Reinforcement Learning-based Receiver for Molecular Communication with Mobility

Lisa Y. Debus[‡], Pit Hofmann[*], Jorge Torres Gómez[‡], Frank H.P. Fitzek[*][†], and Falko Dressler[‡]
[‡]School for Electrical Engineering and Computer Science, Technische Universität Berlin, Berlin, Germany
[*]Deutsche Telekom Chair of Communication Networks, Technische Universität Dresden, Dresden, Germany
[†]Centre for Tactile Internet with Human-in-the-Loop (CeTI), Dresden, Germany
debus@campus.tu-berlin.de, {pit.hofmann,frank.fitzek}@tu-dresden.de,
{torres-gomez,dressler}@ccs-labs.org

***Abstract*—Molecular communication (MC) is getting closer to becoming a next-generation communication technology with many applications in life sciences and other industrial applications. Multiple techniques have been proposed on how to design MC receivers depending on the channel characteristics. Experimentally, first testbeds also demonstrate the potentialities for communication using molecules as carriers. In this paper, we focus on developing a reinforcement learning (RL)-based receiver, targeting a realistic scenario with testbed measurements, and addressing transmitter mobility. Leveraging on reported solutions for machine learning (ML) methods, we demonstrate the usability of an RL agent to synchronize the receiver to the received signal. We evidence the learning capabilities of the agent to compensate for the impact of mobility, achieving a low probability of missed detection and small misalignment with the symbol time.**

***Index Terms*—Molecular Communications, Macroscale Molecular Communication Testbeds, Reinforcement Learning, Synchronization**

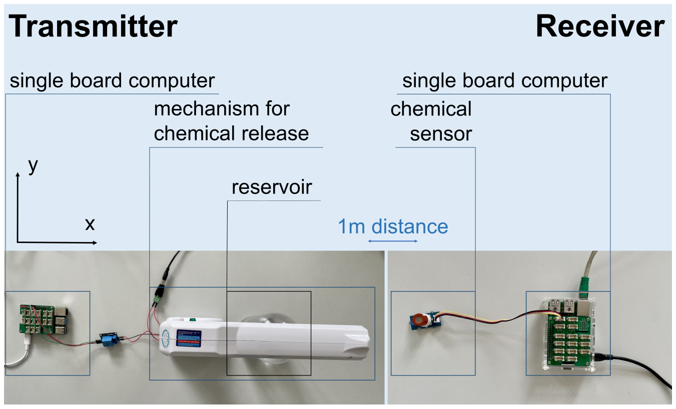

Fig. 1. Molecular SISO communication testbed, see also [18].

I. Introduction

Communication among nanosensors based on molecules still feels like fiction, but today's lab experiments already show that science is overcoming many of the hurdles [1]. In particular, macroscale testbeds demonstrate the capabilities of molecular communication (MC) using information carriers such as nanoparticles like superparamagnetic iron oxide nanoparticless (SPIONs) [2], water droplets [3], or fluorescence proteins [4]. Meanwhile, components like transmitters and receivers for communications are devised with synthetic biology compound [5], [6]. Modified proteins control the opening and closing of gates, allowing the implementation of various modulation schemes like concentration shift keying (CSK).

Using such synthetic components, digital modulation schemes are implemented in MC to connect nodes [7]. For instance, the CSK scheme encodes bits in the amplitude of emitted pulses, where the pulses refer to the concentration level of the released molecules at the transmitter [8]. On the receiver side, pulses are decoded after synchronizing with the received symbols, i.e., the receiver needs to identify the symbol time to decode the message.

To decode the incoming messages, synchronization mechanisms are critical to realizing digital modulation schemes. Symbol timing, frequency, and sampling clock offset result in the main synchronization errors preventing the decoding of binary information. In the case of MC, synchronizing the receiver and the emitter becomes particularly challenging due to random propagation delays and the inherent distortion produced by diffusion processes in MC channels [9]. Optimal receivers operate on the assumption that symbol timing is achieved to decode the received sequence. Furthermore, nanonetworks are inherently mobile, as MC scenarios assume a fluid medium. Due to such mobility, synchronization mechanisms must operate under time-varying conditions, where the MC channel is hard to estimate, or such estimation is just impractical.

At the same time, machine learning (ML) methods have been successfully used in other scenarios to design very flexible receivers, particularly in MC environments [10], [11]. ML models have been reported for symbol detection using recurrent neural networks (RNN) [12] or artificial neural networks (ANN) [13], without implementing channel estimation and equalization techniques. Also, various papers directly target mobile scenarios and report on supervised training of RNN [14] and neural networks (NN) models that additionally perform low pass filtering to reduce the impact of diffusion noise [15], [16]. Most recently, the explainability of the used models also moved into the research focus [17].

