2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC) | 978-1-6654-3161-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CCNC49033.2022.9700709

Offloading Robot Control with 5G

Peter Sossalla*, Justus Rischke*§, Giang T. Nguyen†‡, and Frank H. P. Fitzek*‡
*Deutsche Telekom Chair of Communication Networks, TU Dresden §Dr. Ing. h.c. F. Porsche AG
†Haptic Communication Systems, TU Dresden
‡ Centre for Tactile Internet with Human-in-the-Loop (CeTI)
E-mails: {peter.sossalla|justus.rischke|giang.nguyen|frank.fitzek}@tu-dresden.de

***Abstract*—Simultaneous Localization and Mapping (SLAM), among other critical functions of mobile robots, such as navigation, are computationally expensive. When deployed at the robot, those functions demand high energy consumption and result in shorter operation time. Offloading SLAM to an Edge Cloud (EC) can significantly reduce the robot's computing demand and resources, subsequently reducing energy consumption. We offload intelligence of mobile robot control functionality, i.e., navigation, localization, and control to an EC. The EC processes sensor data and sends the robot the directional velocities. Meanwhile, a 5G wireless connection ensures the necessary low latencies and high throughputs. We demonstrate the feasibility of offloading SLAM and navigation in an EC based on a use case in automotive production. Additionally, we developed a digital twin of the robot and visualized its current sensor data.**

I. Introduction

The concept of Industry 4.0 and digitization has driven automation in factories ever further. The number of mobile robots in factories has increased every single year. Examples are Automated Guided Vehicles (AGVs), a mobile robot type, which can transport components in factories. They can also be integrated with robotic arms to perform several work steps, such as gripping, gluing, or screwing, on other objects. When equipped with an intelligent camera, mobile robots can also perform automated quality control of produced products. For mobile robots to navigate autonomously, reliable localization plays a key role. Among various localization techniques, SLAM which stands for Simultaneous Localization and Mapping (SLAM) has become increasingly popular. In SLAM, a robot creates a map of its surroundings and uses it to localize itself. Due to the constantly sensing and localizing, SLAM and navigation demand very high computing power. Early systems integrate powerful computers directly on the robots.

A typical example of the above setup is a *Kuka KMR iiwa*, which is a sensitive collaborative robot (cobot) for Human-robot collaboration (HRC), consisting of a mobile base (or so-called KMR) and a robotic arm (or so-called LBR). The *KMR* uses two PCs to control the robot fully: one for processing control commands of the *KMR* and the *LBR*, the other, referred to as navigation PC, for localization and navigation. A local area network connects these two computers and other devices, such as the laser scanners. Figure 1 shows the interior of the *KMR iiwa*. The over-provisioned setup with dedicated hardware, on the one hand, ensures that devices always have the computing resource they need. On the other hand, this dedicated setup has several drawbacks. First, it requires more space for any added device and the spacing between them for cooling. Second, the setup demands more energy since all added computing devices need electric power to operate. This results in a shorter operating time for the same battery capacity. In addition, the over-provisioned hardware leads to higher costs. Last but not least, a large number of components in a device also increases the probability of a failure.

A logical step is to offload the computing resources from the robot to a nearby cloud computing service. In this case, the robots streams sensor data via a wireless connection to a computing cloud for processing. Offloading could thereby eliminate the need for a navigation PC. Hence, the offloading approach can mitigate the previously described disadvantages of having more computing hardware on the device.

The drawback of the traditional cloud service is latency since cloud computing data centers are typically hundreds of kilometers away from a particular factory. The typical latency, in this case, is in the order of 10s of milliseconds. For autonomous driving in a factory, this latency is way too high since the control of a robot needs to react quickly to changes. In the development of 5G, uRLLC [1] should provide the possibility to realize latency-critical and reliable communication. With the introduction of 5G, Edge Cloud (EC) has become more attractive. Here, the computing resource is placed at end-users proximity, i.e., at the network's edge, thus reducing latency significantly.

This paper presents a framework to control a mobile industrial robot from an EC of a 5G network. We offload the localization and navigation functionality of a mobile industrial robot to an EC. Our proof-of-concept with the industrial robot *Kuka KMR iiwa* operates in a factory of a car maker. The robot performs a typical use case in the automotive industry, which is the visual quality control of car components.

