Development of a Cyber-Physical System for an Autonomous Indoor Transportation Service

Matthias Dziubany*‡, Lars Creutz*‡, Sam Kopp*, Jens Schneider*,
Anke Schmeink† and Guido Dartmann*
*Institute for Software Systems (ISS), Trier University of Applied Sciences,
Birkenfeld, Germany
Email: {m.dziubany, l.creutz}@umwelt-campus.de
†ISEK Research and Teaching Area, RWTH Aachen University, Aachen, Germany
‡These authors contributed equally to this work

Abstract—**One of the key features of industry 4.0 is the automation and optimization of transportation tasks. Especially autonomous guided vehicles meet the required high flexibility of future enterprises. In this paper, a concept of an autonomous indoor post service is proposed. The whole cyber-physical system from the autonomous vehicle, which is equipped with a Lidar sensor and programmed, using a real world automotive framework, over the secure communication design, to the dynamic time-critical decision and routing optimization is explained.**

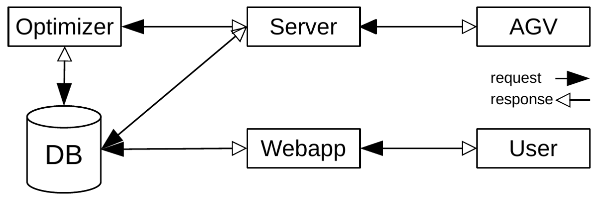

Fig. 1. Components of the cyber-physical system and their communication.

I. Introduction

The development of cyber-physical systems (CPS) is a very complex task and requires expertise in various domains. Especially the design of accurate interactions and synchronisation between the different system components is challenging. With the description of an autonomous indoor post service, this paper provides a detailed concept for the implementation of a secure CPS-based transportation system. Since the aim of the paper is to give a complete overview of the components and their secure interaction, only the basics of the components are described. The overall project with extended autonomous driving functions and vehicle constructions will serve as a demonstrator of flexible autonomous transportation systems.

In our post service, users can order an autonomous vehicle, that picks up mail at their office and then delivers it to another office within the same building. The pickup location s, pickup time window $[a, b]$ and destination d is specified via a web app by the user. After the system decides to accept the request, a self-imposed delivery time window $[a', b']$ with certain length l is assigned to the receiver.

The cyber-physical system can be divided into user, server, optimizer, web app and autonomous guided vehicle (AGV). Requests are created in the user component via a web interface and sent to the server. The server takes care of the whole communication, like the exchange of vehicle status information, triggering the optimizer, which acts as decisional component, and sending the actual destination to the vehicle. The four components and their interactions are briefly illustrated in Figure 1.

After giving some related literature examples in Section 2, the overall system, including the communication, is explained in detail in Section 3. In Section 4, the decision optimization is described. Localization and navigation of the autonomous vehicle can be found in Section 5. At last, the paper is concluded and some further ideas are given.

II. Related Works

The automated transportation within buildings is a highly researched topic. The literature ranges from restrictive conveyor belts [1], over stacker cranes [2], to flexible automated guided vehicles [3]. All transportation methods advance from CPS implementations, as they enable monitoring, interaction with other components and adaption to the production process [4]. In order to recognise faults, the authors of [1] propose to monitor a roller of a conveyor belt, which provides information like rotation speed, bearing temperature and contact pressure. In [5] and [6], the online scheduling and routing of stacker cranes or monorail systems is studied.

In contrast to conveyor belts and stacker cranes, AGV are much more flexible in transportation and can therefore perfectly adapt to changes of the production process [7]. However, the employment of AGVs is technically more challenging, since localisation and navigation is required. The Oregon Institute of Technology implemented robots, that deliver materials to a series of working stations and are tracked by a ceiling-mounted camera [7]. In [8], a Robotic delivery service in combined outdoor-indoor environments is researched. A further practical implementation of an autonomous robotic system for transportation can be found in [9]. Beside navigation and localisation, the coordination of a fleet of AGVs [3] and the

978-1-7281-6949-1/20/$31.00 ©2020 IEEE

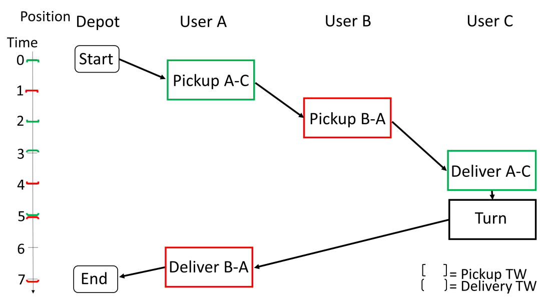

Fig. 2. Actions in an post service.

