Enhanced Precision Time Synchronization for Modular Robots

Jad Bassil
Univ. Bourgogne Franche-Comté
FEMTO-ST Institute, CNRS, France
jad.bassil@femto-st.fr

Benoît Piranda
Univ. Bourgogne Franche-Comté ,
FEMTO-ST Institute, CNRS, France
benoit.piranda@femto-st.fr

Abdallah Makhoul
Univ. Bourgogne Franche-Comté ,
FEMTO-ST Institute, CNRS, France
abdallah.makhoul@femto-st.fr

Julien Bourgeois
Univ. Bourgogne Franche-Comté ,
FEMTO-ST Institute, CNRS, France
julien.bourgeois@femto-st.fr

***Abstract*—As in all distributed systems, having an accurate access to a global notion of time is vital for a modular robot's modules to coordinate their activities and accomplish their goal. In this paper, we present and compare two methods for clock skew compensation to enhance the performance of network-wide time synchronization in modular robots with neighbor-to-neighbor communication. The first one, Adaptive Rate Search (ARS), uses a light weight adaptive search method to adapt the drift rate of local clocks. The second one, combines Linear Regression with Bayes estimation (LR+) to reduce the accumulative error induced during the propagation of synchronization messages on large number of hops. We evaluate both methods with *Blinky Block* robots: using a real modular robots system and simulation. The results show that both methods LR+ and ARS present a significant error reduction compared to least-square linear regression used in previous state of the art synchronization protocol for modular robots with neighbor-to-neighbor communication.**

***Index Terms*—time synchronization, modular robots, distributed algorithms**

I. INTRODUCTION

An autonomous modular self-reconfigurable robot is a distributed system composed of multiple connected modules that can communicate to coordinate tasks and rearrange their connections to change the overall shape of the system to adapt to a specific application providing versatility and robustness [1]. Each module has its own sensors and actuators, computation and communication capabilities and notion of time. In this paper, we consider homogeneous modular robots with identical modules that can communicate only with their directly connected neighbors.

Many applications with distributed control require having the same notion of a global time at each module. It is required for various operations and applications in modular robots (e.g. self-reconfiguration [2, 3], clustering [4], security [5], etc.). For instance, a self-reconfiguration algorithm in a 2D plane is proposed in [6]. In this algorithm, modules reconfigure their shape by forming lines that are translated from the initial shape to target positions in the goal shape. Another interesting application is modular shape-changing user interfaces using modular robots [7] where modules making up the interface are required to be synchronized to efficiently handle interactions between humans and the interface.

Unfortunately, local hardware clocks are insufficient to achieve a global notion of time since the accuracy of hardware clocks is affected by environmental conditions such as voltage, temperature, aging, etc. So, they tend to drift apart quasi-linearly over time even in a perfect environment. Therefore, each module needs to have an estimation of a common global time determined by a distributed time synchronization protocol.

Time synchronization algorithms and protocols have been proposed for computer networks [8, 9, 10, 11]. In addition time synchronization has been widely studied in wireless sensor networks (WSN) that form peer-to-peer networks for resource constrained devices similarly to modular robots. Many time synchronization protocols have been proposed for WSNs such as [12, 13, 14, 15]. As for modular robots, for the best of our knowledge, the Modular Robot Time Protocol (MRTP) [16] is the only protocol that has been proposed for time synchronization in modular robots with neighbor-to-neighbor communication. It is the most adapted protocol for large modular robotic systems with low precision hardware clocks that use low-bitrate neighbor-to-neighbor communication and can form networks with large diameters. It aims to achieve network-wide tight synchronization by keeping a small offset between any local clock and a global reference clock situated at a reference module. The reference module periodically sends a synchronization message that propagates hop-by-hop through a breadth-first communication tree to all modules using a predictive method to compensate for communication delays. Modules compensate for clock skew with least-square linear regression on a window of fixed size containing previously received synchronization points. However, the performance of MRTP decreases when the diameter of the modular robot network increases. The reason is that a small error induced by non-deterministic delays that can be looked at as the distance from each synchronization point to the best-fit line is accumulated which can reduce the efficiency of least-square linear regression on large scale modular robots with a network that spans on large number of hops.

In this paper, we present and compare two new methods to compensate for clock skew in the aim of enhancing the performance of MRTP on modular robot networks with large diameter. The first, Adaptive Rate Search (ARS) is a lightweight dynamic search method that can provide at any time the adapted rate of a module's clock by adjusting its provided rate value according to the received global time of the reference module. The second, Enhanced Linear Regression (LR+), uses a Bayesian approach to reduce the uncertainty error induced at each hop along the synchronization path by recursively combining knowledge of a module's parent with a module's local knowledge to calculate an improved estimation of the global time.

