Multi-view Traffic Sign Localization with High Absolute Accuracy in Real-time at the Edge

ABM Musa
musaabm@amazon.com
Amazon
Bellevue, WA, USA

ABSTRACT

In this paper, we present a system to localize traffic signs with high accuracy in real-time at the edge using 3D reconstruction of an environment. We use Structure-from-Motion (SfM) based 3D reconstruction with only a small number of tracked feature points in video frames. The 3D model obtained from SfM alone has an arbitrary scale and orientation, which is not suitable for localizing traffic signs absolutely. Therefore, we use the GNSS locations of the images to scale and orient the 3D model with the real world. Next, we compute the latitude-longitude location of a traffic sign using the mapping between the traffic sign bounding box and corresponding 3D points in the 3D model. We also present the design of other relevant system components such as vehicle location, timing, data synchronization, and sensor calibration, which are required to achieve high accuracy localization. We evaluated our system in city, suburb, and highway scenes on meticulously annotated ground truth datasets. According to evaluation, our system achieves the median localization accuracy of 2.76 meters with high localization success rate and runs on average 15× faster than a comparable baseline system.

CCS CONCEPTS

• **Computing methodologies** → **Reconstruction**; **Object identification**; **Vision for robotics**; • **Information systems** → **Location based services**; **Geographic information systems**.

KEYWORDS

Localization, 3D Reconstruction, Mapping, Traffic sign

ACM Reference Format:
ABM Musa. 2022. Multi-view Traffic Sign Localization with High Absolute Accuracy in Real-time at the Edge. In *The 30th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '22), November 1–4, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3557915.3561020

1 INTRODUCTION

Detection and localization of traffic signs are the first steps in understanding road semantics. This is important for a wide variety of applications such as Advanced Driver Assistance Systems (ADAS), high-definition mapping, autonomous driving, navigation, safety, route planning, and map error correction. This also has both real-time and offline use-cases. For example, we can alert a driver if she is about to run over a stop sign through accurate detection and localization of the stop sign in real-time. An example of an offline use case is routing where a one-way sign can confirm the road directionality, and we can route drivers to the correct direction only. Additionally, traffic signs can often change in the real world, thus continuous detection and localization of them to update map data is important. While there has been a lot of research on traffic sign detection and recognition, the research on traffic sign localization is limited.

Accurate absolute localization of traffic signs has many challenges. First, traffic signs are relatively small compared to their surrounding; thus detection, recognition, and localization can be hard. Second, an image can contain multiple traffic signs that have varying locations on the street, and it can be difficult to infer their spatial distribution. Third, traffic signs are often dependent on other contexts such as a particular lane. Finally, traffic signs from an adjacent street or even a perpendicular street can be visible in an image and can be detected, which have no relation to the current street.

The localization accuracy requirement for traffic signs depends on the environment and street topology. For example, a 50-meter localization error might be good enough to understand a sign's context in a highway or suburb street where there is no nearby sign of the same type. However, in dense urban areas, where there are many signs close to each other, in a multi-lane street, and at a complex intersection, the localization accuracy requirement is a few meters.

In this paper, we present a system that addresses these challenges and localizes traffic signs with high accuracy using a single camera, and runs in real-time at an edge device with low computing power. Here, we use Structure-from-Motion (SfM) [2] to reconstruct an area in 3D so that we have the accurate relative locations of various traffic signs in the scene. The benefit of SfM is that it enables the accurate 3D reconstruction of a scene. However, SfM algorithms are generally very compute-intensive and unsuitable for real-time applications at the edge. At the edge, we need to process the video and finish all computations in real-time as the vehicle drives and the video stream arrives, which is usually very challenging for traditional SfM pipeline. In this work, we optimize various stages of our SfM pipeline and apply many techniques to achieve real-time runtime on a low-compute edge device with ample compute capacity preserved. Another limitation of SfM is that it does not provide the absolute scale of the 3D reconstruction To address this, we use the Global Navigation Satellite System (GNSS) locations of

SIGSPATIAL '22, November 1–4, 2022, Seattle, WA, USA

ACM ISBN 978-1-4503-9529-8/22/11.
https://doi.org/10.1145/3557915.3561020

the vehicle to obtain the true scale and orientation of the reconstructed scene. Additionally, geolocation accuracy of the vehicle and video frames, accurate synchronization between GNSS locations and video frames, intrinsic camera calibration, and multi-sensor extrinsic calibration, which are important to achieve high accuracy localization. Furthermore, signs typically get detected and localized repeatedly in different video frames as a vehicle moves as well as during different drives on the same street. Therefore, it is important to consider the de-duplication and localization of signs at the instance level. In this work, we consider all these topics combinedly and present an end-to-end system. The primary contributions of this paper are:

- To the best of our knowledge, we present the first system that combines a very high-accuracy low-cost GNSS system and a single camera for traffic sign localization, which achieves a few-meter accuracy in the real-world.
- A 3D reconstruction-based traffic sign localization system that achieves high accuracy and fast runtime on the edge.
- Design and analysis of various components of an effective localization system consisting of high accuracy GNSS location, multi-sensor data synchronization, and multi-sensor calibration.
- A complete and deployable system with both per-detection and per-instance localization.
- Extensive evaluation with meticulously annotated ground-truth data.

2 RELATED WORK AND BACKGROUND

There has been a significant amount of work in traffic sign detection and recognition. German Traffic Sign Dataset [34] is one of the earliest and most widely used datasets in this area. The past work [27] primarily used classical vision techniques such as edge and shape detection to recognize signs. Recent works on detection and recognition primarily uses deep learning methods [25] such as Retinanet [19], YOLO [31] or some form of custom architecture [8]. Despite significant research on traffic sign detection and recognition, it is not yet a solved problem at a large-scale and in various geographic regions, which is evident by the recent release of large datasets by Mapillary [8] and others [38, 42] .

Compared to traffic sign detection and recognition, there has been inadequate research on traffic sign localization. Due to recent growth in autonomous driving and high-definition mapping [3, 20], there has been an increased interest in large scale and robust traffic sign localization systems. However, many of these systems [38, 41] use a combination of LIDAR [18] and multiple cameras, which are very expensive.

It is also possible to use stereo camera systems [6] to compute depth and thus localize traffic signs or other objects. However, it is hard to calibrate the stereo cameras and keep the calibration accurate over time, especially for a large baseline, which is desired for outdoor scenes. Stereo cameras are more costly as well. In contrast, we use a low-cost single-camera solution and still achieve very high accuracy.

Most of the vision-based techniques use some sort of 3D information such as triangulation [29, 35] for localization. It is possible to reconstruct a 3D environment using multiple images/views from different perspectives using SfM [2]. However, SfM provides an up-to-a-scale 3D model, which does not align with the real world. To obtain the actual scale and orientation, an external sensor such as GNSS [15] or Inertial Measurement Unit (IMU) [33] is required.

