Performance Evaluation of Automated Guided Vehicles:
Use Case of using OMNeT++ in a Master’s Thesis*

Kaushik Harihanagarajan, Leo Kröger, Koojana Kuladinithi,
Andreas Timm-Giel

Institute of Communication Networks, Hamburg University of Technology, Germany
[kaushik.harihanagarajan | leo.krueger | koojana.kuladinithi | timm-giel]@tuhh.de

Abstract

OMNeT++ as a discrete event simulator is widely used in combination with the INET framework to simulate communication networks. For students, the limited time frame designated for their master’s thesis is usually a challenge. In particular, because not only an implementation but a scientific contribution is expected as well. Within this paper we present an approach on how OMNeT++ can be used to solve the research question given in the context of a thesis. The exemplary application scenario under consideration here is based on Automated guided vehicles (AGV) which use a WLAN network to exchange information about new obstacles and their localization as such with a central server. As the performance indicators of communication networks are not yet considered while planning such setups, the effect of the network on the operation of the AGVs has to be evaluated. Especially in scenarios with a large number of AGVs the scalability of the WLAN is of interest. We show how the simulation model is parametrized based on a field analysis. With the aid of the experimental data, the different types of traffic are distinguished and modelled according to probability distributions. Based on the scenario under consideration and the envisioned large scale setups, a simulation model is implemented. In the evaluation we present exemplary results. We conclude with an explanation of the grading scheme of master’s theses to highlight how the scientific approach presented is taken into account here.

1 Introduction

The OMNeT++ simulator together with the INET framework is widely used within the research community [9]. In this paper we want to highlight how OMNeT++ can be used in the context of a master’s thesis. Here, one of the main challenges is the limited time frame of usually six months. In addition, students are expected to not only focus on an implementation but should also include a scientific contribution. Based on an exemplary topic we show the approach that can be taken to tackle a research question with the OMNeT++ simulator considering the challenges mentioned.

The main challenge of this work was the modelling of an application scenario in a networking simulator. Our application scenario is based on Automated guided vehicles (AGV), which are mobile robots used in factory environments for varied purposes ranging from material logistics to indoor localization. AGVs use the concept of simultaneous localization and mapping (SLAM) to navigate an indoor environment. In our scenario the AGVs in use are so-called Turtle bots equipped with a Light Detection and Ranging (LIDAR) sensor [2]. On a central server, a map of the environment is continuously updated by the data acquired through the LIDARs on the Turtle bots. In the map, each pixel represents a square with an edge length of 5cm.

*The work presented here was mostly from the Master’s thesis of the first author
Continuous efforts are being taken to improve the localization accuracy and to increase the efficiency, but little care has been taken to evaluate the effects of communication network performance on AGV-based applications operating on a large scale. To the best of our knowledge, Key Performance Indicators (KPIs) of communication networks are not taken into consideration while setting up an AGV-based localization environment. Therefore, this work focuses on modelling of AGV-based applications to evaluate the effects on communication network performance in terms of throughput, round trip time, and packet loss.

AGV-based applications, considered in this work, generate Messaging Queuing Telemetry Transport (MQTT) messages where the traffic pattern depends on the respective scenario. It is very tedious to evaluate a large-scale scenario in an experimental setup, because of higher cost, more time, and resource consumption. Therefore, simulating AGV-based applications is the most feasible way for scalability analysis.

In this paper, we provide a clear insight on how each objective of this master’s thesis is accomplished. There are four main research objectives:

**Analysis of scenarios:** In order to understand the existing setup, a field analysis is done with the aim to study the varied data traffic patterns generated by the AGVs. Sufficient test cases are identified to capture all possible types of traffic.

**Traffic modelling:** Both periodic and event-driven messages are modelled in terms of message length and send interval according to probability distributions to be used in the simulator configuration. A hypothesis test is used to verify that the measured data follows the expected distributions.

**Simulator implementation:** The test setup taken into consideration is then implemented in OMNeT++, a discrete event simulator [8]. Parameters of the access point and Turtle bot radio modules are configured according to the test setup.

**Evaluation:** The Segment latency rate of the MQTT traffic and the UDP packet drop rate are considered as Quality of Service (QoS) constraints. Four scenarios are identified: First, an investigation on the scalability of the existing setup. Second, an assessment on how additional access points can improve the performance for larger numbers of Turtle bots. Third, an evaluation on the impact of MQTT traffic on the other users in the network. Fourth, the performance without considering specific messages.