Leveraging on the extended use of ML models, in this work, we report implementing a synchronizer using reinforcement learning (RL). With the appropriate data, RL is able to train an agent especially adept at reacting to changing surroundings, which makes it perfect for use in mobile scenarios [19]. In this work, we target mobile scenarios where the transmitter moves according to Brownian motion while the receiver is static (details of the system model are presented in Section II). As a baseline, we use a macroscale MC testbed as depicted in Fig. 1. Although this testbed was developed for static environments (e.g., fixed distance between the sprayer and the sensor), we use it to generate the channel impulse response (CIR) as a spatiotemporal grid by interpolating fixed distance measurements. The spatiotemporal CIR enables modeling the received molecules at the receiver's time-varying positions in an equivalent manner as a mobile scenario. We train an RL model, which is deployed at the receiver, to set the proper threshold level and detect the incoming frame for synchronization. Our results clearly show a high probability for correct frame detection and low symbol-time offset.

II. System Model

As for the system model, we target an MC scenario where the emitter is moving according to Brownian motion and the receiver stays at a fixed position. Such a setting takes place, for example, in monitoring applications, where mobile transmitters report collected data to a stationary gateway node. Using the testbed in Fig. 1, we artificially implement mobility by interpolating the CIR for various distances. In the following, we provide details on the macroscale MC testbed and the used mobility model.

A. Macroscale MC Testbed

We make use of the macroscale single-input single-output (SISO) MC testbed as introduced in our previous work [18]. All used devices and sensors in the testbed setup are commercial off-the-shelf products: The Inoxi Air Atomizer for the sprayer, the MQ-3 Gas Sensor as the chemical sensor at the receiver, and two single board computers (Raspberry Pi) to control the sprayer electronically and for forwarding the electrical signal from the chemical sensor to a local PC.

The transmitter consists of a sprayer for the mechanical release of molecules and a reservoir of Ethanol. We employ on-off keying (OOK) modulation for data transmission while controlling the sprayer electronically. Transmission of a bit-1 activates the sprayer to release molecules (about 3.9×10^{21} in total) and bit-0 accounts for no emissions. The molecules' propagation from the sprayer to the sensor is also subject to additional drift.[1] The receiver consists of an electrochemical sensor measuring the concentration level of molecules with a sampling frequency of $5\,\mathrm{Hz}$.

B. Mobility Model

To introduce mobility into our macroscale MC environment, we consider three components: The emitter's mobility model, the spatiotemporal CIR, and the sampling of the CIR according to the emitter's position to evaluate the total number of received molecules.

[1]To generate drift, we use a commercial off-the-shelf fan, see [18].

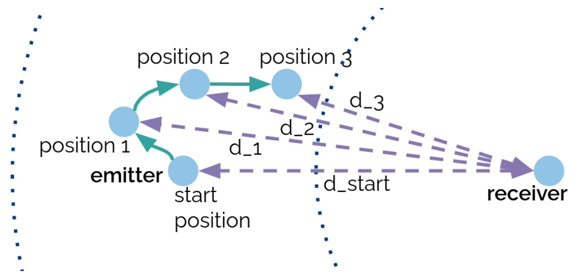

Fig. 2. Mobility model. The emitter moves within a given radius while the receiver remains static. The change in distance results in a change of CIR.

1) Emitter Movement: We simulate mobility in our system by moving the emitter according to Brownian motion. A schematic of the approach is shown in Fig. 2. By changing the position, we change the distance to the receiver and, thus, the CIR of the MC channel. In the Brownian motion model, we set a diffusion coefficient for the emitter, denoted as D_{Tx}, which defines the distance it can move within a given time step as $\delta = \sqrt{2D_{\mathrm{Tx}}\Delta t}$ [9].

