Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Improving efficiency of DNN-based relocalization module for autonomous driving with server-side computing

Dengbo Li[1,4], Hanning Zhang[2,3], Jieren Cheng[1,4*] and Bernie Liu[5]

Abstract

The substantial computational demands associated with Deep Neural Network (DNN)-based camera relocalization during the reasoning process impede their integration into autonomous vehicles. Cost and energy efficiency considerations may dissuade automotive manufacturers from employing high-computing equipment, limiting the adoption of advanced models. In response to this challenge, we present an innovative edge cloud collaborative framework designed for camera relocalization in autonomous vehicles. Specifically, we strategically offload certain modules of the neural network to the server and evaluate the inference time of data frames under different network segmentation schemes to guide our offloading decisions. Our findings highlight the vital role of server-side offloading in DNN-based camera relocation for autonomous vehicles, and we also discuss the results of data fusion. Finally, we validate the effectiveness of our proposed framework through experimental evaluation.

Keywords Autonomous Driving, Offloading, Relocalization, Edge Cloud Collaboration

Introduction

Autonomous driving technology is a longstanding challenge in academia and the automotive industry, necessitating precise environmental maps and accurate self-positioning for a safe and stable driving experience. The emergence of mobile edge computing has prompted exploration for its application in intelligent vehicles [1–5]. Relocalization plays a crucial role in the application of autonomous vehicles, allowing them to achieve precise orientation within high-precision maps. Currently, edge computing and artificial intelligence technologies are extensively employed in this domain [6–12].

The traditional relocalization method involves building a feature descriptor database based on the existing map data and matching the query image with the feature descriptor [13–15]. Currently, many efficient retrieval methods exist, such as deep learning-based descriptor construction [16, 17], word bag models [18, 19], and VLAD [20]. Previous researchers have used priority correspondence search and 3D point division to improve positioning efficiency [21, 22]. Additionally, they have utilized SIFT features and correspondence between 3D points and 2D points in the scene to enhance positioning accuracy [23, 24]. The traditional camera relocalization approach relies heavily on shallow feature information in the scene, which can lead to poor relocalization accuracy, large drift, and other failures.

In recent years, there has been a growing interest in using deep learning-based scene construction models to perform pose regression. PoseNet [25] was the first to propose using the GoogleNet [26] depth neural network

*Correspondence:
Jieren Cheng
cjr22@163.com
[1] School of Computer Science and Technology, Hainan University, Haikou 570228, China
[2] China Unicom(Hainan) Innovation Research Institute, Haikou 570100, China
[3] Hainan Branch, China United Network Communications Group Co. Ltd ("China Unicom"), Haikou 570100, China
[4] Hainan Blockchain Technology Engineering Research Center, Haikou 570228, China
[5] Hainan Shuyi Technology Co., Ltd, Sanya 572025, China

to directly return the 6-DoF pose of an input image. This method leverages the powerful feature extraction capabilities of neural networks to capture contextual features. The accuracy of pose estimation can be further improved by adding uncertainty measures and geometric loss to the PoseNet framework [27, 28]. RobustLoc [29] obtained robustness to environmental disturbances from differential equations, extracted feature maps using CNN, and estimated vehicle attitude using a branch attitude decoder with multi-layer training. This method achieved robust performance in various environments. Simply using sparse descriptors can also regress scene coordinates without the need for dense RGB images [30]. The basic connection and coupling between multiple tasks can improve the model's understanding of the scene, further improving positioning performance [31, 32].

Despite the significant advancements achieved by deep learning-based relocation methods in terms of performance and scene adaptability, certain inherent challenges remain to be addressed. For instance, while the problem of memory resource allocation is mitigated by representing the map as a deep neural network, the reasoning process still relies heavily on computational resources. Considering that computing on self-driving cars is usually limited by factors such as power consumption and cost (higher-performance hardware brings higher costs), this will in turn affect the autonomy of the car. Edge computing technology brings opportunities to solve these confusions [33–36].

We use mobile edge computing offloading to solve these problems. Our framework enables autonomous vehicles to perform complex model reasoning and provide relocation information for the vehicle without relying on on-board high-performance computing equipment. The main contributions of this article are:

- a cross-device edge-cloud collaborative offloading framework for autonomous vehicle camera relocalization eases the requirements of autonomous vehicles for high-performance computing equipment,
- the advantages of our proposed framework in terms of computational efficiency are demonstrated through simulation experiments on an advanced MapNet series relocalization model.
- by improving the two performance indicators of frequency and route, we demonstrate the prospects of our framework in multi-source information fusion.

Related work

MapNet series camera relocation scheme

This series comprises an enhanced version of PoseNet [25] and two newly proposed architectures, MapNet [37] and MapNet++. The backbone network of PoseNet adopts GoogleNet [26], and the representation of the direction when returning to posture is a four-dimensional unit quaternion. The authors of the MapNet series have pinpointed shortcomings in this approach, as detailed in their paper. To address these issues, they replaced the backbone network GoogleNet with ResNet34 [38], and changed the direction representation to the logarithm of a unit quaternion.

In this method, both the absolute pose loss of each input image frame and the relative pose loss between two image pairs related to this frame are minimized. The learned pose feature distribution using this method has a strong correlation with the actual value, and the combination of absolute and relative loss ensures global consistency in pose estimation, leading to a significant improvement in relocation performance.

Due to the labor-intensive and costly nature of labeling data, a significant amount of unannotated data exists in the real world, such as robot trajectory and GPS data obtained via visual odometer calculations. MapNet++leverages these unmarked data to fine-tune the supervised training network in a self-supervised manner, the performance of pose estimation has been improved.

Upon obtaining the absolute pose of the targeted frame through MapNet++, the proposed approach utilizes the pose graph optimization (PGO) [39] algorithm to fuse the absolute pose with the relative pose between frames obtained by the VO algorithm, thereby establishing a longer constraint on the corresponding trajectory for the targeted frame. The integrated method enhances the accuracy of pose estimation.

Mobile edge computing offloading computing resources

The CORA [40] algorithm leverages reinforcement learning technology to tackle challenges posed by dynamic environments in edge cloud collaboration.The adaptive DNN inference acceleration framework [41] utilizes neural networks to learn features associated with inference delay and identify the optimal partitioning point. The model training strategy based on edge cloud collaboration [42] can achieve power-sharing in edge cloud computing, ensuring ample computing power for model training. SwarmMap [43] extends collaborative visual SLAM services in edge offloading settings to minimize data redundancy.Utilizing an edge cloud diversion mechanism ensures traffic service quality and facilitates efficient allocation of edge-side resources [44]. The delay in mobile edge computing can be minimized by decomposing the edge cloud resource allocation problem to calculate optimal task allocation ratios and resource allocation strategies, respectively [45]. RecSLAM [46] employs layered map fusion technology to ensure workload balancing among edge servers. EDDI framework [47] supports

collaborative on-demand inference acceleration, addressing optimization challenges and meeting user latency requirements.Implementing a Block Reuse Mechanism (CRM) [48] for cloud and remote nodes can reduce the data required for image construction and enhance container update efficiency.

