Bejarano, Ronal; Atmojo, Udayanto Dwi; Blech, Jan Olaf; Vyatkin, Valeriy

Towards enhanced live visualization based on communication delay prediction for remote AGV operation

Published in:
Proceedings - 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2021

DOI:
10.1109/ETFA45728.2021.9613513

Published: 30/11/2021

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Bejarano, R., Atmojo, U. D., Blech, J. O., & Vyatkin, V. (2021). Towards enhanced live visualization based on communication delay prediction for remote AGV operation. In *Proceedings - 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2021* Article 9613513 IEEE. https://doi.org/10.1109/ETFA45728.2021.9613513

Towards enhanced live visualization based on communication delay prediction for remote AGV operation

Ronal Bejarano*, Udayanto Dwi Atmojo *, Jan Olaf Blech*, Valeriy Vyatkin*,**
* School of Electrical Engineering, Department of Electrical Engineering and Automation, Aalto University, Espoo, Finland
** Luleå University of Technology, Luleå, Sweden
Email: {ronal.bejarano, udayanto.atmojo, valeriy.vyatkin}@aalto.fi

***Abstract*—This paper presents the development progress of a methodology to enhance the dynamic visualization of remote Automated Guided Vehicles (AGVs) using open source frameworks as Gazebo and ROS. The approach is based on a deterministic path pre-calculated by edge computing, accessible for a 3D web visualization cloud platform. The trajectory displayed is verified by a live position streaming from the AGV and a predicted communication delay value. By succeeding on the methodology proposed, it is expected to enhance the fidelity of the 3D live representation for every trajectory executed dynamically and autonomously by AGVs in the shop floor, leaving behind an initial scenario with low visualization fidelity due non-deterministic behavior of long-distance communication channels (including wireless networks essential for AGVs). This work aims to have an impact on improving the user experience of remote interfacing tools, supporting a web-based paradigm to visualize the behavior of cyber-physical systems in agile manufacturing.**

***Index Terms*—Automated Guided Vehicles (AGVs), remote visualization, cyber-physical systems**

I. Introduction

The Industry 4.0 paradigm, based on seamless integration of data between business, management, and production, is pushing the interests of industries to adopt and apply tools benefiting from IT concepts as an immersive experience, distributed computing, semantic web applications [1], wireless communications, and many more. In this case study, the Autonomous Mobile Robots (AMRs) or AGVs with technology readiness above level 8 (System complete and qualified) were selected as an excellent example of robust coexistence between IT and industrial controls with high IT demands. Nevertheless, this methodology is expected to be applicable to other similar devices, as drones or robotic manipulators.

Nowadays, commercial models of AGVs for industrial use are often present in use cases comprising material handling or warehouse logistics. Robust sensing and on-board edge processing provides necessary resources for the autonomous control algorithms to navigate the vehicle through pre-calculated paths in dynamical factory floor environments (including traffic of pedestrians and other vehicles). However, the operation of these systems is not as deterministic as any other traditional industrial machine, because the environment can be extremely dynamical. Also, a large-scale deployment scenario for AGVs in factories around the world is still in progress, growing through conceptual designs and demonstrators. From a European perspective, the conceptual designs for factories of the future should consider adaptability as a relevant feature for manufacturing sites [2]. The changes in the factory layout, process, products, batch sizes, and recipes are considered prevalent, bringing critical challenges on every technological front.

The main challenge considered by the authors in this work is to enable an optimized method to represent virtually the physical behavior of an AGV fleet through tools inspired by evoluted HMI or SCADA. This paper presents the progress towards rethinking the architecture of human - machine interfacing, considering visualization fidelity indicators as the main performance metrics. In that way, the problem of a reliable interface of autonomous vehicles, translates into requirements for IT tools compatible with AGVs, capable of providing less jerky and lagged visual interface of the vehicle's behavior to human operators located away of the operation site. The sections in this document are structured as follows: In Sect. II, some relevant resources and previous experiences are presented along the contextual description of fidelity and the experimental framework. Sect. III presents details about the main components in the proposed algorithm and finally, the preliminary results are presented in Sect. IV.