2) Spatiotemporal Channel Impulse Response: The random motion produced by the emitter changes the CIR with time. To evaluate the spatiotemporal CIR, we collect testbed measurements for fixed distances of $0.5\,\mathrm{m}$, $0.75\,\mathrm{m}$, $1.0\,\mathrm{m}$, $1.25\,\mathrm{m}$ and $1.5\,\mathrm{m}$ and then interpolate the measured CIR values to approximate the CIR for any distance between the recorded steps.[2] We chose the given distances to cover the range achievable with our testbed in equal steps. For each distance, the number of received molecules over $20\,\mathrm{s}$ was measured 10 times at a rate of 5 measurements per second. We then calculated the average of all measurements per distance. For the interpolation, we used the spline interpolation `interp2` provided in Matlab. The interpolation is performed with a grid relating its step size to the step width possible to be taken by the emitter. Fig. 3 shows the resulting surface of the CIR for a distance range of $0.5 - -1.5\,\mathrm{m}$ and a time interval of $0 - -20\,\mathrm{s}$. This surface renders the spatiotemporal representation of the CIR we use to evaluate the total of received molecules at the receiver.

3) CIR Sampling: Using the spatiotemporal CIR surface in Fig. 3, we sample it with the varying distance between the emitter and the receiver for the given simulation time. Each time the emitter changes its position, we evaluate the distance to the receiver and accordingly read the sample on the surface. We use this reading to calculate the total of the received molecules at the receiver, accounting for the emitter's mobility.

4) Frame emission strategy: The frame emission is integrated into the movement of the emitter in a way that avoids the influence of the Doppler Effect on the performed

[2]The minimum distance of $0.5\,\mathrm{m}$ is due to the propulsive release of the molecules influencing the achievable communication range within the testbed setup in Fig. 1.

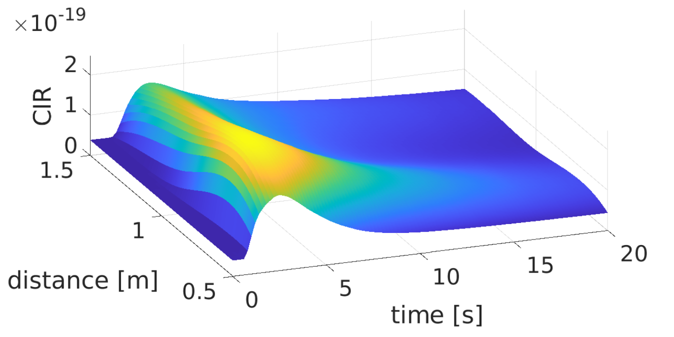

Fig. 3. Average CIR of the testbed created by interpolating the recorded measurements.

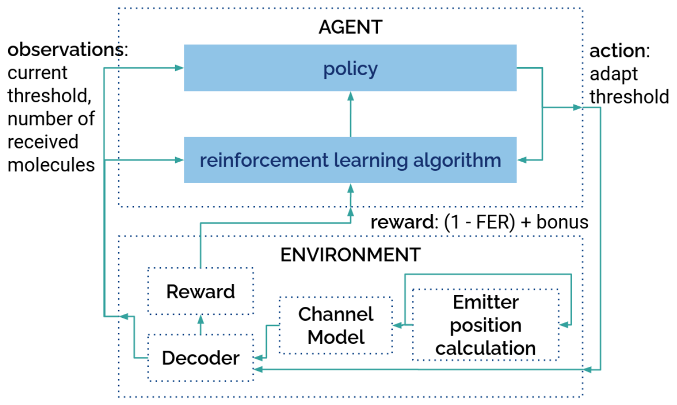
AGENT
observations: current threshold, number of received molecules
policy
reinforcement learning algorithm
action: adapt threshold
reward: (1 - FER) + bonus
ENVIRONMENT
Reward
Channel Model
Emitter position calculation
Decoder

Fig. 4. Training loop used for the training of RL-based threshold setting with a mobile emitter in the presented testbed.

transmissions. This means that the emitter has to stop to perform the transmission of a frame. Only after all molecules for the current synchronization frame are emitted does the emitter start moving again in our scenario.

III. RL-based Synchronizer

To implement and study RL-based threshold setting for our MC receiver, we implemented the setup as an environment in Matlab/Simulink. In the following, we will describe the general structure of the RL procedure, the implemented environment and agent, and we explain the training process and results.

A. Reinforcement Learning-Loop

Fig. 4 shows the training loop we used to train RL-based threshold setting. In each loop, a new emitter position is calculated based on its last position. The distance between the new emitter position and the static receiver is provided to the channel model, where it is used to look up the CIR (based on the measured and interpolated testbed results). Based on this time series, the number of received molecules per symbol (here one bit is encoded in one symbol) is calculated and forwarded to the decoder. The decoder uses the threshold set by the RL agent to decode the number of molecules into a received bit sequence.

