https://doi.org/10.3311/PPci.21500
Creative Commons Attribution

Periodica Polytechnica Civil Engineering, 67(2), pp. 457–472, 2023

Building Maps Using Monocular Image-feeds from Windshield-mounted Cameras in a Simulator Environment

Mátyás Szántó[1*], Sándor Kobál[1], László Vajta[1], Viktor Győző Horváth[2], János Máté Lógó[2], Árpád Barsi[2]

[1] Department of Control Engineering and Information Technology, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, 2. Magyar tudósok Blvd., H-1117 Budapest
[2] Department of Photogrammetry and Geoinformatics, Faculty of Civil Engineering, Budapest University of Technology and Economics, 3 Műegyetem rkp., K building first floor 31. Budapest, H-1111, Hungary
[*] Corresponding author, e-mail: mszanto@iit.bme.hu

Received: 10 November 2022, Accepted: 16 January 2023, Published online: 01 February 2023

Abstract
3-dimensional, accurate, and up-to-date maps are essential for vehicles with autonomous capabilities, whose functionality is made possible by machine learning-based algorithms. Since these solutions require a tremendous amount of data for parameter optimization, simulation-to-reality (Sim2Real) methods have been proven immensely useful for training data generation. For creating realistic models to be used for synthetic data generation, crowdsourcing techniques present a resource-efficient alternative. In this paper, we show that using the Carla simulation environment, a crowdsourcing model can be created that mimics a multi-agent data gathering and processing pipeline. We developed a solution that yields dense point clouds based on monocular images and location information gathered by individual data acquisition vehicles. Our method provides scene reconstructions using the robust Structure-from-Motion (SfM) solution of Colmap. Moreover, we introduce a solution for synthesizing dense ground truth point clouds originating from the Carla simulator using a simulated data acquisition pipeline. We compare the results of the Colmap reconstruction with the reference point cloud after aligning them using the iterative closest point algorithm. Our results show that a precise point cloud reconstruction was feasible with this crowdsourcing-based approach, with 54\% of the reconstructed points having an error under 0.05 m, and a weighted root mean square error of 0.0449 m for the entire point cloud.

Keywords
environmental reconstruction, crowdsourcing, SfM, automotive simulator, image feed processing

1 Introduction

In the last decade, the attention surrounding autonomous vehicles (AVs) has grown to a previously unprecedented level. The incredible interest shown towards the technologies involved has carried considerable momentum towards the research of methods enabling the emergence of smart cities such as traffic automation or self-driving cars. One of the cornerstones for the successful and robust functionality of AVs is their ability to understand their surroundings by use of sensors, such as cameras, light detection and ranging (Lidar) sensors, radio detection and ranging (Radar) sensors.

A new trend emerged together with the development of these technologies: one that provides solutions for processing raw sensor data into meaningful information and that enables reliable, and real-time navigation-related decision making. Modern image and data processing algorithms have been gravitating toward soft computing designs in the past five years, such as machine learning (ML) and Deep Learning (DL). Although such novel methods have proven to perform en-par with and often exceeding the efficiency of their traditional counterparts, and even human actors in some specific use-cases, a big shortcoming has been taking form in connection with these solutions. The required amount of training data – used for parameter optimization – is immense for learning algorithms; this is especially true in the case of networks trained using the supervised learning paradigm, not uncommon in the case of AV navigation solutions.

However, the compilation of training databases involves data gathering and labelling processes that are costly and utterly resource-intensive. Simulation-to-Reality (Sim2Real) solutions present a prominent direction for the mitigation of such problems. For fruitful Sim2Real applications, precise, and realistic simulations are required that are able to exhibit excellent similarity with the physical domain. To achieve good results, therefore, robust modelling methods of real-world scenarios are in demand. In recent publications surrounding this topic, crowdsourcing applications appeared as sources of viable solutions: volunteered geographical information (VGI) methods and research on 3-dimensional (3D) modelling solutions – e.g., Simultaneous Localization and Mapping (SLAM) and structure-from-motion (SfM) – performed on "images from the wild" have been brought into the spotlight. In the case of crowdmapping – i.e., crowdsourced mapping –, the quantity over quality approach is adopted, wherein maps are constructed from data acquired by low-cost sensors – e.g., red-green-blue (RGB) raster cameras and location information from the Global Positioning System (GPS) and inertial measurement units (IMUs).

In this paper, we validate our approach, namely that by using crowdsourcing data acquisition techniques (with cheap sensor units) accurate 3D representations – e.g., point clouds – can be created of realistic environments. These point clouds meet sufficiently high-quality requirements to be used in simulator frameworks as a basis for Sim2Real data amassing. To show the viability of our proposal, we simulate a crowdmapping project and reconstruct a complex piece of road geometry within the Carla simulator environment using an industrial-grade SfM solution. We also introduce our solution for generating ground-truth data using the Python API of the Carla Simulator. We compare the results of our point cloud-generation method against the obtained ground-truth data.

Our contributions in this paper are as follows:

- We create a novel solution for simulating a crowdsourcing-like geographical data gathering solution using the open-source Carla simulator environment and synthesize raw, realistic, and monocular image data that can be used as input for point cloud generation methods;
- We propose a crowdmapping pipeline that yields dense point clouds using an SfM technique based on the previously generated images;
- We demonstrate the validation process of the functionality of our novel pipeline by comparing the outputs to ground truth data exported from the simulation environment.

This paper is structured as follows: First, we present our review on the relevant literature, then we introduce the methodology we employed during our research – including the crowdsourcing simulation, the ground-truth synthesis, and the point cloud reconstruction and comparison procedures. After that, we introduce the results obtained during the validation of our approach. Finally, we conclude our paper with summarizing our work and listing future opportunities.

2 Related work

In this section, we present the state-of-the-art technologies employed in our research. First, we introduce automotive maps, followed by crowdsourcing techniques used in mapping. We then focus on software applying the presented technologies and give an overview of autonomous vehicle support with data synthesis.

2.1 Automotive maps

We first summarize the general terms in maps and their production methodology, then navigation maps and their relation to the vehicular assistants. The subsection "Modern maps" covers the requirements and development timeline. The short introduction of the rapidly developing simulator-oriented map format OpenDRIVE is closing the section.

2.1.1 Maps in general and their production

Since the start of transport, network maps of possible ways have been produced to help travelers choose and follow the best route. Maps must include all the elements of the road network, as well as the navigation. In addition to geometrical accuracy, topology is equally essential for these maps.

The map reference and projection systems, furthermore all the important mapping features of maps have gradually moved towards ever higher quality. Nowadays, to meet global needs, standardized solutions are chosen, such as the WGS84 (World Geodetic System) standard reference system and the UTM (Universal Transverse Mercator) projection system [1, 2]. GPS and other satellite-based positioning systems also use these systems, so standardization is inevitable.

Perhaps the most important aspect of using maps is the format and storage solution that is used to access the content. In the case of automotive maps, there are two possible formats:

- *Physical format*: a solution that can store all object types together with all their descriptive data for a given task. Beyond storage, efficient data access and manipulation are naturally also part of the technology.

The physical formats are usually associated with the mapping companies (intellectual property) and are therefore diverse and often not publicly documented. An exceptional example is the NDS (Navigation Data Standard), which is a standard physical format used by several providers [3];
- *Exchange format*: a format created to allow interoperability between physical formats. Since the aim here is to provide a precise description of all formats involved, it is rather complex, complicated, and voluminous, but at the same time, its description is available (in some cases for a fee). The best-known example is GDF (Geographic Data File).

To create map content, reality needs to be measured, which is where the various field survey technologies come in. Data collection and processing based on surveying, photogrammetric, remote sensing and laser scanning methods, whose results are digital map databases, are widespread. One of the most effective methods today is mobile mapping, which describes a vehicle-mounted unit with positioning and sensing instruments – together with data recording and processing software [4]. Modern surveying systems generate camera images and laser-scanned point clouds on the ground, which are postprocessed to extract the objects to be mapped (e.g., traffic lanes) together with their descriptive characteristics (e.g., width, pavement type, pavement marker color).

2.1.2 Maps for navigation and assistants

Map data can be used at different levels for vehicle navigation. A DIS (Driver Information System) can provide the driver with different types of information, one of which is onboard navigation, which is also related to map information. Navigation displayed via DIS requires map information in addition to position data. Thus, the map and the current traffic information provide the basis for the vehicle to reach its destination via the optimal route.

