjust the camera's yaw and pitch angles by adding the calculated angles:

$$\theta_{yaw,eff} = \theta_{yaw} + \alpha_x, \theta_{pitch,eff} = \theta_{pitch} + \alpha_y \tag{3}$$

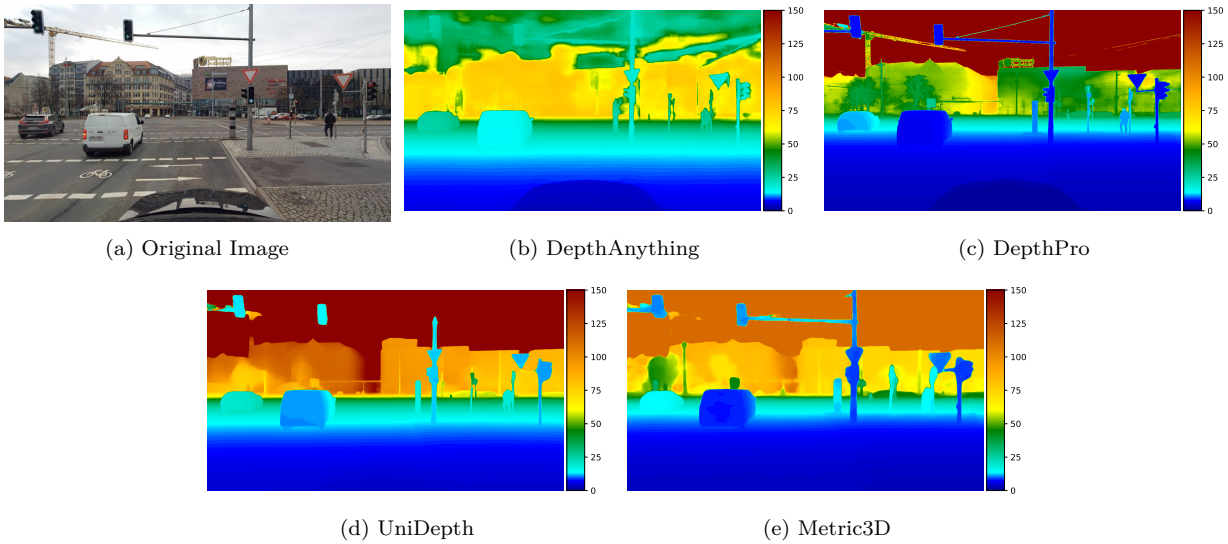
Using the predicted metric depth d of the object, we first compute the horizontal distance $d_{horizontal}$ on the ground plane. We can then calculate the spatial displacements Δx (eastward distance) and Δy (northward distance) relative to the camera:

$$d_{horizontal} = d \cdot \cos(\theta_{pitch,eff}) \tag{4}$$

$$\Delta x = d_{horizontal} \cdot \cos(\theta_{yaw,eff}) \tag{5}$$

$$\Delta y = d_{horizontal} \cdot \sin(\theta_{yaw,eff}) \tag{6}$$

Finally, we convert the displacements into changes in latitude and longitude using the Earth's radius R and add



(a) Original Image (b) DepthAnything (c) DepthPro

(d) UniDepth (e) Metric3D

Figure 1. Comparative visualization of four SOTA Monocular Metric Depth Estimation models. Scales are in meters. For better visibility, we clip the estimations at 150m. Note how DepthPro (c) maintains sharper edges on the poles compared to the over-smoothed output of Metric3D (e), which impacts the precision of the resulting geocoordinates.

them to the camera's coordinates:

$$\Delta\text{lat} = \frac{\Delta y}{R}, \quad \Delta\text{lon} = \frac{\Delta x}{R\cos(\text{lat}_{camera})} \tag{7}$$

$$\text{lat}_{object} = \text{lat}_{camera} + \left(\frac{180}{\pi}\right) \cdot \Delta\text{lat} \tag{8}$$

$$\text{lon}_{object} = \text{lon}_{camera} + \left(\frac{180}{\pi}\right) \cdot \Delta\text{lon} \tag{9}$$

Based on these calculations, there is a linear relationship between the deviation in the depth determination and the error of the calculated coordinate.