The reward is calculated by comparing the decoded bit sequence and the original synchronization frame. It is passed on to the agent, while the number of received molecules and the currently used threshold are forwarded to it in parallel as the observation for the current environment state. Using the reward and the observations, the RL agent then decides how to adapt the threshold for the next run of the training loop and forwards the change of the threshold via its action to the environment. By repeatedly interacting with the environment in this fashion, it learns to set a fitting threshold for the correct decoding of the synchronization frame.

B. Simulink Environment Model

Based on the training loop described in Section III-A, the simulation model implements the testbed as an environment for RL. One execution of the simulation loop implements the transmission of the defined synchronization frame via an MC link. In the environment, the new emitter-receiver distance is calculated based on the emitter's position. With the distance, the CIR for this distance is sampled from the interpolated experiment values (using the surface in Fig. 3). The number of molecules per time step recorded in the channel is then calculated based on the CIR and the number of molecules remaining in the channel from the last transmission. With this value and the threshold set by the RL agent, the transmitted bits are decoded. The values of the decoded bits are compared to the originally transmitted values, and a reward describing the success of the decoding process is produced.

The reward is calculated by comparing the decoded and the source synchronization frame. Using the frame error rate (FER) defined as

$$\text{FER} = \frac{\text{number of incorrectly received frames}}{\text{total number of transmitted frames}}, \tag{1}$$

and a bonus set to

$$\text{bonus} = \begin{cases} 1 & \text{FER} = 0 \\ 0 & \text{FER} \neq 0, \end{cases} \tag{2}$$

we calculate the reward as

$$\text{reward} = (1 - \text{frame error rate}) + \text{bonus}, \tag{3}$$

which resolves to reward $\in [0, 2]$.

By subtracting the FER from 1 in Eq. (3), we get the success rate of the transmission, aiming to reward the Agent with increasing synchronization success. Additionally, we add a bonus of 1 if the FER equals 0. By adding the bonus, we introduce an additional reward, which lets the agent distinguish more easily between right and wrong actions. The calculated reward is forwarded to the agent together with the number of received molecules and the used threshold as an observation. Based on those values, the agent decides the next threshold for the MC link.

To make it possible for the agent to learn to fit the threshold to the emitter mobility accurately, the setting of matching simulation parameters is crucial. In our setup, we use the synchronization frame `[11001]` (as follows from [20]) with

TABLE I
SIMULATION PARAMETERS USED FOR THE TRAINING OF RL-BASED THRESHOLD SETTING WITH A MOBILE EMITTER

Parameter	**Value**
Synchronization frame	`[11001]`
Total number of transmitted frames	700
Bit rate	$0.25\,\mathrm{Hz}$
Emitter start distance	$1\,\mathrm{m}$
Emitter diffusion coefficient	$D_{\mathrm{Tx}} = 8.4 \times 10^{-5}\,\mathrm{m}^2/\mathrm{s}$
Molecule diffusion coefficient	$D = 8.4 \times 10^{-10}\,\mathrm{m}^2/\mathrm{s}$
Total number of emitted molecules	3.9×10^{21}
Sampling rate	$5\,\mathrm{Hz}$

a bit time of $4\,\mathrm{s}$ aiming to reduce the impact of inter-symbol interference (ISI) (as follows from our observations on the CIR). Additionally, important parameters for the simulation environment concern the movement of the emitter. The start distance of the emitter in our setup is set to $1\,\mathrm{m}$. The speed with which it moves is defined by its diffusion coefficient. To make it possible for the agent to learn a fitting threshold, the diffusion coefficient has to be appropriately low so that the agent is able to react to the changed position. Based on our experiments, we were able to determine that a diffusion coefficient of $D_{\mathrm{Tx}} = 8.4 \times 10^{-5}\,\mathrm{m}^2/\mathrm{s}$ or lower would give the agent enough time to react to the changed channel behavior. With this diffusion coefficient, the emitter is able to move up to around $4\,\mathrm{cm}$ between transmissions. The used parameters are shown in Table I.

C. Matlab RL Agent

The RL agent is implemented in Matlab using the RL Toolbox. Based on the received reward, the agent evaluates the current threshold and decides on the necessary change to it based on the observed number of received molecules. For our experiments, we implemented a proximal policy optimization (PPO) agent with an RNN to learn the setting of the threshold with a mobile emitter.

We implemented the PPO RL agent using a `rlPPOAgent` with a `rlValueRepresentation` for the critic and an actor created by a `rlDiscreteCategoricalActor` in Matlab. Both the actor and the critic network include a long short-term memory (LSTM) layer as a recurrent neural network layer, which supports the learning of long-term dependencies. We realized during our experiments that the inclusion of this layer lets the agent adapt more dynamically and accurately to the changing environment.