While SfM is primarily an offline process, Simultaneous Localization and Mapping (SLAM) is more real-time. SLAM [28, 30] constructs a map from a sequence of frames while simultaneously localizing all frames with respect to that map. However, SLAM encounters drift and accumulates errors because of dead-reckoning. The primary way of detecting and correcting the accumulated error is loop-closing [37], where the previously visited places are identified. Image matching is used for identifying the same location. One major drawback with SLAM techniques is that the localization is only with respect to the locally constructed map, and global localization is not considered except for the loop closing. SLAM also often fuses IMU [4] to reduce the dead-reckoning error and obtain an absolute scale.

Most of the SfM and SLAM solutions generally evaluate themselves in terms of relative localization accuracy, error growth with respect to the total movement length, etc. In contrast, we combine a high-accuracy GNSS system, 3D reconstruction, and accurate data synchronization to achieve high absolute accuracy localization in real-time. This makes our system readily deployable and usable for real-world mapping needs.

Deep learning-based depth estimators [10, 13] are a growing area of research nowadays, which can be potentially used for traffic sign localization. While these methods work reasonably well for large objects, they often do not work for small objects such as traffic signs. They are often prohibitively expensive to run on an edge device with low computing power. Additionally, these methods often perform poorly if the test data significantly differs from the training data.

3 LOCALIZATION SYSTEM

We use SfM-based 3D reconstruction for sign localization. Now, there are some failure scenarios in SfM in the real-world outdoor scene. First, SfM can fail if the video/image quality is bad. Some examples of bad image quality are rain and wiper movement, strong reflections, and exposure problems. Second, SfM relies on matching features between image pairs and matching can fail due to repetitive textures (e.g., brick wall). Third, the camera view can be blocked by some objects such as a large truck on the road. Fourth, it is possible to have no 3D point at all in the reconstruction that corresponds to a traffic sign. This might happen due to low resolution or blurriness of an image, or triangulation, or bundle adjustment filtering. Figure 1 shows some examples of these types of failure scenarios. We designed our system to minimize the effect of these failure scenarios.

Furthermore, it is very challenging to achieve both high accuracy and fast runtime in a SLAM or SfM-based system. SLAM algorithms typically use only triangulation and avoid expensive bundle adjustment process to run in real-time. This results in high drift over time causing high localization error. They correct for the high error, using loop closure, which is expensive and unsuitable for real-time use cases. On the other hand, SfM algorithms use bundle adjustment and they are a lot more robust against noise. However, they are very

expensive to run and not suitable for real-time use cases. In this work, we combine the ideas from both SLAM and SfM techniques and optimize various stages of our 3D reconstruction pipeline to achieve both high accuracy and a fast runtime. These include the computation of a small number of features, feature tracking instead of feature matching, and small-scale reconstruction. Additionally, we present an end-to-end system that considers other important details at the system level such as vehicle location accuracy, time synchronization, intrinsic and extrinsic calibration.

Note that we focus on sign localization alone in this paper, and assume that sign bounding boxes in video frames are provided by a sign detection and classification algorithm.

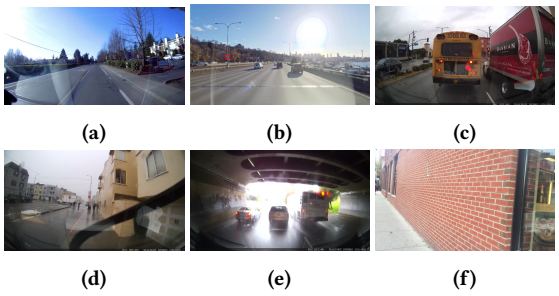


Figure 1: Challenging scenarios. (a) High reflection, (b) Sun flare, (c) View blockage, (d) Rain and wiper movement, (e) Exposure problem, (f) Repeated pattern.

3.1 Overview

Figure 2 shows a high-level overview of our system. We take sign bounding boxes in a video frame, subsequent video frames, and the GNSS location of all frames as input, and we output the geo-locations (latitude, longitude, altitude) of the traffic signs. During SfM 3D reconstruction, we compute a small number of features/key-points within sign bounding boxes and the rest of the frame. We track these features in subsequent frames to get the correspondences between frames. We then reconstruct a very sparse 3D model of the world from these correspondences using SfM. Since the 3D model from SfM is up to a scale, we use the GNSS geotags of the frames to geo-reference and align the images in the real world. This also scales and orients the sparse point cloud to the real world and geo-references them accurately. Next, from a bounding box of a traffic sign on the frame, we find corresponding 3D points in the geo-referenced point cloud and obtain the geo-locations of those points. These geo-locations usually form a tight cluster, and we take the cluster center as the final location for a traffic sign. We discuss all these steps in detail in the sections below as well as other relevant details.

3.2 Features and tracking

In the traditional SfM/SLAM process, the goal often is to generate as many 3D points as possible to reconstruct the scene in detail. Accordingly, they compute as many features as possible, track them as long as possible, and compute new features for each incoming frame, making the process compute-intensive and slow. We can successfully localize a sign if we get at least one 3D point reconstructed for the features from a sign bounding box. Therefore, we only compute a small number of features within each sign bounding box. Additionally, we compute a small number of features outside since 3D reconstruction works best if the features are well distributed over a frame. We use ORB [32] features here, but experimented with other features as well.

Computing and retaining features within sign bounding boxes specifically is very useful to get 3D reconstructed points for a sign thus successful localization. This resulted in a high success rate of sign localization in our system compared to the generic SfM process that does not specifically compute features for a sign (see §4.6). Figure 3 shows a visualization of our computed features within sign bounding boxes and outside.

We use optical flow [21] to track the features in subsequent frames to get the correspondences between the frames. This is unlike the traditional SfM process, where feature matching is typically used, which is significantly more expensive. We adaptively track the feature points up to a maximum duration (2 to 4 seconds) or a maximum distance (10 to 20 meters) or until there are no more tracked features for a sign. Unlike traditional SfM/SLAM, this short tracking duration keeps the reconstruction size small and achieves faster runtime, which is sufficient for sign localization. Note that feature tracking with the optical flow is noisier compared to feature matching. To mitigate this to a large extent, we also use bi-directional optical flow between a pair of frames.

3.3 3D reconstruction

The tracked features provide the correspondences between video frames, which is required for SfM-based 3D reconstruction. Now, we need correspondences between each pair (not only adjacent) of frames. However, the optical flow tracking provides correspondence between only the adjacent pair of frames. To solve this, we maintain an efficient graph data structure with the complete track of correspondences among a sequence of frames and generate correspondences between any pair of frames required by the SfM process. This is again unlike the traditional SfM process where features are typically matched between an arbitrary pair of frames directly and there is no need to keep track of transitivity.

Next, we run both triangulation and bundle adjustment to obtain an accurate 3D model of the scene. This is different from traditional SLAM algorithms, which avoid expensive bundle adjustment to achieve real-time performance. However, SLAM algorithms [28, 30] encounter incremental drift due to avoiding bundle adjustment, which often is corrected by loop-closure [37]. Loop closure uses image feature matching to find the same place again the correct the overall drift. However, loop closure itself is expensive, and visiting the same place again is not guaranteed in outdoor scenarios like ours.

Since we only use a small number of correspondences, strategically taken from sign bounding boxes and outside as well as a few seconds of tracking, we can run bundle adjustment, which results in very fast and high accuracy 3D reconstruction.