The scientific approach is twofold here: On the one hand, the student shows the ability to abstract and create traffic models for a simulator. On the other hand, the student conducts a scientific analysis discussing on the reliability and trustworthiness of the results. In short, we show how the OMNeT++ simulator can be used for any master’s or bachelor’s thesis while fulfilling the requirement for the application of scientific methodology.

The paper is organised as follows. We address the aforementioned four objectives of the master’s thesis in subsequent sections. Due to space constraints, we do not cover the complete thesis in detail. Instead, the modelling of only two message types (Object List & Map Update) and two scenarios (scalability analysis & analysis with existing video conferencing traffic) are explained in this paper. At last, we discuss our evaluation criteria to show how we grade a thesis considering implementation work, scientific contribution, writing, discussion, and presentation.

## 2 Understanding of Application Scenario

The chosen field of study is a research testbed in Hamburg for AGV-based indoor localization algorithms using SLAM. Figure 1 shows the floor map of the testbed where Turtle bots are used to track objects and localize the environment. The communication network includes both
wired and wireless communication. The wired communication is based on \textit{IEEE 802.3 ethernet} interface and the wireless communication is realised through \textit{IEEE 802.11(b,g,n) WLAN}.

Data traffic in an \textit{AGV} based localization environment can be classified into two types depending on the frequency and type of generation as in [6]. They are \textit{periodic update} and \textit{event driven} messages.

\textbf{Periodic Update Messages}: Periodic update messages are those messages that are generated periodically at a fixed rate by the Turtle bot.

\textbf{Event Driven Messages}: Event driven messages are those messages that are generated based on the occurrence of a certain event. The Table 1 shows messages generated from the chosen setup:

<table>
<thead>
<tr>
<th>Message</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pose</td>
<td>Periodic update from Turtle bot at 12.5 Hz</td>
<td>Indicate current position to the cloud</td>
</tr>
<tr>
<td>Object list</td>
<td>Periodic update from Turtle bot at 12.5 Hz</td>
<td>Information to cloud about tracked objects</td>
</tr>
<tr>
<td>Map update</td>
<td>Event driven from Turtle bot</td>
<td>Pixel information due to pixel occupation/freeing by semi dynamic objects</td>
</tr>
<tr>
<td>Destination</td>
<td>Event driven from Cloud Turtle bot</td>
<td>Destination information for Turtle bot to go</td>
</tr>
<tr>
<td>Waypoints</td>
<td>Event driven from Turtle bot</td>
<td>Way point information to reach destination set from Cloud</td>
</tr>
</tbody>
</table>

Table 1: Message Types

2.1 Data Sniffing

The characteristics of the above mentioned messages differ in different situations like stationary, movements and detection with obstacles. Therefore, various test cases are considered for this,
with setting up different scenarios like static and mobile Turtle bots to have a clear understanding of its data generation behaviour. It is vital to know how to sniff data and on what basis the data has to be sniffed. With an aid of data sniffed, traffic models are to be generated with which the state of current setup and applications running on them can be explained. One of the data sniffing locations is chosen as the Turtle bot to understand the data generation and other Round Trip Time RTT comprising of wireless transmission, queuing and processing delays.

3 Traffic Modelling

Figure 2 gives a general overview of the traffic modelling procedure. A particular message type is filtered from the data traffic sniffed. The distributions of send interval and payload length are calculated from the filtered data set. The Send Interval shows an Inter Arrival time between two messages of the same type to the sniffing point. The Payload Length is the application layer MQTT payload generated based on the type of message.

![Traffic Modelling - General Overview](image)

Figure 2: Traffic Modelling - General Overview

Modelling of only object list and map update messages are considered here for their variations in frequency of generation and payload.

3.1 Modelling of Object List and Map Update Messages

Though object lists are generated periodically and map updates are event driven, traffic modelling should verify their generation frequency and payload distribution with the Chi-Square goodness of fit test. A clear analysis is made to understand the relevance of payload length chosen with corresponding coherence to number of pixels in map updates and the number of objects tracked in object list messages during an experiment conducted for half an hour.