3.3 Remove Duplicates or match with Database

To identify duplicates resulting from overlapping images, we group predictions that share the same object class and fall within a specified geographic radius. The radius size that represents the best trade-off between correct matches and error rate should be determined for each object type. A basic approach to determining this radius is to analyze the minimum distance between two distinct ground-truth objects of the same class. When duplicates are found within this range, we take the center of the predicted coordinates as the final prediction.

While our experimental prototype uses a straightforward radius search to assess localization accuracy, we acknowledge that this $O(n^2)$ brute-force comparison does not scale well. For practical, city-scale deployments involving massive datasets, integrating spatial indexing structures (such as R-trees) into the spatial database is essential for efficiently processing spatial queries. Furthermore, to match our finalized predictions with existing municipal database entries, we use the calculated angular offsets (added to the recording direction) to find entries of the matching class within a reasonable distance, accounting for slight angle deviations.

4. Experiments

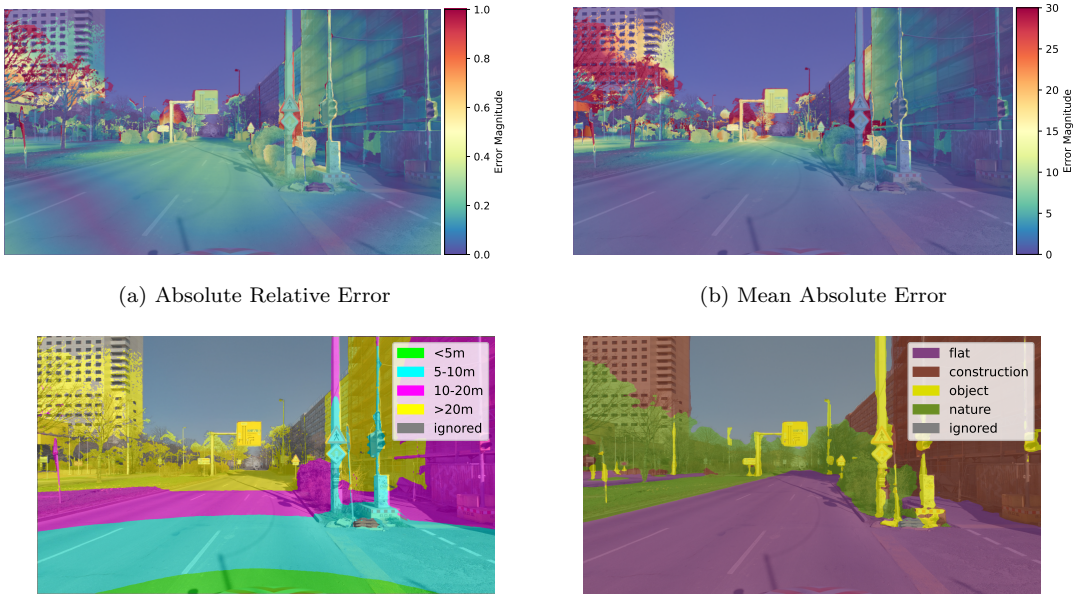
We first analyze the extent to which the predicted depth deviates from the actual distance measured by a LiDAR sensor. For a comprehensive evaluation, we test various models, distance range, and semantic segments in the images. Then, we demonstrate the proposed pipeline in two use cases: traffic signs and road pavement damage, as these issues are common on most roads.

4.1 Evaluation of the Depth Models

We evaluate four models that were recently published: DepthAnything (Yang et al., 2024), DepthPro (Bochkovskii et al., 2024), UniDepth (Piccinelli et al., 2024), and Metric3D (Yin et al., 2023). DepthAnything offers three different model sizes, does not require any camera intrinsic information, and was trained on the synthetic Virtual KITTI dataset (Cabon et al., 2020). DepthPro was trained on a combination of real-world and synthetic datasets to improve its generalization. It uses the focal length as an input parameter for scaling, but can also operate without this parameter by estimating it. We test both versions. UniDepth (Version 2) also does not rely on external information and can estimate the camera's intrinsic parameters. It has a ViT backbone and is also available in three sizes. Metric3D was trained on 18 datasets, including over 16 million images from thousands of different camera types. It always requires the camera intrinsic, and there are versions with a ViT backbone and another with a ConvNeXt backbone. Figure 1 shows predicted depth maps from the original image in (a). It is important to note that sky pixels are usually mapped to the highest possible value. The Metric3D model only predicts depths up to 100 meters. Thus, the sky appears orange (its maximum) instead of red. DepthAnything (b) is the only model that struggles with the sky and the background, but it produces good results for objects close to the camera. The depth map of Metric3D (e) appears blurred, making it difficult to recognize contours, such as the pedestrian on the right. Depth Pro (c) produces very fine contours, even for the pipe of the traffic light, which