II. SYSTEM MODEL

Each module u is equipped with a read-only hardware clock H_u. At any time t, H_u can be modeled as:

$$H_u(t) = \int_0^t h_u(\tau)\, d\tau$$

Where $h_u(\tau)$ denotes the hardware clock rate of u at τ. We denote as $G(t)$ the global time across the system at time t. $G(t)$ is determined as the hardware clock value of a reference module chosen by the synchronization protocol. Each module u maintains a logical clock that provides the value of $G_u(t)$: an estimation of $G(t)$ determined by the synchronization algorithm and the skew compensation method. The rate h is not stable so modules clocks tend to drift apart. The objective is to synchronize modules clocks using skew compensation methods to maintain at each module u a minimal relative synchronization error at real time t: $\epsilon_u(t) = G_u(t) - G(t)$.

III. ADAPTIVE RATE SEARCH

Adaptive Rate Search (ARS), inspired from [17, 15] is a search method that aims to track and find a dynamic search value in a given search space which is a real interval $[r_{min}, r_{max}] \subset \mathbb{R}$. An ARS proposes a value to the environment in which it is implemented, then the environment sends back a feedback that guides the ARS towards the searched value. In the synchronization context, since hardware clocks rates tend to drift apart in a quasi-linear fashion, the ARS is implemented on each module to adapt the speed rate of its logical clock to reduce the module's relative synchronization error ϵ_u. The logical clock of a module u can be modeled as:

$$G_u(t) = G_u(t_{up}) + (1 + r_u(t)) \times (H_u(t) - H_u(t_{up}))$$

Where t_{up} is the last time a synchronization message was received from u's parent and r_u is the value of the adjustment of the rate of the logical clock provided by ARS$_u$.

The objective of ARS is to provide at any real time t the adapted value of the logical clock rate $r_u(t)$ of the module in which it is implemented. In order to adapt the value of ARS, at each synchronization message reception, a module u (The ARS$_u$ environment) will calculate its clock skew by subtracting its clock estimate G_u from the clock value G_p received from its parent p. Then, send a feedback f_t from the set $\mathcal{F} = \{f \uparrow, f \downarrow, f \approx\}$ to ARS_u. If the skew is negative, the ARS$_u$ will receive the increase feedback $f \uparrow$ and forwards its value $r(t)$ a certain adjustment step $\Delta \in [\Delta_{min}, \Delta_{max}] \subset [0, |r_{min}, r_{max}|]$ accelerating the clock rate of its logical clock G_u. Similarly, if the skew is positive, module u sends a decrease feedback $f \downarrow$ so ARS$_u$ can move back its value to decelerate the logical clock rate. Otherwise, if the skew is null, module u sends the feedback $f \approx$ and no adjustment is made. Then, the logical clock G_u value is set to the one received G_p.

In order to make the search process more efficient, if an ARS receives successive feedback in the same direction, the adjustment is accelerated by increasing the adjustment step by a factor of λ_{incr}. Conversely, when the ARS receives successive feedback in opposite directions, the ARS value is oscillating around the target value noted as r_u^*. Therefore, the adjustment step size is decreased by a factor of λ_{decr} to get closer to the target value.
As shown in [15], if $\lambda_{incr} > 1$ and $\lambda_{decr} = 1/(1 + \lambda_{incr})$, an ARS will always reach the target value r_u^* for any initial value r_0.

IV. ENHANCED LINEAR REGRESSION

Enhanced Linear Regression (LR+) was first proposed in [18], it combines least-square linear regression with Bayes estimation. Using least-square linear regression, as in MRTP, we can model the logical clock of a module u at any time t as follow: $G_u(t) = a_u(W_t) \times H_u(t) + b_u(W_t)$, where $a_u(W_t)$ and $b_u(W_t)$ are the coefficient of the best-fit line determined by linear regression calculated on a window W_t containing the last w synchronization points. They respectively represent the estimated skew and offset relative to the global time $G(t)$. However, non-deterministic delays such as the instability of clock frequency and synchronization message delivery delay can induce an error that propagates and increases hop by hop along the synchronization path which negatively affect the performance of the synchronization algorithm especially in a modular robot network with neighbor-to-neighbor communication and a long multi-hop synchronization path.