3.4 Geo-referencing a 3D model

The 3D reconstructed model of the scene has an arbitrary scale and orientation and does not correspond to the real world, which is

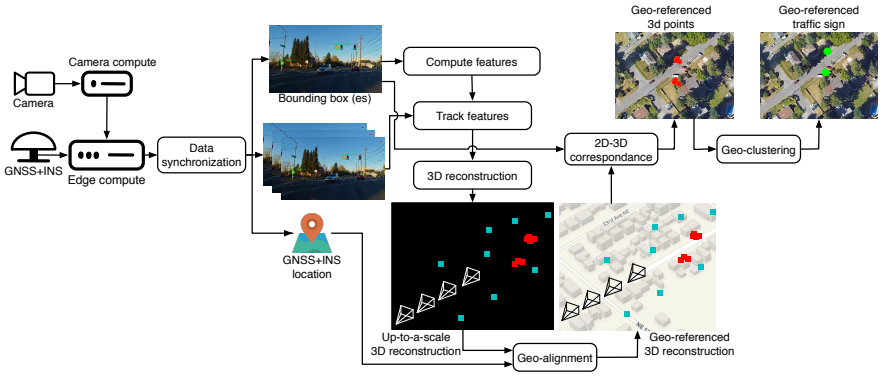


Figure 2: System Architecture.

Figure 3: Visualization of features (red: sign features, green: outside features).

not suitable for localizing traffic signs absolutely. Note that the 3D model is relatively consistent and provides a correct representation of the scene. However, the model is not absolute and the model does not correspond to the real-world dimensions and orientation.

Now, the reconstructed 3D model has 6 Degree-of-Freedom (DoF) poses of the frames (also referred to as cameras) in the arbitrary (but relatively correct) coordinate frame. We transform this arbitrary coordinate frame according to the GPS locations of all the frames, which aligns the cameras accurately to the real world. Since the whole 3D model is rigid, this transformation also scales and orient all 3D points representing the scene to the real world. Note that we only need to take 3-DoF (XYZ or latitude-longitude-height) from the GPS as the camera orientation from the original reconstruction does not change. Furthermore, the 3D scene points do not have any orientation.

The absolute sign localization accuracy is highly dependent on the accuracy of the video frame locations, which are used for accurate geo-referencing of the 3D model. Thus, it is important to consider high accuracy location for the vehicle. Additionally, we need careful design and implementation for accurate time synchronization and video encoding-decoding pipeline so that we do not have an error in video frame location despite the vehicle location being accurate. We discuss these in details in §3.7.

Note that the GNSS localization for the vehicle is still the only way to automatically and absolutely localize and map objects such as traffic signs. These apply to any traditional SLAM or SfM algorithm, both real-time and offline and for both small and large-scale reconstruction. Any advanced techniques such as loop-closure can only improve relative accuracy. Sometimes, these systems use human annotation to manually align and scale the model accurately to the real world, which is often time-consuming and non-salable. However, here we achieve automatic high-accuracy absolute localization with a very high-accuracy but low-cost precision GNSS system that includes Real-Time Kinematic (RTK) [7] correction and Inertial Navigation System (INS) [9].

3.5 Sign localization

Once we have geo-referenced camera poses and point cloud, we can localize the traffic signs. We know the 3D points that are generated from a particular image. Accordingly, we extract the points that come from a traffic sign bounding box. Figure 4 shows a visualization of correspondence between traffic sign bounding boxes from an image and sparse 3D points in the model. Note that the 3D points shown here correspond to the 2D features shown in 3.

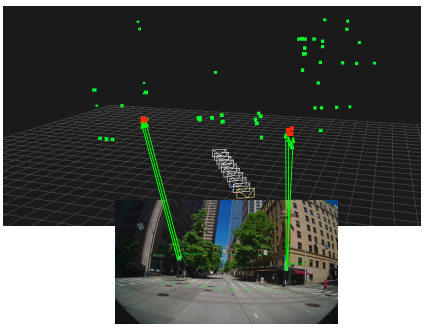


Figure 4: Correspondence between sign 2D bounding boxes and 3D points. Red points are from sign bounding boxes and green points are from outside. The camera icons show the frame locations.

The camera poses and the 3D points are initially in a Cartesian coordinate frame. Hence, we transform the points corresponding to the bounding box from the Cartesian frame to the geo-coordinate frame (latitude, longitude, height). These points usually fit in a very tight cluster as they come from the same traffic sign, and we take the center of this cluster to obtain the geo-reference coordinate

of the traffic sign. Note that we output the sign height in addition to latitude-longitude, which can be useful for semantics and to disambiguate among nearby signs.

Figure 5 shows a few qualitative examples of the final localization of traffic signs. It shows images with multiple traffic signs and their localization on the map. The combined image and map view show that our system achieves very high accuracy localization. Note that we can even localize signs at night if there is some light in the scene and we detect some features as shown by the right-bottom image example.

3.6 De-duplication

We discussed the localization of signs detected in a single frame until now. However, as the vehicle travels, the same sign gets detected and localized multiple times at different frames of the same video. We ultimately want to localize and map a sign in the real world uniquely at the instance level. Hence, we want the multiple localization of the same sign with high accuracy so that they are tightly clustered. Additionally, it is a lot better to not localize a sign rather than localize it incorrectly. A sign will be visible in many frames and it is sufficient to localize successfully once for any of those frames. Therefore we actively try to abort the localization of a sign if there is a chance of poor quality localization. Aborting quickly also reduces computation and runtime. We discuss these abort and failure cases more in §4.4.

Similar to the same sign being localized multiple times in a single drive, the same sign can be localized multiple times for different drives too, and we need high accuracy localization to successfully group and cluster them. Note that the same drive de-duplication can be also achieved with instance-level tracking of the sign, but the only way to achieve de-duplication for multiple drives is by localizing a sign with high accuracy each time.

Figure 6 shows the complete picture of our localization system with multiple detections of the same signs at different frames and different drives, their de-duplication to obtain an instance-level location. It shows that we achieve high accuracy across all frame-level localization, which forms a tight cluster, resulting in accurate instance-level localization.

3.7 Location, timing, and calibration

The SfM reconstruction provides a 3D model where the camera poses and sign locations are relatively consistent and accurate. However, the absolute location accuracy of them in the real world depends on the accuracy of the vehicle/camera location, synchronization of camera frames and GNSS locations through accurate timing, and calibration. We discuss them in detail below.

3.7.1 Location. The absolute sign localization accuracy depends on the GNSS location accuracy of the vehicle and the camera frames since we use the frame locations to geo-reference the reconstructed 3D model (see §3.4). Typical GNSS receivers found in commodity solutions (e.g., smartphones, vehicle trackers, etc.) can have 5 to 10-meter error [39] in good conditions (e.g., highway), and can have over 100-meter error in sub-optimal conditions (e.g., downtown). There are survey-grade GNSS localization systems that combine RTK [7] correction data from a nearby base station along with GNSS carrier phase [11] to achieve centimeter-accurate locations in ideal conditions. These GNSS systems can be fused with Inertial Navigation Systems (INS) [9] to achieve very high localization accuracy in difficult conditions such as GNSS-denied downtown. We use an industry-standard reference survey system that achieves centimeter to decimeter accuracy in all areas, and we use the location data from this system as ground truth. The reference systems are in general very expensive. However, there is a lot of progress recently in low-cost systems that combine GNSS, INS, and RTK. We use such a system (we call this precision system) in this work, which achieves centimeter to decimeter accuracy in highway scenarios and around 5-meter median accuracy in dense downtown and costs a few hundred US dollars.