The main purpose of Driver Assistance Systems (DAS) is to make driving safer. Advanced Driver Assistance Systems (ADAS) make better use of the data from the vehicle's sensors [5]. Navigation-based ADAS elements can be, but are not limited to:
- *Congestion warning systems*, which allows to avoid or approach congestion at the appropriate speed;
- *Curve warning*, which warns the driver if the chosen speed is too high for the next bend in the road;
- *Danger spot warning*, indicating dangerous traffic situations, which may include potential collision zones (e.g., where two roads join in a curve and continue together; difficult to see intersection), high attention facilities (kindergarten, school), etc.;
- In the case of *wrong-way driving* (or ghost driving), the driver is warned that he is not trying to enter the road from the direction of travel.

For assistants, an evolutionary curve can be observed: first the intersection collision warning (ICW) assistant, which provides support in case of crossing traffic directions, and then the intersection collision avoidance (ICA) assistant. Likewise, the fuel efficiency advisor (FEA), which only gives suggestions, and the hill descent control (HDC), which provides continuous control [6].

2.1.3 Modern maps

The development of traffic maps, and even automotive maps, has followed a very similar trajectory to that of the assistants. The gradual increase in automation has also changed the requirements for maps. Increasingly sophisticated map support, therefore, demanded more and more content: road descriptions became more refined and detailed, elevation (3D) information was included in map content, POIs (Points of Interest) with their own additional data were introduced and commonly used, buildings and other infrastructure elements were included as spatial models, and finally lanes as segments of the road network became the basic elements. Of course, the main driver of this development has been surveying technology, so fieldwork based on images and laser-scanned point clouds has become inevitable. This enrichment of detail has raised a number of problems:
- The amount and complexity of the data to be stored has increased significantly: effectively accessing and managing heterogeneous content requires new, specific data models;
- The increased volume of data has led to a rethinking of map representation and content management; the total mass of information has grown to an unmanageable, opaque size, and therefore the use (visualization) for machine and human purposes has become separated. The full-detail map became known as a high-definition map (HD map), the reduced version as a small-definition map (SD map) [7, 8]. A. Karpathy, the former head of AI at Tesla calls "HD maps a part of the infrastructure for self-driving cars" and declares

that "self-driving features require HD maps to precisely localize themselves and to accurately navigate on a lane-level basis" NVIDIA also has its own HD map format named DriveWorks HD map, in which they collect data with their own mapping system, but unlike Tesla, they also provide support to integrate different formats from other sources into their DRIVE framework;

- The processing of increased amounts of data has started involving computer technologies [9], including crowdsourcing. In 2017, Intel acquired an Israeli company named Mobileye. According to their own data, Mobileye systems are installed in more than 300 vehicle models by 27 OEMs (Original Equipment Manufacturer), which means that most main roads are observed by vehicles equipped with their cameras at least once per day – but often even more frequently. The company harvests the sensor data from customers' vehicles in the cloud, aggregating them into a crowdsourced, constantly updated map database without any contribution from dedicated survey vehicles [10]. Along with this, data with different origins and technologies have often lost their homogeneity of accuracy, and with it, the issue of reliability as data quality has come to the fore [11];
- Efficiently accessing and updating the increased amount of data is critical to having a large number of vehicles to be served. This requirement indicates a fundamental reason for a distributed system architecture or cloud-based service;
- A growing number of solutions are gradually being developed to serve the updating needs. SENSORIS seems to be one of the most promising technologies, which collects field data as "rolling meters" of vehicles in motion, and the processing mechanism that sits on top of it detects and manages the changes [7];
- Thanks to low latency database updates, increasingly rapidly changing phenomena (weather elements, traffic lights and their conditions, even data from moving vehicles themselves) can be incorporated into the map content. A proprietary, specially developed data model (e.g., Local Dynamic Map – LDM) can handle moving traffic, pedestrians, etc. [12, 13];

To meet the needs of the automotive industry, a map database with full detailed terrain data for computer simulations, specific formats (e.g., OpenDRIVE) are needed and there is a growing effort by researchers to ahieve this.

2.2 Crowdsourced data

Crowdsourcing, a term coined by Jeff Howe in 2006 [14], is a paradigm in which individual contributors – that are not necessarily tied to a single organization or enterprise – are involved in an effort to achieve a well-described goal. The members of the *crowd* are not required to work together – neither spatially, nor temporally; thus, the steps taken towards the common goal can and often must be well-distributable. Crowdsourcing is a term often used in connection with research activities that involve platforms that enable the reaching of large amounts of humans, who thus have the ability to participate in solving a complex problem or a component of a problem – these platforms include social media [15], medical research facilities [16], and traffic systems.

Crowdsourcing techniques for traffic data collection have seen a surge in the past decade thanks in large to the increasing number of sensors utilized in modern vehicles. Amongst others use-cases, intelligent transportation systems [17], traffic accident post impact monitoring schemes [18] have all employed crowdsourced data collection and processing for solving their given tasks. In our research, we have scrutinized data acquisition techniques, whose results are linked to a certain – actual or virtual – geographical location. Gathering data for map-related usage is also not new in the literature; using crowdsourced data for achieving this goal is often referred to as VGI in the literature [19, 20]. Arguably, the most widely spread VGI platform is OpenStreetMap [21], which is an online platform that became an industry standard for VGI schemes in the past decade.

Crowdmapping, a subcategory of VGI applications, has appeared in the technical literature in the last 5 years, and have shown great usability for traffic automation and passenger as well as freight vehicle automation. For robust navigation of autonomous vehicles (AVs), it is crucial to sense and interpret their surroundings correctly, while recognizing its pose – i.e., location and orientation – and the traffic situation in its environment. The use of crowdmapping has appeared in several research endeavors [22, 23] as well as industrial applications (Waymo [24], Lyft [25], Uber [26], and Google [27]). A prime example is Mapillary [28]; a company that created a crowdsourcing platform, whose main purpose is to map out every corner of the world. Maintenance of the maps is performed using computer vision and artificial intelligence (AI). Members of the community can expand the framework's database by uploading raw street imagery.

In 2009, a conference paper by Agarwal et al. attracted a lot of attention, which was later published in a journal in 2011 [29]. Using Structure-from-Motion technology [30], they created a sparse point cloud, then densed it and derived a 3D model based on MultiView Stereo (MVS) algorithms, with input from a rich repository of photos shared on social networking sites. The word "Rome", for example, is associated with more than 3 million photos on Flickr, so this collection contains enough images of every landmark in the city. For such a city-scale project, the authors used a smart solution to handle the huge number of matches in a cluster of 62 dual quad-core processors with 496 cores. 150 000 images were processed in nearly 21 hours, building a digital model of Rome really in a day. The authors later applied the technology to other urban projects (Venice, Dubrovnik, etc.) with large numbers of images.

Following the idea of Agarwal et al., our team performed similar research: using images of Budapest uploaded to Flickr, we built models, loaded them into a CAD system, and finally took geometric measurements and derived sections [31]. We reconstructed the Lion of the Chain Bridge from 63 images, the Parliament building from 173 images, and Heroes' Square from 1 626 images. The calculation times on an average Intel i7 processor desktop PC are given in Table 1.

The reconstruction was also carried out using our SLR and mobile phone camera images and terrestrial laser scanning (TLS) [31].

2.2.1 Structure-from-Motion

The Structure-from-Motion technique is less than 50 years old. The phrase originates from Simon Ullman, who used it in his doctoral thesis in 1977. Ullman's Structure-from-Motion theorem states: "Given three distinct orthographic views of four non-collinear points in a rigid configuration, the structure and motion compatible with the three views are uniquely determined" [32].

There are several approaches to Structure-from-Motion. In incremental SfM, camera positions are solved and added to the collection one by one [32]. The incremental reconstruction algorithm as a practical interactive pipeline reconstructs the three-dimensional (3D) model depending on the corresponding image points. This incremental SfM strategy is widespread nowadays as global methods need many images which will allow related estimations to create closed loops [33]. In global SfM [34], all camera poses are solved simultaneously. An intermediate approach is out-of-core SfM, where several partial reconstructions are computed and integrated into a global solution [35]. Nowadays, there is a wide range of software available to apply the SfM method in different fields.

Table 1 Processing times [sec] of crowd-sourced imagery in Budapest

	Pairwise matching	3D reconstruction	Bundle adjustment	Dense reconstruction
Lion on the Chain Bridge	86	57	7	22
The Parliament Building	875	102	7	113
Heroes Square	3342	104	5	278

However, SfM is fundamentally different from traditional photogrammetry in that the object's geometry, position, and camera orientation are solved automatically without the need to specify a predefined set of "ground control" targets/points with known 3D positions. Instead, these are solved simultaneously using a highly redundant iterative node alignment procedure based on a database of features automatically extracted from multiple overlapping images. To determine the 3D location of points within an object, conventional photogrammetric methods require the cameras' 3D location and orientation or the 3D location of a pair of control points [36]. In the former case, the object's geometry can be reconstructed by triangulation in the absence of a GPS and electronic compass mounted on the camera. In contrast, in the latter case, the control points must be manually identified in the input photographs, and the camera position determined by a process called resection or camera pose estimation.

