2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Integration of an Automated Valet Parking Service into an Internet of Things Platform

Louis Touko Tcheumadjeu[1], Franz Andert[1], Qinrui Tang[1], Alexander Sohr[1], Robert Kaul[1], Jörg Belz[1], Philipp Lutz[2], Moritz Maier[2], Marcus G. Müller[2], Wolfgang Stürzl[2]

Abstract— **This paper presents an architecture for Automated Valet Parking (AVP) connected to cloud-based IoT services and mobile user interfaces. The goal is to enable AVP services for automatic vehicles. From the user perspective, automatic car drop-off and pick-up are activated via smart phone application, and the user will be able to continuously monitor the vehicle status together with additional services as cleaning or recharge during the parking phase. Further, the IoT platform allows the integration of live services that will interact with automatic driving and parking. As an example, the presented AVP setup includes the operation of service drones to automatically guide a vehicle to the best parking spot. The demonstration in this paper comprises a parking car and a micro aerial vehicle (MAV) connected in real-time through the IoT platform as well as the smart phone application where the car is controlled and supervised.**

I. INTRODUCTION

Today one of the main research efforts in the automotive field is concerned with the design and development of automated valet parking (AVP) applications [1]. However, until now the integration of such telematic services into the Internet-of-Things (IoT) platform, which is designed to support added-value services in smart cities [2], is not yet realized. To fullfil this gap and benefit from the value chains provided by the IoT, the large scale project AUTOPILOT [3] aims at the development and integration of mobility applications and services into IoT-architectures and platforms so that the service enables safer highly automated driving and parking. Beside urban driving, highway pilot and platooning, AVP is one of the automotive applications in the scope of this project that is currently developed and integrated into the IoT-platform. The AVP service is applied in the scenarios of parking and collecting automated vehicles. The user drives to the drop-off position and leaves the vehicle. After the user requests the available parking spots in the parking place with her/his smart phone (see Fig. 1), the vehicle can automatically drive into a parking spot. In the parking place, the vehicle can charge and re-park if it is necessary. When collecting the vehicle, the user goes to the pick-up position and sends the request with the smart phone, and then the vehicle automatically drives to the pick-up position. Fig. 2 illustrates the overall scenario.

*This work has been supported by the European Commission within the European Union's Horizon 2020 research and innovation programme funding, Project AUTOPILOT under Grant Agreement No.731993

[1]Institute of Transportation Systems, German Aerospace Center (DLR), 12489 Berlin, Germany Louis.ToukoTcheumadjeu@dlr.de
[2]Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Wessling, Germany

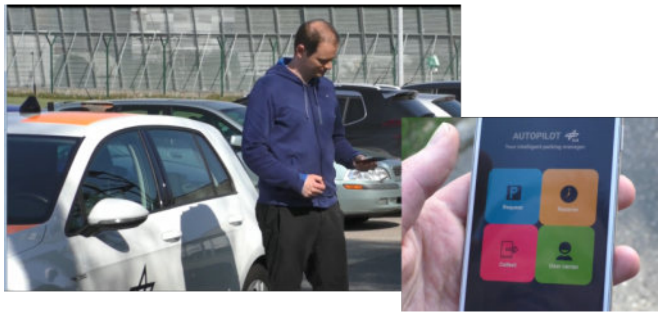


Fig. 1: Mobile-controlled automated valet parking

The AVP service offers parking space reservations, advices automated vehicles where to park and return to their users upon request using a smart phone. The system consists of the parking management service, the routing service and the user management service by integrating data using the IoT-platform. Furthermore, the parking space infrastructures are also part of the AVP system and consist of Micro Aerial vehicles (MAVs) and cameras used for the detection of parking spot occupancy as well as obstacles in the parking zone. The parking management system can manage the operations of parking, maneuvering of the car in the parking area, retrieving and possibly other additional services, e.g. refueling, recharging or cleaning.

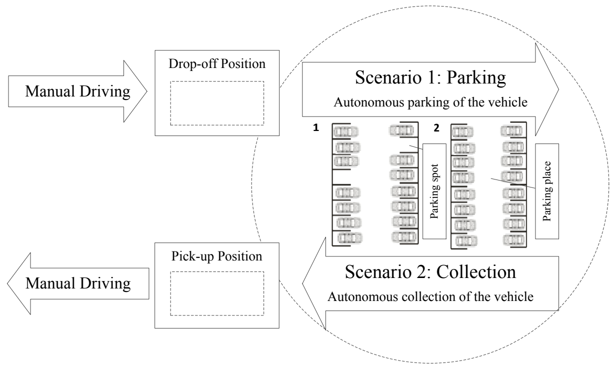


Fig. 2: Scenarios of automated valet parking

The paper presents the technical aspects of the software prototypic AVP system designed and developed by the German Aerospace Center (DLR). The focus is on the integration of the AVP system into to the IoT platform, which combines several sources of information. All IoT devices (AVP-vehicle, smart phone, MAV and camera) and the parking management application can publish their data to the IoT platform and subscribe necessary information, such that the AVP-vehicle can perform its parking task in the most efficient way. The AVP use case demonstration will take place in the AUTOPILOT pilot site [3].The first results of the evaluation and the investigation of the influence of IoT on the AVP-efficiency will be also presented in this paper.