3.7.2 Timing. The GNSS location and camera frames are independent data streams, and they come at different frequencies. In our system, the GNSS locations come at 10Hz, and camera frames come at 30Hz. The only way to synchronize these two streams is through their timestamps. Hence, accurate timing is a fundamental requirement for accurate data synchronization. However, achieving highly accurate timing is very hard in reality. The best source of accurate timing is GNSS [16], which typically has an accuracy of a few microseconds. However, the difficult part is to get that accurate timing propagated to the camera frames. We need an accurate timestamp for each frame trigger, but the computers and operating systems can introduce large and variable latency.

Figure 7 shows our setup for accurate timing. The Network-Time-Protocol (NTP) [26] is the first step for time synchronization, which corrects the computer clock using the NTP servers over the internet. However, NTP can have 50 to 100-millisecond errors. To fix this high error at the edge computer, we use GNSS Pulse-Per-Second (PPS) [17] signal, which corrects the high error by the regular NTP. Now, our camera is connected to a separate computer (used for video encoding and other basic image processing) over Ethernet. The camera computer has a separate clock and it can have a different time. To synchronize the time of the camera computer, we use Ethernet-based Precision-Time-Protocol (PTP) [1]. Next, we need to put the accurate time from the camera computer to camera frames, and we use custom Supplemental Enhancement Information (SEI) [14] messages in H.264 protocol to embed the timestamps in the encoded video. This video stream gets transmitted to the edge computer over Ethernet, and we decode the video frames and use them in our system. According to our estimation, we achieved end-to-end timing accuracy within 2 milliseconds by combining all these methods.

If the timestamps are inaccurate, the frame location will be ahead or behind the actual GNSS location depending on the timing error. Moreover, the location error is proportional to the vehicle speed for a fixed timing error. For example, a 100 milliseconds timing error will cause a 3.13-meter location error at 70 miles/hour speed on a highway and a 1.56-meter error at 35 miles/hour speed on a city street. Appendix A shows examples of location error due to timing error.

3.7.3 Calibration. To achieve high accuracy sign localization, both intrinsic camera calibration and extrinsic calibration between sensors are important. Intrinsic camera calibration [40] provides the camera pinhole model parameters and lens distortion parameters, which are required to accurately map the real-world projection

Figure 5: Qualitative localization accuracy of traffic signs. Traffic sign bounding boxes at an image and corresponding geo-referencing locations are shown on the map pointed by the arrow. On the map view, we show the frame location with a green square and sign locations with blue circles. The map view also shows white lines between a frame and localized signs. The right-bottom image shows a night example.

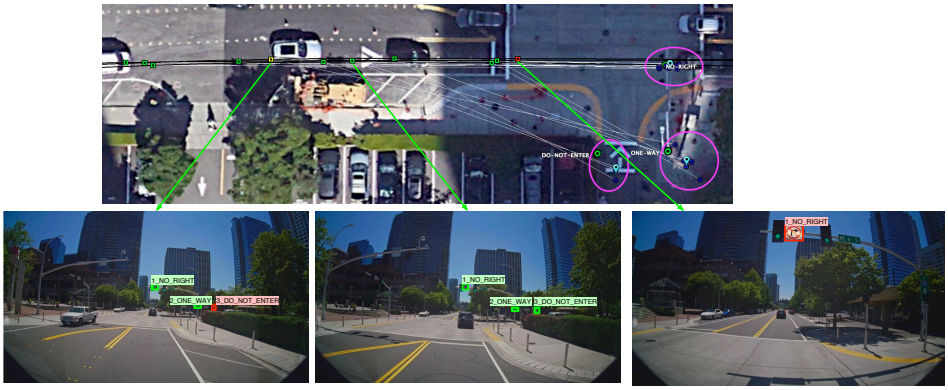


Figure 6: Multiple localization from different frames of a single drive, different drives, and de-duplication. The blue circles, light blue pins, and larger green circles show frame-level sign localization, instance-level sign localization, and ground truth sign locations respectively. Pink ovals mark them in groups. The squares show the frame locations (red square: no detected sign is localized, yellow square: a subset of all detected signs are localized, green square: all detected signs are localized). We also show example images for a red, green, and, yellow square with pointed arrows.

to the sensor plane. Additionally, extrinsic calibration (x-y-z-roll-pitch-yaw) between two sensors is important. In our case, GNSS locations are generated at the antenna center, but we need locations at the camera. Therefore, we need to know the extrinsic calibration between the GNSS antenna and the camera. Note that extrinsic calibration error introduces a fixed error in sign localization due to a fixed offset. This is unlike fixed timing error as the sign localization error due to timing error depends on the vehicle speed.

We currently have partial intrinsic calibration (only focal length) for our camera. We use an 8-megapixel camera with a large field of view (130 degrees). Due to both high megapixels and large field-of-view, the calibration for distortion parameters is complex and requires a very large flat checkerboard, which is hard and expensive to manufacture. However, we are currently working on this calibration process and in the future, we expect to have our localization error to be further reduced by using accurate intrinsic calibration. We also use extrinsic calibration to remove the fixed error in the location of video frames.

4 EVALUATION

We present the evaluation of our system in this section. Here, we discuss evaluation challenges, our evaluation dataset, and detailed system performance such as localization accuracy, runtime, and

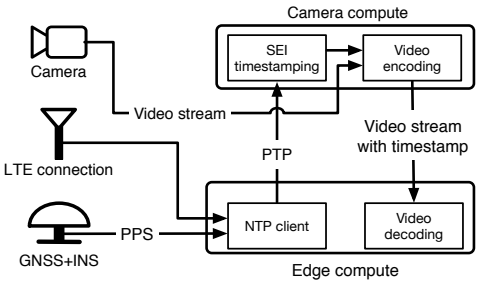


Figure 7: Setup for accurate timing.

the impact of algorithm parameters. We also compare our system with a baseline SfM method and a learning-based method.

4.1 Evaluation challenges

Collecting ground-truth data of sign locations at a large scale is very challenging. To the best of our knowledge, there is no large-scale public traffic sign location dataset available today. The largest dataset on traffic signs today is available from Mapillary [23], but it only provides sign bounding box annotations for detection and recognition but not sign locations.

One possible source of large-scale traffic sign location data is one of many autonomous driving datasets available today such as KITTI [12], nuScenes [5], Lyft [22], and Waymo [36]. However, these datasets primarily have annotations for large objects such as vehicles and pedestrians, but not traffic signs. Only the Waymo dataset has traffic signs annotated in the LIDAR point cloud. It is possible to re-project the LIDAR annotation on the image plane to get sign bounding boxes on the image. However, the Waymo dataset does not provide GNSS coordinates of the images, which is an essential requirement for our system.