The SfM approach does not require any of the above before reconstructing the scene. Multiple views of an object are captured with a (non-metric) digital camera from different positions, ensuring a high degree of overlap between images. The common points of the images must then be identified to establish a spatial relationship between the locations of the original images in an arbitrary three-dimensional coordinate system. In this way, the position and orientation of the camera are automatically determined. The image features are tracked from frame to frame, allowing an initial estimate of the camera position and object coordinates, which are then iteratively refined by nonlinear least squares minimization, and finally by bundle adjustment.

2.3 Software

The popularity of the SfM method lead to it being utilized in numerous fields of application [37], where 3D environmental reconstruction is required from 2D images taken from numerous viewpoints: photogrammetry [38], robotics [39], augmented reality (AR) applications [40]. Moreover, a considerable amount of valuable SfM solutions can be found both in the literature as well as in industrial practice:

- *Colmap*: an open-source incremental SfM implementation, whose purpose is to offer a general tool for 3D reconstruction pipelines. A graphical user interface (GUI) is also available as part of the implementation [41];
- *Theia*: an open-source – and thus openly extendable – function library containing SfM building blocks. The author created the library using C++ [42];
- *OpenMVG*: an open-source function library primarily for MultiView Stereo (MVS) algorithms. The implementation contains SfM algorithms too, and it enables the usage of GPS coordinates for the estimation of external camera parameters [43];
- *VisualSFM*: an open-source solution that implements incremental SfM algorithms, while offering a GUI for users [44];
- *Metashape*: Agisoft Metashape software, formerly known as Photoscan, provides tools for photogrammetry pipelines such as processing digital images and generating 3D objects. Metashape could be called an industry standard, its fields of application range from entertainment media to archeology to surveying. It supports both local and cloud processing with many settings, and a modern graphical user interface. It can generally be described as a well-maintained and supported software for photogrammetric applications;
- *RealityCapture*: RealityCapture is a quite recent software from 2016 with fields of an application similar to Metashape. RealityCapture is more aimed towards being a one-button solution;
- *Recap*: Autodesk ReCap is from the Autodesk software lineup; its basic functions include handling and registering point clouds. Only the Pro version is usable for photogrammetric applications, as ReCap Photo is not included in the base version.

From the above list, we have opted to use Colmap version 3.8 during our experiments thanks to its flexible functionality, robust algorithms, and open availability.

2.4 Synthesis of training data for autonomous vehicles

In the previous decade, the technology of self-driving cars has seen an unprecedented surge in popularity and interest. The SAE (Society of Automotive Engineers) has defined the levels of autonomy in these vehicles on a scale from 0 to 5 [45].

Vehicles residing on levels 1, 2, and 3 reach their respective levels through employing certain driver assistance systems, or – in the case of camera-aided solutions – advanced driver assistance systems. The functionality of such systems is understandably safety-critical, and as such, they must meet serious quality and availability requirements. Novel ADAS software often apply deep learning-based machine vision algorithms to support the functionality [46]. Many state-of-the-art visual deep learning (DL) solutions show the ability to surpass their classical counterparts for solving image-processing tasks when trained successfully and properly using supervised and unsupervised learning [47]. During supervised learning, a predefined neural network structure aims at finding the best set of weights and biases through error (or loss) minimization on a previously defined dataset equipped with ground truth information.

In order to satisfy the demanding specifications set towards ADAS solutions, the deep-learning algorithms must generalize well to many scenarios. In the case of supervised learning, this generalization ability is – among other factors – proportional to the size and heterogeneity of the dataset fed to the system. This, however, poses a great problem: obtaining the data and generating the ground truth are utterly costly processes, thus causing one of the key bottlenecks for database creation and for successful deep learning-based technology usage. For image processing DL applications, obtaining data requires precise measurements from well-calibrated sensors – e.g., cameras, and Lidars – and resource intensive – i.e., human labor demanding – tasks for labelling data [48].

A solution for data amassing that has gained significant attention recently is simulation-to-reality or Sim2Real. This technique has been used predominantly in the field of robotics [49], but has recently gained significant attention in connection with rendering training datasets for Machine Learning solutions in general [50] and in particular for AV development [51, 52]. In this scenario, data acquired in a simulation environment is used to reach better results in an actual physical working environment, thus decreasing the cost and effort needed for database creation and ground truth labelling.

One key problem facing developers when applying Sim2Real techniques is the difference between the simulation environment and the physical environments the autonomous vehicles or agents will encounter once used in the real world. Solutions have been proposed to this problem, such as in [53], wherein the authors attempt to make the robotic agent "feel at home", and thus achieve better results via translating the realistic model of the physical environment into the simulated domain.

2.5 Simulators

IUsing Real2Sim (reality to simulation), we can overcome the problematic nature of unrealistic environments, while with Sim2Real, the problem of insufficient training data can be mitigated. In this paper, we present our solution that uses a simulation environment that is commonly utilized for autonomous driving training. These simulators enable the synthesis of driving environments (urban and suburban roads) and the introduction of autonomous agents, such as vehicles, and pedestrians. For data generation, a flexible selection of sensors including passive (monocular and stereo RGB) cameras and active depth sensors (RGB-Depth – RGB-D – cameras, Lidars, Radars, etc.) may be added to the simulation. Rendered images can then be exported through real-time rendering. Recently, several drive simulators have appeared both for research as well as for industrial use – all able to ingest and use OpenDRIVE data.

- *NVIDIA DRIVE Sim*, which is powered by the NVIDIA Omniverse platform. As of the publication of this paper, the simulator is only available for participants of an early access program;
- *AIMotive aiSim*, the first ISO26262 ASIL-D certified simulator tool that is aimed at validating autonomous driving-related technologies in a synthesised environment;
- *Carla Simulator*, which is an open-source simulation environment aimed towards autonomous driving research that is based on Unreal Engine 4. The framework realizes a server-client structure, where simulation preferences, agents, sensors and other settings can be customized through a Python API [54];
- *Tass PreScan*, a simulator environment purpose-built for ADAS solution testing in a virtual context [55];
- *IPG CarMaker*, which was specifically developed for the design and testing of passenger and light-duty vehicles [56];
- *Vires Virtual Test Drive*, similar to Tass PreScan, is a simulator that has been designed with the aim of creating a workbench for the validation of the ADAS solutions of road vehicles [57].

Because of its research-geared scheme, availability, and the extensive amount of community-generated materials available online, we opted to use Carla Simulator version 0.9.14 as our data synthesis framework during our research presented in this paper.

3 Methodology

The work described in this paper aims to validate the assumption of crowdsourced visual data usability for low-latency road infrastructure mapping. Our hypothesis is that by using information from several analogous sensors – i.e., cameras, IMU, and GPS – associated with all the data sources, a point cloud with acceptable accuracy can be reconstructed from the environment of the moving data acquisition vehicles. For this, our method makes use of vehicle-mounted monocular camera image sequences synthesized and collected in an autonomous driving training simulator. The proposed framework is built upon the CrowdMapping structure [23].

The crowdsourcing nature of our solution lies in the data acquisition simulation described in Section 3.1. It differs from the approaches used in [29] and [31], as the data sources used in this paper are not handheld or smartphone cameras, but rather windscreen-mounted ones, and the data is not collected from a pre-existing database, but rather volunteered by the operators of the vehicles in real time. Each synthesized data acquisition vehicle represents a member of the crowd, meaning that these vehicles are differently structured – i.e., the cameras are located at different relative positions. Moreover, they operate separately from each other, and voluntarily provide data to a central mapping platform. The monocular imagery provided by these agents are then processed and a 3D point cloud is reconstructed in the data centre. The flow of data realized in our method is shown in Fig. 1.

3.1 Data simulation

As written above, because of its versatility, availability and community support, during this research, we opted to use the Carla Simulator as the source of the simulated as well as the ground-truth data. All software components used during data and ground truth exports were implemented using Python.