II. Related work and context

From user experience to radio technologies, numerous authors have contributed to enrich the toolboxes for building industrial applications in mining, public transport, heavy machines, warehouse management, etc. T. Mirfakhrai and S. Payandeh presented in [3] an approach for remote control of a virtual robot through a remote haptic interface. Nazari *et al.* presented an analysis about the integration of communication parameters into a controller driving an automated vehicle towards a traffic intersection [4]. Lozoya *et al.* presented in [5] the simulation of a remote wireless path tracking control for an AGV. Zhan and Yu presented in 2018 a survey on wireless communication technologies for AGVs in [6]. Ohori *et al.* exalted the importance of handling correctly the wireless communication inherent in AGVs [7]. The authors

978-1-7281-2989-1/21/$31.00 ©2021 IEEE

presented a survey about delay and throughput for long-distance communication of AGVs in [8].

From a different perspective, the entertainment industry have developed for decades animation methodologies. As a valuable reference from a non-scientific resource, we would like to include the 12 basic principles of animation from O. Johnston and F. Thomas [9]. This work determined essential characteristics of animation technologies that are a valuable contribution to the main objective of this work, enhance the remote visualization from an animation perspective. The straight-ahead action and pose to pose animation principle describe the basic dynamics behind animation. Straight-ahead actions are animated frame by frame from beginning to end. In contrast, the pose to pose method involves drawing the most relevant frames (i.e beginning, middle, end), and then fills the intervals with progressive movements across the gaps. Taking into account this concept, probably, one of the most important principles is timing. Timing refers to the number of frames for an action, regulating the speed of the actions during the playback. For consistent animation, the timing between frames should be constant.

The concept of fidelity in this study can be defined as the extent to which the detail and quality of the AGV environment are represented by a remote view component, judging from a human perspective. Two main indicators were formulated to assess the methodology proposed. The most perceptible fidelity deficiencies in the current interface systems are related to reaction delays (popularly known as lag) and lack of fluency on the animation dynamics (known as choppy or jerky playback). Then, the fidelity indicators are defined as Average Rendering Delay (ARD) in Eq. 1 and Uniformity of Animation Detail (UAD) in Eq. 2, both based in the time between changes of position t_p. In a simulation or even a in-situ real time live setup, the adverse effects from data transfer delays in ARD and UAD are barely noticeable, but they become problematic in remote rendering use cases.

$$ARD = \frac{1}{n}\sum_{k=1}^{n} t_p(k) \tag{1}$$

$$UAD = \frac{1}{n-1}\sum_{k=1}^{n} (t_p(k) - \overline{t_p})^2 \tag{2}$$

The highest-fidelity corresponds to the lowest possible values for ARD and UAD, so the delay is minimum assuming low jitter values (measured as the coefficient of variation for latency). Nevertheless, high fidelity becomes technically challenging when the only available communication methods behave in a non-deterministic way, influenced by several real-world factors such as weather, maintenance, crowded spectrum, and many more.

The framework for this study is based on the Model View Control pattern. The MVC pattern is resilient to changes on individual components that might represent minor disturbances over the general setup. The model component uses the publish-subscribe messaging pattern implemented in ROS, by accessing predefined nodes and topics. A ROS Driver, produced by M. Günther, M [10] is used to gain access to the vendor-unsupported ROS platform in the vehicle used for this study, a MIR-100. The main visualization tool is Gazebo, an open-source 3D robotics simulator that integrates the Open Dynamics Engine (physics) and OpenGL (Open Graphics Library). Gazebo provides a WebGL client called GZweb, for a front-end graphical interface accessible and cross platform compatible through a web browser. A REST API based on web services is hosted by the vehicle controller as main control interface.