The send interval of more than 8% of object list messages is found to be greater than 1s as shown in Figure 3. This behaviour is concluded to be a glitch in application programming after further analysis through different test cases. Mathematically they also follow normal distribution with a mean of 1.1263 s and sigma of 0.0017 s for 95% acceptance level, whereas send intervals of map updates cannot be embodied under a single distribution owing to their multiple peaks. Therefore the data points are filtered peak wise and modelled. The traffic contribution of each peak is given in Table 2. Predominantly, peaks follow a normal distribution as verified by the Chi-Square goodness of fit test using [5] and [1].

3.1.1 Payload Length Analysis

The experiment tracked a maximum of four objects as seen in Figure 4 at different points of time. In order to see the payload pattern and understand them by payload parsing, efforts were taken to track at least a single object during the most part of the experiment. After modelling the payload distribution, denoting ’k’ as the number of tracking objects, the payload pattern is given by
Performance Evaluation of AGVs

1. for \( k = 0 \); payload length is constant of 213 bytes with 147 bytes of MQ topic and device information

2. for \( k \) greater than 0; payload length is 147 bytes of MQ topic and device information + \((k*1945)+\text{uniform}(0;1..27)\)

The Table in Figure 4 indicates that for more than 90% of the experiment duration at least an object is tracked. The payload range for objects are in the multiples of 27, indicating 27 different random variables in the object description mathematically described using uniform distribution. In the case of map update messages, up to 23 pixels are carried by them which is a right sample set to describe the payload being chosen. This is shown in Figure 6 with pixels less than 11 appearing in vast quantities than other pixels. The traffic contribution by each pixel during the course of experiment is given by Table in the Figure 6. Mathematically, the payload for map update messages when ‘n’ represents the number of pixels are modelled as

\[
\text{For } n = 1 \text{ to } 23; \text{ payload is MQ topic and device information } + (n*113) + \text{uniform}(0;1...2n)
\]
4 Simulator Implementation

The application layer implementation of the chosen setup can be implemented using the TcpBasicClientApp in INET framework as a template, generating different types of messages. The functionality of cloud in sending destination message is implemented as the TestApp using the TcpGenericServerApp. As shown in Figure 7, every message is generated with the general procedure, but the payload length and the send interval of every message is chosen as per the

<table>
<thead>
<tr>
<th>Peak Number</th>
<th>Distribution</th>
<th>Distribution (%)</th>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Normal</td>
<td>81.2</td>
<td>1.0453</td>
<td>0.0098</td>
</tr>
<tr>
<td>2</td>
<td>Normal</td>
<td>11.5</td>
<td>1.1251</td>
<td>0.0103</td>
</tr>
<tr>
<td>3</td>
<td>Normal</td>
<td>3.6</td>
<td>1.2066</td>
<td>0.0073</td>
</tr>
<tr>
<td>4</td>
<td>Generalized Extreme value</td>
<td>3.1</td>
<td>1.3315</td>
<td>0.1016</td>
</tr>
</tbody>
</table>

Table 2: Send Interval Peak wise - Map Updates

<table>
<thead>
<tr>
<th>Pixel Count</th>
<th>Payload (Bytes)</th>
<th>Distribution (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>385-387</td>
<td>1.74</td>
</tr>
<tr>
<td>2</td>
<td>498-502</td>
<td>6.83</td>
</tr>
<tr>
<td>11</td>
<td>1514-1526</td>
<td>5.03</td>
</tr>
<tr>
<td>12</td>
<td>1693-1717</td>
<td>2.27</td>
</tr>
<tr>
<td>22</td>
<td>2765-2809</td>
<td>0.14</td>
</tr>
<tr>
<td>23</td>
<td>2878-2924</td>
<td>0.14</td>
</tr>
</tbody>
</table>

Table 6: Pixel Count Payload and its distribution
distribution they follow. Probe classes like *Tcp emit scalar* are created to probe various delays like TCP smoothed round trip time delay for every acknowledgement received.

![Diagram of message generation procedure](image)

**Figure 7: Message Generation Procedure**

### 4.1 Network Set-up in OMNeT++

Once the simulation network is setup as shown in Figure 8, simulation parameters such as batch size are decided through *Batch Means method* from multiple long runs. Batch sizes are fixed at 200 s with 30 batches. An ideal warm-up period of 20 s is also selected to make sure all the simulation results are taken in steady state after initial test runs.