II. INTEGRATION INTO IOT PLATFORM

A. System Architecture

The overall AVP system includes a backend system, a IoT platform and some IoT devices such as smart phone, IoT vehicle, parking spot, MAV and camera (See Figs. 3 and 4). The operations on the IoT platform can be publish/subscribe events, and publish/subscribe commands. With the IoT AVP applications in the backend system, the backend system can handle all operations about IoT platform. However, the IoT devices can only publish events and subscribe commands. With the IoT platform, the backend system and the IoT devices can efficiently exchange information by using the protocol MQTT or HTTPS.

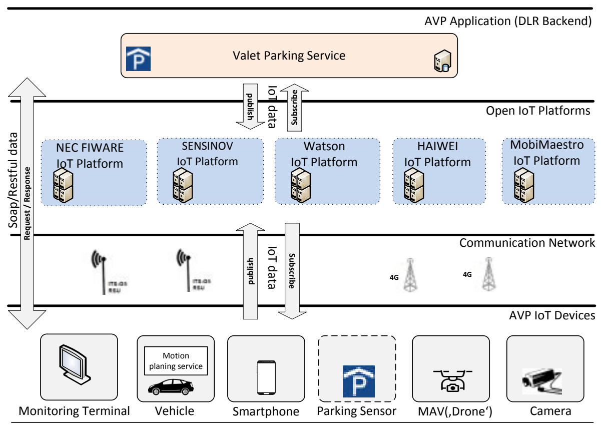


Fig. 3: System architecture of the integration of AVP into IoT platform

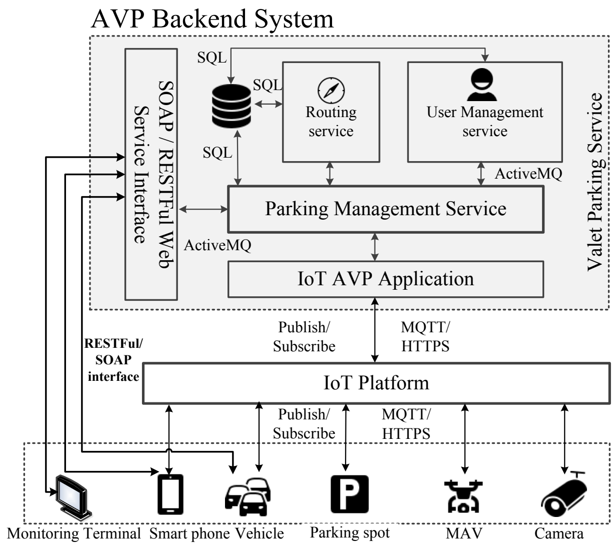


Fig. 4: System components and interfaces of the AVP system

B. AVP Backend System

The main function of the AVP backend system is parking management service. As the routes from the drop-off position to a parking spot and the routes from a parking spot to the pick-up position need to be handled, a routing service is also included. Meanwhile, the user information needs to be managed.

The backend system has two interfaces: the IoT AVP Application and the web service interface. The IoT AVP application handles the operations on the IoT platform because IoT devices cannot directly send messages to each other; instead, IoT devices publish events to the IoT platform and the IoT AVP application handle these events and publish them to relevant IoT devices in the form of commands. When publishing commands, the IoT AVP application has to figure out which device it publishes to. The web service interface is responsible for rest of necessary operations.

C. IoT Platform

An IoT platform is an broker and device manager, which can be connected to a wide variety of devices, gateways and applications exists on the market. Today many IoT platforms [4, 5, 6, 7, 8] provide about the same core functionalities (API Gateway, cloud service, device integration, device management, data management, data analysis, data publication, data subscription, service operation and management) and key concepts customized for different requirements. However, they provide different communication protocols like MQTT, TLS, REST API or CoAPS. The valet parking service designed and developed by DLR has been integrated in the first step into the Watson IoT platform (see Fig. 5). Furthermore the AVP can be easily extended to support other existing IoT-platforms (e.g. oneM2M, FIWARE, HUAWEI OceanConnect platform). Using IoT can enhance the AVP use case because the connected vehicle to the IoT platform can use information published by other IoT devices and sensors to enhance its environment model and its autonomous driving functions.