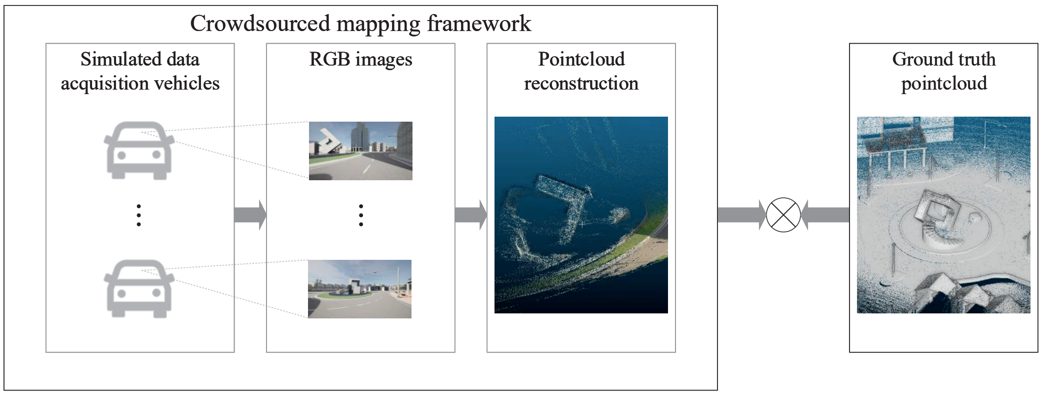

Crowdsourced mapping framework
Simulated data acquisition vehicles
RGB images
Pointcloud reconstruction
Ground truth pointcloud

Fig. 1 Our proposed data flow. In a crowdsourcing-like manner, individual data acquisition vehicles supply RGB images from a selected road environment. The images are preprocessed and transferred to the point cloud reconstruction section. The resulting point clouds are then compared with the ground truth.

During our preliminary experiments, we set up the Carla environment to be able to export useful data. We used one of the built-in maps of the Carla Simulator, namely 'Town03'. This choice proved to be convenient because of a unique feature of this setting, namely that it has a roundabout in the middle with some complex geometry in it – i.e., the model of a concrete sculpture (Fig. 2).

For data gathering, we spawned an ego-vehicle near the entrance of the roundabout and turned autopilot mode on for it. We simulated an RGB camera, a depth sensor, as well as a semantic segmentation camera that was all fixed onto the vehicle at the same exact relative position measured from the origin of the coordinate system fixed to the vehicle. We set up the simulator environment for synchronous functionality – i.e., data exporting is performed on fixed time steps. – so that the simulation results were fully reproducible. The chosen time-step was Δt = 1/30 s. The simulated weather conditions were the default settings for Town03.

One of the key conditions we aimed to simulate during the data acquisition pipeline was crowdsourcing. To accomplish this, we spawned data acquisition vehicles near the 4 entrances of the roundabout. For the simulation to mimic the heterogeneity of real-world vehicles as best as possible, we started the simulation with numerous different settings for the relative camera locations – random perturbations have been applied to the sensor-mounting position for all spawned ego-vehicle instances.

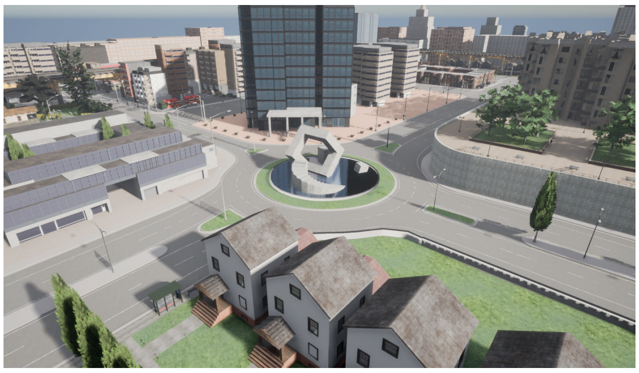


Fig. 2 Town03, a built-in map of Carla Simulator with the examined roundabout

One run consisted of a vehicle being spawned near one of the entrances and driving into the roundabout with autopilot for 240 time steps – or the equivalent of 8 s of simulated world time. The first 99 frames exported by the vehicle have been disregarded for data reduction purposes. According to our experiments, this mitigation step has not resulted in the loss of useful data, since the first few time-steps after the vehicle and sensor initialization were not spent with useful movement by the autopiloted car and the distance from the spawn-point to the roundabout has been covered in this time period. With this data reduction, one resulting run consists of 141 consecutive images taken by the synthetic RGB, depth, and semantic segmentation cameras. The output of a randomly selected time-step can be seen in Fig. 3. In addition, the camera positions have been assigned to every run as a text file containing x, y, and z coordinates and yaw, pitch, and roll angles at every time step.

The depth and semantic segmentation images were used for masking the raw RGB images. This poses two advantages:

- The sheer amount of data is reduced so that less network capacity is used for communication and data transportation;
- The input of the reconstruction algorithm is simplified, thus less calculation is carried out on areas of reduced interest – e.g., background structures, sky.

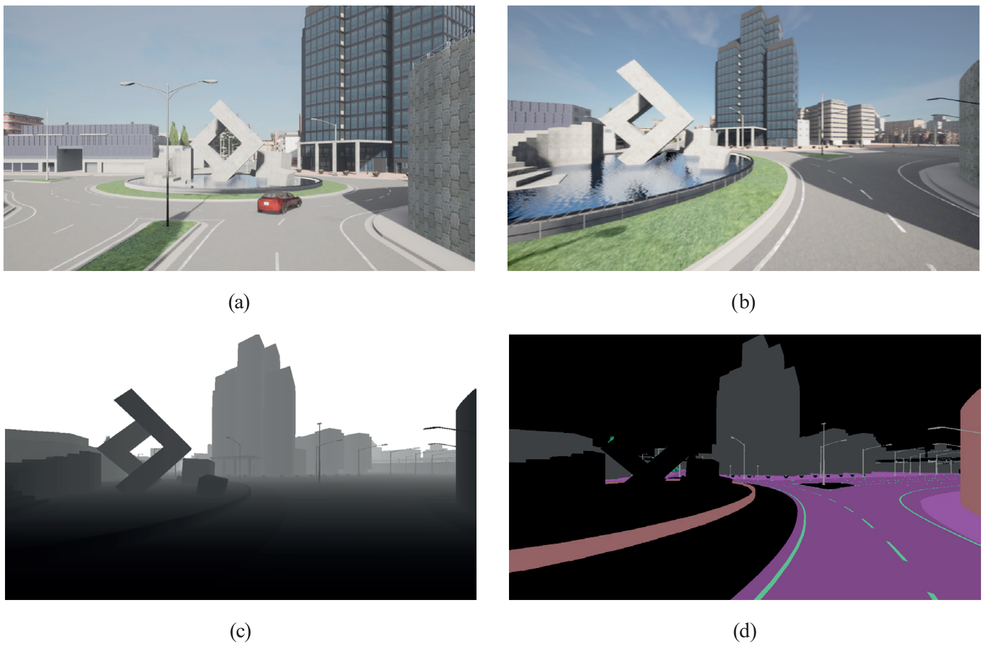


Fig. 3 One time-step of a data acquisition run: (a) birds-eye view of the data acquisition vehicle; (b) synthesized RGB image; (c) synthesized depth image with logarithmically transformed depth values for better visual clarity; (d) synthesized semantic segmentation image

This masking step is also useful if dynamic objects – e.g., other vehicles, pedestrians, etc. – are also recorded by the RGB camera, as it rids the image from unwanted objects. Such a masked output of the time-step shown in Fig. 3 can be observed in Fig. 4.

During the experiments described below, we have used a data package consisting of four runs at data acquisition vehicles spawned at each entrance of the roundabout. In total, this provided us with 4 × 4 runs, each run consisting of 141 RGB, depth, and semantic frames. The masked output (shown in Fig. 4) was also created for each image, and the camera positions were recorded for the run. The amassed data package has a size of 5.3 GB. The size of the masked images, which were used for the reconstruction demonstrated below amounts to 0.636 GB, 12% of the original data package.

3.2 Ground truth

Since the aim of our algorithm is to produce point clouds based on crowdsourced images, the ground truth (GT) – against which the output can be examined – should also be a point cloud. For the creation of GT, we also used the Carla simulator.

We have created a solution that allows us to fly around in any Carla environment and while doing so create a dense point cloud of the geometric model surrounding the spectator's viewpoint. This solution employs a synthetic Lidar sensor fixed to the spectator actor of the simulator. The parameters of this sensor – shown in Appendix A – were set up to facilitate the ground truth point cloud synthesis as required by our workflow. The spectator's movement can be controlled manually by mouse movements or by keystrokes.

The synthetic Lidar sensor yields a 360◦ point cloud for every time step. The generated point clouds are to be merged using the Open3D Python library [58]. The GT point cloud synthesized from the roundabout region of the built-in Town03 map is shown in Fig. 5.

3.3 Point cloud generation

Colmap – described in Section 2.3 – provides a pipeline for 3D reconstruction from overlapping images. The input images can either be taken in sequential order (a video) or by randomly taking pictures of the target object.