Methods

In this section, we aim to introduce the DNN-based relocation module and the challenges it encounters, followed by the presentation of the advanced design of the DNN-based relocation module, which constitutes an upgraded version of the MapNet [37] series relocation methods that supports the offloading of the computationally intensive reasoning process to the server. Our goals for this upgraded version are two-fold: first, to enhance the reasoning speed of the network on the autonomous vehicle without requiring high-performance computing equipment; second, to ensure that our proposed method achieves the same accuracy as the original relocation model, particularly in terms of attitude regression. We note that we will not delve into the optimization of vehicle posture using PGO [39] in a sliding window, which is a separate step from the MapNet reasoning model, since it necessitates posture estimation from multiple frames of images. Our focus remains solely on the computation and reasoning involved in the original MapNet series model for autonomous vehicle posture regression.

Overview of DNN-based Relocalization Module

The left part of Fig. 1 depicts the pose reasoning flowchart based on the MapNet [37] series camera relocalization method. It can be observed that the relocalization module of the autonomous vehicle first acquires an environmental image (or frame) via the visual sensor, which is subsequently fed into a deep neural network (DNN)-based scene representation model. The system accepts regular RGB images, though it can also accept color and depth images simultaneously. Ultimately, the network reasoning outputs a pose.

Challenges in DNN-based Relocalization Module

The current deep neural network model consists of multiple hidden layers, and the processing of each frame is performed through each layer in turn, which requires a lot of calculations on the terminal device. This is especially challenging for devices without high-performance computing capabilities. During the posture estimation process of autonomous vehicles, the visual sensor captures environmental frames in a sequential manner. If the previous frame has not completed processing within the system, subsequent frames will not be estimated until processing for the previous frame has completed. This results in a delay in obtaining posture data and poses challenges for accurate vehicle posture correction. As shown in Fig. 2, the visual sensor may capture additional image data while processing a particular frame. In practical applications, it is not feasible to wait for the completion of processing for the previous frame before commencing processing for the subsequent frame, as this would not meet real-time processing requirements.

Offloading strategy

We propose a solution to address the time-consuming nature of the relocation module by offloading its computation to the server, while allowing subsequent processing to be carried out on the mobile device. The detailed reasoning process of the upgraded version is depicted in the right part of Fig. 1. We propose a strategy in which the less computationally demanding parts of the module

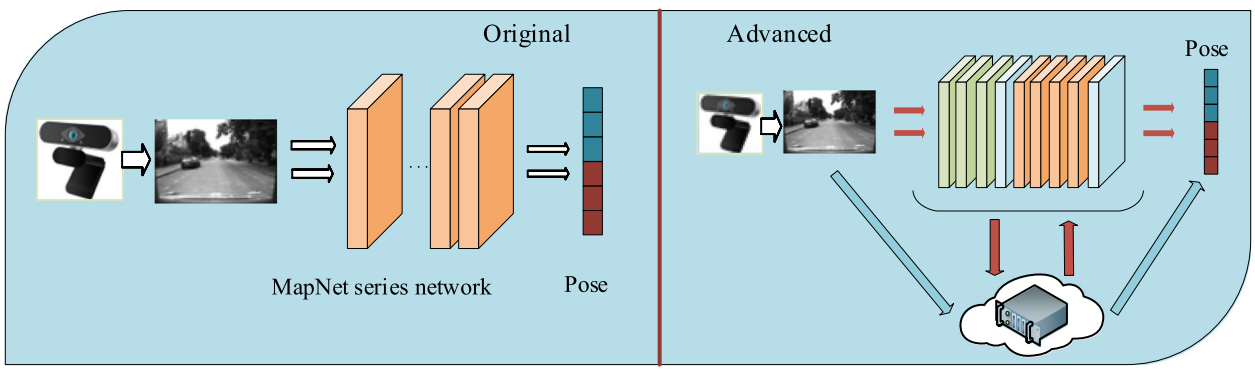

Fig. 1 Original: The relocation framework for the MapNet series. Feed the environmental photos collected by the sensors into the network for reasoning, and finally obtain the pose. Advanced: The upgraded version of the relocation framework splits the network, uploads the portion that consumes computing resources to the server for calculation, and finally returns the results to the mobile terminal for subsequent processing(The network types available for both frameworks are PoseNet-new version, MapNet, and MapNet++)

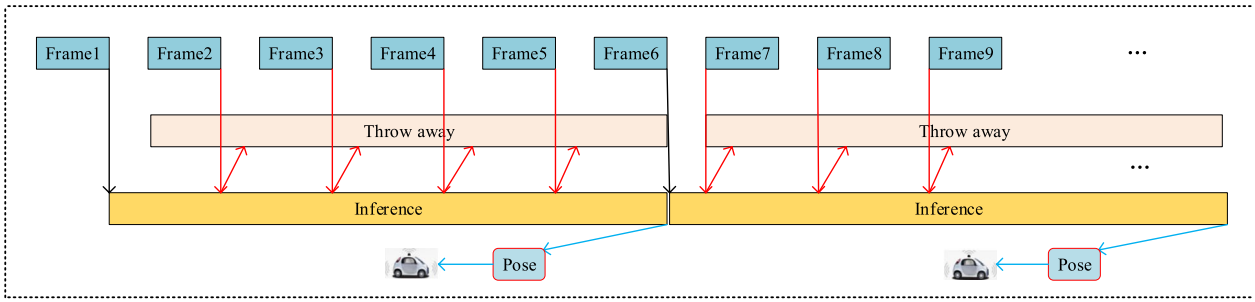

Fig. 2 The simulation pipeline of the original relocation framework, with red lines indicating discarded frames. Due to the huge amount of time spent in reasoning each frame, the environment frames captured by the sensor will be discarded during this period, and the vehicle will only receive a small amount of pose information

are computed locally, while the more intensive ones are offloaded to the server for processing. The server's higher computing power enables it to complete these calculations efficiently, after which it returns the results to the mobile device.

In response to the imperative for specific preprocessing of environmental frames and the potential requirement for terminal devices to conduct partial inference for heightened efficiency in specific scenarios, our terminal devices necessitate a certain level of computing power. However, the performance criteria are significantly diminished.

The network layering strategy for the MapNet [37] series of backbone networks is determined by calculating the offload times of several key layers. Our framework is designed to facilitate cloud-edge collaborative computing across three entities: local mobile devices, servers, and LAN. The model is deployed on both local mobile devices and servers. To conduct posture inference based on the offload strategy, we first establish data communication between the mobile device and server.