Due to the unavailability of a large-scale traffic sign dataset, we painstakingly and manually annotated the location of a large number of signs observed in our recorded videos. In our annotation process, we jointly look into our camera image, satellite view, and sometimes Google street view to estimate the location of signs. This is illustrated in Figure 8. Accurate annotation of traffic sign location on the map, in reality, is hard and time-consuming even from combining all these viewpoints. For example, we need to sometimes look into the shadow of a sign or a pole to estimate the sign location on the map. However, The shadows can vary depending on the Sun's location in the sky and can introduce ambiguity. Additionally, we sometimes use Google Street View to look into a sign from different perspectives to better estimate its location. Since this annotation process is hard and ambiguous, we worked with two annotators and they need to agree on a sign location within one meter.

Note that the accuracy of this annotation process depends on the accuracy of Google Earth's image tile geo-referencing accuracy. We verified that the accuracy of Google Earth is quite high in general. For verification, we used a survey-grade GNSS receiver and compared the location of some landmarks (e.g., a roof corner, crosswalk center) obtained from both the GNSS receiver and Google Earth, and they generally agreed within decimeter accuracy.

4.2 Ground Truth

We evaluate our system with over 1000 annotated and detected signs in total from different datasets and experimental setups. First, we annotated traffic signs and their location in a 42-mile route that takes approximately two hours to drive and combines a dense downtown, a small downtown, suburbs, and highways. In this dataset, we annotated the location of more than 250 signs from different frames that are well distributed in different areas, located at different distances from the vehicle, and have varying lighting conditions. Second, we annotated the location of around 150 signs from dashcam videos. Finally, we used a sign detector model, which was trained to detect 7 regulatory signs (e.g., one-way, do-not-enter, stop, etc.). This detector detected 570 signs in our 42-mile route, and we used these detections and our localization for various statistical analyses.

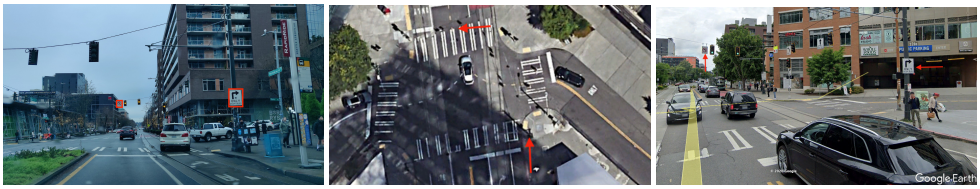
We evaluate our system for the exact same video frames for which annotations are available. This enables tracking the system performance exactly as we know annotation and detailed information for all individual video frames. However, the end goal of a practical and deployed system is to successfully localize all signs in the real world uniquely or at an instance level with high accuracy. Therefore, an equally important evaluation criterion is to compare the performance of a localization system for a drive relative to all existing signs in the real world. The existing signs in the real world are unique and constant (except for signs that can change in a longer time period) and can be annotated from a past drive. The benefit of this type of evaluation is that any number of drives can be evaluated without repeated annotation of traffic sign locations in that drive's video frames. Accordingly, in addition to the exact annotated video, we evaluate our system for a drive with respect to the existing ground truth real-world sign locations.

4.3 Experimental setup

We run and evaluate our system on an edge computer with a 6-core CPU and 8 GB memory. We perform both offline and online evaluations. In offline evaluation, we run our system on pre-recorded video frames in batch mode. In online evaluation, we run our system in real-time as the vehicle drives and captures video. All our computation currently is CPU only. Our system use only one core, but optical flow and SfM bundle adjustment stages in our pipeline can take a maximum of 3 cores. Our system consumes approximately 250MB to 500MB of memory for 2s to 4s maximum tracking duration. However, we keep all tracked frames in the memory currently, which is not an essential requirement. We will be able to easily move to a few megabytes of constant memory if we don't keep all tracked frames in the memory. Note that not consuming all available CPU and memory is important for other tasks running in a deployed system. We use an automotive-grade camera and 720p video frames.

4.4 Localization success and accuracy

It is important to consider both success rate and accuracy for localization. As discussed in §3.6, we consider both frame level and instance level localization. Since a unique sign in the real world gets detected at multiple frames, it is sufficient to localize them for only one of those detections.



(a) Image with sign bounding boxes. **(b) Satellite view.** **(c) Google Street view.**

Figure 8: Ground truth sign-location annotation process. We jointly look into all three views to annotate the sign locations on the map.

4.4.1 Success rate. Table 1 shows the percentage of localization success and abort/failure for all detections at frame-level for our annotated 42-mile route. Here, we successfully localized 75.69% of all detected signs at the frame level. Table 1 also shows various abort and failure cases and their percentage.

As discussed in §3.6, we actively try to abort early in the localization pipeline to save computation and reduce runtime if the end localization is likely to be unsuccessful. For example, we abort without attempting 3D reconstruction if the vehicle did not move a minimum distance, since the SfM 3D reconstruction process needs some perspective change for the localization to be successful. The vehicle can often be stopped at a stop sign or traffic light or can be parked. We also abort if features are tracked for only a few frames, making 3D reconstruction highly likely to be unsuccessful. Feature tracking can fail only after a few frames for various reasons such as blockage of the camera view by another vehicle, sun flare, wiper movement during rain, etc. Our final abort reason is no feature detection for a sign. This can happen occasionally in very low-light conditions or due to sun flare.

It is not always possible to abort quickly, and 3D attempted reconstruction can eventually fail. This can happen if the feature points are not well distributed on the frame or if most of the feature points come from moving objects in the scene such as vehicles. Finally, even if the 3D reconstruction is successful, we may not get a 3D point for a sign to be able to successfully localize it. For example, this can happen if a sign is at the corner of the frame and quickly goes out of camera view where we continue to track feature points of some other signs.

Overall, Table 1 shows that we abort in 9.42% cases, and fail in 14.9% cases. The majority (12.94%) of the failures are due to "no-sign-3d-point", which frequently happens when a sign is at the corner of a frame and quickly goes out of view.

Now, the important point to note here is that the failure at the frame level is not the most important fact and we need to look into our success at the instance level. As discussed in §3.6, a sign is detected at many video frames and it is enough to successfully localize the sign for one of those detections. Specifically, our sign localization success rate increased to 83.2% at the instance level from the success rate of 76.69% at the frame level.

4.4.2 Accuracy. Figure 9 shows the sign localization error in details. Here, we show our localization accuracy for dense downtown and outside (a combination of a small downtown, highway, and suburb) and for both reference GNSS system and low-cost precision GNSS system. Our localization accuracy with the precision GNSS is very similar to the reference GNSS outside. Specifically, the median localization errors are 2.76m and 2.95m for reference GNSS and precision GNSS respectively. However, the accuracy drops in dense downtown for precision GNSS, and this is due to the high GNSS error in dense downtown. We show an example of a high sign localization error due to a high GNSS error in Appendix B.

Figure 10 shows the localization error for both frame-level and instance-level. Instance-level error is significantly lower, especially at a higher percentile as some outlier signs are thrown away by geo-clustering.

