CF '18, May 8–10, 2018, Ischia, Italy

Distributed Learning-Based State Prediction for Multi-Agent Systems with Reduced Communication Effort

Daniel Hinkelmann
RWTH Aachen University
Research Area ISEK
Aachen, Germany
daniel.hinkelmann@rwth-aachen.de

Anke Schmeink
RWTH Aachen University
Research Area ISEK
Aachen, Germany
anke.schmeink@rwth-aachen.de

Guido Dartmann
University of Applied Sciences Trier,
Institute of Software Systems
Trier, Germany
g.dartmann@umwelt-campus.de

ABSTRACT

A novel distributed event-triggered communication for multi-agent systems is presented. Each agent predicts its future states via an artificial neural network, where the prediction is solely based on own past states. The approach is therefore scalable with the number of agents. A communication is triggered if the discrepancy between actual and predicted state exceeds a threshold. Numerical results show that this approach reduces the communication effort remarkably compared to existing methods.

KEYWORDS

Multi-agent systems, event-triggered communication, machine learning, artificial neural networks, formation control, consensus

ACM Reference Format:
Daniel Hinkelmann, Anke Schmeink, and Guido Dartmann. 2018. Distributed Learning-Based State Prediction for Multi-Agent Systems with Reduced Communication Effort. In *CF '18: CF '18: Computing Frontiers Conference, May 8–10, 2018, Ischia, Italy.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3203217.3203230

1 INTRODUCTION

In multi-agent systems, distributed agents cooperate in order to accomplish tasks that are beyond the capabilities of each individual. An agent can be defined as an entity which is able to sense and act in an environment. Compared to a single complex agent, a whole system of simple agents is often more robust, scalable and cost efficient. Applications can be found in areas such as robotics and transportation systems. Another example is the formation control for micro satellite swarms, see Figure 1. The technology of those swarms has made significant progress recently, and installations are already possible [1]. Agents are often equipped with low computing power and capability-limited communication modules. Due to a large number of agents and limited communication ranges, a centralized controller is considered as unpractical. Thus, agents have to be controlled decentralized.

CF '18, May 8–10, 2018, Ischia, Italy

ACM ISBN 978-1-4503-5761-6/18/05…$15.00
https://doi.org/10.1145/3203217.3203230

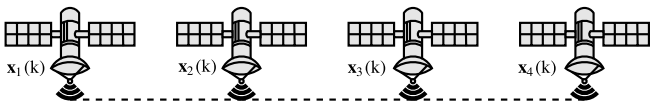

Figure 1: Satellites in a formation flight with a line communication topology.

Group tasks can often be formulated as consensus problems. In order to reach consensus, see [9], agents have to exchange information. The communication topology defines which agents can talk to each other. Based on information transmitted from neighbors, an agent can update its state.

Communication over wireless networks offers flexibility and thus, is used in many applications. Current topics such as Industry 4.0 and the Internet of Things (IoT) introduce a significant number of agents to wireless communication networks. The challenge with an increasing number of users lies in the limited communication resources. How limited communication resources effect consensus problems has been investigated in [3]. In order to save communication resources and energy, communication needs to be efficient.

The communication triggering condition needs to minimize the number of transmissions and also leads to the completion of the group task. The trade-off of a reduced communication is in general a consensus error and slower convergence.

The communication can be triggered periodically. Disadvantage of this method is an unnecessary exchange of information if the agent's state has not changed. Alternatively, it may be event-triggered. An event is triggered whenever the discrepancy between the agent's current and last broadcasted state exceeds a threshold. As shown in [4, 7, 8, 10], an event-triggered approach decreases the number of transmissions.

In biological groups, e.g., swarms, individuals have predictive mechanisms. Based on bio-inspired prediction mechanisms, [15–17] presented a decentralized predictive consensus protocol which reduces communication costs. An event-trigger where each agent estimates all other agents' states has been proposed in [12, 13].

Contribution: In this paper, we propose a novel distributed and event-triggered communication method. Solely on its past states, an agent predicts its future states, independently of its neighbors. This supports independence of negative scaling effects with an increasing number of agents. As a prediction mechanism, we propose an artificial neural network. A communication is triggered if the discrepancy between actual and predicted state exceeds a threshold.