Figure 3 depicts the pipeline of the repositioning module in the upgraded autonomous vehicle proposed in this study. During the reasoning process, the local mobile device's visual sensor initially captures the environmental photograph, which is subsequently preprocessed to conform to the predefined input shape of the network. Meanwhile, the server opens a port to await the unloading request transmitted from the mobile device. Upon receiving the processing request from the mobile terminal, the server executes DNN to reason about the relevant data. Finally, the reasoning result is returned to the mobile device and used for vehicle auxiliary positioning. As can be seen, the upgraded relocalization pipeline produces more pose data than the method in Fig. 2 that only performs inference locally, because the inference time for a single frame is greatly reduced. While these changes are simple in theory, integrating these aspects to operate in a coordinated manner presents significant challenges. While this study proposes a conceptual offload design for the network, integrating the framework into a specific relocation scheme is imperative to determine the

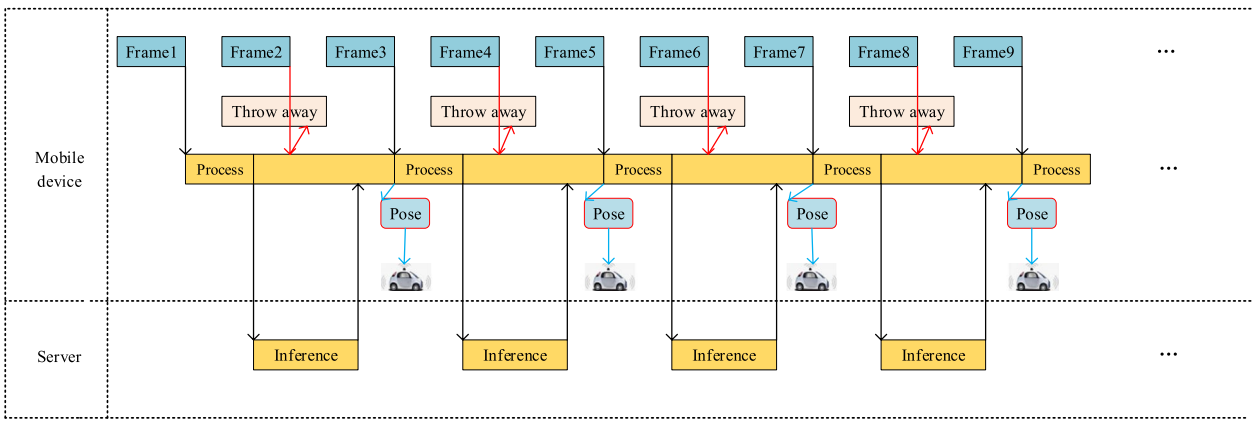

Fig. 3 The upgraded version repositions the simulation pipeline of the framework, with red lines indicating discarded frames. After offloading the computation with the help of the server, the inference time for each frame is significantly reduced. Compared to the previous framework, vehicles will receive more pose information at the same time

appropriate implementation of the offload. To validate the feasibility of our concept, we implemented the relocation framework based on MapNet [37] series.

Experimental Results

Setup

To evaluate our proposed offloading strategy, we conducted simulation experiments using two different computing devices to represent the mobile and server ends respectively. A low-cost development board, the NVIDIA Jetson Nano development kit with Quad-core ARM A57 CPU, 128-core Maxwell GPU (1.43 GHz), and 4 GB memory, was used as the computing center for simulating an autonomous vehicle. For the server module, we used a Dell notebook computer equipped with Intel (R) Core (TM) i5-7300HQ CPU (2.50 GHz, 4-core), NVIDIA GeForce GTX 1050 GPU, and 8 GB memory. The main difference between the server module and the mobile terminal is the higher performance processor of the former. Both devices run on Ubuntu 18.04LTS operating system and are connected to the same LAN. We chose L-

AN for data transmission because the focus of this work is on the impact of computing power on system performance.

Dataset

We employed the Oxford Robot dataset [49] and the 7Scenes dataset [50] as the input sources for our experiments, which were also used in the original MapNet [37] series. The Oxford Robot dataset was captured by a vehicle driving twice a week in the center of Oxford, over the course of more than a year. This dataset includes almost 20 million images and encompasses various weather conditions, allowing us to study the performance of autonomous vehicles in real-world and dynamic urban scenes. We selected the loop and full sequences mentioned in the original MapNet series paper, both of which comprise a large number of continuously captured road photos. The 7Scenes dataset includes RGB-D image sequences of seven indoor scenes, which were recorded by handheld Kinect RGB-D cameras. Each sequence in this dataset contains 500–1000 frames, and each frame includes an RGB image, depth image, and pose.

Experimental details

To ensure the applicability and effectiveness of the proposed framework, we employed a different approach compared to loading data in batches into the network. we utilized OpenCV to read the photos in the dataset frame by frame at the mobile end, which simulates the real-world data reading scenario of autonomous vehicles on the road.

Network split

Given that the network structure of MapNet [37] is derived from the PoseNet(new version) network with a ResNet34 [38] backbone, and the modification has only a slight impact on the computation, we conducted a network splitting experiment on the middle layer of the PoseNet(new version) network to represent our approach. To account for network transmission rate and fluctuation, we selected 100 consecutive images from the 7Scenes [50] dataset for the experiment and calculated the inference time for each image. The results were averaged to examine the effect of different layers of the neural network on inference time, as show in Fig. 4(d). In order to simulate realistic scene, we also conducted single-frame splitting inference. Tables 1 and 2 summarizes the final results for the two approaches. The main reason for the fluctuation in inference time from bn1 to fc is that the max-pool layer reduces the size of the tensor. At this time, the data transmission time between devices is greatly reduced. Table 3 shows the parameter amounts of the main layers. It can be found that after bn1, as the parameter amount increases, the inference time gradually increases. Our findings demonstrate that unloading all images is more conducive to on-device inference. we choose to offload the entire image and subsequently upload it to the server for inference.

Communication delay analysis

Since in actual situations, we also need to consider the impact of communication on inference time, we calculated the local computing time and server-side computing time of the data set during the inference process. The remaining space in the middle is their communication time. Figure 4(d) shows the correspondence between communication time and total inference time. The total size of the model parameters is 85.25MB. From the figure we can find that no matter where the local inference is, At one level, the communication time only fluctuates within a small range.

Configuration difference comparison

Figure 5 shows a comparative experiment of two sequences on the 7Scenes dataset. By using GeForce GTX for relocalization model inference, we observed that advanced devices can infer the entire model in 369 s, but if the model is only run on the local device, the model only infers few trajectories in the same time.