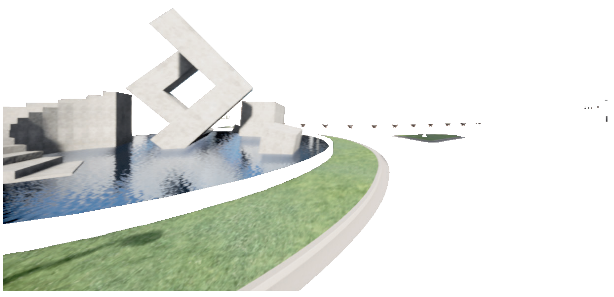


Fig. 4 Masked image of the same timestep as shown in Fig. 3. The masking is performed on the raw RGB image (Fig. 3(b)) using the depth data (Fig. 3(c)) and the semantic information (Fig. 3(d))

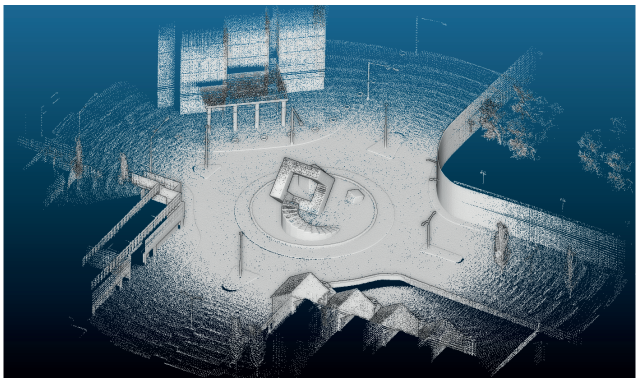


Fig. 5 The dense point cloud from a simulated Lidar survey used as ground truth in our experiments. Screen Space Ambient Occlusion shading has been applied to the point cloud during display for better visual clarity

In the first few steps of the pipeline, the algorithm performs feature extraction and matching using the Scale-Invariant Feature Transform (SIFT) method. It is followed by a sparse reconstruction step, during which the camera poses, and intrinsic values are calculated for each input image. The main feature points are then positioned in the 3D space. After a sparse model is successfully created, it is possible to continue with the dense reconstruction step, the result of which is a dense point cloud of the scene.

Although Colmap provides high flexibility and multiple options for fine-tuning the reconstruction process, the default parametrization did not satisfy our initial expectations regarding the precision of the result. With the parameter values left untouched, the reconstructed model had the four roads leading into the roundabout merged into one. As a result, different profiles of the central sculpture were merged as well. The result of reconstruction with the default parameter set can be seen in Fig. 6.

To resolve this problem, camera pose data was added to each input image. The data exported from Carla Simulator needed conversion to match the Colmap input interface. While the exported data contains camera orientation in Euler angles, Colmap expects the rotation to be supplied using quaternions. The order of the applied transformations also differs. In the synthesized data, the camera model is first rotated and then translated; however, in Colmap the transformation is implemented in the opposite order: translation then rotation. Finally, in the exported data the camera poses are given in the "world" coordinate frame, meaning that the X axis points to the front, the Y axis to the right, and the Z axis upwards. Colmap on the other hand expects inputs in the "camera" coordinate frame, meaning that the Z axis points to the front, the X axis to the right, and the Y axis downwards. Camera intrinsic values were also provided as input because the automatic reconstruction process could not learn them correctly.

With the camera position data provided to Colmap, it was able to yield promising results. A snapshot of the dense point cloud can be seen in Fig. 7.

The full reconstruction took 171 minutes on a server computer containing an NVIDIA Titan X GPU with 12 GB graphical memory, and an Intel i5 8600K CPU running at 2.8GHz.

3.4 Metrics and comparison

Generally, due to the absence of markers and the difference in the number of points, it is impossible to establish correspondence between two surfaces directly; furthermore, the transformation matrix is unavailable immediately. With the help of the Iterative Closest Point (ICP) algorithm [59], the closest point in the reference is considered as the corresponding point of the test surface iteratively adapting the transformation matrix. Several searching strategies exist to avoid a complete search between the two surfaces.

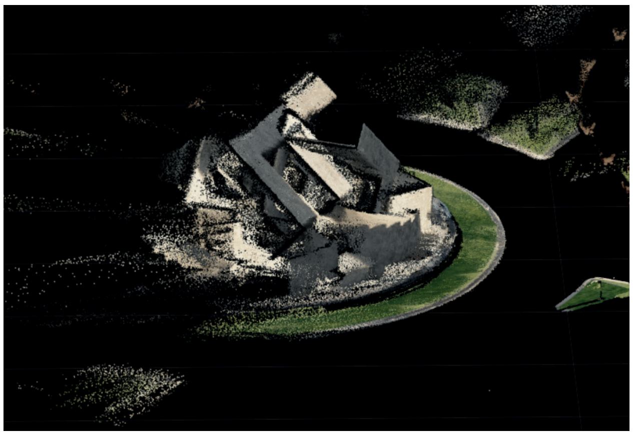


Fig. 6 Colmap dense reconstruction result with default parameter set

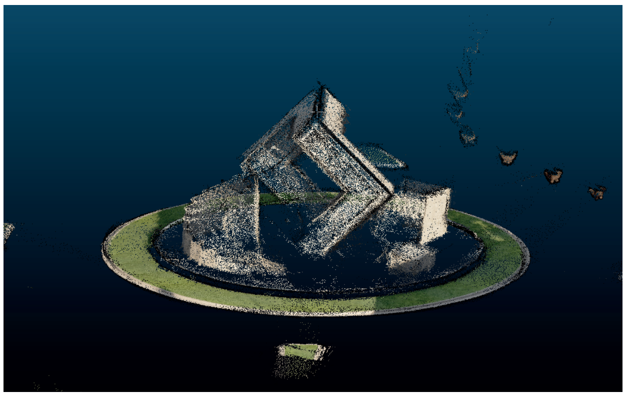


Fig. 7 Reconstructed dense point cloud after providing camera position data

The last step of the ICP algorithm measures the error between the point clouds and minimizes the distance between them. Either a "point-to-point" or a "point-to-plane" error metric is applied.

In the case of a "point-to-point" error metric, if p_i is a source point and q_i is the corresponding point in the target point cloud and M is the transformation matrix, then the sum of squared distances must be calculated and minimized as

$$i_{\min} = \underset{i}{\operatorname{argmin}} \sum_i \left(M \cdot p_i - q_i \right)^2 , \tag{1}$$

and

$$M_{\min} = M_{i_{\min}} . \tag{2}$$

Closed-form solutions for this kind of error metric exist, such as singular value decomposition (SVD), dual quaternions, quaternions, and orthonormal matrices [60].

The point-to-plane error metric converges better than the point-to-point error metric. It minimizes the sum of squared distances between source points and the tangent plane at the target point which is orthogonal to the unit normal vector of that point. Mathematically, if p_i is a source point and q_i is the corresponding point in the target point cloud and $\boldsymbol{n}$ is the normal vector at q_i then the ICP algorithm estimates the rigid transformation matrix by the minimizing function:

$$i_{\min}^{\text{norm}} = \underset{i}{\operatorname{argmin}} \sum_i \left(\left(M \cdot p_i - q_i \right) \cdot \boldsymbol{n} \right)^2 , \tag{3}$$

and

$$M_{\min}^{\text{norm}} = M_{i_{\min}^{\text{norm}}} . \tag{4}$$

No closed form solutions exist for point-to-plane error metric, so it is usually solved iteratively by nonlinear methods such as Levenberg-Marquardt or it can be linearized considering some approximation for the rotation matrix. The problem of the point-to-plane error metric is that it is sensitive to noise and that it does not converge well if the distance between point clouds in question are large spatially [59].

The comparison can be done using CloudCompare, a 3D point cloud processing software that provides a set of basic tools for manually editing and rendering 3D points clouds and triangular meshes1. It also offers various advanced processing algorithms, among which methods for performing projections, registration, distance computation, segmentation, etc. In our analysis, we used CloudCompare v. 2.12.4.

CloudCompare supports most file formats both in input and output. Our main use for CloudCompare is to compare the original point cloud with the one generated from the images taken from the simulation with the use of SfM. The comparison is made with the registration functionality with the use of ICP algorithm. The registration was controlled using the point-to-point error metrics of the ICP algorithm. The results of our method were evaluated using cloud-to-cloud (C2C) distance of CloudCompare.