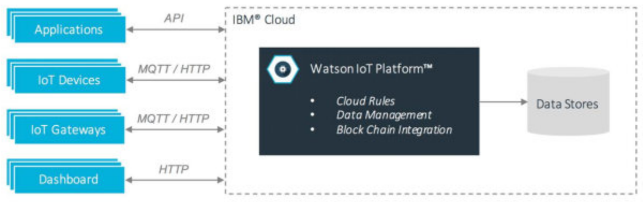


Fig. 5: Architecture of the Watson IoT Platform[4]

D. IoT Device: IoT Vehicle

Additional to the AVP functions, an IoT vehicle can also publish its events and subscribe the commands from the IoT platform. Fig. 6 shows the integration of IoT vehicle into the IoT platform. The IoT vehicle publishes an event, e.g., its current status, to the IoT platform. The IoT AVP application subscribes this event and this event information can be published to other devices. Meanwhile, the vehicle can also subscribe commands if the IoT AVP application publishes any commands to it. The events related to the IoT vehicles are the vehicle AVP status and vehicle AVP charging status. As the vehicle is equipped with sensors, it can also detect the occupancy of parking spots. It is possible that the parking spots are occupied by manual vehicles. Thus, the detection of parking spot occupancy is necessarily real-time.

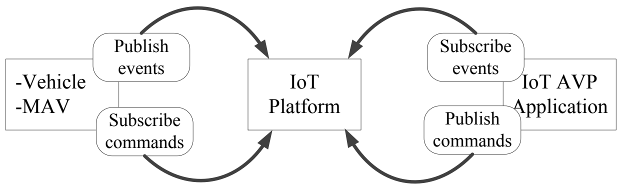


Fig. 6: Integration of vehicle and MAV into IoT platform

E. IoT Device: Smart phone

For user interaction with the AVP backend, a smart phone application was implemented. Similar to the AVP backend system, the App in the smart phone consists of a web service module and a IoT module, but the App also has its GUI. The web service module is responsible for requesting information about user management service, parking management service and routes in the parking place. The IoT service mainly focuses on listening to the current status of the IoT vehicle. In this manner, the relevant status of the IoT vehicle can be traced in real time.

Different from the IoT vehicle, the smart phone need not publish its events to the IoT platform, but only subscribe commands (See Fig. 7). For example, the current status of the IoT vehicles is published as a command from the IoT AVP application, and with the subscription, the smart phone can display the current status of the vehicle and notify the user. The commands which the smart phone can subscribe are the vehicle AVP status and vehicle AVP charging status.

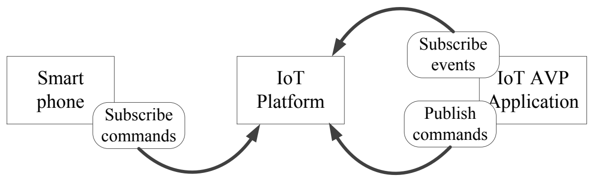


Fig. 7: Integration of smart phone into IoT platform

F. IoT Device: MAV equipped with cameras

In the presented automated valet parking use case, driverless cars should ideally avoid aimless search, thus knowledge of available parking spots is of major importance. Since Micro Aerial Vehicles (MAVs) equipped with cameras are able to collect data about parking spot occupancy autonomously, reliably, and fast, they are perfectly suitable, but not limited to parking places without additional infrastructure, like stationary cameras or ultrasonic sensors. An MAV can update the available parking spots on request or as soon as a driverless car leaves the parking area. In addition to that, obstacles on the route between drop-off point and the aimed parking spot can be detected using the cameras and their presents and exact location can be reported. Compared to stationary cameras, MAVs are more flexible and can help to find optimal routes for the autonomous vehicle in real-time.

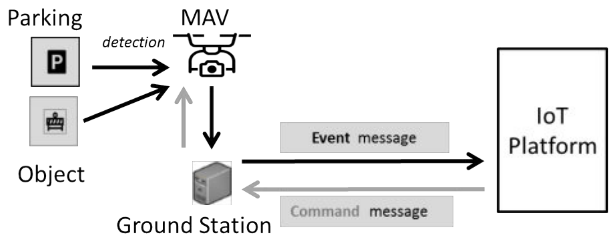


Fig. 8: Communication between MAV and IoT-platform.

The combination of MAV and ground station (See Fig. 8) is considered as one IoT device, because this solution provides extended processing power as well as reliable communication infrastructure and data storage. Therefore, once the MAV has collected images via its on-board camera, it can directly process or stream them to the ground station for further processing. Similar to the IoT vehicles, MAVs publish events and subscribe to commands. If the AVP systems requests the current situation of the parking place, it publishes a command to which the MAV subscribes to. The MAV will then execute the command autonomously, publish its current state and, upon completion of the task, the result of the parking spot detection. These events are in turn subscribed by the backend AVP system. The complete data model is presented in detail in the upcoming section.