III. Methodology and system architecture

From the animation theory presented in Sect. II, the starting point for this work can be considered as a live animation resembling the straight-ahead drawing method. In practice, it is based on a virtual environment implemented in Gazebo, which renders the localization data coming as a live stream of poses from the AGV controller. Usually the interfacing and control systems consume velocity data, which might be less susceptible to latency, but the cumulative position error becomes a major issue during poor communication quality periods, for that reason the localization data was selected over velocity. Given the initial conditions and the essential path calculation feature available in modern AGVs, it is possible to use the path plans and other data available in the system to enhance the fidelity of a live virtual visualization. This work proposes to upgrade the animation methodology to a new combined technique, inspired in the straight-ahead and pose to pose drawing methods. Fig. 1 presents a general overview of the preliminary results implementing the enhanced methodology discussed in this section. It is important to note that a tight coupling of timing and data transfer delay for a straight-ahead method is one of the most critical factors influencing the current flaws on fidelity indicators (ARD and UAD).

The list of expected positions for every autonomous navigation trajectory is reported by exception from the robot controller. This list, also known as a path plan, is represented in the form of a uni-dimensional array of poses, reported before starting navigation. Each pose includes *X, Y, Z* coordinates in meters and *X, Y, Z, W* orientation in radians (quaternion). A detailed description of the upcoming poses provides enough data to resemble the pose to pose drawing method. The visualization component can display data coming from a simulated position data stream, reconstructed as a controlled playback of the extended path plan. Whereas, the live stream of poses reported from the AGV can work as contrast data to confirm the veracity of the simulated behavior (confidence), instead of driving the robot pose in the animation as it used to be. Monitoring the round trip delay for the communication channel between the AGV and the visualization server provides additional diagnostic information related to the animation timing and predicting lags that might impact the confidence. In the future, the visualization server

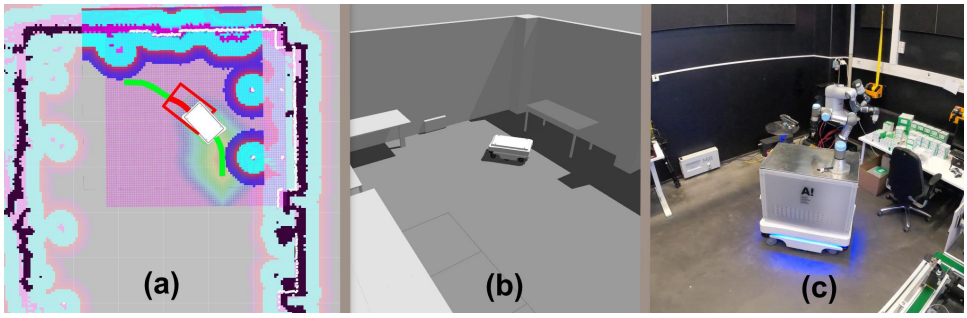

Fig. 1. Preliminary results. (a) Path plan in green line, uncertainty in red line. Red square represents the delayed position report. Image taken from RViz. (b): 3D visualization, taken from Gazebo interface. (c) path execution of AGV

can be deployed through publicly available cloud platforms (Microsoft Azure, Amazon Web services, private clouds, etc.).

The main difference between the simulated and the live stream localization data is that the latter travels a long distance from the robot to the to the visualization server with variable delay (jitter). The benchmark for such delay is approximately two to three times the expected RTT [8], calculated from the geographical separation over the speed of light through fiber optic [11]. Nevertheless, especially while using an internet-based channel, the delay behavior is subject to many external factors. Since there is a non-deterministic behavior of long-distance communication over the internet, there is also a direct impact on the visualization's ARD and UAD. Subsequently, the jitter becomes the core flaw for fidelity in a visualization merely based on straight-ahead animation. In other words, the timing principle is seriously affected by non-uniform position updates.

The method proposed in this article is implemented via two main algorithms, a Position Estimator (PE) Algo. 1 and a Delay Predictor (DP) Algo. 2. Both algorithms use of ROS, each one includes an initialization routine which deals with basic declarations (variables, nodes, topics, and handlers) necessary to exchange and process the data. The functions contained in the algorithms correspond to callbacks executed by exception (on demand) or periodically (timed). The PE handles the localization data reported from the AGV and provides data separately to the web visualization clients. The DP produces a forecast of the expected communication delay for the next second. The prediction is made by monitoring the delay in the communication channel between the vehicle and the cloud server, as well as between the server and client location during a preset period (constant). Sect. IV presents more details about the pre-trained supervised data model used in the DP.