Model	cam	total		<5m		5–10m		10–20m		>20m		flat		constr.		object		nature		time
	intr.	M	A	M	A	M	A	M	A	M	A	M	A	M	A	M	A	M	A	(s)
DAny-S	✗	8.7	0.500	1.3	0.275	3.7	0.517	9.4	0.655	13.9	0.398	4.4	0.465	10.9	0.452	12.4	0.649	14.1	0.562	0.9
DAny-B	✗	9.0	0.524	1.5	0.327	3.9	0.545	9.7	0.676	14.4	0.420	4.7	0.496	10.9	0.463	12.1	0.643	14.7	0.591	2.3
DAny-L	✗	9.4	0.546	1.6	0.350	4.1	0.568	10.0	0.691	15.1	0.447	5.0	0.527	11.5	0.488	12.0	0.632	14.7	0.595	6.9
DPro-1	✗	13.7	0.697	2.2	0.484	3.2	0.460	13.8	0.968	26.9	0.805	2.4	0.283	9.9	0.309	10.5	0.474	24.4	0.936	7.1
DPro-2	✓	16.3	0.834	2.9	0.630	4.3	0.628	15.5	1.081	31.6	0.924	4.1	0.469	14.9	0.487	13.6	0.598	25.6	0.938	7.2
UniD-S	✗	6.9	0.331	**1.2**	**0.253**	2.0	0.271	6.1	0.424	13.3	0.353	2.2	0.232	7.5	0.244	14.5	0.718	16.2	0.582	**0.4**
UniD-B	✗	5.7	0.267	1.3	0.288	**1.4**	**0.198**	5.0	0.344	**11.0**	0.283	**1.5**	0.166	**5.5**	**0.176**	11.4	0.562	14.6	0.516	0.9
UniD-L	✗	6.5	0.298	**1.2**	0.263	1.5	0.199	5.9	0.410	12.5	0.329	**1.5**	**0.154**	5.8	0.181	15.1	0.781	17.6	0.644	2.5
M3D-V	✓	**5.6**	**0.251**	1.7	0.360	1.5	0.219	**3.5**	**0.237**	11.4	**0.271**	**1.5**	0.202	6.4	0.213	**6.5**	**0.278**	**13.0**	**0.350**	3.2
M3D-C	✓	16.7	0.767	2.4	0.511	4.2	0.598	11.8	0.807	34.3	0.927	5.6	0.601	27.0	0.915	19.3	0.864	25.6	0.829	1.8

Table 1. Mean Absolute Error (M) and Absolute Relative Error (A) evaluated across depth ranges and semantic groups, alongside inference time per image. Model abbreviations: DAny (DepthAnything), DPro (DepthPro), UniD (UniDepth-v2), M3D-V (Metric3D-ViT), M3D-C (Metric3D-ConvNext). S/B/L denote Small, Base, and Large variants.



(a) Absolute Relative Error

(b) Mean Absolute Error

(c) Distance Ranges

(d) Semantic Groups

Figure 2. Visualization of depth estimation and segmentation metrics. Top: Errors produced by UniDepth. Bottom: Evaluation groupings from Segformer predictions (vehicles, people, and sky are ignored).

UniDepth (d) does not find at all. Overall, UniDepth and Metric3D seem to agree on how far objects are away from the camera. For example, both estimate the background buildings to be about 80 meters away, whereas DepthPro predicts about 50 meters.

To evaluate these models on our data, we match 20 sample images with LiDAR point clouds recorded at the same position. To quantify the deviation between the measured and predicted distances, we calculate the mean absolute error (MAE) in meters (how far off is the estimation) and the absolute relative error (ARE):