The non-deterministic error can be looked at as the distance from each synchronization point to the best-fit line of regression. It can be described as a Gaussian distribution with zero mean and a known standard deviation $\sim N(0, \sigma^2)$. Consequently, the logical clock value at a module u comes from a Gaussian distribution with a mean equal to the global time G: $p(G_u|G) \sim N(G, \sigma_u^2)$.
Therefore, Bayes theorem can be applied to improve the estimation of the global time G: $p(G|G_u) \sim N(G'_u, \sigma'^2_u)$.

The prior knowledge of module u's parent $p \sim N(G'_p, \sigma'^2_p)$, the local estimation of the global time G_u and the uncertainty of u's global time estimation $\sim N(0, \sigma_u^2)$ are combined together using Bayes theorem to improve the global time estimation at module u as follow:

$$\sigma_u^2 = \frac{\sigma_p'^2 \sigma_u^2}{\sigma_p'^2 + \sigma_u^2} \text{ and } G_u' = \frac{\sigma_u^2}{\sigma_p'^2 + \sigma_u^2} G_p' + \frac{\sigma_p'^2}{\sigma_p'^2 + \sigma_u^2} G_u$$

Initially, at each synchronization round, $G_r' = G_r = G$ and $\sigma_r = \sigma_r' = 1$ where r is the reference module. Then, at each module u, the improved estimation of global time G_u' is calculated and is sent to its children. Therefore, at each hop, modules are recursively synchronized with the improved estimated global time received from their parents until the leafs modules are reached. Hence, reducing the dissemination error induced from measurements uncertainty accumulated at each hop on the the synchronization path.

V. COMPLEXITY COMPARISONS

In this section, we compare the processing and memory complexity on each module of the three presented skew compensation methods.

The least-square linear regression (LR) method originally used in MRTP needs to store w pairs of previously received synchronization points at each module.

The adaptive rate search method (ARS) needs to store two values: the multiplier rate and the adjustment step.

The enhanced Linear Regression method LR+ uses at least the same memory space as LR: w pairs. We can use additional number n of synchronization records, apart from the ones used for LR, to calculate the uncertainty σ^2 and the improved global time estimation (see Section VI-D). As for the number of operations needed, LR+ needs to calculate the global time estimation, the uncertainty σ^2 and the improved global time estimation.

Overall, ARS is the lightest method with $O(1)$ storage and processing complexities. The storage and processing complexities of LR and LR+ depend on the window size used to calculate the global time estimation. Hence, they have a complexity of $O(w)$.

VI. EVALUATION METHOD AND RESULTS

In this section, we evaluate the improvement that ARS and LR+ can yield comparing to least-square linear regression used in MRTP.

A. Hardware System

We implemented both methods LR+ and ARS on top of MRTP on the *Blinky Blocks* V1 hardware using C language [19]. A *Blinky Block* can be connected to up to 6 neighbors via magnets and can communicate through serial interfaces on each face. They have a poor hardware clock resolution of 1m and an accuracy of 10 000 ppm (parts per million).

B. Evaluation Method

In order to evaluate ARS and LR+, we alter the MRTP protocol to use them as skew compensation methods in place of the originally used least-square linear regression method. The configuration of MRTP including the synchronization period and the number of synchronization points for linear regression where studied in [16]. They show that an optimal performance on *Blinky Blocks* can be obtained using a window of 5 synchronization points and a synchronization period of 2 seconds during calibration phase: the phase until the window of synchronization points is filled, and 5 seconds during runtime. We use the same configuration for our experiments while varying the compensation methods between least-square linear regression (LR) originally used in MRTP and the two previously presented methods: ARS and LR+.

Parameter	r_{min}	r_{max}	r_0	λ_{incr}
Value	-10^{-2}	10^{-2}	0	2

Parameter	λ_{decr}	Δ_0	Δ_{min}	Δ_{max}
Value	1/3	10^{-6}	10^{-10}	10^{-3}

TABLE I: ARS parameters

To calculate the dissemination error, we replayed the experiment depicted in Figure 1 used in [16]. It consists of deploying two virtual modules on each physical block except the last in line. The leaf module M_{2n-1} is hosted on the same physical block as the time reference module $TM = M_1$ so it can access the actual global time which allows us to compute the global dissemination error $G_{M_{2n-1}}(t) - G(t)$ at distance $2(n-1)$ hops. In addition, since the *Blinky Blocks* uses neighbor-to-neighbor communication, it is impossible to access the states of all modules at the same time. So, we conducted simulations to evaluate network-wide synchronization error.