### 5 Scenario Identification

The setup under study is a demonstrator and hence identifying how scalable the setup, is one of the essential attribute for future implementations in different application scenarios. Out of various application scenarios mentioned earlier, two major scenarios are considered in this paper. They are

- **Normal Scalability**: Analysing how scalable the chosen setup is using simulations.
- **Scalability with respect to background traffic**: Impact of *MQTT* traffic from AGVs on the background traffic present already. The background traffic is considered as a UDP bidirectional video conferencing.

As Turtle bots generate *MQTT* traffic running over *TCP/IP*, TCP ensures reliability in data transmission by adapting to the congestion in the network and controls the throughput accordingly by adjusting its retransmission timeouts. This is why round trip delays serve as a major constraint in identifying scalability limits in TCP traffic. With UDP as background
traffic, packet drop due to the impact of TCP traffic is a major contributor for QoS outages. With these difficulties the following QoS parameters are considered with respect to ITU-T recommendations as shown in Table 3 and as explained in [7].

<table>
<thead>
<tr>
<th>Medium</th>
<th>Application</th>
<th>One way delay</th>
<th>Packet Drop rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>Video Conferencing</td>
<td>lesser than 150 ms</td>
<td>lesser than 1%</td>
</tr>
<tr>
<td>MQTT Data</td>
<td>Command/control</td>
<td>lesser than 200 ms</td>
<td>Zero</td>
</tr>
</tbody>
</table>

Table 3: Quality of Service constraint recommendations

Based on the recommendations in Table 3, two QoS constraints considered are

- **TCP Segment Latency Rate**: A segment is considered delayed here when the round trip time (RTT) for the segment is greater than 400 ms. In all TCP applications, the Segment latency rate is maintained to 1% on determining the scalability.

\[
\text{Segment latency rate} = \frac{\text{No. of segments with RTT greater than 400 ms}}{\text{Total number of successful transmitted segments}} \times 100
\]

- **UDP Packet Drop Rate**: UDP is suitable for delay sensitive real time applications where delay is a major constraint. A frame is considered latent when the end to end delay for it is greater than 150 ms.

\[
\text{UDP packet drop rate} = \frac{\text{Number of (UDP packets dropped + latent UDP packets)}}{\text{Total number of sent UDP packets}} \times 100
\]

6 Results and Discussions

Every scenario is simulated with the mentioned simulation parameters for different number of Turtle bots. The scenario with only MQTT traffic considers segment latency rate and the one with background UDP data traffic considers UDP packet drop rate as the major constraint in determining the allowable Turtle bot limit. Apart from segment latency rates and UDP packet drop rates simulation results are analysed using average round trip times in transport layer, average number of retries in link layers and other corresponding buffer overflows.

6.1 Normal Scalability

Normal scalability scenario holds good up to 52 Turtle Bots for the considered QoS constraints regarding segment latency rate of 1% (refer Figure 10). Further, Figure 11 shows how link layer average retries increases with increase in the number of Turtle Bots. Average round trip times of Figure 13 in the transport layer attributes the same as in throughput analysis showing a steep increase from 10 to 20 Turtle Bots after which TCP adjusting the congestion window. Beyond this the throughput is maintained constant with an increase in the number of Turtle Bots. The received throughput at the Access Point shows only TCP acknowledgement (refer Figure 12).

6.2 Scalability with background traffic

For a background UDP video conferencing bidirectional traffic (VideoCon1 and VideoCon2), generating a video packet of 1420 bytes every 40 ms, only up to 9 Turtle bots can be supported.
Beyond which, the background traffic is impacted as per ITU-T QoS constraints resulting in real time video packet drops as shown in Figure 14. Further buffer analysis in the link layer showed an interesting behaviour of decreasing link layer retries mainly due to an increase in Turtle bots as in Figure 15. This is due to an increased buffering in access point side reducing the number of retries required. This buffering activity and buffer overflows can be seen in Figure 18 and Figure 19. Remaining behaviours in transport and link layer of Turtle bots hold same as in basic scalability scenario. There was at least a single retry in link layer with no MQTT traffic as both UDP applications are programmed to start at the same time needing a retry.

Based on the above mentioned results, following are the discussions that can be drawn.