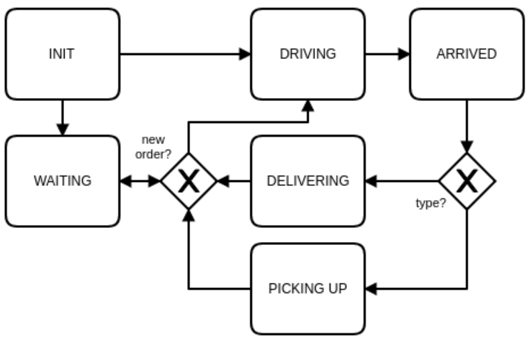

Fig. 3. State machine of the AGV.

integration of IoT devices [10] are researched in autonomous management systems.

III. Description of the Overall System and Communication Design

There is much literature stating the requirements of CPS [11], [12]. However, their implementation in practice is often not explained [3]. Therefore, we give a detailed description of the overall system and communication design, enabling a reliable, autonomous post service.

An example of an operating post service with corresponding actions is illustrated in Figure 2. There, user A requests a transport from his location to user C. The optimizer accepts this request and sends a delivery time window to user C. After the AGV picked up the letter of user A, a new transportation request from user B to user A gets known to the system. Since the optimizer decides to also accept this request, a delivery time window is sent to user A. Finally, the AGV performs the actions, depicted in Figure 2, following the state machine in Figure 3.

Before we describe how the communication and optimization work, we briefly introduce the network environment, the system lives in. The AGV uses the university's Wi-Fi network eduroam to communicate with the server and is therefore only able to send requests to a specific set of ports. Other private ports of the network are used to send control instructions to the server. Lastly, some local ports, which are used for communicating with the database and the optimizer, are entirely filtered. An overview is given in Table 1.

Ports (TCP)	Service	Accessible by
80/443	HTTP/HTTPS	Entire network
8000	AGV state updates	Entire network
7777	AGV control	Institute subnet
7000	Optimizer communication	Local
3306	Standard MySQL Port	Local

TABLE I
Overview of the services, their corresponding TCP port and their access level

The web app, that hosts the post services for our users, forces HTTPS connections, which offers end-to-end encryption for the regular user communication. The remaining parts use raw TCP sockets for communication. To prevent attacks, we use a thin cryptographic layer, which is part of the communication protocol. We also use public key encryption, based on the elliptic curve secp256r1 [13]. An address inside the system consists of a private and a public key. The public key is exchanged between the components and used to encrypt messages for the intended receiver. The private key is used for signing and decrypting. Our server backend is mainly written in C++, therefore we use the Crypto++ library to handle any form of elliptic curve cryptography.[1] Any non-local communication must comply with the protocol, to be able to successfully interact with the server. Before we describe the message format, the basic protocol definition contains:

1) Accept a connection on TCP port 8000 and read the fingerprint of the caller's public key.
2) Generate a random nonce, encrypt it under the known public key and write it to the socket. Using a nonce on each synchronous connection prevents replay-attacks [14].
3) Read until the defined message delimiter or until the used buffer is full. Using a size limit on the buffer prevents any sort of overflow attack on the exposed port, which could cause the application to crash [15].
4) Decrypt the buffer that contains information about the AGV state update and the corresponding signature.
5) Check the signature by testing for all available client public keys.
6) If the signature is correct, parse the AGV message and handle the request.

The server's response, also a predefined message type, will be signed and encrypted before responding to the exposed raw socket.

All exchanged messages are defined using Google protocol buffers[2] (protobuf), serialized as string objects and hex-encoded afterwards. We choose to use an additional layer of conversion, because the encrypted messages might contain some unsupported characters. A regular state update of the AGV would build the data to be sent like the following:

[1]Crypto++® Library 8.2, https://www.cryptopp.com - Accessed 02-26-2020
[2]Google Protocol Buffers, https://developers.google.com/protocol-buffers - Accessed 02-26-2020

1) Create the actual state update
2) Get the nonce and update the state update
3) Serialize the message
4) Hex-encode serialized message
5) Sign the hex-encoded message

The signature and message are packed into a protobuf object afterwards, encrypted by using the server's public key and sent in hex-encoded format via the TCP socket, under the sign-then-encrypt principle, which is sufficient for our use case, because of the synchronous communication and the inclusion of a random nonce [16]. A key feature, besides the design and usability of the components, was to keep the exchanged messages as slim and secure as possible. An average conversation between server and AGV results in approximately 1,2kb of data, transferred over the TCP stream.