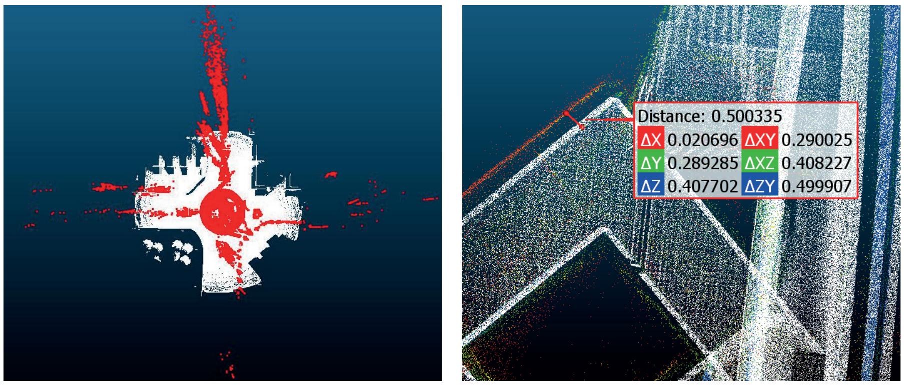
4 Results

The Ground Truth point cloud, which was exported from Carla, and is the synthesized Lidar-collected data, contained around 2 million points. The Colmap-reconstructed point cloud contained about 371 000 points. The point cloud registration was done with the ICP point-to-point method in CloudCompare. The two clouds were imported and then aligned using only translation and rotation; the scale was kept fixed. Since the Colmap-reconstructed point cloud is less coherent, the ICP registration was done by removing the farthest points, resulting in a weighted reliability metric. The weighted RMS (Root Mean Square) error of the registration was 0.0449 m. The theoretical overlap is 100%. Out of 371 269 points of the Colmap-reconstructed cloud, 236 545 were used in the ICP registration, and the rest were ignored.

The registration quality was measured as distances between the point clouds. The incoherence of the Colmap-reconstructed point cloud resulted in points with a distance of more than 200 m (which were therefore dropped). The amount of deleted points was less than 10% of the total points (about 37 000 points were cut). Most of these points were cut because they were outside the ground truth area, as shown in Fig. 8(a).

With Colmap's algorithm, the sides that were on fewer images have a ghost effect (Fig. 8(b)): most of the affected points show an average distance of 0.5 m, but the points describe a surface parallel with the surface of the ground truth shape.

After these qualitative analyses, we have also executed some quantitative tests, during which numerical values were derived to characterize the quality of the Colmap-reconstructed point cloud. One of the objective metrics can be the numerical expression of the difference between the ground truth and the obtained point clouds. The differences vary from point to point, therefore the ICP registration is characterizable by visualizing the distances between the



(a) (b)

Fig. 8 Colmap reconstruction failures; (a) White points of the ground truth cloud while red dots are from the Colmap-reconstructed cloud (b) Ghost effect on one side

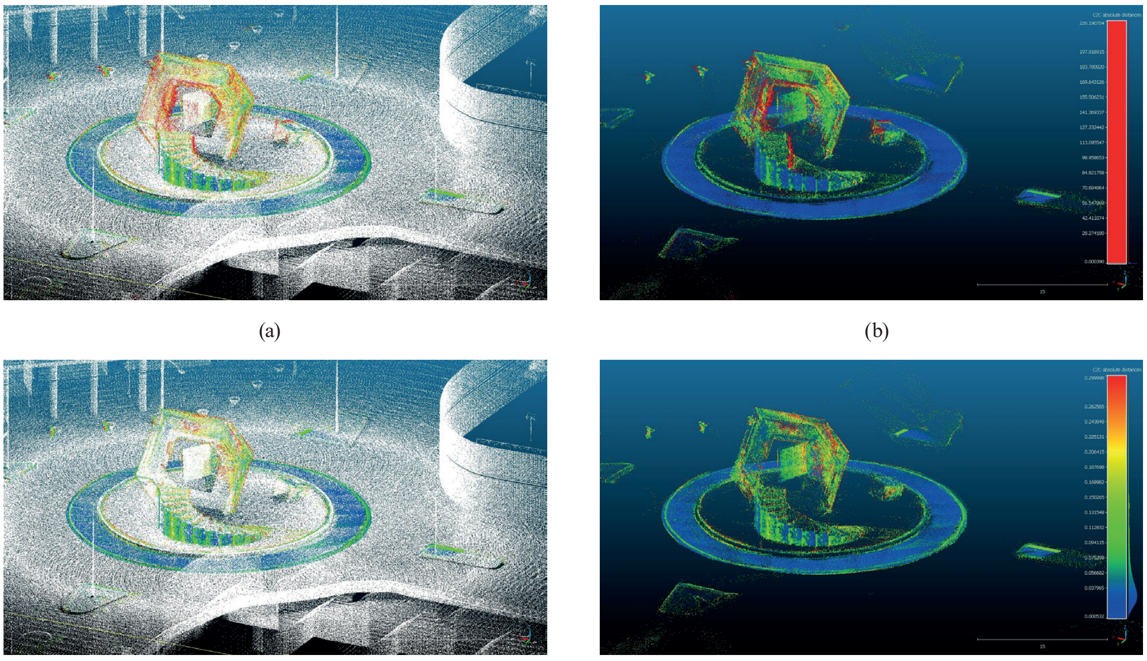
consecutive point regions. Fig. 9 demonstrates these quality measures with and without ground truth data, as well as with and without considering point removals.

With the generation of a histogram of obtained registration distances, further quantitative analysis demonstrates the distribution of registration differences. Analyzing the histogram (Fig. 10), one can notice that 54% of all points were closer than 5 cm, which underlines the reliability and high quality of the obtained synthetic images and their reconstruction in the Colmap software environment.

The point cloud obtained from Colmap is worth comparing to other SfM-based reconstruction techniques. Our previous experience [31, 61, 62] has shown that the available SfM and MVS software packages are very similar in reconstruction methodology and similar outcomes can be predicted. However, to the best of our knowledge, no equivalent workflow exists that would allow direct comparison of our results.

5 Conclusions

Autonomous vehicles place significant demands on map data. This has led to a significant change in the way maps of reality are produced: the emergence of HD maps. At the same time, using computers in the development process has opened up a new field, namely simulation. It makes it cheaper, faster and more efficient to study vehicle



(a) (b) (c) (d)

Fig. 9 Registration quality maps; (a) Registration quality map: lower to higher distances are from blue to red coloring (b) The results of the calculation without the ground truth (c) The distances to ground truth after removing points farther than 30 cm (d) The results of the calculation after removing points

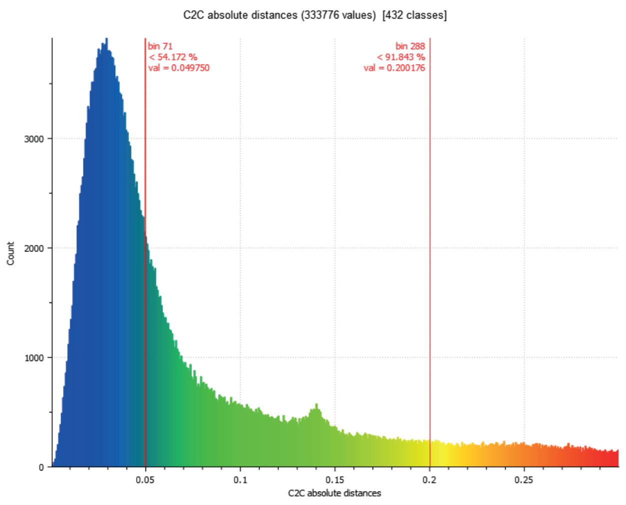


Fig. 10 Histogram after removing points farther than 30 cm with the percentage of points closer than 5 cm and 20 cm

components and even vehicles. Simulation is of similar importance for vehicle management: in addition to common everyday situations, it is also easy to investigate rare cases. Significant efforts are also being made to provide simulators with real or realistic field data. The OpenDRIVE format is the best-known example of this.

Computer simulation has the tools to produce the measurement data required for the test, i.e., onboard equipment can be simulated. This means that an order of magnitude increase in the number of measurements is available to testers. One of the most spectacular simulation applications is the imitation of onboard cameras, however not simply in terms of whether, for example, a camera can detect a cyclist or other road object approaching an ego-vehicle from the cross traffic approaching an intersection, but also for synthesizing raw imagery in a crowdsourced data acquisition system.

The crowdsourcing nature of our solution lies in the data acquisition simulation. The approach uses windscreen-mounted cameras as data sources volunteered by the operators of the vehicles in Carla simulator in realtime. These vehicles represent the crowd of different relative positioned cameras with diverse settings – similarly to the real crowd parameters. Future analyses can cover further variations in parameter settings and tests with additional influencing factors, like speed variations, occlusions, and weather effects.

In our work, we have produced full-fledged camera images from the synthetic environment (computer environment model), frames that are "seen" by the onboard camera of a virtual vehicle with parameterized motion. The stream of location-to-location images can then be acquired and processed using the same Structure-from-Motion and MultiViewStereo-based pipeline as with the real-world windshield camera image streams.

Our paper, therefore, demonstrates the equivalent capabilities of synthetic imagery as a component of a development methodology.

As a validation of the processing, we performed a comparison between the synthetic space-based image sequence and the point cloud from the underlying space. From the image sequence, we derived a point cloud using reconstruction that is comparable to the ground truth point cloud of the artificial environment.