III. AVP IoT DATA MODEL

The AVP IoT data model covers the IoT event and command messages exchanged between IoT devices like vehicle, MAV and IoT platform. For the AVP use case the three following IoT-devices are involved (Vehicle, MAV and camera). The AVP data model of these three IoT devices is specified in this section.

A. AVP IoT event data model

The IoT event message consists of status information (e.g. AVP vehicle status, current vehicle charging status, current vehicle position, parking spot occupancy) published by vehicle, MAV or camera to the IoT platform. The data model specification of the event message for the vehicle, MAV and camera is described in the following section.

1) AVP IoT event message for vehicle: The standardized SENSORIS [9], [10] vehicle data model is adopted to model the IoT raw data from the vehicle in AUTOPILOT project. To support the AVP use cases the valet parking specific status information (AVP status message) is identified, specified and integrated into the existing standized vehicle SENSORIS data model. For the implementation of AVP use case the following three event message *AVPStatus*, *ParkingSpotDetection* and *VehiclePositionEstimation* are relevant.

- The *AVPStatus* message: that contains valet parking specific status information related to the IoT device.

This AVP status message is published into the IoT Platform by the corresponding IoT-device like AVP-vehicle during the valet parking driving phases. The following types of status message are required by the AVP used. All this message can be:
  - *AVPVehicleStatus*: represents the AVP status of the vehicle.
  - *AVPVehicleChargingStatus*: represents the charging status of the vehicle.
- *ParkingSpotDetection*: contains the information about the parking spot detection and the occupancy by the vehicle during the manual and automated driving on the road (onstreet parking spot) or in parking zone or parking garage (offstreet parking spot). The *parkingSpotDetection* model is not only specific to the valet parking use case.
- *VehiclePositionEstimation*: the AD-vehicle need to send continuously his current position to the IoT platform, so that the driver can be able during the valet parking to visualized this information on the AVP mobile app. The SENSORIS vehicle data model contains the *PositionEstimation* data model. That need to be used by the AVP vehicle.

2) AVP IoT event message for MAV: Together with a ground station computer the MAV acts as an IoT device and publishes events such as telemetry messages and it can receive AVP command messages from the AVP backend. In the AVP use case it is used for following purposes:

- Parking spot and occupancy detection: fly to the corresponding AVP parking zone, detect free parking spots and send occupancy information back to the IoT platform.
- Object detection: while flying to the corresponding AVP parking zone it detects obstacles on a given route. (e.g. from drop-off to parking spot).

The MAV data model is based on the SENSORIS [9], [10] vehicle data model and contains additional MAV information. In the AVP use case the following event message are relevant:

- *DroneStatus*: Contains all relevant information for AVP backend decision making such as whether the MAV is currently ready to take AVP commands.
- *ParkingSpotDetection*: Provides all details to detected parking spots, such as geometry and occupancy.
- *ObjectDetection*: Lists all detected objects on the way between drop-off and parking spot such as object type, location and size.
- *ADServiceAndSensorState*: Provides information of MAV specifc capabilities to the AVP backend, such as object and parkingspot recognition.

3) AVP IoT event message for camera: The camera as IoT device is used by the AVP use case to provide the Parking spot and occupancy detection information. For this purpose the camera are installed to the corresponding AVP parking zone with the function to detect the parking spot and publish the occupancy information into the IoT platform. Camera data model based on the SENSORIS[9], [10] vehicle data model. Because the camera IoT data model does not exist yet, a new camera's IoT data model need to be specified. That is based on the SENSORIS vehicle data model and contains AVP specific information.

B. AVP IoT command data model

The AVP command message contains the driver's valet parking instruction required by the IoT-device like vehicle or MAV to execute the valet parking process. To be able to receive the AVP command the vehicle or MAV as IoT-device need to subscribe to such message by the IoT platform. According to the type of IoT device used the AVP command is differentiated in two categories: *VehicleAVPCommand* and *DroneAVPCommand*.

IV. SYSTEM IMPLEMENTATION

A. IoT Device: Vehicle

Test car is a Volkswagen e-Golf operated by DLR's Institute of Transportation Systems. As other vehicles from DLR's FASCar fleet, it includes some modifications for experimental and user-defined vehicle automation and control. The modular and exchangeable equipment comprises e.g. multiple LIDARs and other environment sensors (cameras, RADAR), GPS/INS with RTK capability, on-board computers for data logging, real-time data processing and vehicle control as well as various Car-to-X interfaces as WiFi and 3G/4G mobile network. In the presented tests, data recording and IoT-communication is supervised in the vehicle while driving. Further information about DLR's test car fleet is presented in [11].