II. System Design

A. Localization & Navigation

In order to control a mobile robot from an edge cloud, navigation and localization are two vital building blocks. For localization, there are several approaches. A classical approach is localization via wheel speeds, also called *odometry*. However, different floor coverings and inaccurate measurements can cause the estimation to drift from the actual position. To compensate for this, markers, for example, in the form of QR codes or strips, are installed in factories. However, these solutions require a lot of planning and manual maintenance.

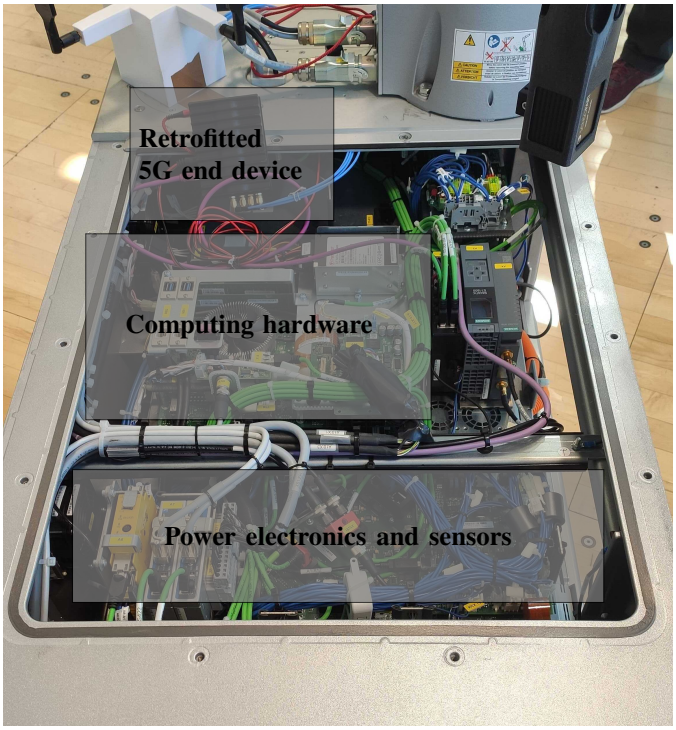

Fig. 1. Insight into the electronics of a *Kuka KMR iiwa* and its components.

Reconstruction of the factory environment therefore always requires replanning of the routes for the mobile robots.

An alternative and increasingly popular approach for localization is SLAM. This method uses spatial data, such as lidar sensors or optical cameras, to create maps and use them for orientation. SLAM operates as follows: First, features are detected in the images and then stored in frames. Second, the SLAM system compares the progression of the position of these features in different time steps and then estimates the change in position and orientation of the robot. In a previously unknown environment, these feature frames can now be assembled to create a map. At the same time, the SLAM algorithm checks whether the environment is known. In that case, the current position, as well as the previously estimated positions, are recalculated. The feature detection, map creation as well as comparison with previously determined data are computationally expensive processes. Due to the limited resources of a mobile robot in terms of energy consumption and computing resources, offloading such a computationally expensive task is a reasonable option.

We use the environment of the Robotic Operation System (ROS) [2] because ROS offers a wide range of frameworks to implement the navigation, the SLAM procedure, and the map generation algorithms. We decided to use the current variant ROS2 because its communication subsystem extends the Data Distribution Services [3], a standardized publish-subscriber architecture. We also use the SLAM toolbox [4], extending graph-based SLAM algorithms.

The second vital building block for robot control is navigation, which depends directly on the localization and the created map, similarly to route planning. For that, computations are also necessary, and hence offloading the calculations to the same instance as the SLAM process is desirable. The navigation is implemented with *Navigation2* [5] of ROS2. *Navigation2* is a state-of-the-art implementation for path planning of mobile robots. Figure 2 visualizes how navigation and localization work in our implementation.