Finally, note that we currently have two limitations in our system. First, we encounter occasional frame drops during video encoding and decoding. Each frame-drop causes 33 milliseconds of timing error and corresponding localization error depending on the vehicle speed. Second, we did not yet fully calibrate and camera. Once these two limitations are removed, we expect our localization accuracy will be higher.

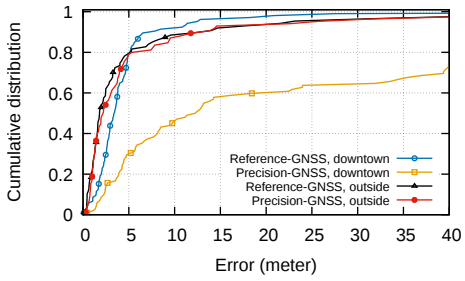


Figure 9: CDF of localization error for downtown and outside.

4.5 Runtime

The end-to-end median runtime of our localization system for one or multiple detections in a video frame is 1.29 seconds, which is very fast. The number of signs detected at a single frame has very little impact on the runtime since we reconstruct all signs together. Note that this is the total runtime for end-to-end localization that combines multiple frames and reconstructs a scene in 3D. We track a sign until it is visible or up to a maximum tracking duration (see §3.2). Note that the sign visibility depends on the relative distance between the vehicle and the sign, the vehicle's speed, and the vehicle's driving pattern (e.g., straight vs turning).

Success/Abort/Failure	Percentage	Category	Description
Success	76.69	-	Successful localization
No-sign-3d-point	12.94	Failure	There is no reconstructed 3D point corresponding to a sign
Very-few-frames	4.71	Abort	Sign and/or outside features are not tracked for many frames
Small-cumulative-distance	3.53	Abort	The vehicle moved a little or not at all
No-3d-reconstruction	1.96	Failure	3D reconstruction is unsuccessful
No-sign-features	1.18	Abort	No feature point is detected at a sign

Table 1: Localization success, abort, and failure for all detections at frame-level.

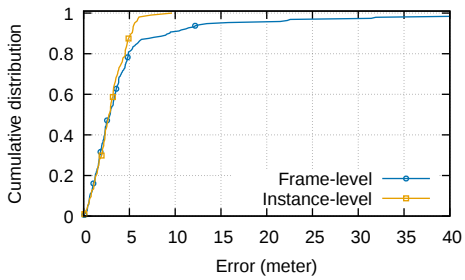


Figure 10: CDF of localization error for instance-level and frame-level.

The fast runtime enables us to run our system in real-time at the edge as a vehicle drives. According to our deployment and experiments in running vehicles, our system can process video at 30 FPS as the frames arrive and never encounter any backlog of frames or dropped frames. In the real world, we do not detect traffic signs in every video frame. Therefore, we do not need to run 3D reconstruction and localization for every frame. This is why a 1.29 second median runtime is very fast and more than sufficient to achieve real-time performance.

The runtime is significantly faster than real-time if we process in batch mode. For example, for the 42-mile two-hour drive, we detected 570 signs, and our system took 14 minutes to localize them. Fast processing and low computing are important to reserve the capacity of the edge computer for other tasks.

4.6 Comparison with a baseline SfM method

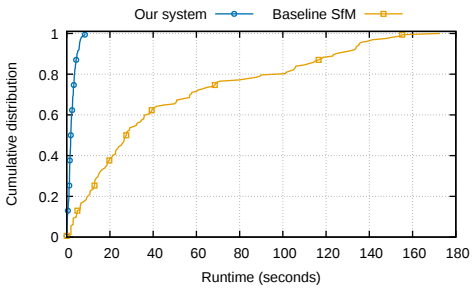


Figure 11: Runtime comparison with a baseline SfM method.

Figure 11 compares the runtime and error of our system to a baseline SfM 3D reconstruction and localization algorithm. The baseline here is the default OpenSfM [24] reconstruction process combing feature extraction, feature matching (instead of tracking), and 3D reconstruction. The runtime here is the total time for processing a frame with detections (the number of detected signs has very little impact on the runtime) along with some next frames for 3D reconstruction. The median runtime of our algorithm is 15× faster, and the 90th percentile runtime of our algorithm is 25× faster compared to the baseline. Note that the localization error of our algorithm is very similar to the baseline.

The localization success rate of our system is also much higher compared to the baseline. As shown in Table 1, our system's success rate is 76.69% at the frame level. The baseline's success rate is only 60.39%. This is due to a high increase of failure for "No-sign-3d-point" cases. Unlike our system, the baseline does not specifically compute and retain features for a sign bounding box, causing no 3D points for a sign.

In summary, our system runs orders of magnitude faster with a higher success rate and achieves very similar localization accuracy compared to the baseline.

4.7 Comparison with a learning-based method

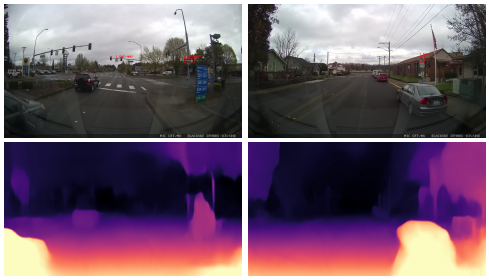


Figure 12: Depth output by Monodepth2 [13]. Detected traffic signs are shown with a bounding box on the top row images. The bottom row shows the estimated depth.

Figure 12 shows the depth output from Monodepth2 [13], which is one of the recent and best learning-based monocular depth estimation method. While these types of learning-based methods work quite well for larger objects (e.g., a car) or in the near distance, they usually do not work well or at all for small objects such as traffic signs as shown in Figure 12. In contrast, our SfM method works almost always even if the sign is very far as long as we can detect a feature.

5 CONCLUSION

In conclusion, we present a system to localize traffic signs accurately using an SfM method with numerous optimizations. Our system achieves a localization error of only a few meters, has a high success rate, and runs faster than real-time on a low-compute edge device. In the future, we plan to improve our system further by reducing noise in feature tracking, improving GNSS accuracy even more with wheel odometry and INS improvements, better intrinsic and extrinsic calibration, and estimation of sign directionality (discussed in Appendix C).

REFERENCES

[1] 2020. IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008)* (2020), 1–499. https://doi.org/10.1109/IEEESTD.2020.9120376

[2] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. 2011. Building Rome in a Day. *Commun. ACM* 54, 10 (Oct. 2011), 8 pages. https://doi.org/10.1145/2001269.2001293

[3] S. Bauer, Y. Alkhorshid, and G. Wanielik. 2016. Using High-Definition maps for precise urban vehicle localization. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 492–497. https://doi.org/10.1109/ITSC.2016.7795600

[4] Michael Bloesch, Michael Burri, Sammy Omari, Marco Hutter, and Roland Siegwart. 2017. Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback. *The International Journal of Robotics Research* 36, 10 (2017), 1053–1072. https://doi.org/10.1177/0278364917728574

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2019. nuScenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027* (2019).