IV. DECISION OPTIMIZATION

As most requests in a post service come in spontaneous, online optimization is necessary for decision-making. In short time, our algorithm does not only decide, which new request to accept or to deny, but it further assigns a self-imposed delivery time window to the receiver. Literature about self-imposed time windows can be found in [17]. Since we have only one vehicle in operation, and the corresponding graph of our building, on which routing takes place, is very small, we use exact IP solutions in our online heuristic. In more detail, the time window assignment and routing problem is formulated as a multicommodity-coupled flow problem in time-expanded networks and solved optimally by Gurobi[3] after each server call. An introduction to multicommodity-coupled flows can be found in [18].

Time-expanded networks are perfectly suited to capture the service time and the turning time of the vehicle. Further, the assignment of time windows can be accomplished. As we want to give an overview over the whole CPS, we do not explain the construction of the time-expanded network in detail, rather we give an example for one hallway with three offices and one request in Figure 4. There, the time to drive from one office to the next office is one, the service time is two and the turning time is one. In order to maximize the number of accepted requests, we put costs of minus one on the arc from s_{A-C} to s^0_{A-C} and give all other arcs zero cost.

V. AUTONOMOUS GUIDED VEHICLE

Autonomous driving in an indoor environment requires at least localisation and navigation. That is why we focus on these two problems. Further desirable properties, like obstacle avoidance and trajectory coordination of a fleet of AGVs can be found in [19]. The vehicle for the transport service, described here, is a 1:8 scale electric model car, which features a high-torque, brushless engine and is controlled with cruise control. It is also equipped with a variety of sensors to perceive its surroundings, such as front-facing almost 180-degree Lidar and fisheye camera, ultrasonic sensors at its sides and back, rear view camera and wheel encoders. The sensor outputs of the Lidar, fisheye camera and the ultrasonic sensors can be seen in Figure 5. The driving functions are implemented using ADTF (Automotive Data and Time-triggered Framework)[4]. This software enables the time-based processing of several data streams to generate driving functions.

A. Localization

Localization is imperative for autonomous driving, as well as routing and scheduling. Since every localization method, like Wifi-based [20], sound-based [21] and vision-based [3] has its drawbacks, the simultaneous usage of different methods is reasonable. The combination of relative and absolute position systems, like in [22], is also followed by us. Bluetooth beacons can be used to correct the cumulative errors of wheel encoders, emerged by limited encoder resolution, uneven floors, slipping wheels etc. Since we have only one vehicle in operation, we use Two-Way-Ranging (TWR), where a request is continuously sent to the Bluetooth beacons and the corresponding response time is used to determine the distance of the beacons to the vehicle.

Another possible method could be using particle filters based on the Lidar data like proposed in [23].

B. Navigation

The indoor environment, which our vehicle is supposed to navigate in, consists mainly of long straight corridors, so following a wall is the core of the car's driving functionality. We have implemented this functionality based on an algorithm proposed by Karl Bayer [24]. Our modified implementation uses a linear least-squares-fitting (LLSF) function, to estimate the position of a wall $\tilde{w}$, based on a random subset S of the 2D point cloud L, generated by the Lidar. Only if more elements of S than a specified proportion p have distance smaller than a specified threshold t to the fitted line, and are therefore called inliers S_{inliers}, $\tilde{w}$ is preserved. If $\tilde{w}$ is already known, it is checked against a new subset S' of L. The new S' is taken from the same section of the Lidar sample as the previous one. This increases the probability of recognizing an already known $\tilde{w}$. Only if $\tilde{w}$ is no longer recognized, S to find a new $\tilde{w}$ is generated from a new random section of L. This procedure is repeated, until a $\tilde{w}$ is found or the maximum number of iterations i_{max} is reached. Subsequently, a second LLSF can be performed, this time, only with the points classified as inliers S_{inliers}, to obtain a more accurate $\tilde{w}$. Since the search for a new $\tilde{w}$ is more computing-intensive than updating an already-known $\tilde{w}$, we always try to update an already-known $\tilde{w}$ first. Only if the known $\tilde{w}$ cannot be confirmed by the inlier check of S, we create a new S' and search for a new $\tilde{w}$ therein.