In future research, we plan to extend this technology by adding different types of noise to the sequence of images that make up the measurement and by inserting disturbances, such as moving objects, into the synthetic model, which will appear in the resulting images – similar to the ones captured in the real world. Similarly, we plan to use 3D environmental data stored in geospatial information systems to produce an OpenDRIVE representation of them, which will naturally form the basis of the artificial space through direct simulator coupling.

Our work has shown that existing environmental models (e.g., city models), which may have been created for other technical or administrative purposes, are suitable to provide the field/onboard data that is essential for developing the automotive industry. We also recognized that the software packages underlying object reconstruction technologies are indifferent between providing results by processing real or synthetic data.

Acknowledgement
The research was supported by the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

The research reported in this paper is part of project no. BME-NVA-02, implemented with the support provided by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021 funding scheme.

References

[1] Hofmann-Wellenhof, B., Legat, K., Wieser, M. "Navigation", Springer, 2003. ISBN 978-3-211-00828-7
https://www.doi.org/10.1007/978-3-7091-6078-7

[2] Torge, W., Müller, J. "Geodesy", De Gruyter, 2012. ISBN: 9783110207187
https://www.doi.org/10.1515/9783110250008

[3] NDS "Navigation Data Standard (NDS) - The worldwide standard for map data in automotive eco-systems", Navigation Data Standard, Gries, Germany, 2022. [online] Available at: https://nds-association.org/

[4] Tao, C. V., Li, J. "Advances in Mobile Mapping Technology", CRC Press, 2007. ISBN 9780429224409
https://www.doi.org/10.4324/9780203961872

[5] Craig, J. "Map data for ADAS", In: Eskandarian, A. (ed.) In Handbook of Intelligent Vehicles, Springer, 2012, pp. 881–892. ISBN 978-0-85729-084-7
https://www.doi.org/10.1007/978-0-85729-085-4_33

[6] Winner, H., Hakuli, S., Lotz, F., Singer, C. "Handbook of driver assistance systems: Basic information, components and systems for active safety and comfort", Springer, 2015. ISBN 978-3-319-12351-6
https://www.doi.org/10.1007/978-3-319-12352-3

[7] Kahl, B. "Autonomous Driving Data Chain & Interfaces", arXiv:2104.01252 [cs.RO], 2021.
https://www.doi.org/10.48550/arXiv.2104.01252

[8] Liu, R., Wang, J., Zhang, B. "High Definition Map for Automated Driving: Overview and Analysis", Journal of Navigation, 73(2), pp. 324–341, 2020.
https://www.doi.org/10.1017/S0373463319000638

[9] Karamanov, N., Andreev, D., Pfeifle, M., Bock, H., Otto, M., Schulze, M. "Map Line Interface for Autonomous Driving", In: Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 2018, pp. 26–33. ISBN 978-1-7281-0321-1
https://www.doi.org/10.1109/ITSC.2018.8569378

[10] Dahlströhm, T. "The road to everywhere: are HD maps for autonomous driving sustainable?", Autonomous Vehicle International, 2021.

[11] Lógó, J. M., Krausz, N., Potó, V., Barsi, A. "Quality Aspects of High-Definition Maps", The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLIII-B4-2021, pp. 389–394, 2021.
https://www.doi.org/10.5194/isprs-archives-xliii-b4-2021-389-2021

[12] Shimada, H., Yamaguchi, A., Takada, H., Sato, K. "Implementation and Evaluation of Local Dynamic Map in Safety Driving Systems", Journal of Transportation Technologies, 5(2), pp. 102–112, 2015.
https://www.doi.org/10.4236/jtts.2015.52010

[13] Szántó, M., Vajta, L. "Towards an intelligent traffic control system using crowdsourcing, based on combined evaluation of weather information and accident statistics", Időjárás - Quarterly Journal of the Hungarian Meteorological Service, 123(3), pp. 295–312, 2019.
https://www.doi.org/10.28974/idojaras.2019.3.3

[14] Howe, J. "The rise of crowdsourcing", Wired Magazine, 14, pp. 1–4, 2016.

[15] Gao, H., Barbier, G., Goolsby, R. "Harnessing the crowdsourcing power of social media for disaster relief", IEEE Intelligent Systems, 26(3), pp. 10–14, 2011.
https://www.doi.org/10.1109/MIS.2011.52

[16] Tucker, J. D., Day, S., Tang, W., Bayus, B. "Crowdsourcing in medical research: concepts and applications", Peer J, 7, e6762, 2019.
https://www.doi.org/10.7717/peerj.6762

[17] Lucic, M. C., Wan, X., Ghazzai, H., Massoud, Y. "Leveraging Intelligent Transportation Systems and Smart Vehicles Using Crowdsourcing: An Overview", Smart Cities, 3(2), pp. 341–361, 2020.
https://www.doi.org/10.3390/smartcities3020018

[18] Lin, Y., Li, R. "Real-time traffic accidents post-impact prediction: Based on crowdsourcing data", Accident Analysis & Prevention, 145, 105696, 2020.
https://www.doi.org/10.1016/j.aap.2020.105696

[19] Goodchild, M. F. "Citizens as sensors: the world of volunteered geography", GeoJournal, 69, pp. 211–221, 2007.
https://doi.org/10.1007/s10708-007-9111-y

[20] Sui, D., Elwood, S., Goodchild, M. "Crowdsourcing geographic knowledge: volunteered geographic information (VGI) in theory and practice", Springer, 2012. ISBN 978-94-007-4586-5
https://www.doi.org/10.1007/978-94-007-4587-2

[21] Haklay, M., Weber, P. "OpenStreetMap: User-generated street maps", IEEE Pervasive Computing, 7(4), pp. 12–18, 2008.
https://www.doi.org/10.1109/MPRV.2008.80

[22] Neuhold, G., Ollmann, T., Rota Bulò, S., Kontschieder, P. "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes", In: Proceedings of the Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 5000–5009. ISBN:978-1-5386-1033-6
https://www.doi.org/10.1109/ICCV.2017.534

[23] Szántó, M., Vajta, L. "Introducing CrowdMapping: a novel system for generating autonomous driving aiding traffic network databases", In: Proceedings of the 2019 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO), Athens, Greece, pp. 7–12, 2019. ISBN 978-1-7281-3573-1
https://www.doi.org/10.1109/ICCAIRO47923.2019.00010

[24] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, …, Anguelov, D. "Scalability in perception for autonomous driving: Waymo open dataset", In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 2020, pp. 2446–2454. ISBN: 978-1-7281-7169-2
https://www.doi.org/10.48550/arXiv.1912.04838

[25] Houston, J., Zuidhof, G., Bergamini, L., Ye, Y., Chen, L., Jain, A., Omari, S., Iglovikov, V., Ondruska, P. "One Thousand and One Hours: Self-driving Motion Prediction Dataset", In: Proceedings of the 4th Conference on Robot Learning, CoRL 2020, Virtual Conference, 2021, pp. 409–418.
https://www.doi.org/10.48550/arXiv.2006.14480

[26] Bârsan, I. A., Wang, S., Pokrovsky, A., Urtasun, R. "Learning to Localize Using a LiDAR Intensity Map", In: Proceedings of the Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 2018, pp. 605–616.
https://www.doi.org/10.48550/arXiv.2012.10902

[27] Anguelov, D., Dulong, C., Filip, D., Frueh, C., Lafon, S., Lyon, R., Ogale, A., Vincent, L., Weaver, J. "Google street view: Capturing the world at street level", Computer, 43(6), pp. 32–38, 2010. https://www.doi.org/10.1109/MC.2010.170

[28] Antequera, M. L., Gargallo, P., Hofinger, M., Bulò, S. R., Kuang, Y., Kontschieder, P. "Mapillary planet-scale depth dataset", In: Proceedings of 16th European Conference on Computer Vision, Online, 2020, pp. 589–604. https://www.doi.org/10.1007/978-3-030-58536-5_35

[29] Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S. M., Szeliski, R. "Building Rome in a Day", Communications of the ACM, 54(10), pp. 105–112, 2011. https://www.doi.org/10.1145/2001269.2001293

[30] Hirschmuller, H. "Stereo Processing by Semiglobal Matching and Mutual Information", IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(2), pp. 328–341, 2008. https://www.doi.org/10.1109/TPAMI.2007.1166

[31] Somogyi, A., Barsi, A., Molnar, B., Lovas, T. "Crowdsourcing Based 3D Modeling", The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLI-B5, pp. 587–590, 2016. https://www.doi.org/10.5194/isprs-archives-XLI-B5-587-2016

[32] Ullman, S. "The interpretation of structure from motion", Proceedings of the Royal Society of London. Series B. Biological Sciences, 203, pp. 405–426, 1979. https://www.doi.org/10.1098/rspb.1979.0006