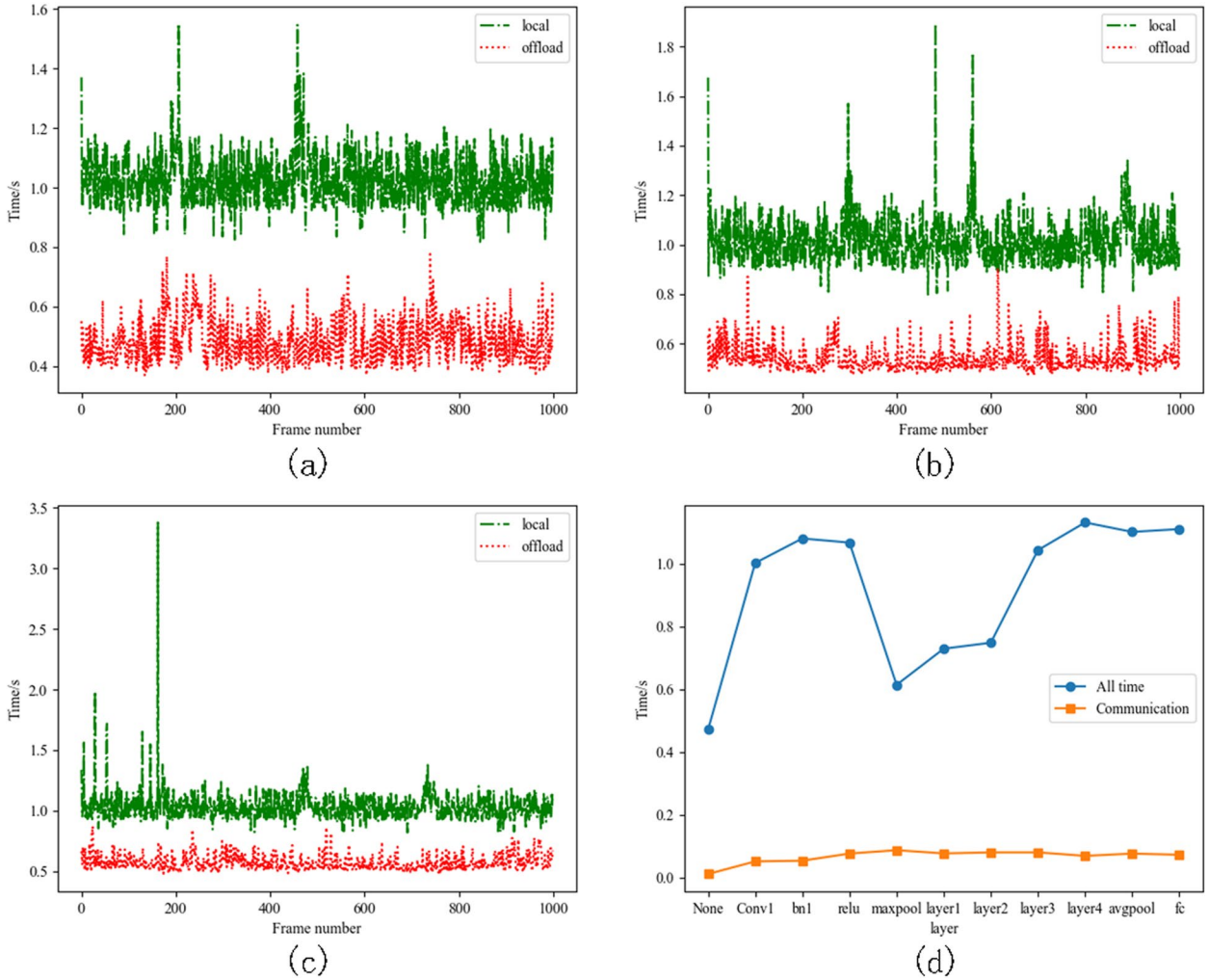

Fig. 4 **a** PoseNet (new version) inference time. **b** MapNet inference time. **c** MapNet + + inference time. **d** Network splitting experiments were conducted on the 7Scenes dataset. The abscissa represents the layer at which the network infers termination on mobile devices. For example, relu indicates that the network infers to this layer on the mobile device, and the remaining network inferences are performed at the server. The blue line represents the time it takes to complete inference, and the orange line represents the time it takes to communicate

Table 1 Single and multi frame network split results on the 7Scenes dataset(null-maxpool)

local inference to	null	conv1	bn1	relu	maxpool
average of 100 frames time/s	0.4710	1.0022	1.0804	1.0672	0.6140
single frame time/s	0.5612	1.2357	1.1516	1.5340	0.6589

Offloading time comparison

Building upon the current configuration, we select a predetermined number of frames from the dataset for our unloading tests and subsequently compare the outcomes of the two distinct configurations.

Local pose calculation Run PoseNet(new version), MapNet, MapNet + + on Jetson nano.

Table 2 Single and multi frame network split results on the 7Scenes dataset(layer1-end of)

local inference to	layer1	layer2	layer3	layer4	avgpool	fc
average of 100 frames time/s	0.7287	0.7480	1.0426	1.1310	1.1010	1.1099
single frame time/s	0.8266	0.7595	0.8537	0.9657	0.8700	0.8609

Table 3 Parameters of each layer of the network

layer	Conv1	Bn1	Layer1	Layer2	Layer3	Layer4	Fc
params	9408	128	221,952	1,116,416	6,822,400	13,114,368	1,050,624

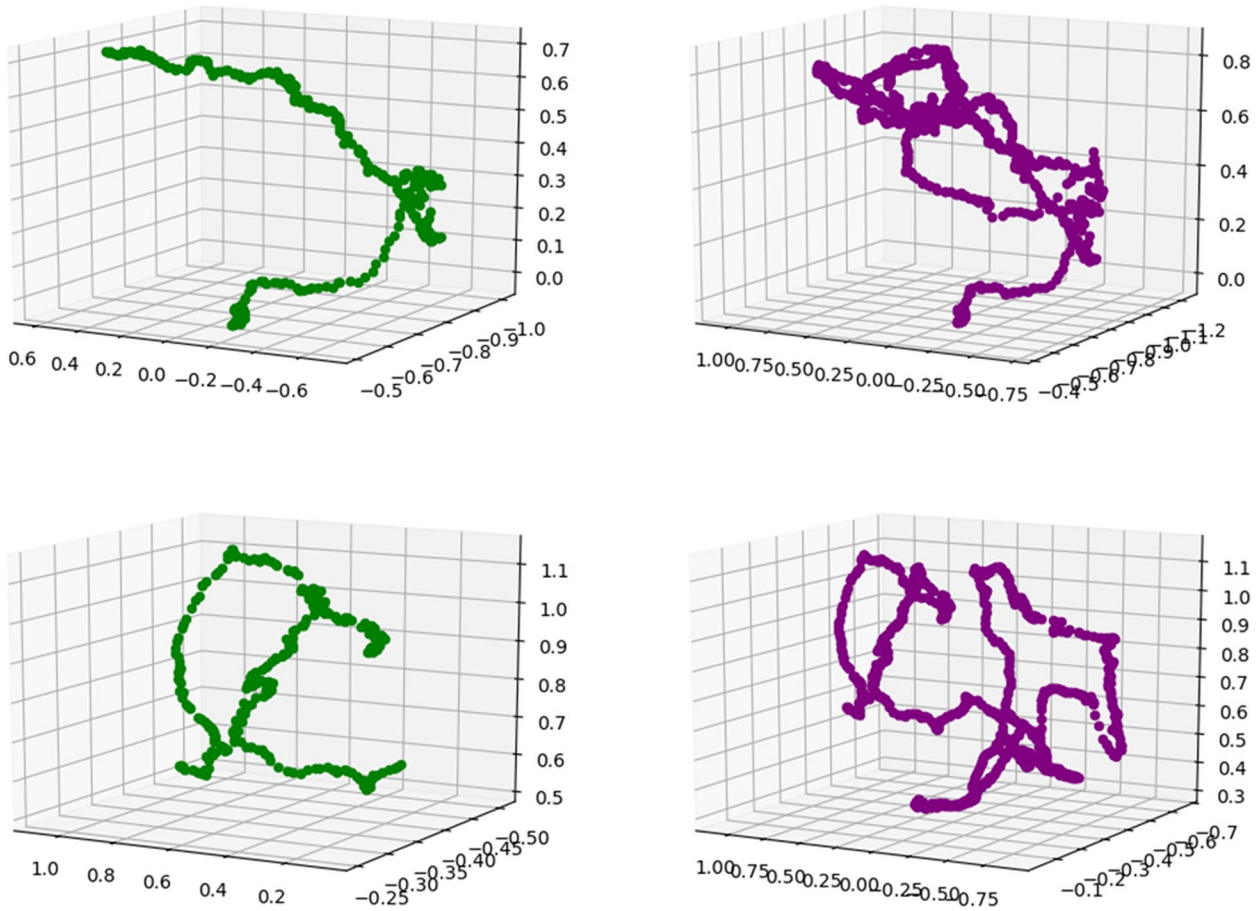

Fig. 5 Row 1: 7Scenes-pumpkin-07. Row 2: 7Scenes-fire-04. The left side is the local posture calculation, and the right side is the posture calculation on the server. The time is 369 s

Server-side computing Use our proposed offloading strategy to unload the three models.