$$ARE = \frac{|depth_{pred} - depth_{true}|}{depth_{true}} \tag{10}$$

The ARE normalizes the mean deviation. Large deviations in the background, therefore, do not influence the metric more than small deviations near the camera. We do not calculate errors for pixels in the image that were segmented as one of the person or vehicle classes, as moving objects were removed from the point clouds, which would otherwise lead to errors in the calculations. In Figure 2, we visualize the ARE (a) and MAE (b) for an example image using the UniDepth model. We can see that the MAE gradually increases with distance from the camera, whereas the normalized ARE remains constant. In addition to the total deviation, we examine the deviation at specific distance intervals: less than 5m, 5-10m, 10-20m, and over 20m. We also analyze different semantic areas in the image by applying a Segformer model trained on the Cityscapes[1] dataset. We group the classes into four semantic groups: flat (road, sidewalk, parking, rail track), construction (building, wall, fence, guard rail, bridge, tunnel), object (pole, pole group, traffic sign, traffic light), and nature (vegetation, terrain) - leaving out vehicles, people, and sky classes. Figure 2 also shows the defined distance ranges (c) and semantic groups (d) for an example image. We also measure the inference time per image on an RTX 4090 GPU.

The results for the different depth ranges are shown in the left half of Table 1. Overall, and for each distance range,

[1] https://huggingface.co/nvidia/segformer-b5-finetuned-cityscapes-1024-1024

(a) Traffic Signs Geo-Localization

	cam	signs		avg dist (m)			
	intr.	all	temp.	total	<10	10–20	>20
Database	–	73% (190)	9% (2)	2.3	–	–	–
Cyclomedia		*of 258*	*of 23*				
DAny-B	✗	52% (134)	26% (6)	6.3	4.6	6.7	6.3
DPro	✓	86% (221)	**100%** (23)	4.8	3.6	4.3	**5.7**
UniD-B	✗	**87% (225)**	**100%** (23)	**4.4**	**3.0**	**3.7**	**5.7**
M3D-V	✓	76% (195)	**100%** (23)	7.3	5.9	7.5	8.8
Mapillary		*of 86*	*of 9*				
DAany-B	✗	28% (24)	11% (1)	6.2	6.4	6.1	**6.2**
DPro	✓	52% (45)	**44%** (4)	6.0	5.9	**5.3**	7.1
UniD-B	✗	**59% (51)**	22% (2)	**5.7**	**5.1**	5.4	6.3
M3D-V	✓	50% (43)	33% (3)	6.6	5.4	7.5	8.8

(b) Ablation Study: Coordinate Error

		Depth	
		Annotation	Estimation
Segm.	Annotation	0	2.17
	Prediction	0.06	2.21

(c) Road Damages Geo-Localization

	cam	avg. distance (m)				
	intr.	total	2–4	4–6	6–8	8–10
DAny-B	✗	**2.34**	2.9	3.2	**2.1**	**2.2**
DPro	✓	3.86	2.5	3.0	4.2	4.6
UniD-B	✗	2.80	**1.5**	**2.2**	2.5	2.8
M3D-V	✓	4.42	2.0	2.5	3.4	3.8

Table 2. Case Studies and ablation. **(a)** Annotated signs present in the database and/or predicted by our system, alongside average distance to the true geo-location. **(b)** Mean coordinate error (in meters) introduced across 20 sample images depending on segmentation and depth sources. **(c)** Coordinate error for road damages across distance intervals. For model abbreviations, see Table 1.

either the UniDepth or the Metric3D model with the ViT backbone produces the results closest to the point clouds. Metric3D has slightly lower overall absolute (MAE) and normalized (ARE) errors. UniDepth achieves better results for smaller distance ranges, while Metric3D is better for objects farther from the camera.

The results for the semantic groups, shown on the right side of Table 1, show a similar overall picture, with UniDepth and Metric3D-ViT achieving the best scores. UniDepth works better for large man-made structures represented by the flat and construction groups, while Metric3D performs better for the object and nature groups. All models struggle with the nature group. They might not catch the details of single branches of a tree. For the models, it is easiest to estimate depth in flat structures. Roads, for example, are very uniform across different datasets.

Regarding inference time, the small UniDepth model is the fastest, followed by the small DepthAnything and the basic UniDepth models. DepthPro takes the longest, even when the focal length is provided. Also, the large version of DepthAnything takes considerable time. DepthPro includes a module to estimate the focal length of the source image. While the camera's true focal length is 960 pixels, the average predicted focal length is 1433 pixels (ranging from 1079 to 1597). Nevertheless, the model that uses the predicted focal length instead of the ground truth achieves better results. Overall, we do not observe a clear advantage for models that use camera intrinsics as input, such as Metric3D, compared to models that do not use them, such as UniDepth.