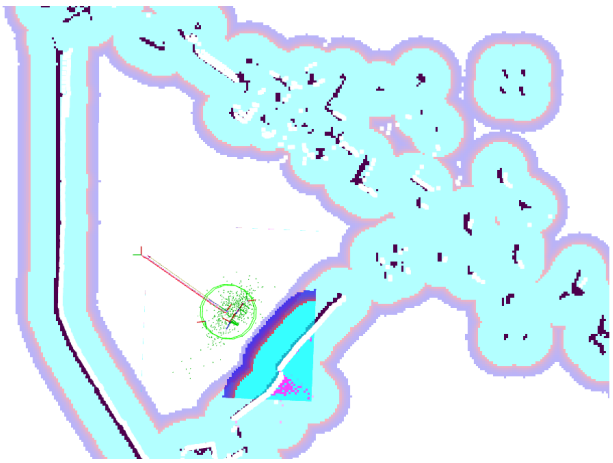

Fig. 2. Navigation and Localization via SLAM. The white line shows the current sensor data. The black line represents the previously created map.

The black line in Figure 2 shows the previously created map, which was recorded with the well-known mapping framework Cartographer [6]. The white line represents the current sensor data sent to the EC through the 5G network. The current position can be estimated by matching the created map (black line) with the sensor data (white line). The navigation is visualized by the thin red line, which depicts the planned route. In addition, the visualization shows the different coordinate systems of the arm, base, and other robot elements.

B. Digital twin

As can now be seen from section II-A, the robot's live sensor data is available at the EC to create a digital twin [7]. The system keeps updating the digital twin by continuous streams of the localization and the coordinate data from sensors. Digital twins have a great potential application in monitoring production lines or pre-simulating processes. For that purpose, the system creates a 3D model of the *KMR iiwa* consisting of its driving platform together with the robot arm. In this work, we used the Unified Robot Description Format (URDF) to define the robot. URDF is compatible with ROS, and thus the digital twin can be visualized with *rviz* as show in Figure 3 on the right side. In addition to the digital twin, the sensor data of the laser scanner relative to the mobile robot is displayed as white dots.

To summarize, offloading the computation of localization and navigation saves resources of the mobile robot and allows to track its behavior as a digital twin simultaneously.

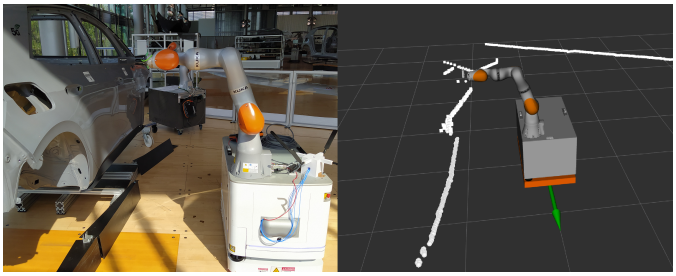

Fig. 3. Demonstration of the digital twin: The left side depicts the footage of the real world. The right side shows the digital twin based on the feedback sent by the robot.

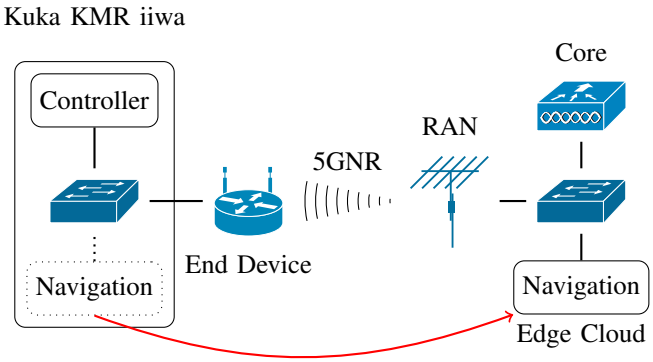

Fig. 4. Physical system setup consisting of a *Kuka KMR iiwa*, 5G network, and an Edge Cloud.

C. 5G Campus Networks

A crucial requirement for offloading computationally intensive tasks is reliable communication. Today, Wifi is typically used in industrial environments to connect mobile robots with the production infrastructure. Since everything in a modern factory, from robots to tools, is today already networked via the free ISM bands, Wifi can not be used for a communication requiring high reliability. Therefore, the mobile robots mainly work autonomously, i.e., applications such as navigation need to be implemented locally on the robot. Wifi is then only used for non-time-critical commands.