In order to learn successfully, the observations and actions of the agent have to be defined in a way that they describe the environment accurately and fit together. In our experiments, we pass two observations per loop to the agent. The current threshold and the number of molecules received during the transmission are used to describe the state of the system. The threshold can be set to values between 200 and 2500 in steps of 100 units and is passed to the agent as is. The number

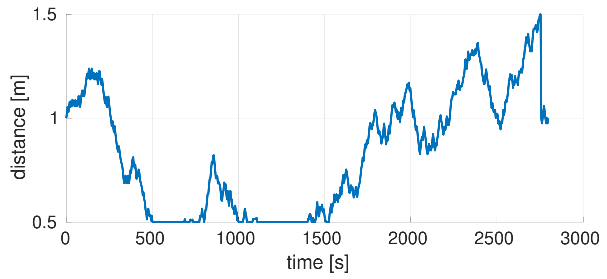

Fig. 5. Distance between the emitter and the receiver over the first 700 transmissions of the simulation.

of received molecules is rounded to the interval between 100 and 2900 in steps of 100 for the observation. The rounding happens because of restrictions on the computation capacity of the machine used for the training.

The agent interacts with the environment via actions. The implemented PPO agent produces action values in the form of integers in the interval $[-8, 8]$. The action chosen by the agent is then multiplied by 100 and added to the current threshold. In each loop, the agent can therefore increase or decrease the threshold by up to 800 molecules. The resulting threshold is then passed as an observation to the agent in the next simulation loop.

D. Training of the RL Agent

The RL agent was trained for 450 episodes in which a total number of 9×10^4 transmissions of the 5-bit synchronization frame was performed at changing distances between the emitter and the receiver. The starting distance and the number of transmissions per episode varied from training episode to training episode, while the agent parameters defining the learning behavior of the agent were left the same.

Fig. 5 shows the changing distance between the emitter and the receiver over the simulation time of 700 times 5-bit transmissions. The emitter's movement shown in Fig. 5 changes the CIR of the MC channel, as we can see in Fig. 6. The figure shows the number of received molecules for the transmitted synchronization frame for three example distances. The number of molecules reaching the receiver side of the system varies significantly. Fig. 6 shows the threshold set by the trained agent for a distance of $1\,\mathrm{m}$, $0.5\,\mathrm{m}$ and $1.5\,\mathrm{m}$. As expected, the threshold varies and is adapted by the RL agent to the changing CIR. These results show the success of the training process, supported by Fig. 7 showing the erroneous decoded bits in the frame of the performed transmissions. While it is obvious, that the FER could still be well improved, it already regularly reaches 0 for several transmitted synchronization frames in a row. The RL agent was therefore trained to set the decoding threshold correctly.

The reliability of the agent setting the correct threshold can be further increased by longer training. As this work only provides a first view on the usability of RL in setting the threshold in MC links with emitter mobility, the training was aborted at this point.

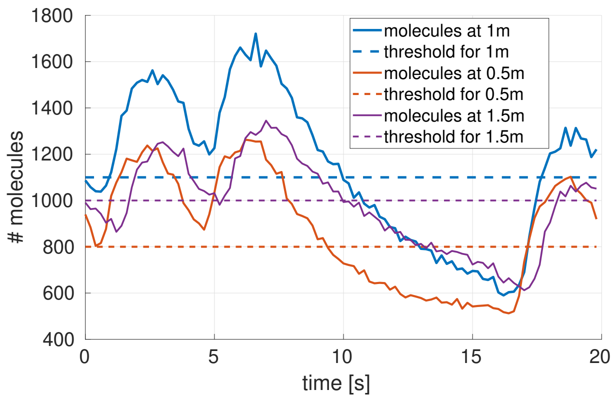

Fig. 6. Number of received molecules over the time of a synchronization frame and the threshold set by the PPO agent for the same transmission for the distances of $1m$, $0.5m$, and $1.5m$.

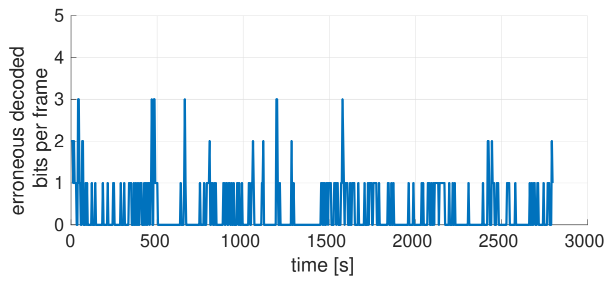

Fig. 7. Number of erroneous decoded bits during the first 700 transmissions.