Figure 4 illustrates the model reasoning time under two different configurations. It is evident that the powerful computing capability of the server has significantly improved the reasoning speed of the model.

Inference in Oxford RobotCar dataset and 7scenes dataset

Based on the aforementioned simulation results, we performed experiments on two datasets, namely RobotCar [49] and 7Scenes [50]. The network model selected for evaluation was the original paper's best-performing model, MapNet++, without any PGO [39] optimization. We evaluated the model's performance from three perspectives: accuracy, route, and inference frequency of two different schemes.

Accuracy

In the experiment conducted on the loop sequence of the RobotCar dataset, the red dots illustrated in Fig. 6 symbolize the inference outcomes derived from the offloading framework, while the green dots denote the groundtruth values. It has been observed that the obtained inference results closely resemble those detailed in the original paper, albeit with potential

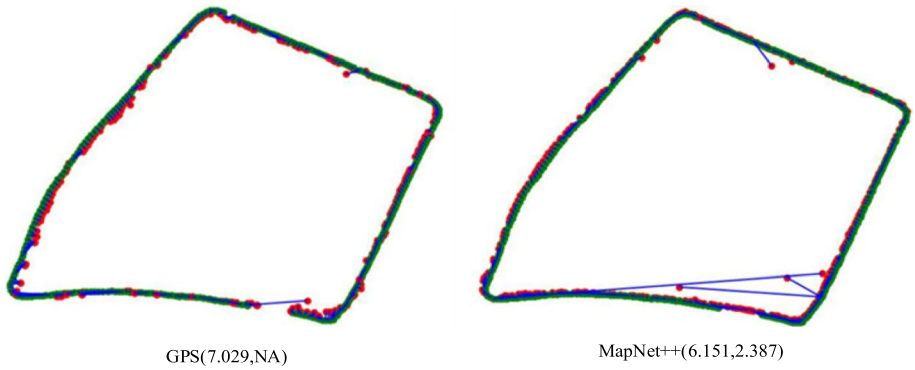

Fig. 6 Comparison of MapNet++inference results and GPS results of loop sequences in RobotCar dataset

influences from modifications in machine specifications. Specifically, the MapNet++inference results exhibit only isolated deviations, exerting no discernible impact on the overall situation and avoiding the generation of cumulative errors. Noteworthy, the DNN-based scene representation model has demonstrated superior performance when compared to GPS.

Inference frequency

In the field of autonomous driving, sensor fusion is an advanced approach for assisting vehicle localization. Our upgraded relocation scheme can serve as both a standalone localization module and as a branch of a multi-sensor scheme. The results depicted in Fig. 7 reveal that our enhanced framework has the capability to generate a larger amount of posture data within the same time period. These data can be integrated with the readings from other sensors to correct the attitude of the vehicle, which is of significant practical value.

Route

If the model needs to output its pose for each frame, our proposed scheme can achieve a longer trajectory within the same time period. As shown in Fig. 8, a comparison of the reasoning track results between the two schemes demonstrates that our upgraded version is capable of covering a greater distance over time, which further underscores the advantages of our proposed framework.

Discussion on data fusion

In the original paper, PGO [39] was utilized by the author to optimize the results generated by MapNet++. This method led to an average translation error that was even less than the average translation error of GPS. Nevertheless, our observation revealed that after averaging the results of GPS and MapNet++, the final result was better than that of MapNet+PGO. Figure 9 displays the overall distribution

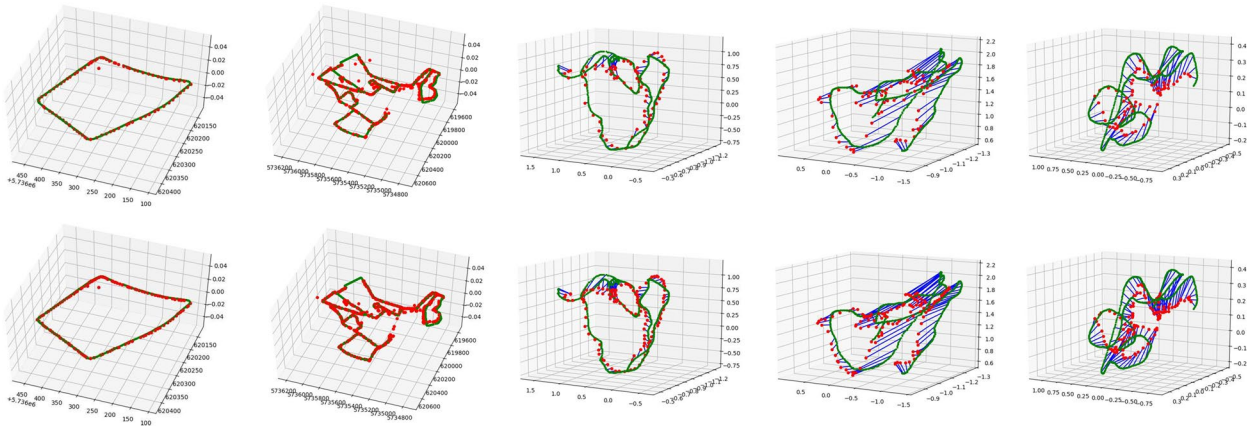

Fig. 7 Row 1: local run frequency results. Row 2: Offloading run frequency results. Green for groundtruth and red for inference result. From left to right are testing sequences:RobotCar-loop,RobotCar-full,7Scenes-Chess-05,7Scenes-Office-06,7Scenes-Heads-01