C. ARS Parameters

We initialize ARS parameters on each module with the values shown in table I. The optimal values for λ_{incr} and λ_{decr} are set respectively as demonstrated in [15].

D. Impact of the Number of Synchronization Points Used for Enhanced Linear Regression

In this section we evaluate the impact of the number of synchronization points used by LR+ to calculate the improved estimation of the global time on the dissemination error. LR+ needs to store a window of previously received synchronization points to calculate the enhanced estimation of global time $G_u'(t)$ at a module u at each synchronization round. The optimal window size used in MRTP to calculate $G_u(t)$ is 5 as mentioned in Section VI-B. For LR+, the same number of synchronization points can be used to calculate $G_u'(t)$. However, Figure 2 shows that using additional number of synchronization points reduces the dissemination error. Hence, for the next sections, we choose 20 as the window size for storing the last received synchronization points.

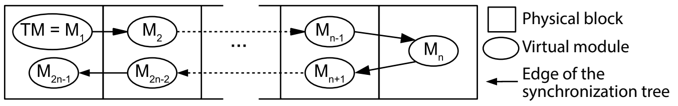

Fig. 1: Virtual line of emulated module on *Blinky Blocks* hardware [16]

E. Impact of Skew Compensation Method on Dissemination Error

The impact of hop distance on global dissemination error for several number of hops is shown in Figure 3. The error distribution seems Gaussian and the precision decreases when the number of hops increases. LR+ provides a better dissemination error comparing to LR and ARS. The dissemination error using ARS is approximately equal to the one using LR but its distribution is less spread. Therefore, both ARS and LR+ are better than LR but using LR+ we can achieve better precision in terms of difference between reference module clock and other modules' clocks since it presents the smaller dissemination error.

F. Impact of Skew Compensation Method on Network-Wide Synchronization Error

To evaluate the network-wide synchronization error, we use *VisibleSim* [20], a discrete-event simulator for modular robots systems. We conducted different experiments on a ball topology [21] with different diameter. All simulations last two hours. Modules remain unsynchronized during the first hour and the synchronization starts at the beginning of the second hour. The reference module is placed at the center of the topology. We used the same configuration for MRTP and the same clock model described in [19]. Each 5 seconds the maximum pairwise synchronization error is recorded.

Figure 4 shows the maximum pairwise synchronization error in the first 15 min after synchronization starts at the beginning of the second hour of simulation until the stabilization of ARS. Figure 5 shows the the maximum pairwise synchronization error for the rest of the simulation. During the first hour modules remain unsynchronized. Compared to ARS and LR+, it is clear that the LR method present the worst precision that decreases more significantly when the diameter of the system increases. Both ARS and LR+ present better precision than LR. ARS requires more time to stabilize than LR+. The reason is that, after the first hour the skew between modules clocks become large, the closest modules to the reference module receive more accurate time information and send an accurate feedback to their ARS. Hence, synchronization using ARS is achieved hop-by-hop. Therefore, both ARS and LR+ are superior to LR in providing network-wide tight

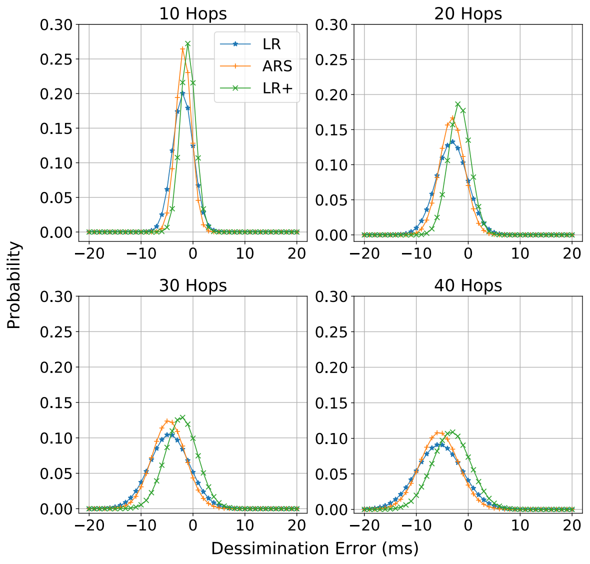

Fig. 3: Impact of number of hops on global time dissemination error.