IV. Results

In the following, we analyze the performance of the RL method for threshold settings. We comparatively evaluate its performance with the filter-based ML synchronizer as an ideal receiver (see [20, Eqs. (7) and (8)]). We plot results for the probability of correct and missed detection of the synchronization frame, as well as the misalignment on the recovered clock signal.

A. Evaluation Setup

We performed our evaluation on a Linux computer (AMD Ryzen 7 3700U, 16 GB RAM) using the trained PPO agent to set the threshold in a simulation of 700 transmissions. The simulation parameters used for the evaluation are listed in Table I.

B. Probability of Missed and Correct Detection of the Synchronization Frame

Averaged over all performed transmissions, the agent achieves a bit error rate (BER) of 0.067. This means that on average, well over $90\,\%$ of the received bits in the frames are decoded correctly. The probability of correctly decoding the sent synchronization frame on the contrary is $72.16\,\%$. The collected values are shown in Fig. 8 with the confidence intervals. The implemented RL scheme is still subject to improvements as upon evaluating the synchronization

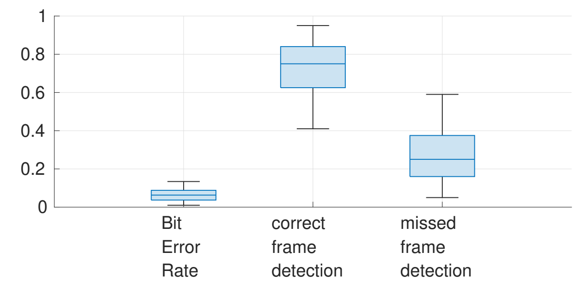

Fig. 8. BER for all transmitted bits and probability of correct or missed detection of synchronization frame achieved by the RL agent in 10^4 transmissions.

capabilities of the filter-based ML, we find that with this method all synchronization frames are detected correctly. This indicates that the system is performing in a high signal to noise ratio (SNR) environment, which may account for the lower FER.

In our results, we see one of the major challenges for correctly setting the threshold for the whole synchronization frame. The emission of molecules for a `1` does not only mean an increase in the number of molecules received for this bit but also influences the number of molecules received over the next bits. In our experiments, we found, that especially the decoding of the `0` in the third position of the used synchronization frame `11001` presents a problem for the threshold setting. The remaining molecules from the first two bits in the current frame and the last bit in the last frame increase the number of molecules received during this bit. They are all `1`s and the ISI with the first `0` in the current frame means that the threshold must be set as high as possible while still decoding all other bits correctly in order to decode the first `0` correctly. This problem is amplified in the case of a bigger distance between the emitter and receiver. As the CIR curve flattens with increasing distance, the ISI increases and leads to even higher numbers of molecules received during the first `0` of a synchronization frame. This can be observed when comparing Fig. 7 showing the FER and Fig. 5 showing the distance between the emitter and the receiver. The smaller the distance, the higher the probability of an FER of 0.

The performance in the probability of correctly detecting the sent synchronization frame is already adequate but shows that some adaptions will still be necessary to improve the performance of this method to a level usable in real-world systems. We can already see its potential, though. The RL agent, even after only this short training, already learned to set the threshold according to the changing dynamics of the system.

C. Misalignment

To see how well the implemented RL agent learns to adapt the threshold to the changing distance between the emitter and the receiver, we consider the misalignment of the synchronization frame. For the misalignment of the synchronization frame, we measure the time offset when the first bit of the sent frame

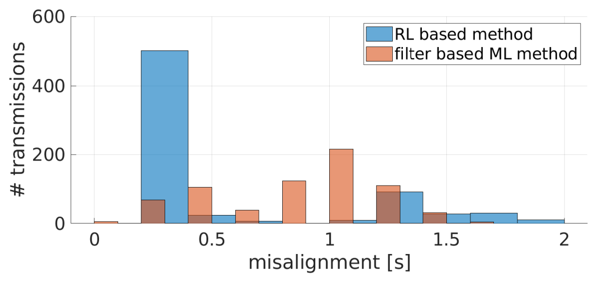

Fig. 9. Histogram of the misalignment of 700 transmissions for the proposed RL-based method and the filter-based ML scheme.

is detected correctly. A histogram showing the misalignment using the proposed RL-based approach compared to the filter-based ML scheme can be seen in Fig. 9.