The PE carries multiple tasks as receiving the path plans from the vehicle controller before starting the navigation, setting up the reconstructed playback for the visualization, and progressively validating the visualization against the live data reported from the AGV while moving. The validation routine consist on finding the live stream coordinates into the path list and verifying the absence of decrements between the last and present path index. The acceptable behavior indicates a null or positive trajectory progress. The validation process also has a secondary certainty component, which aims to inform the user if the position validation is running as expected. The certainty is evaluated in periods dictated by a fixed report period (ΔT) and the predicted communication delay. A certainty rate boundary provides a safe threshold to trigger an error in the visualization. In that way, deviations on communication delays or data loss will work as additional diagnostic information through the certainty rate (C_r) without affecting the visualization fidelity. Common issues can cause deviations in the communication behavior as unreliable channels with temporary high delays or constraints on the VPN tunnel to support User Datagram Protocol (UDP) instead of Transmission Control Protocol (TCP).

A general example of the system architecture for the enhanced visualization methodology is presented in Fig. 2. The deployment is based in shop-floor components interconnected to a visualization cloud server through a point to site VPN tunnel, in such way it is possible to have seamless communication through a secure channel over Internet Protocol v4 (IPv4), according the communication patterns from [8]. As any other web application, the cloud server deployment is expected to be as near as possible to the clients. This practice avoids undesirable side effects in the visualization fidelity due transmission delays from server to client. Finally the clients connect through a web browser accessing the WebView using Hypertext Transfer Protocol Secure (HTTPS).

IV. Implementation

The current implementation of the algorithms presented in Sect. III uses Python as main programming language, interpreted in Azure Cloud, as described in Fig. 2. The cloud server requires a Linux OS to ensure compatibility with ROS and Gazebo. Debian or Ubuntu are recommended, but also it can operate as a Docker container instance.

The ability to verify the visualization progress against the position reports from the AGV is paramount to make the enhanced method reliable and coherent with the physical system. The PE algorithm contemplates two critical aspects. First, the refresh period of the path array (ΔT), which is considered as a constant value. This is based on the assumption of a positive correlation from variable distance traveled during

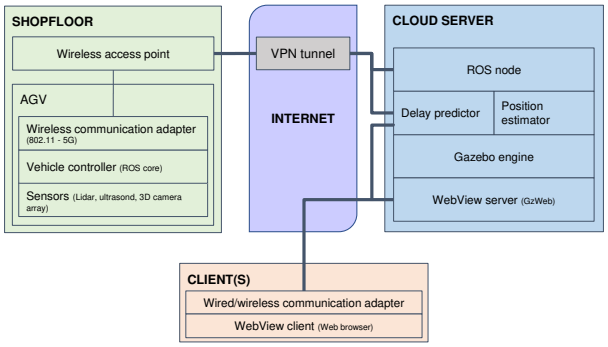

Fig. 2. System architecture for the enhanced visualization method

ΔT along the trajectory (path plan) and the vehicle speed. However, ΔT can be also a variable in function of the estimated pose index n. Second, the certainty rate (C_r) is a critical factor of reliability. Its simple calculation method contrast the deviations in the predicted communication delays. Since the animation is not affected by communication delays and the delays are unlikely to disappear, the C_r values can be presented as a numerical and/or color indicator integrated into the front-end visualization in a separated object implemented through HTML or a Gazebo plugin. Currently the research project targets to find ways to overcome the impact of incomplete path plans, reported with missing orientation quaternions.

The data-driven model included in DP is a critical component of the methodology, since inaccurate predictions can cause false errors. Any event blocking the articulated data transfer for visualization may harm the fidelity indicators. An initial implementation of the delay predictor was formulated as a Recurrent Neural Network (RNN) deployed using TensorFlow-Keras, with 4 layers (LSTM + LSTM + Dense + Dense) of shapes 80, 80, 40, and 1 respectively. A sampling window size (S_s) of 4, trained for 20 epochs provided the best loss ($2.7 * 10^{-4}$) and Root Mean Square Error (2.8). The data-set used for training and test consists of 259200 registers of timestamped RTT captured during 3 days at the rate of 1 sample per second, following the cloud communication pattern from [8]. While this component is relevant for the DP algorithm, this article focuses on the procedural context. Further experimentation and optimization of the RNN parameters can enable the model to give better results.