5G campus networks, however, operate on other frequency ranges designated explicitly for them. In addition, 5G allows for prioritizing different end devices, e.g., a mobile robot control over a non-critical tool. 5G campus networks aim at complementing existing Wifi-based networks to enable new use cases that demand higher communication reliability. A 5G campus network is typically a Non-Public Network (NPN), i.e., the network is not provided by a public telecom company, but rather by the using industrial company itself. For that, the Core for 5G locates itself at a dedicated data center while the local User Plane Functions (UPFs) locates itself at the factory sites. Especially the local UPFs allow placing applications close to the use case, e.g., a mobile robot in a factory. Hence, the application can run directly on a EC connected to the UPF. In this way, the propagation delay is kept low when implementing fast control loop cycles. To summarize, 5G campus networks allow connecting mobile robots with low delay and high reliability. As a result, it is also possible to run critical tasks that require a lot of computing power, such as SLAM for navigation, in an edge cloud connected via 5G.

Figure 4 depicts the system setup of our 5G campus network consisting of three components: (i) the mobile robot *Kuka KMR iiwa* retrofitted with a 5G end device based on a *Telit FN980m*, (ii) a 5G Core and RAN as described in [8], and (iii) the robot control running on a EC connected to the 5G Core.

The *Kuka KMR iiwa* itself consists of multiple components, including computing hardware for navigation and control. Since the software by Kuka expects the navigation to run locally, the end device poses as a navigation PC. However, the end device forwards all communication from the control PC to the edge cloud via 5G. By using Network Address Translation (NAT) with the help of *iptables*, the local *Kuka* network can be bridged into the 5G network. In this way, our solution requires no changes to the existing software or configuration.

III. A Working Prototype and Testbed

A. Demonstrating the proof-of-concept

We implemented a use case in a real manufacturing environment to demonstrate the feasibility of combining offloading and virtualization. For this purpose, we deployed a 5G Standalone network as described in section II-C in the *Transparent Factory* (*Gläserne Manufaktur*) in Dresden, which is an automobile production facility of *Volkswagen*. Afterwards, we measured the Round-Trip Time (RTT) and throughput. Over 1000 measurements with *ping*, an average RTT of 8.2 ms was measured with a standard deviation of 0.8 ms. We measured the throughput with *iperf* using TCP, for download throughput reached 912.7 Mbit/s, while the upload throughput reached 115.6 Mbit/s. The considerably high throughput and low RTTs indicate the possibility of transferring the sensor data and reacting sufficiently fast to changes. An Intel NUC with an Intel i5-6260U CPU @ 1.8 GHz with 4GB of RAM served as an Edge Cloud. The laser scanners of the *KMR iiwa* measured the distances and then uploaded them to the EC via the 5G connection. The localization and navigation functions running on the EC calculated the linear and angular velocities for the moving base and the desired joint angles for the robot arm. Afterwards, the results of the calculations were sent back to the *KMR iiwa* controller as instructions. Figure 5 shows the logical information exchange. In our offloading deployment, we achieved a closed control loop and thus eliminated the need for an additional navigation PC.

As an use case, we implemented a quality control analysis with the mobile robot. The aim is to inspect parts in the automotive industry, usually done by human workers. The robot places itself autonomously in front of a car body and then extends its mounted robot arm and points its camera at the corresponding component. For this specific application, we used the Sensopart VISOR V20 Object Advanced camera.

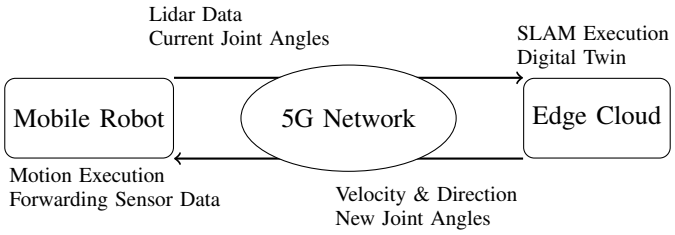

Fig. 5. Logical setup of the system consisting of a mobile robot connected to an Edge Cloud (EC) via 5G.

The camera sent images to an Edge Cloud for analysis by appropriate computer vision algorithms. Figure 3 shows the moment when the robot performs the quality control together with the corresponding digital twin. After inspection, the arm is retracted again and returns to its original position. On the one hand, the successful realization shows the feasibility of introducing 5G into the industry and, on the other hand, the virtualization of robot control in an actual use case.

B. Planned enhancements

We plan to extend our current work with the following steps. First, we plan to use an RGB-D camera on the *KMR iiwa*, since this type of camera has not only an optical but also a depth camera. Then, the Edge Cloud collects the uploaded images to apply object detection algorithms to detect the car's body and its position. The robot can autonomously place itself in front of the car body and align the arm accordingly with the determined position, allowing for integrating this use case into existing workflows in a car factory.