Compared to the results for the filter-based ML scheme, our approach achieves a lower misalignment. The histogram shows that a majority of the transmissions have a misalignment of $0.25\,\mathrm{s}$, which approximately represents the $6\,\%$ of the bit time of $4\,\mathrm{s}$. This is a very short misalignment and means that in most cases the threshold is set low enough to detect the start of the frame immediately after its transmission starts.

A second spike for the misalignment is located at $1.25\,\mathrm{s}$, which represents $31\,\%$ of the bit time. In CIR measured in the testbed, we observe a steep incline in the number of received molecules at this point. Due to the increased number of molecules reaching the receiver at this point in the transmission, a threshold set to a higher value can be crossed at this point.

Even though it is a good result that most of the transmissions have a very short misalignment, this behavior adds to the problem of miss-detection of the complete synchronization frame. To detect the complete frame correctly, the threshold has to be set high enough to detect the transmitted `0`s correctly. Setting the threshold to slightly higher values where necessary and with this, introducing a longer misalignment might therefore be better for a lower overall FER. A possible future improvement to solve this problem will be discussed in the following section.

D. Further Discussion

The implemented RL agent already shows the potential of using RL for threshold setting in MC links with a changing distance between the emitter and the receiver. However, the system can be further improved with the following adjustments, we plan for future work.

During our experiments, we found that the agent has problems setting the threshold correctly for all values, especially for higher distances between the emitter and receiver. In order to improve upon this, a possible adjustment is the extension of the observations passed to the agent by another measurement. Currently, the observations include the current threshold and the number of detected molecules during the first bit. Because the third bit in the used synchronization frame `11001` posed a major challenge to the setting of the correct threshold in our experiments, the addition of the number of molecules counted during this bit as a third observation could lead the agent to set the threshold more accurately. In this case, the misalignment of the synchronization frame might be increased slightly, but at the same time, the probability of correct frame detection could be increased.

In order to further improve the performance of the presented RL agent we consider transferring the agent to a different framework. While Matlab provides basic implementations of well-known RL algorithms, a more evolved and detailed framework such as Tensorflow could enable a more finely tuned agent. With an RL agent better tuned and better able to react to dynamic environments, we can extend the presented scenario to incorporate different channel models, more complex mobility, and more elaborate modulation schemes. We additionally intend to include the concept of relayed transmissions in our future work.

In our experiments, we trained the agent to detect a synchronization frame correctly. Additionally, as an extension of this process, it would be possible to train the agent not only in detecting the suitable threshold for a whole frame but for a single bit. In preliminary experiments, we found that the agent might be trained to change the threshold for each bit. This agent could then be used not only to detect one synchronization frame for the synchronization process but for the detection of the transmitted information as well.

V. Conclusion

In this work, we have shown the usability of reinforcement learning-based threshold setting for MC links with varying distances between the emitter and the receiver to decode a synchronization frame. We implemented our real-world testbed as a simulated environment with an emitter moving around in this environment based on Brownian motion. The trained agent is able to set the threshold in a way that on average, almost three-quarters of all transmitted synchronization frames can be decoded correctly. Overall, we see in this research the possibilities RL can offer when included in MC. While we pointed out future possible improvements to make RL a possible option in real-life deployments, we can already see its potential in this project.

Acknowledgments

We thank Balkis Abidi for her contribution to this work as part of the research project in the master's program at TU Berlin. This work was supported in part by the project NaBoCom funded by the German Research Foundation (DFG) under grant number DR 639/21-2. This work was also supported by the German Research Foundation (DFG) as part of Germany's Excellence Strategy—EXC 2050/1—Cluster of Excellence "Centre for Tactile Internet with Human-in-the-Loop" (CeTI) of Technische Universität Dresden under project ID 390696704 and the Federal Ministry of Education and Research (BMBF) in the program of "Souverän. Digital. Vernetzt." Joint project 6G-life, grant number 16KISK001K.

References

[1] W. Haselmayr, A. Springer, G. Fischer, C. Alexiou, H. Boche, P. A. Hoeher, F. Dressler, and R. Schober, "Integration of Molecular Communications into Future Generation Wireless Networks," in *1st 6G Wireless Summit*, Levi, Finland: IEEE, Mar. 2019.

[2] M. Bartunik, G. Fischer, and J. Kirchner, "The Development of a Biocompatible Testbed for Molecular Communication with Magnetic Nanoparticles," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, pp. 1–1, 2023.