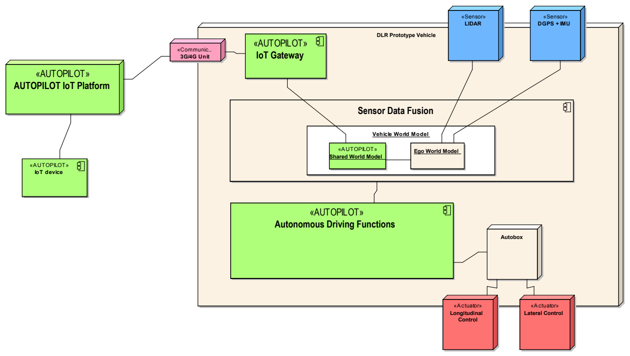


Fig. 9: DLR in-Vehicle Architecture

The in-vehicle architecture (see Fig. 9) comprises two separated computers for the different vehicle functions. First, one computer controls all autonomous driving functions with access to all low-level systems and car control actuators. All additional and less-critical functions are integrated on a sensor data fusion computer. It combines on-board sensors used e.g. for improved and experimental navigation and obstacle avoidance functions with the IoT functionality provided by the IoT platform through wireless data communication. The IoT platform can be considered as cloud service where multiple vehicles and the user front-ends (e.g. the smart phone application) are connected to.

B. IoT Device: MAV

The MAV used for the experiments is a custom built hexacopter with 2.4 kg take-off weight. It was developed from the ground up at the DLR's Institute of Robotics and Mechatronics. It is equipped with multiple cameras providing wide-angle view of the environment, FPGA-based stereo depth estimation and 3D environmental reconstruction. Position estimation on the MAV is performed by an visual-inertial navigation system [12], which consists of two pairs of stereo cameras, one looking down and the other looking up, and an inertial measurement unit (IMU). A sensor fusion algorithm is combining the data of all sensors [13, 14]. This navigation solution enables the MAV to navigate in GPS-denied environments such as indoor parking-structures or underground car parks.

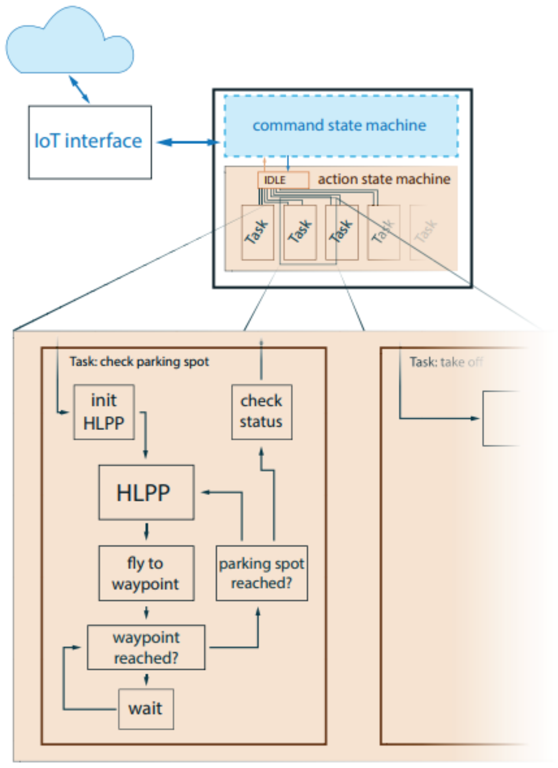


Fig. 10: MAV state machine overview (part).

As mentioned before, the communication is realized via the ground station, on which a custom IoT interface as shown in Fig. 10 is implemented. For the communication between IoT interface and MAV, a custom middleware based on UDP and shared memory is used. Furthermore, all tasks of the MAV are implemented as state machines in our tool RAFCON [15]. All state machines run locally on the MAV for autonomous operation. The top-level state machine consists of two parallel running sub state machines, named command state machine and action state machine as shown in Fig. 10. This concept is not just restricted for MAVs, but can also be used for other robotic agents.

The command state machine is responsible for the coordination of higher level tasks and breaks them down into more simple tasks, which can be executed one by one. It also provides the communication interface for the overall state machine. Via this interface commands can be requested and information can be exchanged with the outside world. The IoT interface for instance is connected to the state machine over this gateway.

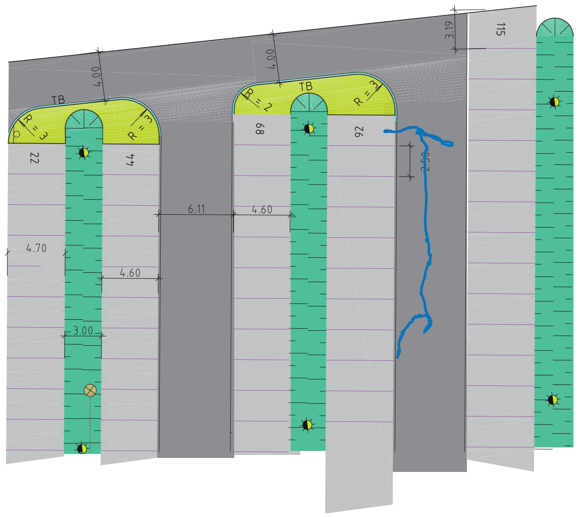


Fig. 11: MAV path flown on DLR test site

The action state machine is the part of the overall state machine, which is executing the actual tasks the MAV is supposed to perform. The action state machine is built up by tasks and an idle state. The MAV awaits commands in the idle state, performs the commanded task, and then always returns to the idle state. Tasks are sequences of different actions. A task might be to take-off, fly to a specific way-point or to return to the home base. If the action state machine finished a task or is not performing any one at the moment, it resides in an idle state. This state notifies the outside world, that the action state machine is ready to perform a new task. If the action state machine is in idle state, the command state machine can request the execution of a specific task.