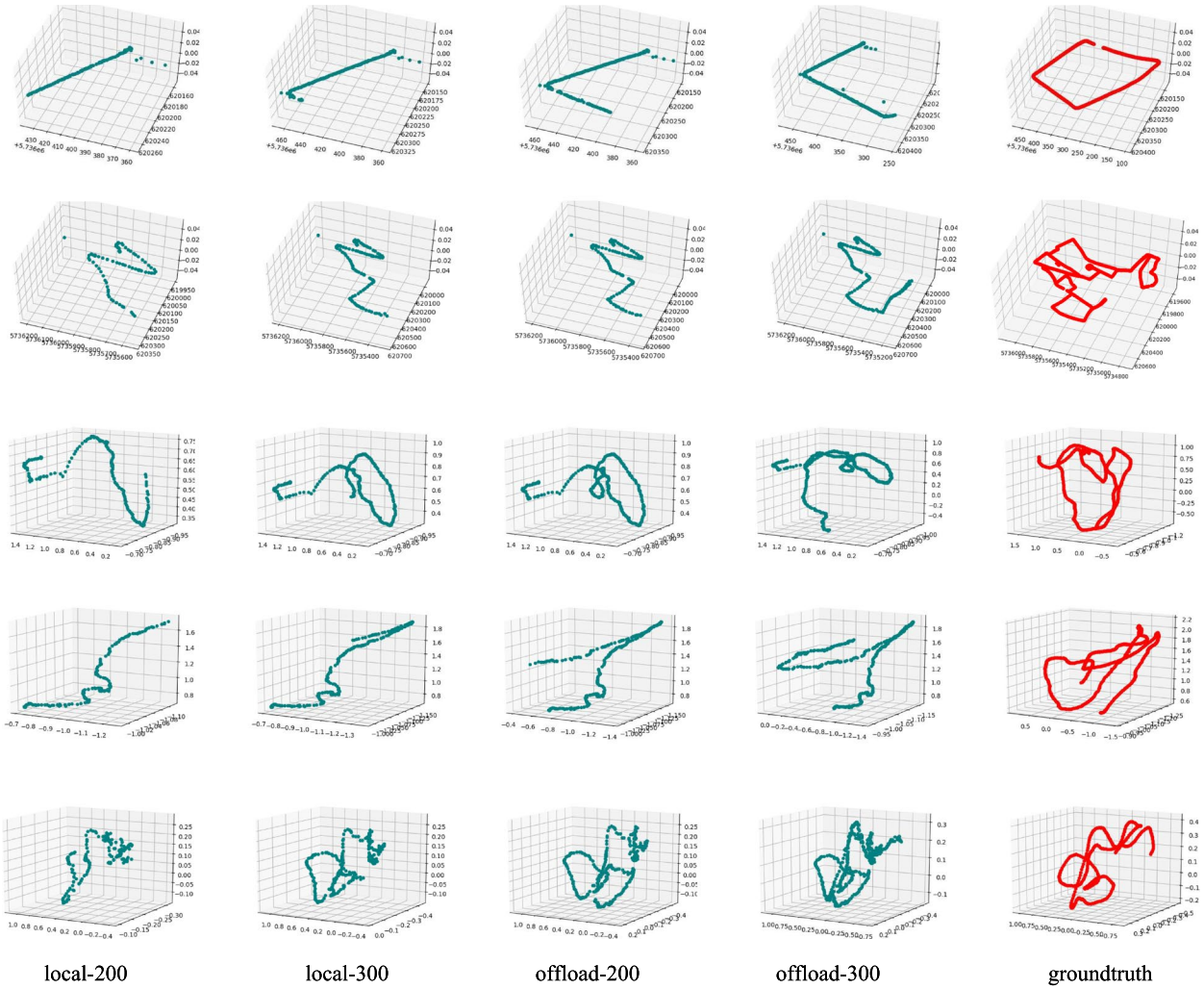

Fig. 8 Local-200: run on the mobile device for 200 s. local-300: Run on the mobile device for 300 s. offload-200: Run using the offloading strategy for 200 s. offload-300: Run using the offloading strategy for 300 s. From top to bottom are testing sequences:RobotCar-loop,RobotCar-full,7Scenes-Chess-05,7Scenes-Office-06,7Scenes-Heads-01. We provide the groundtruth as a comparison in the fifth column

of the error data from both sets. We observed that although MapNet++had some large outliers, the overall variance was smaller than that of the GPS data. Our analysis revealed that the reason why the average result was better is that the average value can reduce noise, particularly when the error of some frames is too large, leading to more stable overall data. This provides a novel approach to DNN-based autonomous vehicle relocation solutions. In practical applications, we can integrate the network output results and GPS data to provide additional auxiliary information for the pose correction of autonomous vehicles.

Conclusion

In this paper, we have presented a novel framework for automatic vehicle relocation based on DNN. Our approach involves offloading the reasoning process to a server, which reduces the computational burden on mobile devices. We have demonstrated the effectiveness of our framework using the MapNet series of relocation schemes. Our experimental results show that our proposed framework can significantly enhance the reasoning efficiency of DNN-based relocation modules in autonomous vehicles. The improved reasoning frequency and route performance highlight