V. Conclusions and Future Work

The approach presented aims to enhance the fidelity of live visualization for AGVs operating remotely. It extends the scope of the IT tools used to run a commercial model of industrial AGV and web architecture to enhance the use of path planning information. By rethinking the way how the communication delay is presented to operators we expect to improve the system usability. In the future, immersive technologies and better tools for visualization and interaction will play a crucial role in technical services provided by companies. Enhancing the interface tools can improve the working conditions for experts facing extraordinary logistic challenges to service facilities in remote locations.

In the future, we aim to improve the validation method, the delay predictor model, and expand the training data set. Also, we will evaluate the algorithms proposed by comparing the ARD and UAD results in different challenging scenarios, including peers separated by thousands of kilometers or producing recurrent updates on playback method for simulation.

VI. Acknowledgements

The authors would like to dedicate this work in memory of Jan Olaf Blech, deceased on 02/2021. This work is part of the EIT MirrorLabs project, which receives financial support from the European Institute of Technology (EIT). This body of the European Union receives support from the European Union's Horizon Europe program.

References

[1] V. Vyatkin, "Software engineering in industrial automation: State-of-the-art review," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1234–1249, 2013.

[2] U. européenne. Direction générale de la recherche, *Factories of the Future: Multi-annual Roadmap for the Contractual PPP Under Horizon 2020*, ser. EDC collection. Publications office of the European Union, 2013. [Online]. Available: https://books.google.fi/books?id=ZC0wngEACAAJ

[3] T. Mirfakhrai and S. Payandeh, "A delay prediction approach for teleoperation over the internet," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, 2002, pp. 2178–2183 vol.2.

[4] M. A. Nazari, T. Charalambous, J. Sjöberg, and H. Wymeersch, "Remote control of automated vehicles over unreliable channels," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6.

[5] C. Lozoya, P. Martí, M. Velasco, J. M. Fuertes, and E. X. Martín, "Simulation study of a remote wireless path tracking control with delay estimation for an autonomous guided vehicle," *The International Journal of Advanced Manufacturing Technology*, vol. 52, no. 5-8, pp. 751–761, 2011.

[6] M. Zhan and K. Yu, "Wireless communication technologies in automated guided vehicles: Survey and analysis," in *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2018, pp. 4155–4161.

[7] F. e. a. Ohori, "Estimating wireless link quality using multiple remote sensors for wireless control of agv in a factory," in *2020 23rd International Symposium on Wireless Personal Multimedia Communications (WPMC)*. IEEE, 2020, pp. 1–6.

[8] R. Bejarano, R. Pääkkönen, J. O. Blech, I. Peake, P. Herrmann, and V. Vyatkin, "Assessing long distance communication alternatives for the remote control of agvs," in *25th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2020, Vienna, Austria, September 8-11, 2020*. IEEE, 2020, pp. 69–76. [Online]. Available: https://doi.org/10.1109/ETFA46521.2020.9211954

[9] F. Thomas, *The Illusion of Life: Disney Animation*, ser. Disney Editions Deluxe (Film). Disney Editions, 1995. [Online]. Available: https://books.google.fi/books?id=3lXFNAAACAAJ

[10] M. Günther, "dfki-ric/mir_robot," https://github.com/dfki-ric/mir_robot, 2017, accessed: 2021-03-03, BSD-3-Clause License, DFKI Robotics Innovation Center.