4.2 Case Study 1: Traffic Signs

We annotated an urban region encompassing 4.74 km of road with 261 traffic signs (23 of which were temporarily mounted). We evaluated our approach using high-resolution imagery from Cyclomedia (786 images from early 2024), commissioned specifically for street recording, as well as crowdsourced images from Mapillary (231 images from October 2023 to April 2024). Based on OpenStreetMap data, the Cyclomedia images provide 100% street coverage, whereas the Mapillary images cover only 28% of streets during the corresponding timeframe. By analyzing the driving direction, we determined the theoretical visibility of the signs: 258 signs should be visible in the Cyclomedia data (including all temporary signs), compared to 86 in the Mapillary data (including 23 temporary signs). A key challenge with crowd-sourced data is GPS inaccuracy. The recorded camera coordinates differ by several meters from their actual positions, resulting in significant localization errors for the objects. For twelve manually measured images, we found a mean deviation of 5.5 meters.

Table 2 (a) details the number of theoretically visible signs that were successfully detected in both image sources. We also compare the existing database entries against these manual annotations. Signs are segmented with a U-Net trained on the A2D2 dataset, then classified with a Vision Transformer (ViT) into the classes of the German Road Traffic Regulations, which are also used in the city's database. The classification model achieved 98.6% test accuracy. We performed an ablation study with 20 images, analyzing how much of the coordinate offset was introduced by the depth estimation vs. segmentation. Results in Table 2 (b) show that the segmentation step contributes only minimally to the overall error. To establish the de-duplication radius, we determined the minimum distance between two signs of the same class. The closest instance was 2.69 meters for two parking signs. Consequently, we adopted a three-meter radius to filter duplicate predictions generated from distinct images in the same area. To match our predictions with the annotations, we assigned each predicted traffic sign to the nearest marked traffic sign of the same class, using a maximum distance threshold of ten meters.

Our results demonstrate that, when applied to high-quality recordings, the proposed system, equipped with the UniDepth model, localizes 87% of all theoretically visible signs. Notably, we detect more signs than are currently recorded in the city's database. The existing database omits nearly all temporarily mounted signs and exhibits a coordinate deviation of over two meters. UniDepth performed best for signs located within 20m of the camera, beyond which its performance is on par with DepthPro. With the crowdsourced Mapillary data, locating all theoretically visible signs is more challenging,

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XII-4-W1-2026-57-2026 | © Author(s) 2026. CC BY 4.0 License.

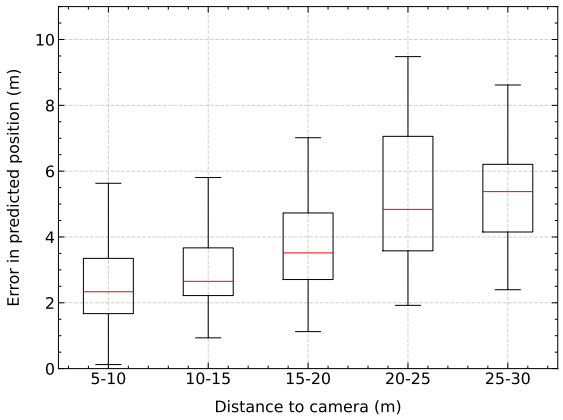


Figure 3. Relationship between true distance (camera-sign) and position error for matched predictions using UniDepth.

although most can still be found. Once again, UniDepth yields the best overall results, particularly at shorter distance intervals. Figure 3 presents the coordinate deviation of the UniDepth model across finer-grained camera-to-sign distance intervals for all matched predictions. As illustrated, the positional error remains notably low for objects in close proximity to the camera, but the deviation increases significantly once the ground-truth distance exceeds 20 meters.

4.3 Case Study 2: Road Pavement Damage

Road pavement damage typically appears at the bottom of the image frame and is situated closer to the camera than traffic signs. We annotated 527 instances of damage across 112 images. For the objects detected in these images, we used our Map Anything Framework to estimate their geo-coordinates and compared them with the ground-truth coordinates.

Table 2 (c) shows that DepthAnything yields the smallest deviation, followed by UniDepth. The localization errors were generally lower than those observed for traffic signs, likely because road pavement damage is closer to the camera, with an average distance of 7.55 meters. This supports our finding that distance estimation is more accurate for nearer objects. UniDepth performs best for damage located within six meters, while DepthAnything is superior for distances exceeding six meters.