IV. Related Work

Offloading the navigation of mobile robots in the cloud is proposed by Okumuş *et al.* in [9]. Only navigation commands in a grid-based environment were sent in their work, and no mapping or localization procedures were offloaded to the cloud. In [10], Fan *et al.* propose a framework to control a mobile robot and a robot arm from an edge cloud. Here, the advantages of edge cloud computing were mainly investigated in terms of image processing for grasping objects by the robotic arm. The localization is still based on the odometry of the wheels and is not performed using SLAM methods. In [11], Mokaram *et al.* present a ROS interface for the *Kuka iiwa LBR* robot arm. The work of Heggem *et al.* [12] presents the control of a *KMR iiwa* with ROS. [11] and [12] did not offload the robot's functions into an edge cloud with 5G but form a solid basis for interacting with robots from *Kuka*. In summary, no offloading of navigation and localization of mobile robots to an edge cloud via 5G has been conducted in state-of-the-art work.

V. Conclusion

In this paper, we presented our work in progress to offload the localization, navigation, and control of a mobile robot via 5G. The intelligence is shifted to the network by edge computing, and the robot only receives basic movement instructions in terms of linear and angular velocities. Offloading removes the need to use an additional navigation PC. We showed that 5G, characterized by low latency and high throughput, can support the offloading of computationally intensive tasks. Offloading has the advantage of longer operation times, lower maintenance efforts, and less space usage inside the robot. The offloading via 5G was carried out as a proof-of-concept with a mobile industrial robot and successfully tested in a typical automotive production use case.

Acknowledgment

This work has been supported by the *German Federal Ministry of Education* and Research (BMBF) as part of the project *5G Insel* under the grant 16KIS0956K and by the German Research Foundation (DFG) as part of Germany's Excellence Strategy EXC 2050/1 – Project ID390696704 – Cluster of Excellence Centre for Tactile Internet with Human-in-the-Loop (CeTI) of Technical University of Dresden. We would also like to thank the teams from Volkswagen, Audi AG and Porsche for supporting our work.

References

[1] Z. Li, M. A. Uusitalo, H. Shariatmadari, and B. Singh, "5g urllc: Design challenges and system concepts," in *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*. IEEE, 2018, pp. 1–6.

[2] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*. Kobe, Japan, 2009.

[3] G. Pardo-Castellote, "Omg data-distribution service: architectural overview," in *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings.*, 2003, pp. 200–206.

[4] S. Macenski and I. Jambrecic, "Slam toolbox: Slam for the dynamic world," *Journal of Open Source Software*, vol. 6, no. 61, p. 2783, 2021. [Online]. Available: https://doi.org/10.21105/joss.02783

[5] S. Macenski, F. Martín, R. White, and J. G. Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 2718–2725.

[6] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1271–1278.

[7] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2019.

[8] J. Rischke, P. Sossalla, S. Itting, F. H. P. Fitzek, and M. Reisslein, "5g campus networks: A first measurement study," *IEEE Access*, vol. 9, pp. 121 786–121 803, 2021.

[9] F. Okumuş and A. F. Kocamaz, "Cloud based indoor navigation for ros-enabled automated guided vehicles," in *2019 International Artificial Intelligence and Data Processing Symposium (IDAP)*, 2019, pp. 1–4.

[10] Z. Fan, W. Chen, G. Zhu, Y. You, F. Deng, Y. Hou, W. Liang, R. Fu, J. Xin, J. Chen, and H. Wang, "Collaborative robot transport system based on edge computing," in *IEEE Annual International Conference on Technology in Automation, Control, and Intelligent Systems*, 2019.

[11] S. Mokaram, J. M. Aitken, U. Martinez-Hernandez, I. Eimontaite, D. Cameron, J. Rolph, I. Gwilt, O. McAree, and J. Law, "A ros-integrated api for the kuka lbr iiwa collaborative robot," *IFAC-PapersOnLine*, 2017.

[12] C. Heggem, N. M. Wahl, and L. Tingelstad, "Configuration and control of kmr iiwa mobile robots using ros2," in *International Symposium on Small-scale Intelligent Manufacturing Systems (SIMS)*, 2020.