We then use the following controller function to compute a steering value ω, in order to hold a predefined distance σ, based on the angle $\tilde{\theta}$ and distance $\tilde{\sigma}$ to $\tilde{w}$:

[3]Gurobi, https://www.gurobi.com - Accessed 02-26-2020

[4]https://www.digitalwerk.net/adtf/ - Accessed 04-25-2020

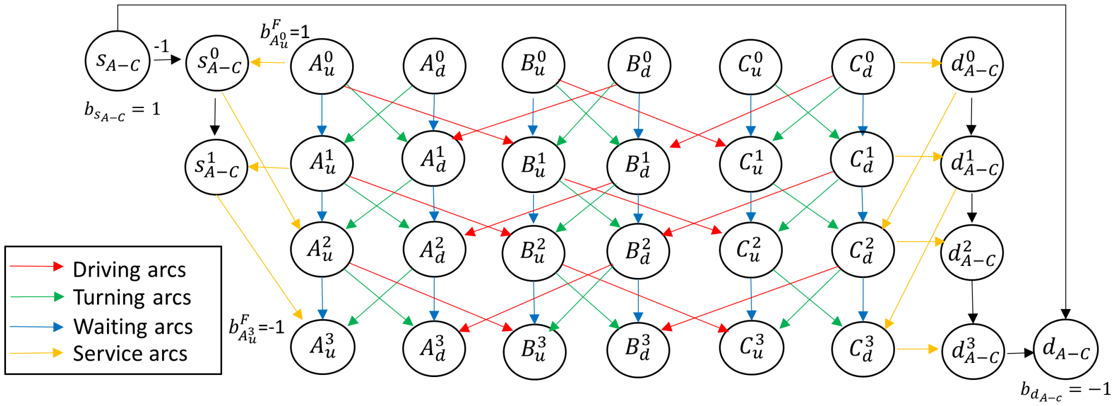

Fig. 4. An example for one hallway with three offices A,B,C and one transport request from A to C with pickup time window [0,2]. The lower indices of the nodes depict the direction (u=up,d=down) and the upper ones the time. All arcs, except the arc from s_{A-C} to s^0_{A-C}, have zero costs. After the optimization problem was solved considering the request the first time, the self-imposed delivery time window is set.