C. User Frontend: Smart Phone Application

The AVP mobile App has four functions: "Request", which refers to request a parking spot; "Collect", which refers to collect the parked vehicles; "Reserve", which refers to reserve a parking spot; and "User center", which refers to

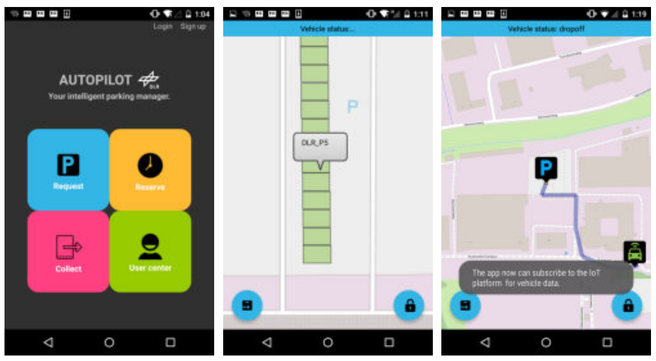


Fig. 12: AVP mobile App

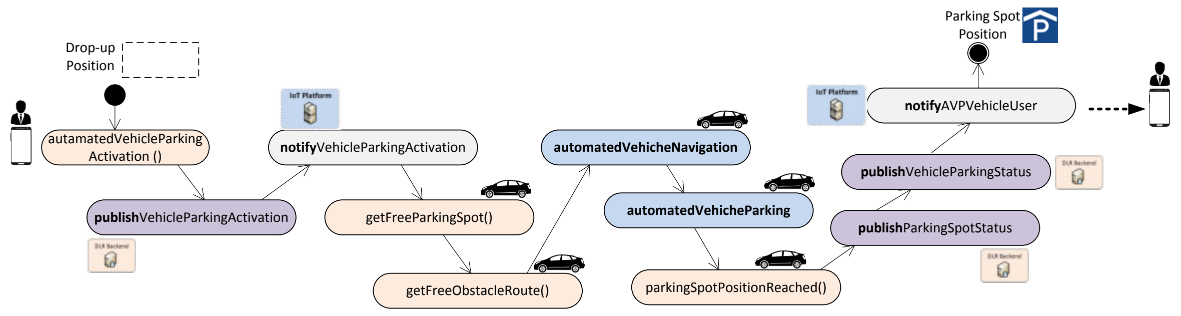


Fig. 13: AVP vehicle parking use case

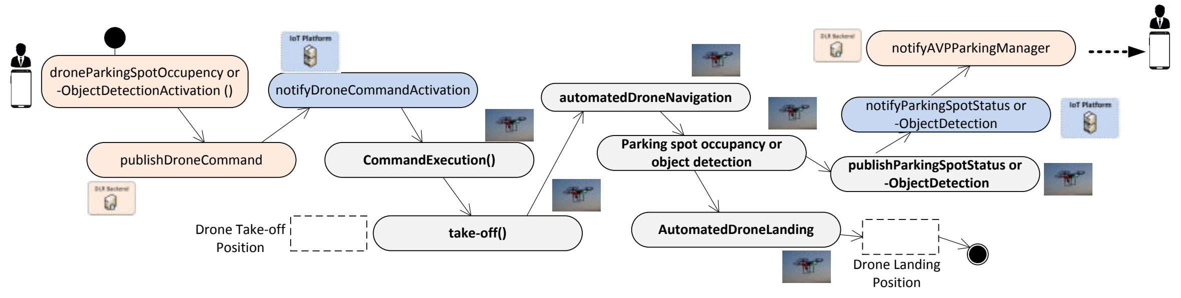

droneParkingSpotOccupency or -ObjectDetectionActivation ()
publishDroneCommand
DLR Backend
IoT Platform
notifyDroneCommandActivation
CommandExecution()
Drone Take-off Position
take-off()
automatedDroneNavigation
Parking spot occupancy or object detection
AutomatedDroneLanding
Drone Landing Position
publishParkingSpotStatus or -ObjectDetection
notifyParkingSpotStatus or -ObjectDetection
IoT Platform
DLR Backend
notifyAVPParkingManager

Fig. 14: Aerial-guided AVP use case

user information management (See Fig. 12). In "Request", once a user requests available parking spots, a list of available parking spots will be returned and displayed on the map. After the user selects one parking spot, the route from the drop-off position to the parking spot will be assigned. Meanwhile, the current status and the current positions of the vehicle will be subscribed from the IoT platform.