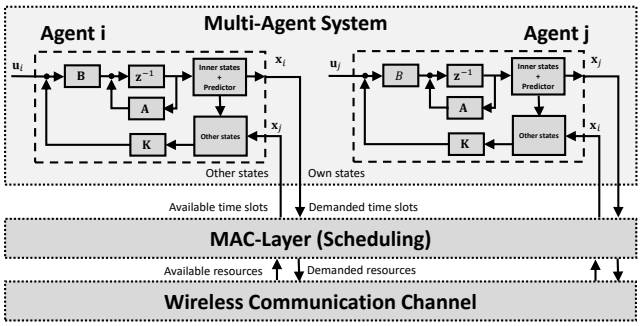

Figure 2: Scenario: Communication layers and system model

When an agent broadcasts, it transmits not only its current state but also its predictions. Thus, in future time instants, neighbors can update their state using those predictions and consequently the communication frequency reduces.

Paper structure: Section 2 contains the theoretical foundations of consensus problems, communication trigger conditions and prediction mechanisms. In Section 3, numerical results for different communication trigger conditions on the example of a formation control are presented. As predictive mechanisms, linear extrapolation and artificial neural networks are compared. Section 4 concludes the paper.

2 PRELIMINARIES

Consider a multi-agent system with an agent as a node $n_i \in V$ in a communication graph $G = (V, E)$. We denote by V the set of all nodes in the system $V = \{1, 2, ..., n\}$ and by E the set of all edges in the multi-agent system $E \subseteq V \times V$. The neighbors of agent i are denoted by $N_i = \{j \in V : (i, j) \in E\}$. If agent i with node n_i communicates with an agent j with node n_j, we call the corresponding connection edge $e_{ij} \in E$ [9]. The dynamics of N homogeneous agents are described by:

$$\begin{aligned} \mathbf{x}_i(k+1) &= \mathbf{A}\mathbf{x}_i(k) + \mathbf{B}\mathbf{u}_i(k) \\ \mathbf{y}_i(k) &= \mathbf{C}\mathbf{x}_i(k), \quad i = 1, \ldots, N, \end{aligned} \tag{1}$$

where $\mathbf{x}_i(k) \in R^n$ is the current state, $\mathbf{x}_i(k+1)$ is the state at the next time instant, $\mathbf{u}_i \in R^p$ is the control input, $\mathbf{y}_i \in R^q$ is the measured output, $\mathbf{A} \in R^{n \times n}$ is the state matrix, $\mathbf{B} \in R^{n \times p}$ is the input matrix and $\mathbf{C} \in R^{q \times n}$ is the output matrix [5]. Figure 2 shows a block diagram of the multi-agent system.

A distributed consensus protocol is defined by [6]:

$$\mathbf{u}_i(k) = c\mathbf{K} \sum_{j=1}^{N} a_{ij}(\mathbf{x}_j(k) - \mathbf{x}_i(k)), \quad i = 1, \ldots, N, \tag{2}$$

where $c > 0$ is the coupling gain, $\mathbf{K} \in R^{p \times n}$ is the feedback matrix and a_{ij} is the (i, j)-th entry of the adjacency matrix. How to determine c and $\mathbf{K}$ is given in [5].

Extension to formation control: Let $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_N) \in R^{n \times N}$ describe a formation structure, where $\mathbf{h}_i \in R^n$ defines the coordinate of agent i. Then, $\mathbf{h}_i - \mathbf{h}_j$ is the relative formation vector between agents i and j. The proposed consensus protocol can be transformed into a formation control problem for agent i described by:

$$\mathbf{u}_i(k) = c\mathbf{K} \sum_{j=1}^{N} a_{ij}(\mathbf{x}_i(k) - \mathbf{x}_j(k) - \mathbf{h}_i + \mathbf{h}_j). \tag{3}$$

In [5], it is shown that agents achieve a formation $\mathbf{H}$, if $\|\mathbf{x}_i(k) - \mathbf{h}_i - \mathbf{x}_j(k) + \mathbf{h}_j\| \to 0$, for $k \to \infty, \forall i, j = 1, \ldots, N$. With $\tilde{\mathbf{x}}_i(k) = \mathbf{x}_i - \mathbf{h}_i, \quad i = 1, \cdots, N$ and (3), (1) can be modified to [6]:

$$\mathbf{x}_i(k+1) = \mathbf{A}\tilde{\mathbf{x}}_i(k) + c \sum_{j=1}^{N} l_{ij}\mathbf{B}\mathbf{K}\tilde{\mathbf{x}}_j(k) + \mathbf{A}h_i, \tag{4}$$

where l_{ij} is the (i, j)-th entry of the Laplacian matrix $\mathbf{L}$.