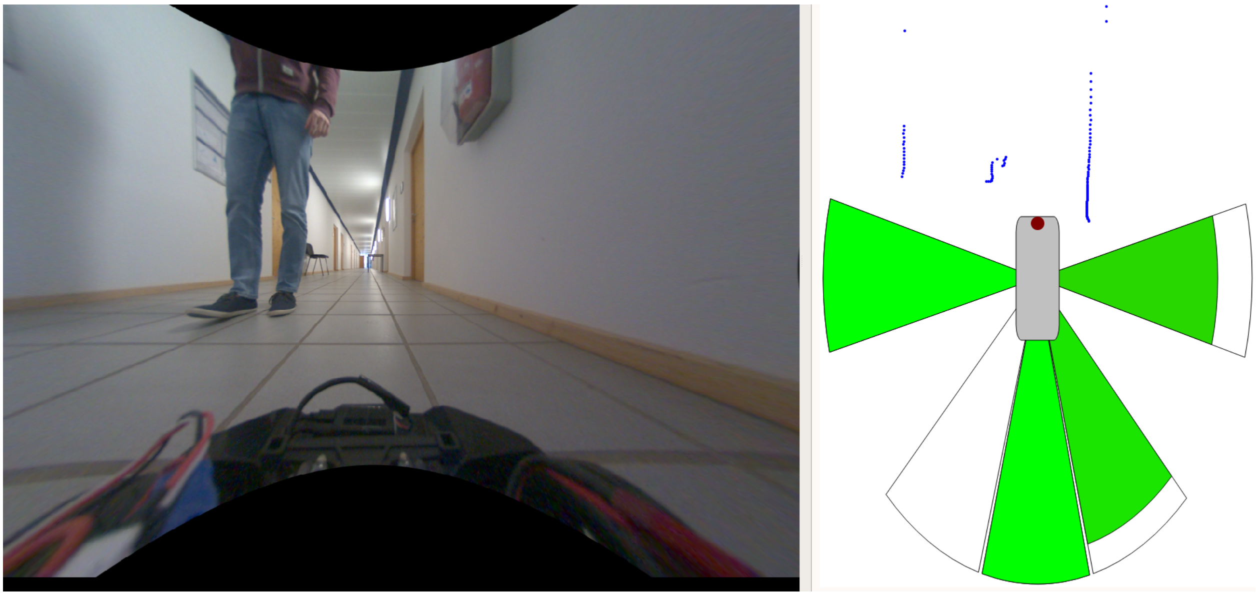

Fig. 5. Left the output of the fisheye camera and right the output of the Lidar and the ultrasonic sensors.

$$\omega = k_1 \cdot \tilde{\theta} + k_2 \cdot (\tilde{\sigma} - \sigma),$$

where k_1 and k_2 are control parameters. This procedure is shown in Algorithm 1, as a pseudo code (based on [24]).

In addition, the Lidar points are also used for an emergency braking function and for detecting and bypassing obstacles. This basic functionality is already sufficient to complete simple driving functions in our test environment.

VI. CONCLUSION AND CURRENT STATUS OF THE PROJECT

We have presented an overview of our ongoing work of implementing an autonomous post service. The following list shows the current status of the project:

- the server and the optimizer are implemented and tested, based on simulation data
- the AGV can perform simple driving tasks, like following a wall and turning as well as bypassing obstacles
- the indoor localization and the navigation to a given destination is in progress.

In the future, we want to use the fisheye camera and implement further autonomous driving algorithms, like obstacle avoidance and intention recognition.

ACKNOWLEDGMENT

This work has been funded by the Federal Ministry of Transport and Digital Infrastructure (BMVI) within the funding guideline "Automated and Connected Driving" under the grant number 16AVF2134C.

Algorithm 1 Wall Following

Input: Vector of 2d Lidar points $L = \{l_1, \ldots, l_n\}$, σ, k_1, k_2

Output: Steering value ω

if wall_known$= 0$ **then**
    i $\leftarrow 0$
    while $i < i_{\max}$ **and** wall_known$= 0$ **do**
        i $\leftarrow i + 1$
        $S = \{l_{\min}, \ldots, l_{\max}\} \leftarrow$ random_excerpt(L)
        $\tilde{w} \leftarrow$ LeastLinearSquaresFit(S)
        $S_{\text{inliers}} \leftarrow$ GetInliers$(\tilde{w}, S, t)$
        if $|S_{\text{inliers}}|/(\max - \min) > p$ **then**
            wall_known $\leftarrow 1$
        end if
    end while
    $\tilde{w} \leftarrow$ LeastLinearSquaresFit(S_{inliers})
else
    $S_{\text{inliers}} \leftarrow$ GetInliers$(\tilde{w}, \{l_{\min}, \ldots, l_{\max}\}, t)$
    if $|S_{\text{inliers}}|/(\max - \min) > p$ **then**
        $\tilde{w} \leftarrow$ LeastLinearSquaresFit(S_{inliers})
    else
        wall_known $\leftarrow 0$
    end if
end if
if wall_known$= 1$ **then**
    $\tilde{\theta} \leftarrow$ angle(position(AGV), $\tilde{w}$)
    $\tilde{\sigma} \leftarrow$ offset(position(AGV), $\tilde{w}$)
    $\omega \leftarrow k_1 \cdot \tilde{\theta} + k_2 \cdot (\tilde{\sigma} - \sigma)$
    return ω
end if

REFERENCES

[1] G. Fedorko. "Implementation of Industry 4.0 in the belt conveyor transport".

[2] X. Sun, Z. Ma, Z. Wang and C. Ai. "The development of stereoscopic warehouse stacker control system based on motion controller". MATEC Web of Conferences, 139, 00038, 2017.

[3] R. Van Parys, M. Verbandt, M. Kotze, P. Coppens, J. Swevers, H. Bruyninckx, J. Philips and G. Pipeleers. "Distributed Coordination, Transportation & Localisation in Industry 4.0". 2018.