The App was developed with Android API 26. The App consists of a SOAP web service interface and an IoT interface, which can connect to the backend system and the IoT platform, respectively. The integrated map is Open Street Map [16].

V. SYSTEM EVALUATION

A. Use Cases

The followings use cases for vehicle parking and collection have been tested:

1) vehicle parking: The vehicle parking scenario is described in in Fig. 13.

2) Parking spot occupancy and obstacle detection with the MAV: The MAV use case for the parking spot occupancy or object detection is described in Fig. 14.

B. Application Area

The backend AVP system together with IoT devices are tested in the DLR test site in Brunswick, Germany. In order to provide routes to the IoT vehicle, we extract and proceed the digital map of the test field from the Open Street Map [16]. Further piloting tests are planed this and next year in the AUTOPILOT test site at the automotive campus in Brainport, the Netherlands.

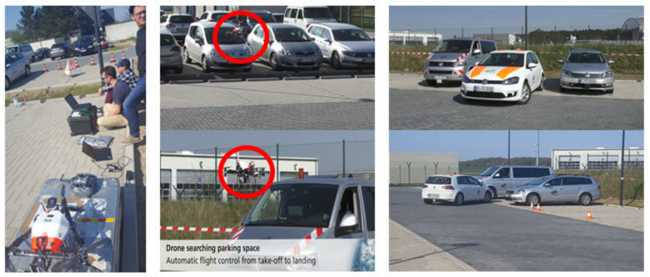
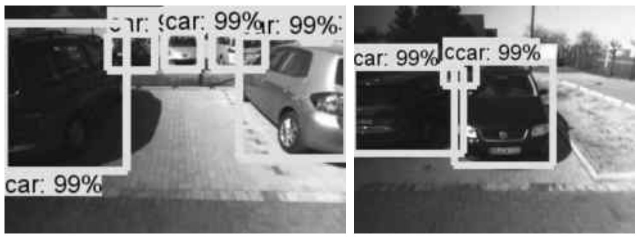

Drone searching parking space
Automatic flight control from take-off to landing

Fig. 15: AVP vehicle demonstration on the DLR test site in Brunswick

C. Testing Processes

1) Vehicle communication and functional testing: The vehicle test (Fig. 15) includes driving and parking area while sensor data is recorded and transmitted to the IoT-platform. The driven path (Fig. 17) includes several rounds at the parking area as well as parking forwards and backwards between two other cars as it would be the standard case in real. LIDAR data is mainly recorded to extend some navigation and obstacle avoidance functionalities (cf. [1]) including future indoor parking. GPS/INS data is used by a variety of experimental functions and positioning reference.

In the presented IoT communication test case, the vehicle position is regularly (approx. once per second) transmitted to the IoT platform (see Fig. 17) while it is checked whether this data has been successfully received by the IoT platform. To reproduce possible data dropouts and the corresponding software behavior, only low-distance Wi-Fi (802.11b/g/n / 2.4 GHz) connectivity is used to transmit vehicle data. As can be seen in the graph, data loss is around 50% at distances between 50 m and 100 m between car and wireless access point.



a) Free parking spot b) Occupied parking spot

Fig. 16: Deep learning based parking spot detection

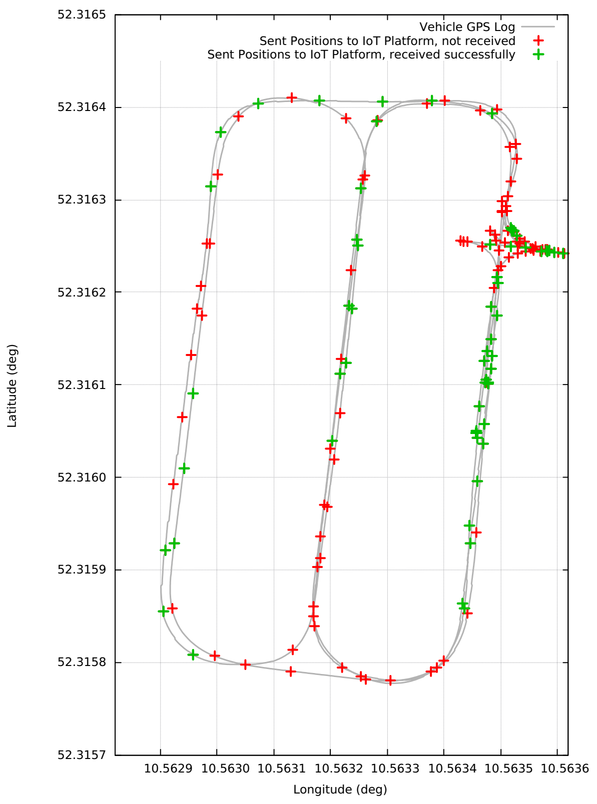


Fig. 17: AVP vehicle trajectories and sent data

2) MAV Communication and Functional Testing: The communication between MAV, IoT platform, and AVP back-end was tested extensively on a parking area at the DLR site in Brunswick. During the test, the area was closed for public traffic. It was successfully demonstrated that the MAV can receive commands from the AVP backend and publish its status during flight in real-time at 0.5 Hz. The resulting path subscribed by the AVP backend is shown in Fig. 11.