[11] F. e. a. Poletti, "Towards high-capacity fibre-optic communications at the speed of light in vacuum," *Nature Photonics*, vol. 7, no. 4, pp. 279–284, 2013, number: 4 Publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/nphoton.2013.45

APPENDIX

Algorithm 1 Position estimator

Input: Path plan (Path), Predicted delay (ms), Maximum rate of unbounded certainty
Output: Estimated pose (Pose2D), Certainty [bounded, unbounded], Certainty rate
Initialisation :
Initialize refresh rate for the visualization pose '$pose\_repost$', maximum rate of unbounded certainty Cr_{Max}
Subscribe to topics
Path plan
Live pose report
Delay prediction
Create empty arrays $posx$ (m), $posy$ (m), T (s) for buffering path coordinates and timing
Call by exception - Path plan :
Get mission duration ETA (s), array length of path L, and refresh period of path array $\triangle$ T = ETA/L
Populate T wit L uniformly distributed time values from 0 to ETA
Create a vector of origin poses P_0 and destination poses P_f = P_0 shifted right 1 position
for a in P_f **do**
    Append x from a pose, position into $posx$, y from a pose, position into $posy$
end for
Initialize playback sequence $n = 0$
Call by exception - Delay prediction:
update D_p
Call by exception - Live pose report:
Obtain current x and y coordinates
Find the index (i) for $posx$ and $posy$ with the nearest values to x and y respectively
Obtain predicted delay D_p from '$Delay\_Prediction$'
Update waiting time for next report in ms T_{n+1} = $T[i]$ + D_p
if ($i_{t-1} > i$ & $i > 0$) **then**
    Raise validation error
end if
$i = i_{t-1}$
Call by period $\triangle$ T ***- Start mission***:
if ($T_0 = 0$) **then**
    Capture real time clock as T_0
end if
Capture current real time clock as T_n
Calculate real time for next confirmation $T_c = T_0 + T_{n+1}$
if ($T_n \leq T_c$) **then**
    Declare bounded certainty
else
    Declare unbounded certainty
end if
Calculate certainty rate ($C_r = |T_c - T_n|/T_{n+1}$)
if ($C_r \geq Cr_{Max}$ & unbounded certainty) **then**
    Stop playback, announce error
end if
Publish $P_f[n]$ as Estimated pose, C as Certainty and C_r as Certainty rate
Increase index ++n
if ($n = L$) **then**
    Reset playback sequence $n = 0$ and start time $T_0 = 0$
end if

Algorithm 2 Delay predictor

Input: RTT delay D (ms), Delay data-set $D_{tr}[:]$ (ms), Data-set size D_s
Output: Delay prediction D_p (ms)
Data collection :
for s in range D_s **do**
    Obtain a RTT delay sample D_n adding the pinging from shop floor to the cloud server and to the web client
    Append D_n into D_{tr}
    Wait the remaining time until next sample (sleep)
end for
Training :
Scale D_{tr} using '$Min - Max$' and export scale as Sc
Separate D_{tr} into 80% training Tr of size Tr_s, 20% test Te of size Te_s
Declare sample window size S_s
Prepare data structure:
Training data input I_{tr} as [Tr_s]x[S_s]x[1]
Test data input I_{te} as [Te_s]x[S_s]x[1]
Training supervisory signal O_{tr} as [Tr_s]
Test supervisory signal O_{te} as [Te_s]
Populate objects:
for x in range (S_s, Tr_s) **do**
    Append $Tr[x - S_s : x, 0]$ into I_{tr}
    Append $Tr[x, 0]$ into O_{tr}
end for
for y in range (S_s, Te_s) **do**
    Append $Te[y - S_s : y, 0]$ into I_{te}
    Append $Te[y, 0]$ into O_{te}
end for
Create a RNN sequential model M (LSTM + Dense)
Train M (fit)
Predict the delays feeding I_{te} into M
Scale the predicted results P_D using Sc
Calculate RMSE comparing P_D and O_{te} and verify M
Initialisation :
Create node '$Latency\_Node$'
Create topic '$Delay\_Prediction$' in node '$Latency\_Node$'
Load pre-trained model 'M' and scale Sc
Prepare the data structure for prediction D_h as [0] x [S_s] x [0]
Call by period 1s - Prediction :
Read RTT delay with AGV D
Append new D as $D_h[0, 0, 0]$ and pop $D_h[0, 4, 0]$
Predict the delay for the next second D_p feeding D_h scaled in M
Publish D_p