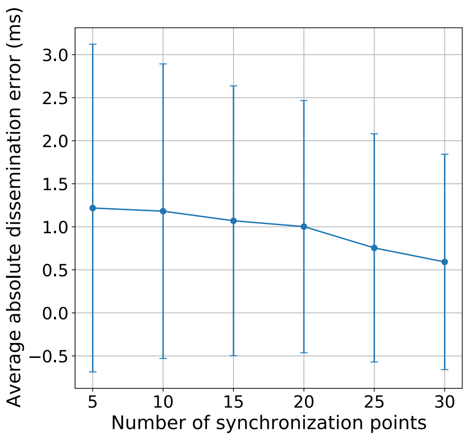

Fig. 2: Impact of Window Size on the Dissemination Error of LR+ on 10 Hops

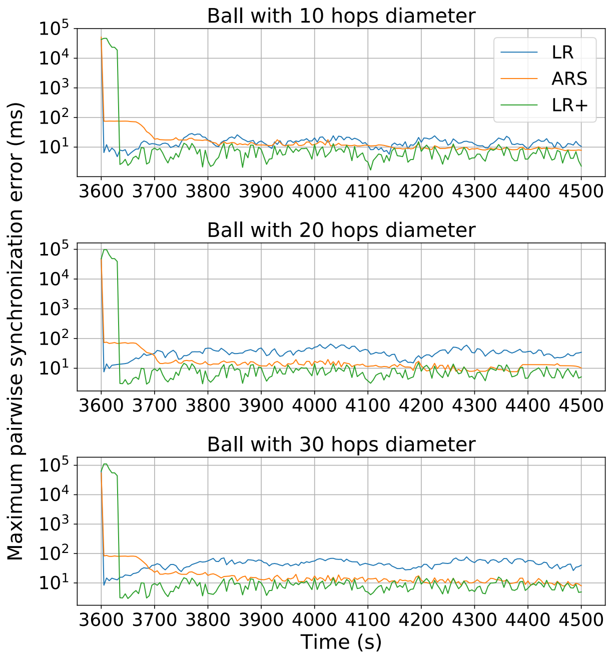

Fig. 4: Maximum pairwise synchronization error in the first 15 minutes after synchronization starts.

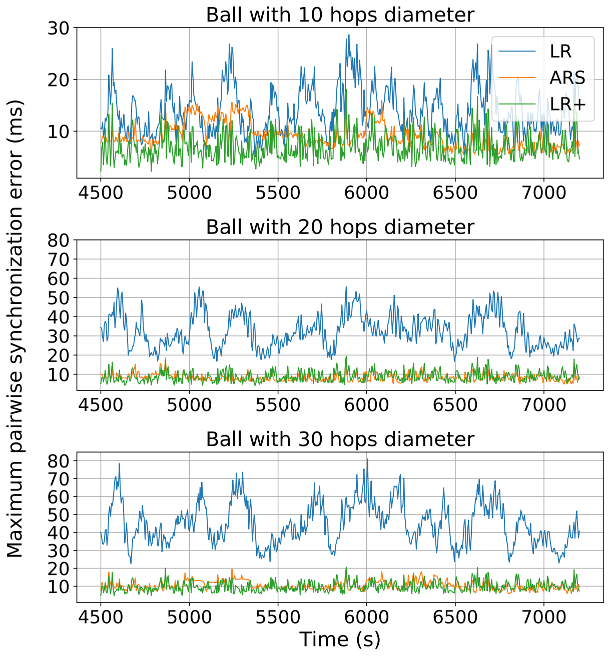

Fig. 5: Maximum pairwise synchronization error after ARS stabilization.

synchronization especially when the modular robot network becomes large. Once stabilized, ARS can provide an equal performance to LR+ in a very light-weight manner. Using LR+, we can directly achieve tight synchronization but using more resources. We can choose the best method according to the application.

VII. Conclusion

In this paper, we presented ARS and LR+, two skew compensation methods to enhance the performance of synchronization protocol for large-scale modular robots with neighbor-to-neighbor communication. ARS uses a light-weight adaptive search method. LR+ combines least-square linear regression with Bayes estimation to reduce the error accumulated on the synchronization path. We showed on real hardware that both methods provide a better dissemination error and in simulation that the network-wide synchronization error can be significantly decreased using both ARS or LR+.
In future work, we will test the performance of both methods on other systems equipped with more accurate hardware clocks. In addition, we aim to study a new method combining LR+ and ARS, i.e. using LR+ in the first synchronization rounds until modules' clocks become synchronized then switch to ARS for more optimized network-wide synchronization.