[6] Gabriel Noya Doval, Abdulla Al-Kaff, Jorge Beltrán, Fernando García Fernández, and Gerardo Fernández López. 2019. Traffic Sign Detection and 3D Localization via Deep Convolutional Neural Networks and Stereo Vision. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 1411–1416. https://doi.org/10.1109/ITSC.2019.8916958

[7] Ahmed El-Mowafy. 2000. Performance Analysis of the RTK Technique in an Urban Environment. 45 (06 2000).

[8] Christian Ertler, Jerneja Mislej, Tobias Ollmann, Lorenzo Porzi, and Yubin Kuang. 2019. Traffic Sign Detection and Classification around the World. arXiv:1909.04422 [cs.CV]

[9] J. A. Farrell, T. D. Givargis, and M. J. Barth. 2000. Real-time differential carrier phase GPS-aided INS. *IEEE Transactions on Control Systems Technology* 8, 4 (2000), 709–721. https://doi.org/10.1109/87.852915

[10] Aikaterini Fragkiadaki, Bryan Seybold, Cordelia Schmid, Rahul Sukthankar, Sudheendra Vijayanarasimhan, and Susanna Ricco. 2017. Self-Supervised Learning of Structure and Motion from Video. In *arxiv (2017)*. https://arxiv.org/abs/1704.07804

[11] Yang Gao and Xiaobing Shen. 2002. A New Method for Carrier-Phase-Based Precise Point Positioning. *Navigation* 49, 2 (2002), 109–116.

[12] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. 2013. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)* (2013).

[13] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. 2019. Digging into Self-Supervised Monocular Depth Prediction. (October 2019).

[14] H.264. 2022. Advanced video coding for generic audiovisual services. https://www.itu.int/rec/T-REC-H.264-201906-P/en

[15] M.R. James, S. Robson, S. d'Oleire Oltmanns, and U. Niethammer. 2017. Optimising UAV topographic surveys processed with structure-from-motion: Ground control quality, quantity and bundle adjustment. *Geomorphology* 280 (2017), 51 – 66. https://doi.org/10.1016/j.geomorph.2016.11.021

[16] Elliott Kaplan and Christopher J. Hegarty. 2017. *Understanding GPS/GNSS: Principles and Applications, Third Edition* (3rd ed.). Artech House, Inc., USA.

[17] Karl Kovach, Jason Taylor, Andrew Elliott, and David Steare. 2007. The Precise Positioning Service (PPS) Performance Standard (PS). *Proceedings of the 20th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2007)* (2007).

[18] Y. Li and J. Ibanez-Guzman. 2020. Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems. *IEEE Signal Processing Magazine* 37, 4 (2020), 50–61. https://doi.org/10.1109/MSP.2020.2973615

[19] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. 2020. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 2 (Feb 2020), 318–327. https://doi.org/10.1109/TPAMI.2018.2858826

[20] Rong Liu, Jinling Wang, and Bingqi Zhang. 2020. High Definition Map for Automated Driving: Overview and Analysis. *Journal of Navigation* 73, 2 (2020). https://doi.org/10.1017/S0373463319000638

[21] Bruce D. Lucas and Takeo Kanade. 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2* (Vancouver, BC, Canada) *(IJCAI'81)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 674–679. http://dl.acm.org/citation.cfm?id=1623264.1623280

[22] Lyft. 2022. Lyft Level-5 Dataset. https://self-driving.lyft.com/level5/data

[23] Mapillary. 2022. Mapillary Traffic Sign Dataset. https://www.mapillary.com/dataset/trafficsign

[24] Mapillary. 2022. OpenSfM. https://www.opensfm.org

[25] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool. 2013. Traffic sign recognition – How far are we from the solution?. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*. 1–8. https://doi.org/10.1109/IJCNN.2013.6707049

[26] D. L. Mills. 1985. RFC0958: Network Time Protocol (NTP).

[27] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund. 2012. Vision-Based Traffic Sign Detection and Analysis for Intelligent Driver Assistance Systems: Perspectives and Survey. *IEEE Transactions on Intelligent Transportation Systems* 13, 4 (Dec 2012), 1484–1497. https://doi.org/10.1109/TITS.2012.2209421

[28] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE transactions on robotics* 31, 5 (2015), 1147–1163.

[29] Abm Musa and Jakob Eriksson. 2020. Feasibility of Video-based Sub-meter Localization on Resource-constrained Platforms. *arXiv preprint arXiv:2002.08039* (2020). arXiv:2002.08039 [cs.CV]

[30] K. Pirker, M. Ruther, and H. Bischof. 2011. CD SLAM - continuous localization and mapping in a dynamic world. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 3990–3997. https://doi.org/10.1109/IROS.2011.6094588

[31] Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An Incremental Improvement. arXiv:1804.02767 [cs.CV]

[32] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An Efficient Alternative to SIFT or SURF. In *Proceedings of the 2011 International Conference on Computer Vision (ICCV '11)*. IEEE Computer Society, Washington, DC, USA, 2564–2571. https://doi.org/10.1109/ICCV.2011.6126544

[33] S. Song, M. Chandraker, and C. C. Guest. 2016. High Accuracy Monocular SFM and Scale Correction for Autonomous Driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 4 (2016), 730–743. https://doi.org/10.1109/TPAMI.2015.2469274

[34] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. 2011. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *The 2011 International Joint Conference on Neural Networks*. 1453–1460. https://doi.org/10.1109/IJCNN.2011.6033395

[35] R. Timofte, K. Zimmermann, and L. V. Gool. 2009. Multi-view traffic sign detection, recognition, and 3D localisation. In *2009 Workshop on Applications of Computer Vision (WACV)*. 1–8. https://doi.org/10.1109/WACV.2009.5403121

[36] Waymo. 2022. Waymo dataset. https://waymo.com/open/

[37] Brian Williams, Mark Cummins, Jose Neira, Paul Newman, Ian Reid, and Juan Tardos. 2009. A comparison of loop closing techniques in monocular SLAM. *Robotics and Autonomous Systems* 57, 12 (2009), 1188 – 1197. https://doi.org/10.1016/j.robot.2009.06.010 Inside Data Association.

[38] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. 2018. BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling. arXiv:1805.04687 [cs.CV]

[39] Paul A Zandbergen and Sean J Barbeau. 2011. Positional accuracy of assisted gps data from high-sensitivity GPS-enabled mobile phones. *Journal of Navigation* 64, 03 (2011), 381–399.