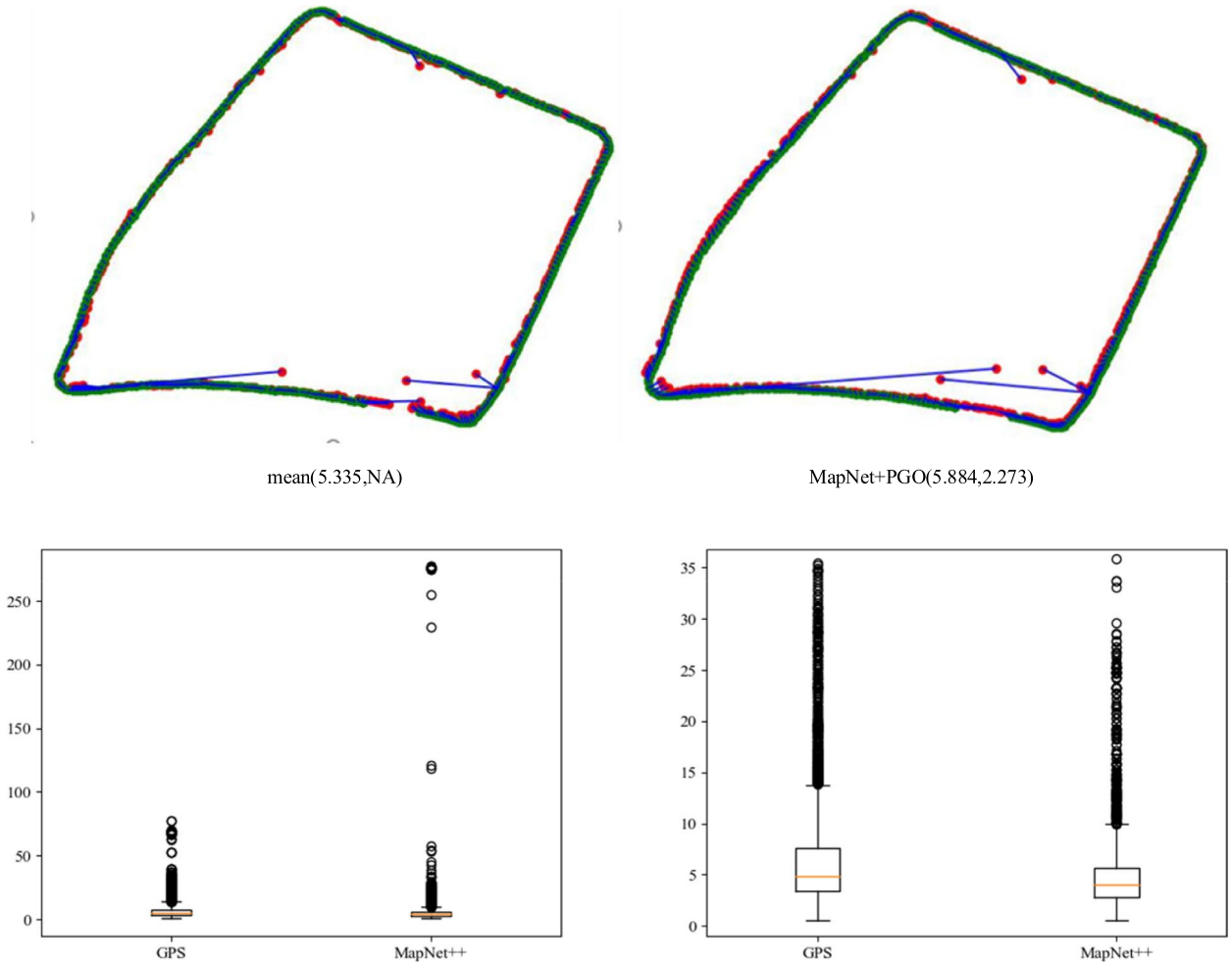

Fig. 9 Top: Comparison of the results obtained by averaging MapNet + + and GPS data with MapNet + PGO. Bottom: Loss distribution of GPS data and MapNet + + results

the practical significance of our approach. This work also have potential values in some other fields as cloud robotics [51–54]. In future research, we will emphasize uncertainty estimation and privacy security in edge cloud collaboration, and will continue to explore communication issues in offloading and offloading of large model architectures. Our goal is to address these challenges through innovative learning methods, contributing significantly to the development of edge cloud collaboration technology.

Authors' contributions

LDB performed the experimental design and validation and wrote the manuscript. ZHN conducted background checks and participated in data analysis. CJR supervised the experimental design and participated in the writing. LB performed data analysis and participated in writing the manuscript. All authors reviewed the paper.

Funding

This work was supported by National Natural Science Foundation of China (NSFC) (Grant No. 62162024,62162022),the Key Research and Development Program of Hainan Province (Grant No.ZDYF2021GXJS003, ZDYF2020040),the Major science and technology project of Hainan Province (Grant No. ZDKJ2020012), Hainan Provincial Natural Science Foundation of China (Grant No. 620MS021,621QN211),Science and Technology Development Center of the Ministry of Education Industry-university-Research Innovation Fund (2021JQR017).

Availability of data and materials

Publicly available datasets were analyzed in this study.

Declarations

Ethics approval and consent to participate

Not applicable.

Competing interests

The authors declare no competing interests.

Received: 10 October 2023 Accepted: 4 January 2024
Published online: 25 January 2024