Acknowledgment

This work has been supported by the EIPHI Graduate School (contract ANR-17-EURE-0002).

References

[1] M. Yim, W. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, and G. S. Chirikjian, "Modular self-reconfigurable robot systems [grand challenges of robotics]," *IEEE Robotics Automation Magazine*, vol. 14, no. 1, pp. 43–52, 2007.

[2] H. A. Akitaya, E. M. Arkin, M. Damian, E. D. Demaine, V. Dujmović, R. Flatland, M. Korman, B. Palop, I. Parada, A. van Renssen *et al.*, "Universal reconfiguration of facet-connected modular robots by pivots: the o (1) musketeers," *Algorithmica*, pp. 1–36, 2021.

[3] P. Thalamy, B. Piranda, and J. Bourgeois, "3D coating self-assembly for modular robotic scaffolds," *IEEE International Conference on Intelligent Robots and Systems*, pp. 11 688–11 695, 2020.

[4] J. Bassil, M. Moussa, A. Makhoul, B. Piranda, and J. Bourgeois, "Linear distributed clustering algorithm for modular robots based programmable matter," 2020, pp. 3320–3325.

[5] E. Hourany, B. Habib, C. Camille, A. Makhoul, B.Piranda, and J. Bourgeois, "Prolisean: A new security protocol for programmable matter," *ACM Trans. on Internet Techno. (TOIT)*, vol. 21, no. 1, pp. 1–29, 2021.

[6] B. Piranda and J. Bourgeois, "A distributed algorithm for reconfiguration of lattice-based modular self-reconfigurable robots," in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, 2016, pp. 1–9.

[7] L. Pruszko, C. Coutrix, Y. Laurillau, B. Piranda, and J. Bourgeois, "Molecular hci: Structuring the cross-disciplinary space of modular shape-changing user interfaces," *Proc. ACM Hum.-Comput. Interact.*, vol. 5, no. EICS, May 2021.

[8] R. Gusella and S. Zatti, "The accuracy of the clock synchronization achieved by tempo in berkeley unix 4.3bsd," *IEEE Transactions on Software Engineering*, vol. 15, no. 7, pp. 847–853, 1989.

[9] F. Cristian, "Probabilistic clock synchronization," *Distributed Computing*, vol. 3, no. 3, p. 146–158, Sep. 1989.

[10] D. Mills, "Internet time synchronization: the network time protocol," *IEEE Trans. on Communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[11] IEEE, "Ieee standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pp. 1–300, 2008.

[12] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. New York, NY, USA: Association for Computing Machinery, 2004, p. 39–49.

[13] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, p. 147–163, Dec. 2003.

[14] C. Lenzen, P. Sommer, and R. Wattenhofer, "Pulsesync: An efficient and scalable clock synchronization protocol," *IEEE/ACM Transactions on Networking*, vol. 23, no. 3, pp. 717–727, 2014.

[15] K. S. Yildirim and Ö. Gürcan, "Efficient time synchronization in a wireless sensor network by adaptive value tracking," *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3650–3664, 2014.

[16] A. Naz, B. Piranda, S. C. Goldstein, and J. Bourgeois, "A time synchronization protocol for modular robots," in *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, 2016, pp. 109–118.

[17] L. Sylvain, C. Valérie, and G. Pierre, "Principles and properties of a MAS learning algorithm: A comparison with standard learning algorithms applied to implicit feedback assessment," *Proceedings - 2011 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2011*, vol. 2, pp. 228–235, 2011.

[18] Q. Gao, K. J. Blow, and D. J. Holding, "Simple algorithm for improving time synchronisation in wireless sensor networks," *Electronics Letters*, vol. 40, no. 14, pp. 889–891, 2004.

[19] A. Naz, B. Piranda, J. Bourgeois, and S. C. Goldstein, "A time synchronization protocol for large-scale distributed embedded systems with low-precision clocks and neighbor-to-neighbor communications," *Journal of Network and Computer Appli.*, vol. 105, pp. 123–142, 2018.

[20] P. Thalamy, B. Piranda, A. Naz, and J. Bourgeois, "Visiblesim: A behavioral simulation framework for lattice modular robots," in *15th International Symposium on Distributed Autonomous Robotic Systems*. Jun, 2021.

[21] A. Naz, B. Piranda, T. Tucci, S. Copen Goldstein, and J. Bourgeois, *Network Characterization of Lattice-Based Modular Robots with Neighbor-to-Neighbor Communications*. Cham: Springer International Publishing, 2018, pp. 415–429.