1. Care has to be taken in choosing proper radio module implementations
2. Buffers/queues in all nodes are very important as a smaller queue length result in more packet drops and a larger queue length result in more queuing delays
3. Prior analysis on implementation area and background traffic is needed
(a) Knowledge on type and priority of background traffic is needed and Quality of service constraints are to be set accordingly

<table>
<thead>
<tr>
<th>Scalability Scenarios</th>
<th>Description</th>
<th>Number of Turtle bots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic</td>
<td>1 Access point</td>
<td>52</td>
</tr>
<tr>
<td>With background traffic</td>
<td>UDP background traffic, 9 Access point</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 4: Scenario scalability - overview

7 Conclusion

This paper discusses the use of the OMNeT++ simulator in a master’s thesis work. The main challenge was to identify network traffic patterns common in an AGV based application and use that knowledge to evaluate the performance in a larger scenario. As the traffic consist of
Performance Evaluation of AGVs

Hariharanagarajan et al.

![Figure 18: Packet Drops in Access Point due to Queue Overflow](image)

![Figure 19: Maximum Queue Length in Access Point Queues](image)

different patterns (i.e., generation intervals, payload size, etc.), the student had to spend extra time, to understand the actual AGV testbed setup and then, to collect traces to analyze the traffic behaviour. The analysis was done using Chi-Square goodness of fit test. All new traffic models were implemented in INET as MQTT messages to evaluate real-life scenarios. This work was completed within a 6-month period as the student had the necessary background information to perform all tasks successfully. The background information of simulating with OMNeT++, statistical tools and methods (e.g., Chi-Square goodness of fit test, results analysis with confidence intervals, knowledge of WLAN networks, etc.) were taught at our Simulation and Modelling of Communication Networks lecture for master students, where students learn how to model a communication network using OMNeT++ simulator [4].

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Contribution from this work</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Engineering work - 40%</strong></td>
<td></td>
</tr>
<tr>
<td>Implementation work (in simulator/testbed), Reproducing results of an existing work, Different scenario setups,</td>
<td>New traffic modeling in INET, New statistics in INET, Proper understanding of OMNeT++, good coding, Analysis of Wireshark traces, Independent piece of work</td>
</tr>
<tr>
<td><strong>Scientific work - 40%</strong></td>
<td></td>
</tr>
<tr>
<td>Suggesting own ideas, Parameter justifications, Analysis of results using testbeds/simulations/mathematical models, Extended scenarios, Proper analysis</td>
<td>Ability to dig into details, Verification with Chi-Square, 4 scenarios and verifications Own ideas for 3rd &amp; 4th scenarios, Justified - CI, warmup period, simulation runs, etc., Looking at standards for MQTT QoS</td>
</tr>
<tr>
<td><strong>Final presentation &amp; discussion - 20%</strong></td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5: Thesis Evaluation Criteria

We expect our master students to not only focus on pure implementation related work but also to include a scientific contribution in their work. Therefore, we discuss here our evaluation
criteria for a master’s thesis work. As shown in Table 5, both *Engineering work* as well as *Scientific work* are graded equally. The 20% of the grade is weighted based on a student’s ability to present his/her work within a 30-minute period to an audience and another 20-30 minutes of answering questions. The final evaluation of the report and the presentation is done by two professors. The first professor should be an expert in the topic area and should also be involved in the supervision during the work. The second professor should join only the final presentation and his evaluation is based only on the final presentation and the report. We encourage each master student to place a greater emphasis on the scientific contribution in order to obtain a very good grade. The work described here was graded as a very good work. If this work would have addressed more analytical evaluation to show the scalability in a larger scenario capturing the behaviour of MQTT traffic or showing the impact of changing *WLAN* parameters like the AP’s queue length, this could have been graded as an excellent work. Finally, the simulation model will be online [3].

**Acknowledgments:** We appreciate the assistance we received from Christoph Hansen, Tim Brencher and Christoph Reinke from SICK AG, Hamburg, Germany during the data collection phase (from the *AGV* testbed) and for the guidance given to complete this work successfully. This work was carried out under the Erasmus framework of the EPIC project, where students from different universities and different discipline work together to fulfill the objectives of a common project. For the *AGV* focussed project, the TUHH master student analysed the technical aspects of *AGV* based applications, while a bachelor student from Stavanger university, Norway focused on analysing business aspects in deploying *AGV* based applications. Finally, we acknowledge our EPIC partners for all the support we received during this work.

**References**