References

1. Chen C, Yao G, Wang C et al (2022) Enhancing the robustness of object detection via 6G vehicular edge computing. Digital Communications and Networks 8:923–931. https://doi.org/10.1016/j.dcan.2022.10.013
2. Kumar A, Abhishek K, Ghalib MR et al (2022) Intrusion detection and prevention system for an IoT environment. Digital Communications and Networks 8:540–551. https://doi.org/10.1016/j.dcan.2022.05.027
3. Deng D, Li X, Menon V et al (2022) Learning-based joint UAV trajectory and power allocation optimization for secure IoT networks. Digital Communications and Networks 8:415–421. https://doi.org/10.1016/j.dcan.2021.07.007
4. Tan K, Bremner D, Le Kernec J et al (2022) Machine learning in vehicular networking: An overview. Digital Communications and Networks 8:18–24. https://doi.org/10.1016/j.dcan.2021.10.007
5. Wang R, Jiang X, Zhou Y et al (2022) Multi-agent reinforcement learning for edge information sharing in vehicular networks. Digital Communications and Networks 8:267–277. https://doi.org/10.1016/j.dcan.2021.08.006
6. Dai H, Yu J, Li M, et al (2022) Bloom Filter with Noisy Coding Framework for Multi-Set Membership Testing. IEEE Trans Knowl Data Eng 1–14. https://doi.org/10.1109/TKDE.2022.3199646
7. Hu C, Fan W, Zeng E et al (2022) Digital Twin-Assisted Real-Time Traffic Data Prediction Method for 5G-Enabled Internet of Vehicles. IEEE Trans Ind Inf 18:2811–2819. https://doi.org/10.1109/TII.2021.3083596
8. Miao Y, Bai X, Cao Y, et al (2023) A Novel Short-Term Traffic Prediction Model based on SVD and ARIMA with Blockchain in Industrial Internet of Things. IEEE Internet Things J 1–1. https://doi.org/10.1109/JIOT.2023.3283611
9. Wang F, Li G, Wang Y et al (2023) Privacy-Aware Traffic Flow Prediction Based on Multi-Party Sensor Data with Zero Trust in Smart City. ACM Trans Internet Technol 23:1–19. https://doi.org/10.1145/3511904
10. Yang Y, Yang X, Heidari M et al (2023) ASTREAM: Data-Stream-Driven Scalable Anomaly Detection With Accuracy Guarantee in IIoT Environment. IEEE Trans Netw Sci Eng 10:3007–3016. https://doi.org/10.1109/TNSE.2022.3157730
11. Xu X, Tang S, Qi L et al (2023) CNN Partitioning and Offloading for Vehicular Edge Networks in Web3. IEEE Commun Mag 61:36–42. https://doi.org/10.1109/MCOM.002.2200424
12. Xu X, Fang Z, Qi L et al (2021) TripRes: Traffic Flow Prediction Driven Resource Reservation for Multimedia IoV with Edge Computing. ACM Trans Multimedia Comput Commun Appl 17:1–21. https://doi.org/10.1145/3401979
13. Sattler T, Leibe B, Kobbelt L (2017) Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization. IEEE Trans Pattern Anal Mach Intell 39:1744–1756. https://doi.org/10.1109/TPAMI.2016.2611662
14. Sattler T, Leibe B, Kobbelt L (2012) Improving Image-Based Localization by Active Correspondence Search. In: Fitzgibbon A, Lazebnik S, Perona P et al (eds) Computer Vision – ECCV 2012. Springer, Berlin, Heidelberg, pp 752–765
15. Hao Q, Cai R, Li Z, Zhang L et al (2012) 3D visual phrases for landmark recognition. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Providence, RI, pp 3594–3601
16. Tolias G, Sicre R, Jégou H (2016) Particular object retrieval with integral max-pooling of CNN activations
17. Babenko A, Lempitsky V (2015) Aggregating Deep Convolutional Features for Image Retrieval
18. Sivic, Zisserman (2003) Video Google: a text retrieval approach to object matching in videos. In: Proceedings Ninth IEEE International Conference on Computer Vision. pp 1470–1477 vol.2
19. Mur-Artal R, Montiel JMM, Tardós JD (2015) ORB-SLAM: A Versatile and Accurate Monocular SLAM System. IEEE Trans Rob 31:1147–1163. https://doi.org/10.1109/TRO.2015.2463671
20. Jégou H, Douze M, Schmid C, Pérez P (2010) Aggregating local descriptors into a compact image representation. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp 3304–3311
21. Sattler T, Leibe B, Kobbelt L (2011) Fast image-based localization using direct 2D-to-3D matching. In: 2011 International Conference on Computer Vision. pp 667–674
22. Lei J, Wang Z, Wu Y, Fan L (2014) Efficient pose tracking on mobile phones with 3D points grouping. In: 2014 IEEE International Conference on Multimedia and Expo (ICME). pp 1–6
23. Li Y, Snavely N, Huttenlocher DP (2010) Location Recognition Using Prioritized Feature Matching. In: Daniilidis K, Maragos P, Paragios N (eds) Computer Vision – ECCV 2010. Springer, Berlin, Heidelberg, pp 791–804
24. Bansal M, Daniilidis K (2014) Geometric Urban Geo-Localization. pp 3978–3985
25. Kendall A, Grimes M, Cipolla R (2015) PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. pp 2938–2946
26. Szegedy C, Liu W, Jia Y et al (2015) Going Deeper With Convolutions. pp 1–9
27. Kendall A, Cipolla R (2016) Modelling uncertainty in deep learning for camera relocalization. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). pp 4762–4769
28. Kendall A, Cipolla R (2017) Geometric Loss Functions for Camera Pose Regression With Deep Learning. pp 5974–5983
29. Wang S, Kang Q, She R et al (2023) RobustLoc: Robust Camera Pose Regression in Challenging Driving Environments. AAAI 37:6209–6216. https://doi.org/10.1609/aaai.v37i5.25765
30. Bui TB, Tran D-T, Lee J-H (2022) Fast and Lightweight Scene Regressor for Camera Relocalization. https://doi.org/10.48550/ARXIV.2212.01830
31. Valada A, Radwan N, Burgard W (2018) Deep Auxiliary Learning for Visual Localization and Odometry. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp 6939–6946
32. Chen Y, Huang S, Yuan T, et al (2019) Holistic++ Scene Understanding: Single-View 3D Holistic Scene Parsing and Human Pose Estimation With Human-Object Interaction and Physical Commonsense. pp 8648–8657
33. Liang S, Wu H, Zhen L et al (2022) Edge YOLO: Real-Time Intelligent Object Detection System Based on Edge-Cloud Cooperation in Autonomous Vehicles. IEEE Trans Intell Transport Syst 23:25345–25360. https://doi.org/10.1109/TITS.2022.3158253
34. Becker PHE, Arnau JM, Gonzalez A (2020) Demystifying Power and Performance Bottlenecks in Autonomous Driving Systems. In: 2020 IEEE International Symposium on Workload Characterization (IISWC). IEEE, Beijing, pp 205–215
35. Hafiz MI, Abdelawwad M, Muller W et al (2021) SoC approach for low cost and low power consumption based on ARM Cortex M3 according to ISO 26262. In: 2021 18th International Multi-Conference on Systems, Signals & Devices (SSD). IEEE, Monastir, pp 777–783
36. Yu Y, Lee S (2022) Measurements of the Benefits of Edge Computing on Autonomous Driving. In: 2022 13th International Conference on Information and Communication Technology Convergence (ICTC). IEEE, Jeju Island, pp 2155–2159
37. Brahmbhatt S, Gu J, Kim K et al (2018) Geometry-Aware Learning of Maps for Camera Localization. pp 2616–2625
38. He K, Zhang X, Ren S, Sun J (2016) Deep Residual Learning for Image Recognition. pp 770–778
39. Forsgren B, Brink K, Ganesh P, McLain TW (2023) Incremental Cycle Bases for Cycle-Based Pose Graph Optimization. IEEE Robotics and Automation Letters 8:1021–1028. https://doi.org/10.1109/LRA.2023.3236580
40. Huang J, Wan J, Lv B et al (2023) Joint Computation Offloading and Resource Allocation for Edge-Cloud Collaboration in Internet of Vehicles via Deep Reinforcement Learning. IEEE Syst J 17:2500–2511. https://doi.org/10.1109/JSYST.2023.3249217
41. Liu G, Dai F, Xu X et al (2023) An adaptive DNN inference acceleration framework with end–edge–cloud collaborative computing. Future Gener Comput Syst 140:422–435. https://doi.org/10.1016/j.future.2022.10.033
42. Li C, Yang H, Sun Z et al (2023) High-Precision Cluster Federated Learning for Smart Home: An Edge-Cloud Collaboration Approach. IEEE Access 11:102157–102168. https://doi.org/10.1109/ACCESS.2023.3315771
43. Xu J, Cao H, Yang Z, et al (2022) {SwarmMap}: Scaling Up Real-time Collaborative Visual {SLAM} at the Edge. pp 977–993
44. Hsieh C-Y, Ren Y, Chen J-C (2023) Edge-Cloud Offloading: Knapsack Potential Game in 5G Multi-Access Edge Computing. IEEE Trans Wireless Commun 22:7158–7171. https://doi.org/10.1109/TWC.2023.3248270

45. Liu F, Huang J, Wang X (2023) Joint Task Offloading and Resource Allocation for Device-Edge-Cloud Collaboration with Subtask Dependencies. IEEE Trans Cloud Comput 1–13. https://doi.org/10.1109/TCC.2023.3251561
46. Huang P, Zeng L, Chen X et al (2022) Edge Robotics: Edge-Computing-Accelerated Multirobot Simultaneous Localization and Mapping. IEEE Internet Things J 9:14087–14102. https://doi.org/10.1109/JIOT.2022.3146461
47. Yang L, Shen X, Zhong C, Liao Y (2023) On-demand inference acceleration for directed acyclic graph neural networks over edge-cloud collaboration. Journal of Parallel and Distributed Computing 171:79–87. https://doi.org/10.1016/j.jpdc.2022.09.005
48. Zhang H, Lin W, Xie R, et al (2023) An optimal container update method for edge-cloud collaboration. Softw Pract Exp spe.3232. https://doi.org/10.1002/spe.3232
49. Maddern W, Pascoe G, Linegar C, Newman P (2017) 1 year, 1000 km: The Oxford RobotCar dataset. The International Journal of Robotics Research 36:3–15. https://doi.org/10.1177/0278364916679498
50. Shotton J, Glocker B, Zach C et al (2013) Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. pp 2930–2937
51. Zheng Z, Zhou Y, Sun Y et al (2022) Applications of federated learning in smart cities: recent advances, taxonomy, and open challenges. Connect Sci 34:1–28. https://doi.org/10.1080/09540091.2021.1936455
52. Liu B, Wang L, Liu M, Xu C-Z (2020) Federated Imitation Learning: A Novel Framework for Cloud Robotic Systems With Heterogeneous Sensor Data. IEEE Robot Autom Lett 5:3509–3516. https://doi.org/10.1109/LRA.2020.2976321
53. Liu B, Wang L, Liu M (2019) Lifelong Federated Reinforcement Learning: A Learning Architecture for Navigation in Cloud Robotic Systems. IEEE Robot Autom Lett 4:4555–4562. https://doi.org/10.1109/LRA.2019.2931179
54. Liu B, Wang L, Chen X et al (2021) Peer-Assisted Robotic Learning: A Data-Driven Collaborative Learning Approach for Cloud Robotic Systems. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, Xi'an, pp 4062–4070

Publisher's Note