Communication-trigger: The consensus protocol depends on the state of agent i and all other agents it is connected to. In order to find out the other agents' states, communication is necessary. A simple method is the periodic time-triggered communication. Agents communicate in constant intervals. In between, agents calculate the consensus update based on the last received value. A disadvantage of this method is an unnecessary communication if an agent's state has not changed. An event-triggered communication is based on a threshold δ. The trigger condition is defined by [4]:

$$\delta < \|\mathbf{x}_i(k) - \mathbf{x}_i(b)\|, \tag{5}$$

where $\mathbf{x}_i(k)$ is the current state of agent i and $\mathbf{x}_i(b)$ its last broadcasted state with $b < k$. Only if the state difference exceeds δ, a broadcast is triggered. This assures broadcasts only if new information is available.

With a state predictor, an agent could broadcast its state prediction besides its current state. The goal is to predict all values $\widehat{\mathbf{x}}_i(k+1), \widehat{\mathbf{x}}_i(k+2), \cdots, \widehat{\mathbf{x}}_i(k+h_p)$ up to a prediction horizon $h_p \in N$. A prediction is based on the current and past states of an agent. The Signal History Length (SHL) defines the depth of past states used for the prediction. The new trigger condition can be rewritten as:

$$\delta < \|\mathbf{x}_i(k) - \widehat{\mathbf{x}}_i(k)\|, \tag{6}$$

where $\widehat{\mathbf{x}}_i(k)$ is the predicted state. Each agent predicts and evaluates its own state. A communication is triggered when the difference of prediction and actual value at a time instant exceeds the threshold δ.

Prediction mechanisms: A common prediction mechanism is the one-step ahead prediction, where multiple-inputs of past values are mapped to a single-output value for the next time instant $k+1$. For multi-step ahead predictions there are various strategies. In a direct approach, multiple-inputs of past values are mapped to a single-output value for the time instant $k + h_p$. In many cases, an output vector of all values is of interest. Therefore, a multiple-input multiple-output model can be used. Another method is an iterative usage of the one step-ahead prediction by feeding back the output as an input for the next prediction. For long prediction horizons this method suffers from error accumulation [2].

A simple direct multi-step ahead predictor for the state x_i of an agent i is a linear extrapolation:

$$\widehat{x}_i(k + h_p) = x_i(k) + h_p \cdot (x_i(k) - x_i(k-1)). \tag{7}$$

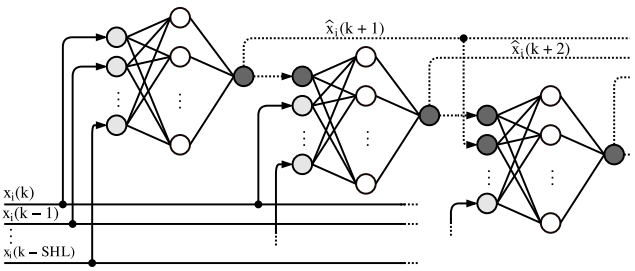

Figure 3: Iterated prediction with a single one-step ahead artificial neural network.

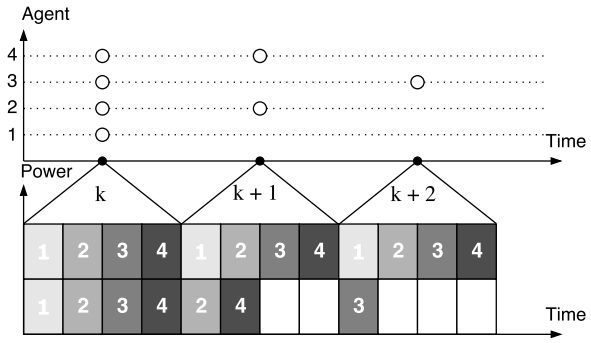

Figure 4: Event graph and the resulting channel access. The latter shows the slots scheduled with TDMA (upper row) versus CSMA/CA (lower row).

Since the extrapolation is linear, this method is only useful for short prediction horizons. In order to make long term predictions, more advanced methods are needed.