Furthermore parking spot occupancy detection was done in post-processing using state of the art deep learning methods [17], see results in Fig. 16.

VI. CONCLUSION AND FUTURE WORK

In the context of automated valet parking, this paper presents an IoT architecture where the vehicle can be connected to mobile end user applications and assistance services here provided by micro aerial vehicles. The paper presents the different components which are operating all together in a complete driving and application test. The test shows all the components being run together, including real-time data exchange while driving and parking.

Future work will integrate all these components to already existing automatic driving functions as well as IoT integration together with new automatic capabilities. For example, this includes LIDAR-based parking on places with occupancy uncertainties and also accurate driving without satellite links such as indoor parking. It also considers manual or non-cooperative vehicles driving in the same area. Within the project AUTOPILOT, all the components will be further integrated to interact with other vehicles using the standards being established now.

REFERENCES

[1] C. Loper, C. Brunken, G. Thomaidis, S. Lapoehn, P. P. Fouopi, H. Mosebach, and F. Köster, "Automated valet parking as part of an integrated travel assistance," in *Intelligent Transportation Systems-(ITSC), 16th Intl. IEEE Conf. on.* IEEE, 2013, pp. 2341–2348.

[2] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[3] ERTICO, "Autopilot project official web site," 2018, [Online; accessed 30-April-2018]. [Online]. Available: http://autopilot-project.eu/

[4] IBM, "Getting started with watson iot platform," 2018 (accessed 30-April-2018). [Online]. Available: https://console.bluemix.net/docs/services/IoT/index.html.

[5] HUAWEI, "Huawei ocean iot platform," 2018 (accessed 30-April-2018). [Online]. Available: http://www.iotocean.org/.

[6] FIWARE, "Fiware iot platform," 2018 (accessed 30-April-2018). [Online]. Available: https://www.fiware.org/.

[7] OneM2M, "Onem2m iot platform," 2018 (accessed 30-April-2018). [Online]. Available: http://www.onem2m.org/.

[8] J. Pillmann, C. Wietfeld, A. Zarcula, T. Raugust, and D. C. Alonso, "Novel common vehicle information model (cvim) for future automotive vehicle big data marketplaces," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[9] HERE, "Sensoris web page," 2018, [Online; accessed 30-April-2018]. [Online]. Available: https://www.here.com/en/vision/innovation/sensoris.

[10] HERE., "Sensoris vehicle data model specification," 2016, [Online; accessed 30-April-2018]. [Online]. Available: https://lts.cms.here.com/static-cloud-content/Company_Site/2015_06/Vehicle_Sensor_Data_Cloud_Ingestion_Interface_Specification.pdf.

[11] C. Kaschwich and L. Wölfel, "Experimental vehicles FASCar-II and FASCar-E," *Journal of large-scale research facilities*, vol. 3, 2017.

[12] K. Schmid, P. Lutz, T. Tomić, E. Mair, and H. Hirschmüller, "Autonomous vision-based micro air vehicle for indoor and outdoor navigation," *Journal of Field Robotics*, vol. 31, no. 4, pp. 537–570, July 2014. [Online]. Available: http://elib.dlr.de/93577/

[13] K. Schmid, F. Ruess, and D. Burschka, "Local reference filter for life-long vision aided inertial navigation," in *17th International Conference on Information Fusion*, 2014.

[14] M. G. Müller, F. Steidle, M. Schuster, P. Lutz, M. Maier, S. Stoneman, T. Tomic, and W. Stuerzl, "Robust Visual-Inertial State Estimation with Multiple Odometries and Efficient Mapping on an MAV with Ultra-Wide FOV Stereo Vision," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[15] S. G. Brunner, F. Steinmetz, R. Belder, and A. Dömel, "Rafcon: A graphical tool for engineering complex, robotic tasks," in *IEEE Intl. Conf. on Intelligent Robots and Syst. (IROS)*, 2016, pp. 3283–3290.

[16] OpenStreetMap, "open street map web page," 2018 (accessed 30-April-2018). [Online]. Available: http://www.openstreetmap.org/.

[17] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," *ArXiv e-prints*, Nov. 2016.