[33] Hu, Q., Luo, J., Hu, G., Duan, W., Zhou, H. "3D Point Cloud Generation Using Incremental Structure-from-Motion", Journal of Physics: Conference Series, 1087(6), 062031, 2018. https://www.doi.org/10.1088/1742-6596/1087/6/062031

[34] Tomasi, C., Kanade, T. "Shape and motion from image streams under orthography: A factorization method", International Journal of Computer Vision, 9, pp. 137–154, 1992. https://www.doi.org/10.1007/BF00129684

[35] Pantoja-Rosero, B. G., Achanta, R., Kozinski, M., Fua, P., Perez-Cruz, F., Beyer, K. "Generating LOD3 building models from structure-from-motion and semantic segmentation", Automation in Construction, 141, 104430, 2022. https://www.doi.org/10.1016/j.autcon.2022.104430

[36] Kraus, K. "Photogrammetry", De Gruyter, 2011. ISBN 978 3 11 089287 1 https://www.doi.org/10.1515/9783110892871

[37] Wei, Y., Kang, L., Yang, B., Wu, L. "Applications of structure from motion: a survey", Journal of Zhejiang University SCIENCE C, 14, pp. 486–494, 2013. https://www.doi.org/10.1631/jzus.CIDE1302

[38] Westoby, M. J., Brasington, J., Glasser, N. F., Hambrey, M. J., Reynolds, J. M. "'Structure-from-Motion' photogrammetry: A low-cost, effective tool for geoscience applications", Geomorphology, 179, pp. 300–314, 2012. https://www.doi.org/10.1016/j.geomorph.2012.08.021

[39] Sturm, J., Burgard, W., Cremers, D. "Evaluating egomotion and structure-from-motion approaches using the TUM RGB-D benchmark", In: Proceedings of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS), Vilamoura-Algarve, Portugal, 2012, pp. 573–580.

[40] Mooser, J., You, S., Neumann, U., Wang, Q. "Applying robust structure from motion to markerless augmented reality", In: Proceedings of the 2009 Workshop on Applications of Computer Vision (WACV), Snowbird, UT, USA, 2009, pp. 1–8. ISBN 978-1-4244-5497-6 https://www.doi.org/10.1109/WACV.2009.5403038

[41] Schönberger, J. L., Frahm, J. M. "Structure-from-Motion Revisited", In: Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 4104–4113. https://www.doi.org/10.1109/CVPR.2016.445

[42] Sweeney, C., Hollerer, T., Turk, M. "Theia: A fast and scalable structure-from-motion library", In: Proceedings of the Proceedings of the 23rd ACM international conference on Multimedia, Brisbane, Australia, 2015, pp. 693–696. ISBN 978-1-4503-3459-4 https://www.doi.org/10.1145/2733373.2807405

[43] Moulon, P., Monasse, P., Perrot, R., Marlet, R. "OpenMVG: Open multiple view geometry", In: Proceedings of the International Workshop on Reproducible Research in Pattern Recognition, Cancún, Mexico, 2016, pp. 60–74. ISBN 978-3-319-56413-5 https://www.doi.org/10.1007/978-3-319-56414-2_5

[44] Wu, C. "Towards linear-time incremental structure from motion", In: Proceedings of the 2013 International Conference on 3D Vision-3DV 2013. IEEE, Seattle, WA, USA, 2013, pp. 127–134. ISBN 978-0-7695-5067-1 https://www.doi.org/10.1109/3DV.2013.25

[45] On-Road Automated Driving (ORAD) Committee "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles", Ground Vehicle Standard J3016_202104, 2014. https://www.doi.org/10.4271/J3016_202104

[46] Moujahid, A., Tantaoui, M. E., Hina, M. D., Soukane, A., Ortalda, A., ElKhadimi, A., Ramdane-Cherif, A. "Machine Learning Techniques in ADAS: A Review", In: Proceedings of the 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE), Paris, France, 2018, pp. 235–242. ISBN 978-1-5386-4486-7 https://www.doi.org/10.1109/ICACCE.2018.8441758

[47] Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E. "Deep learning for computer vision: A brief review", Computational Intelligence and Neuroscience, 2018, 7068349, 2018. https://www.doi.org/10.1155/2018/7068349

[48] Heaton, J. "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep Learning", Genetic Programming and Evolvable Machines, 19, pp. 305–30, 2018. https://www.doi.org/10.1007/s10710-017-9314-z

[49] Sadeghi, F., Toshev, A., Jang, E., Levine, S. "Sim2Real Viewpoint Invariant Visual Servoing by Recurrent Control", In: Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 4691–4699. ISBN 978-1-5386-6421-6 https://www.doi.org/10.1109/CVPR.2018.00493

[50] Zhao, W., Queralta, J. P., Westerlund, T. "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey", In: Proceedings of the 2020 IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, ACT, Australia, 2020, pp. 737–744. ISBN 978-1-7281-2548-0 https://www.doi.org/10.1109/SSCI47803.2020.9308468

[51] Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P. P., Barron, J. T., Kretzschmar, H. "Block-nerf: Scalable large scene neural view synthesis", In: Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 2022, pp. 8248–8258. ISBN 978-1-6654-6947-0
https://www.doi.org/10.48550/arXiv.2202.05263

[52] Gupta, A., Anpalagan, A., Guan, L., Khwaja, A. S. "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues", Array, 10, 100057, 2021.
https://www.doi.org/10.1016/j.array.2021.100057

[53] Zhang, J., Tai, L., Yun, P., Xiong, Y., Liu, M., Boedecker, J., Burgard, W. "Vr-goggles for robots: Real-to-sim domain adaptation for visual control", IEEE Robotics and Automation Letters, 4, pp. 1148–1155, 2019.
https://www.doi.org/10.1109/LRA.2019.2894216

[54] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V. "CARLA: An Open Urban Driving Simulator", In: Proceedings of the Proceedings of the 1st Annual Conference on Robot Learning, Mountain View, CA, USA, 2017, pp. 1–16.
https://www.doi.org/10.48550/arXiv.1711.03938

[55] Wang, C.-S., Liu, D.-Y., Hsu, K.-S. "Simulation and application of cooperative driving sense systems using prescan software", Microsystem Technologies, 27, pp. 1201–1210, 2021.
https://www.doi.org/10.1007/s00542-018-4164-z

[56] Sieberg, P. M., Schramm, D. "Ensuring the Reliability of Virtual Sensors Based on Artificial Intelligence within Vehicle Dynamics Control Systems", Sensors, 22(9), 3513, 2022.
https://www.doi.org/10.3390/s22093513

[57] Schmidt, S., Henning, J., Wambera, T., Kronimus, S. "Early PC-based Validation of ECU Software Using Virtual Test Driving", ATZelektronik worldwide, 10, pp. 14–17, 2015.
https://www.doi.org/10.1007/s38314-015-0508-y

[58] Zhou, Q.-Y., Park, J., Koltun, V. "Open3D: A Modern Library for 3D Data Processing", arXiv:1801.09847 [cs.CV], 2018.
https://www.doi.org/10.48550/arXiv.1801.09847

[59] Krell, G., Saeid Nezhad, N., Walke, M., Al-Hamadi, A., Gademann, G. "Assessment of iterative closest point registration accuracy for different phantom surfaces captured by an optical 3D sensor in radiotherapy", Computational and Mathematical Methods in Medicine, 2017, 2938504, 2017.
https://www.doi.org/10.1155/2017/2938504

[60] Eggert, D. W., Lorusso, A., Fisher, R. B. "Estimating 3-D rigid body transformations: a comparison of four major algorithms", Machine Vision and Applications, 9, pp. 272–290, 1997.
https://www.doi.org/10.1007/s001380050048

[61] Somogyi, Á., Lovas, T., Barsi, Á. "Comparison of spatial reconstruction software packages using DSLR images", Pollack Periodica, 12(2), pp. 17–27, 2017.
https://www.doi.org/10.1556/606.2017.12.2.2

Appendix

Simulated Lidar sensor paramaters

The parameters of the simulated Lidar sensor introduced in Section 3.2, can be seen in Table A1. Italic parameter names show that the used value differs from the default settings of a Lidar sensor in the Carla simulation environment. Default settings are listed here: https://carla.readthedocs.io/en/latest/ref_sensors/ [54].

Table 1 Parameters of the synthesized Lidar sensor

Parameter	Value
Number of lasers	32
Range	50 m
Points per second	1×10^6
Rotation frequency	30 Hz
Upper field of view	50°
Lower field of view	–30°
Number of channels	64
Horizontal field of view	360°
Atmosphere attenuation rate	4×10^{-3}
Dropoff general rate	4.5×10^{-1}
Dropoff zero intensity	8×10^{-1}
Sensor tick	0
Noise standard deviation	0