[3] M. Hamidović, U. Marta, G. Fink, R. Wille, A. Springer, and W. Haselmayr, "Information Encoding in Droplet-Based Microfluidic Systems," in *6th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2019)*, Dublin, Ireland: ACM, Sep. 2019.

[4] L. Brand, M. Scherer, S. Lotter, S. Lotter, A. Burkovski, H. Sticht, K. Castiglione, and R. Schober, "Switchable Signaling Molecules for Media Modulation: Fundamentals, Applications, and Research Directions," arXiv, cs.NI 2302.10356, Feb. 2023. [Online]. Available: https://arxiv.org/pdf/2302.10356.pdf.

[5] C. A. Soldner, E. Socher, V. Jamali, W. Wicke, A. Ahmadzadeh, H.-G. Breitinger, A. Burkovski, K. Castiglione, R. Schober, and H. Sticht, "A Survey of Biological Building Blocks for Synthetic Molecular Communication Systems," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2765–2800, 2020.

[6] M. Kuscu, E. Dinc, B. A. Bilgin, H. Ramezani, and O. B. Akan, "Transmitter and Receiver Architectures for Molecular Communications: A Survey on Physical Design With Modulation, Coding, and Detection Techniques," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1302–1341, Jul. 2019.

[7] M. S. Kuran, H. B. Yilmaz, I. Demirkol, N. Farsad, and A. Goldsmith, "A Survey on Modulation Techniques in Molecular Communication via Diffusion," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 7–28, Jan. 2021.

[8] M. Ş. Kuran, H. B. Yilmaz, T. Tugcu, and I. F. Akyildiz, "Modulation Techniques for Communication via Diffusion in Nanonetworks," in *IEEE International Conference on Communications (ICC 2011)*, Kyoto, Japan, Jun. 2011.

[9] V. Jamali, A. Ahmadzadeh, W. Wicke, A. Noel, and R. Schober, "Channel Modeling for Diffusive Molecular Communication - A Tutorial Review," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1256–1301, Jul. 2019.

[10] A.-A. A. Boulogeorgos, S. E. Trevlakis, S. A. Tegos, V. K. Papanikolaou, and G. K. Karagiannidis, "Machine Learning in Nano-Scale Biomedical Engineering," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 7, no. 1, pp. 10–39, Mar. 2021.

[11] Y. Huang, F. Ji, Z. Wei, M. Wen, and W. Guo, "Signal Detection for Molecular Communication: Model-Based vs. Data-Driven Methods," *IEEE Communications Magazine*, vol. 59, no. 5, pp. 47–53, May 2021.

[12] N. Farsad and A. Goldsmith, "Neural Network Detection of Data Sequences in Communication Systems," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5663–5678, Nov. 2018.

[13] X. Qian and M. Di Renzo, "Receiver Design in Molecular Communications: An Approach Based on Artificial Neural Networks," in *15th IEEE International Symposium on Wireless Communication Systems (ISWCS 2018)*, Lisbon, Portugal: IEEE, Aug. 2018.

[14] N. Farsad and A. Goldsmith, "Detection Over Rapidly Changing Communication Channels Using Deep Learning," in *52nd Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA: IEEE, Oct. 2018.

[15] U. K. Agrawal, A. K. Shrivastava, D. Das, and R. Mahapatra, "Neural Network Detector in Mobile Molecular Communication for Fast Varying Channels," in *International Conference on Connected Systems & Intelligence (CSI 2022)*, Trivandrum, India: IEEE, Aug. 2022.

[16] A. K. Shrivastava, D. Das, and R. Mahapatra, "Performance Evaluation of Mobile Molecular Communication System Using Neural Network Detector," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1776–1779, Aug. 2021.

[17] J. Torres Gómez, P. Hofmann, F. H. P. Fitzek, and F. Dressler, "Explainability of Neural Networks for Symbol Detection in Molecular Communication Channels," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, Jul. 2023, to appear.

[18] P. Hofmann, J. Torres Gómez, F. Dressler, and F. H. P. Fitzek, "Testbed-based Receiver Optimization for SISO Molecular Communication Channels," in *5th IEEE International Balkan Conference Communications and Networking (BalkanCom 2022)*, Sarajevo, Bosnia and Herzegovina: IEEE, Aug. 2022, pp. 120–125.

[19] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of Deep Reinforcement Learning in Communications and Networking: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[20] V. Jamali, A. Ahmadzadeh, and R. Schober, "Symbol Synchronization for Diffusion-Based Molecular Communications," *IEEE Transactions on NanoBioscience*, vol. 16, no. 8, pp. 873–887, Dec. 2017.