[4] Y. Cohen, M. Faccio, F. Pilati and X. Yao. "Design and management of digital manufacturing and assembly systems in the Industry 4.0 era". The International Journal of Advanced Manufacturing Technology, 2019.

[5] Q. Gao and X. Lu. "The complexity and on-line algorithm for automated storage and retrieval system with stacker cranes on one rail". Journal of Systems Science and Complexity, 29, 2016.

[6] K. Gutenschwager, C. Niklaus, S. Voss. "Dispatching of an Electric Monorail System: Applying Metaheuristics to an Online Pickup and Delivery Problem". Transportation Science, 38, 2004.

[7] D. Culler and J. Long. "A Prototype Smart Materials Warehouse Application Implemented Using Custom Mobile Robots and Open Source Vision Technology Developed Using EmguCV". Procedia Manufacturing, 5, 2016.

[8] R. Limosani, R. Esposito, A. Manzi, G. Teti, F. Cavallo and P. Dario. "Robotic delivery service in combined outdoor-indoor environments: Technical analysis and user evaluation". Robotics and Autonomous Systems, 103, 2018.

[9] A. Bouguerra, H. Andreasson, A. Lilienthal, B. Astrand and T. Rognvaldsson. "An Autonomous Robotic System for Load Transportatio". IEEE Conference on Emerging Technologies and Factory Automation, 2009.

[10] F. Yao, A. Keller, M. Ahmad, B. Ahmad, R. Harrison and A. Colombo. "Optimizing the Scheduling of Autonomous Guided Vehicle in a Manufacturing Process". 2018.

[11] M. Sadiku, Y. Wang, S. Cui and S. Musa. "Cyber-Physical Systems: A Literature Review". European Scientific Journal, 13, 2017.

[12] A. Darwish and A. E. Hassanien. "Cyber physical systems design, methodology, and integration: the current status and future outlook". Journal of Ambient Intelligence and Humanized Computing, 2017.

[13] S. Turner et al.. Elliptic Curve Cryptography Subject Public Key Information, https://www.ietf.org/rfc/rfc5480.txt, 2009

[14] G.M. Køien. "A Brief Survey of Nonces and Nonce Usage". SECURWARE International Conference on Emerging Security Information, Systems and Technologies. 2015

[15] C. Cowan, P. Wagle, C. Pu, S. Beattie and J. Walpole. "Buffer overflows: attacks and defenses for the vulnerability of the decade". Foundations of Intrusion Tolerant Systems, 2003 [Organically Assured and Survivable Information Systems].227-237. 2003

[16] D. Davis. "Defective Sign Encrypt in S/MIME, PKCS7, MOSS, PEM, PGP, and XML". USENIX Annual Technical Conference. 2001.

[17] O. Jabali, R. Leus, T. Woensel and T. Kok. "Self-imposed time windows in vehicle routing problems". OR Spectrum, 37, 2, 2010.

[18] S. Bsaybes, A. Quilliot and A. Wagler. "Fleet management for autonomous vehicles using multicommodity coupled flows in time-expanded network". 17th International Symposium on Experimental Algorithms (SEA 2018), Leibniz International Proceedings in Informatics (LIPIcs), 103, 2018.

[19] D. Herrero-Perez and Humberto Barberá. "Decentralized coordination of automated guided vehicles". 2008.

[20] H. Blunck, T.S. Prentow, S. Temme, A. Thom and J. Vahrenhold. "Deviation Maps for Robust and Informed Indoor Positioning Services". SIGSPATIAL Special, 2017.

[21] M. Dziubany, R. Machhamer, H. Laux, A. Schmeink, K.-U. Gollmer, G. Burger and G. Dartmann. "Machine learning based indoor localization using a representative k-nearest-neighbor classifier on a low-cost iot-hardware". 26th European Signal Processing Conference (EUSIPCO), IEEE, 2018.

[22] HS. Kim, W. Seo and K. Baek. "Indoor Positioning System Using Magnetic Field Map Navigation and an Encoder System". Sensors, 17, 651, 2017.

[23] Xiao, Yue and Ou, Yongsheng and Feng, Wei. "Localization of indoor robot based on particle filter with EKF proposal distribution". 2017 IEEE international conference on cybernetics and intelligent systems (CIS) and IEEE conference on robotics, automation and mechatronics (RAM), IEEE, 2017.

[24] K. Bayer. Wall Following for Autonomous Navigation. 2012.