5. Discussion

Consultations with local municipal experts revealed that the tolerable spatial error for urban assets depends largely on object density and classification granularity. For instance, densely distributed traffic signs with high class variance are more easily matched to existing database entries than sparsely distributed items, as these factors independently reduce ambiguity. Furthermore, practical insights indicate that even manually measured coordinates in existing municipal databases can deviate by up to five meters while retaining substantial operational value. Based on these practical constraints, our evaluation demonstrates that the MapAnything pipeline is highly effective for mapping objects within a 20-meter distance, where spatial deviations consistently remain below this five-meter operational threshold.

While monocular depth estimation offers a cost-effective, high-frequency alternative to infrequent LiDAR mapping, several practical limitations remain. Real-world street-view imagery is often subject to occlusions, motion blur, and adverse environmental conditions (e.g., poor lighting, rain), which can degrade the accuracy of depth estimation. Future work must systematically assess the precise impact of these environmental factors, alongside inherent sensor inaccuracies in crowd-sourced data, such as minor heading or yaw deviations, on the final coordinate calculations. Finally, while the inference times of the evaluated models (e.g., 0.4s for UniDepth-S) demonstrate near real-time potential, scaling this pipeline for digital twin integration will require standardized spatial data models (e.g., CityGML) and robust spatial indexing to ensure seamless interoperability.

6. Conclusion

In this study, we systematically evaluated MapAnything, a highly scalable, low-cost pipeline that uses monocular metric depth estimation and standard camera geometry to extract 3D geo-coordinates of urban assets from a single 2D image. By validating four SOTA depth models against high-precision LiDAR data, we demonstrated that standard imagery can effectively support the budget-friendly, continuous maintenance of urban digital twins without relying on expensive, specialized hardware.

Furthermore, this evaluation highlights the pipeline's potential to detect temporal changes in the urban environment. By continuously processing imagery from varying dates, city administrations can automatically log spatial discrepancies to flag missing or damaged assets. Whether processing routine municipal street recordings or integrating with citizen reporting platforms, this approach offers an automated, accessible tool to maintain continuously updated spatial overviews of urban infrastructure.

Acknowledgement

The authors acknowledge the financial support by the Federal Ministry of Research, Technology and Space of Germany and by Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the program Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification number: ScaDS.AI

We would also like to thank the German Federal Ministry for Digital and Transport for funding the DiGuRaL project through the mFund program.

References

Bhat, S. F., Birkl, R., Wofk, D., Wonka, P., Müller, M., 2023. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint*.

Bochkovskii, A., Delaunoy, A., Germain, H., Santos, M., Zhou, Y., Richter, S. R., Koltun, V., 2024. Depth pro: Sharp monocular metric depth in less than a second. *arXiv preprint*.

Boller, D., Moy De Vitry, M., D. Wegner, J., Leitão, J. P., 2019. Automated localization of urban drainage in-

frastructure from public-access street-level images. *Urban Water Journal*, 16.

Branson, S., Wegner, J. D., Hall, D., Lang, N., Schindler, K., Perona, P., 2018. From Google Maps to a fine-grained catalog of street trees. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135.

Cabon, Y., Murray, N., Humenberger, M., 2020. Virtual Kitti 2. *arXiv preprint*.

Campbell, A., Both, A., Sun, Q. C., 2019. Detecting and mapping traffic signs from Google Street View images using deep learning and GIS. *Computers, Environment and Urban Systems*, 77.

Cheng, L., Yuan, Y., Xia, N., Chen, S., Chen, Y., Yang, K., Ma, L., Li, M., 2018. Crowd-sourced pictures geolocalization method based on street view images and 3D reconstruction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 141.

Cordts, M., Omran, M., Ramos, S., Scharwächter, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2015. The cityscapes dataset. *CVPR Workshop on the Future of Datasets in Vision*, 2.

Eigen, D., Puhrsch, C., Fergus, R., 2014. Depth map prediction from a single image using a multi-scale deep network. *Neural Information Processing Systems*.

Garg, R., Kumar, B. V., Carneiro, G., Reid, I. D., 2016. Unsupervised cnn for single view depth estimation: Geometry to the rescue. *ECCV*.

Geyer, J., Kassahun, Y., Mahmudi, M., Ricou, X., Durgesh, R., Chung, A. S., Hauswald, L., Pham, V. H., Mühlegg, M., Dorn, S. et al., 2020. A2d2: Audi autonomous driving dataset. *arXiv preprint*.