[40] Z. Zhang. 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 11 (2000), 1330–1334. https://doi.org/10.1109/34.888718

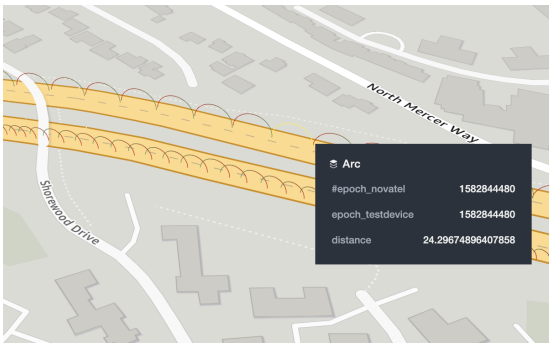
[41] L. Zhou and Z. Deng. 2014. LIDAR and vision-based real-time traffic sign detection and recognition algorithm for intelligent vehicle. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. 578–583. https://doi.org/10.1109/ITSC.2014.6957752

[42] Zhe Zhu, Dun Liang, Songhai Zhang, Xiaolei Huang, Baoli Li, and Shimin Hu. 2016. Traffic-Sign Detection and Classification in the Wild. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

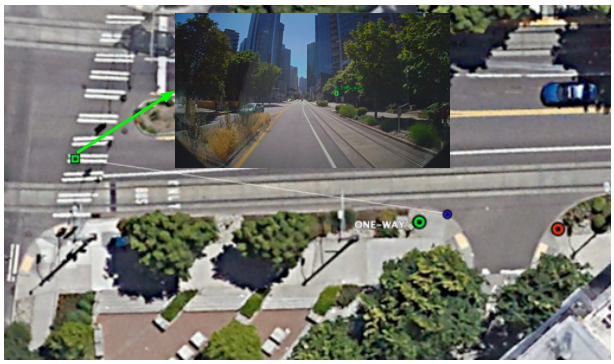
A IMPACT OF TIMING ERROR

We show a couple of examples of location errors due to timing errors here. Figure 13a shows the timing error and corresponding location error for a commodity smartphone. Here, we draw arcs from the smartphone's locations (red) to our reference system's locations (green). The error is larger at a higher speed, and the higher error is represented by a larger arc length and height. Even if the GNSS timing is accurate, the smartphone hardware and operating system introduce large and variable latency and cause high error in system time. Note that this type of high error in the direction of motion is hard to catch as there is a low error in the sideways of the travel direction and the location trace visually looks smooth.

Figure 13b shows an example of the effect of the system timing error on our sign localization. Here, the sign gets shifted towards the vehicle's direction of motion and results in a large localization error.



(a) Vehicle location error due to the timing error for a smartphone. The arcs are drawn from the smartphone's locations (red) to our reference system's locations (green).



(b) Sign location error due to system timing error (green square: frame location with the correct time, blue circle: sign location with the correct time, red circle: sign location with the incorrect time.)

Figure 13: Location error due to timing error.

B IMPACT OF GNSS ERROR

We discuss the impact of the GNSS vehicle location error on our sign localization error here. Figure 14 shows an example of a high sign localization error due to GNSS error. Here, the trajectory started to deviate in our low-cost precision GNSS system, which caused inaccurate absolute sign localization. Note that the relative sign localizations in two different frames are accurate here as can be seen in Figure 14. If the vehicle GNSS location would have been accurate, our absolute sign localization would have been accurate as well.

It is very hard to obtain very high accuracy GNSS positioning all the time, especially using a low-cost system. However, the accuracy of our low-cost GNSS system is still very high most of the time. In the future, we plan to improve our accuracy further with the integration of wheel odometry and improved INS performance to cope with highly GNSS-denied areas like dense downtowns.

C SIGN DIRECTIONALITY

Some traffic signs that are detected and localized during a drive are not applicable for that direction of travel. This is a hard problem to solve automatically. However, we preliminary investigated the solution to this problem and found that the distribution of all 3D point locations of a sign can be utilized to infer a sign's direction with respect to the vehicle's direction of travel. Figure 15 shows an example of this where we show the localized sign from a video frame, a close-up view of all 3D points for that sign, and the frame. Here, Figure 15a shows that the points are distributed perpendicularly to the travel direction of the vehicle and thus applicable for the vehicle. Figure 15b shows that the points are distributed in the same direction of the vehicle's travel direction and thus not applicable. We will evaluate this method in detail in the future. Another possible approach to solve this problem is to learn the sign directionality as a part of the sign detection model.

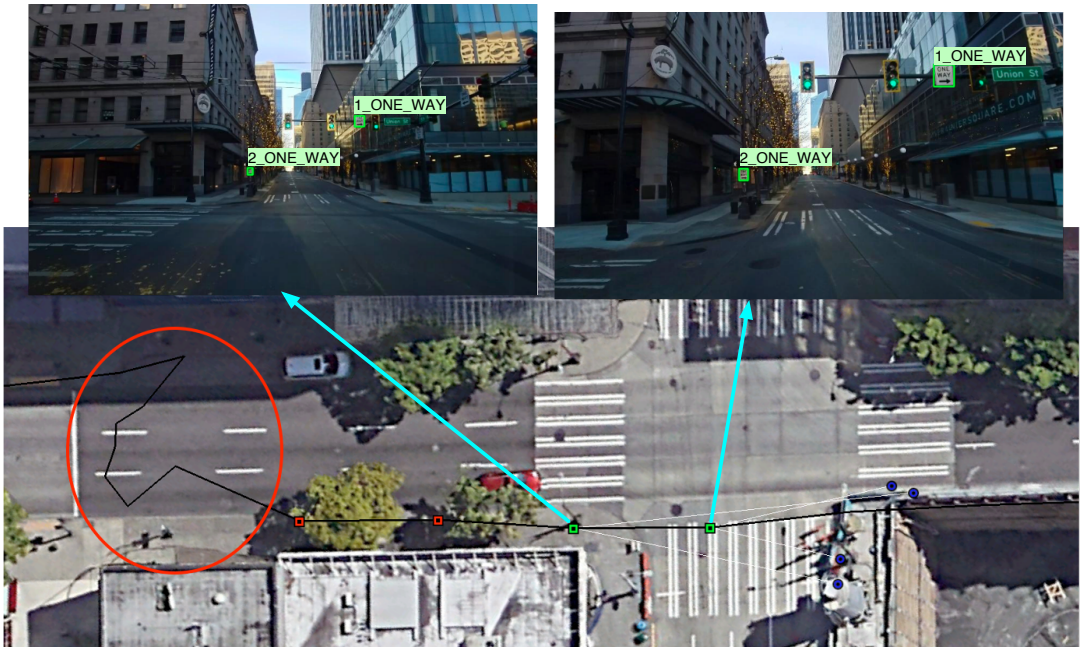
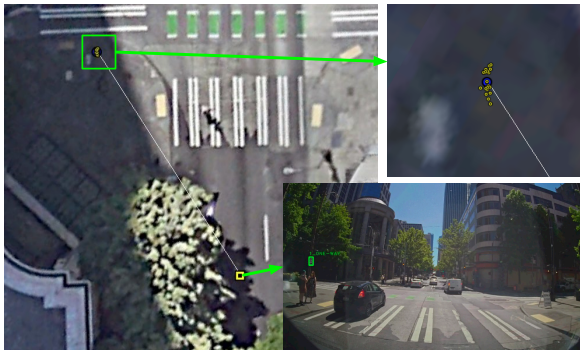


Figure 14: Example of inaccurate sign localization due to inaccurate GNSS vehicle localization in dense downtown. The GNSS location trajectory (black line) started to deviate around the red oval although the vehicle kept driving straight (can be seen on the camera images on the top). This resulted in inaccurate absolute localization of signs although the relative localization of them is correct at two different video frames.

(a) Sign is applicable for the direction of travel.



(b) Sign is not applicable for the direction of travel.

Figure 15: Sign applicability in the direction of travel based on the distribution of sign points.