Godard, C., Aodha, O. M., Brostow, G. J., 2016. Unsupervised Monocular Depth Estimation with Left-Right Consistency. *CVPR*.

Hebbalaguppe, R., Garg, G., Hassan, E., Ghosh, H., Verma, A., 2017. Telecom inventory management via object recognition and localisation on google street view images. *WACV*.

Hu, M., Yin, W., Zhang, C. X., Cai, Z., Long, X., Chen, H., Wang, K., Yu, G., Shen, C., Shen, S., 2024. Metric3D v2: A Versatile Monocular Geometric Foundation Model for Zero-Shot Metric Depth and Surface Normal Estimation. *TPAMI*, 46.

Krylov, V. A., Dahyot, R., 2019. Object Geolocation from Crowdsourced Street Level Imagery. C. Alzate, A. Monreale, H. Assem, A. Bifet, T. S. Buda, B. Caglayan, B. Drury, E. García-Martín, R. Gavaldà, I. Koprinska, S. Kramer, N. Lavesson, M. Madden, I. Molloy, M.-I. Nicolae, M. Sinn (eds), *ECML PKDD Workshops*, Cham.

Krylov, V. A., Kenny, E., Dahyot, R., 2018. Automatic Discovery and Geotagging of Objects from Street View Imagery. *Remote Sensing*, 10.

Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., Li, Z., 2023a. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *AAAI*.

Li, Z., Gao, Y., Hong, Q., Du, Y., Serikawa, S., Zhang, L., 2023b. Keypoint3D: Keypoint-Based and Anchor-Free 3D Object Detection for Autonomous Driving with Monocular Vision. *Remote Sensing*.

Lin, Z., Liu, Z., Xia, Z., Wang, X., Wang, Y., Qi, S., Dong, Y., Dong, N., Zhang, L., Zhu, C., 2024. Rcbevdet: Radar-camera fusion in bird's eye view for 3d object detection. *CVPR*.

Masoumian, A., Rashwan, H. A., Cristiano, J., Asif, M. S., Puig, D., 2022. Monocular Depth Estimation Using Deep Learning: A Review. *Sensors*, 22.

Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P., 2017. The mapillary vistas dataset for semantic understanding of street scenes. *ICCV*.

Novack, T., Vorbeck, L., Lorei, H., Zipf, A., 2020. Towards detecting building facades with graffiti artwork based on street view images. *ISPRS International Journal of Geo-Information*, 9.

Piccinelli, L., Yang, Y.-H., Sakaridis, C., Segu, M., Li, S., van Gool, L., Yu, F., 2024. UniDepth: Universal Monocular Metric Depth Estimation. *CVPR*.

Qi, C. R., Liu, W., Wu, C., Su, H., Guibas, L. J., 2018. Frustum PointNets for 3D Object Detection from RGB-D Data. *CVPR*.

Qin, Z., Wang, J., Lu, Y., 2018. Monogrnet: A geometric reasoning network for monocular 3d object localization. *AAAI*.

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V., 2020. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44.

Vishnani, V., Adhya, A., Bajpai, C., Chimurkar, P., Khandagle, K., 2020. Manhole detection using image processing on google street view imagery. *Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*.

Wegner, J. D., Branson, S., Hall, D., Schindler, K., Perona, P., 2016. Cataloging public objects using aerial and street-level images — urban trees. *CVPR*, Las Vegas, NV, USA.

Yang, L., Kang, B., Huang, Z., Xu, X., Feng, J., Zhao, H., 2024. Depth anything: Unleashing the power of large-scale unlabeled data. *CVPR*.

Yin, W., Zhang, C., Chen, H., Cai, Z., Yu, G., Wang, K., Chen, X., Shen, C., 2023. Metric3D: Towards Zero-shot Metric 3D Prediction from A Single Image. *ICCV*.

Zhang, G., Chen, J., Gao, G., Li, J., Liu, S., Hu, X., 2024. Safdnet: A simple and effective network for fully sparse 3d object detection. *CVPR*.

Zhu, R., Wang, C., Song, Z., Liu, L., Zhang, T., Zhang, Y., 2024. Scaledepth: Decomposing metric depth estimation into scale prediction and relative depth estimation. *arXiv preprint*.