A comparative study of multi-step ahead predictions using machine learning methods such as multiple linear regression, recurrent neural networks and a hybrid of a hidden Markov model with multiple linear regressions was made in [2]. A nearest neighbor approach was suggested in [11] and decision trees were applied in [14].

Artificial neural networks (ANN) can be used for one-step ahead predictions. Based on the assumption that sufficient prior data is available, an ANN can be trained. The signal history length SHL defines the dimension of the input to a feed-forward network $x_i(k), x_i(k-1), \cdots, x_i(k-SHL)$. Target is the value at the next time step $\widehat{x}_i(k+1)$.

The trained network can make a one-step ahead prediction for a set of observed values. In order to make a multi-step ahead prediction $\widehat{x}_i(k+j), 1 \geq j \geq h_p$ for all states up to a prediction horizon h_p, an ANN with h_p output nodes can be trained. Another way is to iteratively use a one-step ahead prediction network as shown in Figure 3. This method offers a flexible way to predict all values for variable h_p without a retraining process.

When multiple agents want to use the same transmission medium, the access needs to be controlled. Therefore, in the medium access control (MAC)-layer which is shown in Figure 2, slots get scheduled. Time-division multiple access (TDMA) allocates each transmitter a slot defined by a time period. Carrier-sense multiple access with collision avoidance (CSMA/CA) allows a more flexible slot scheduling.

Figure 4 shows an event graph for an event-based communication of four agents and the resulting channel access with TDMA (upper row) versus CSMA/CA (lower row). With TDMA, each agent has a predefined slot at each time instant. However, this might be unnecessary if an agent has no information to transmit. This is the case of agent 1 at time step $k+1$ and $k+2$. With CSMA/CA, the slots get scheduled individually, since agents who need to transmit compete for slots. For example at time step $k+1$, solely agent 2 and 4 need to transmit, and thus, just two slots are used. The other slots remain free.

3 NUMERICAL RESULTS

In this section, numerical simulation results are illustrated, where we used MATLAB as a simulation environment. A system of 6 agents and a line communication topology is considered. The state of the i-th agent is defined by the position and the velocity in x- and y-direction. All states are initialized by zero. Goal for the agents is to reach a circle formation. It is a time discrete space with 300 time instants chosen. All agents update themselves at each time instant with no time delay.

In a first simulation, the communication between agents was triggered periodically at every second time instant, as it can be seen in Figure 6. In total, 900 communication events happened. Figure 5 shows the formation flight. Due to the unfavorably line topology, the agents converge slowly.

During this simulation with a periodic communication, different event-triggered methods were tested passively. All event-triggered methods received the same input, meaning the trajectories of Figure 5, but communication events were only logged in a file and not broadcasted. Figure 7 shows the logged events based on an event-triggered communication as in Equation (5) with $\delta = 0.1$. With around 350 events in total, this is already a significant reduction in comparison to the periodic trigger. Figure 8 and 9 are showing the event-triggered approach with an additional state prediction via linear extrapolation and an ANN. The total number of communication events is around 200 in the case of linear extrapolation and around 100 in the case of an ANN. Both predictions were based on an prediction horizon $h_p = 25$.

The training data for the ANN, meaning velocities of agents, was collected during several formation control cycles with random communication topologies and formations and a permanent communication. The latter ultimately means that the signals are smooth. The training dataset contained 53100 signals. Each signal holds six values. The first five are considered as the input and the sixth as the target. The MATLAB neural network toolbox was used to train the ANN ahead of operation.

A crucial variable using event-based communication with state prediction is the prediction horizon h_p, which influences the total number of communication events. Figure 10 shows how the number of events declines with an increasing h_p. Both state prediction approaches outperform the regular event-trigger, which is independent from h_p. The ANN approach reduces the communication events most, especially for a large h_p. It can also be seen

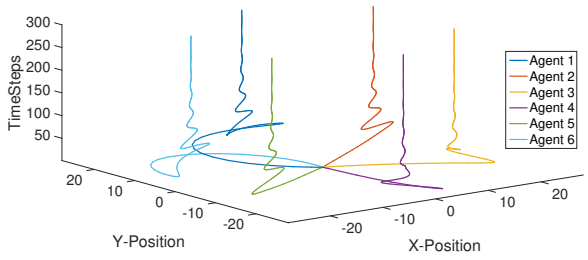

Figure 5: Formation control of 6 agents with a line topology and a periodic communication.

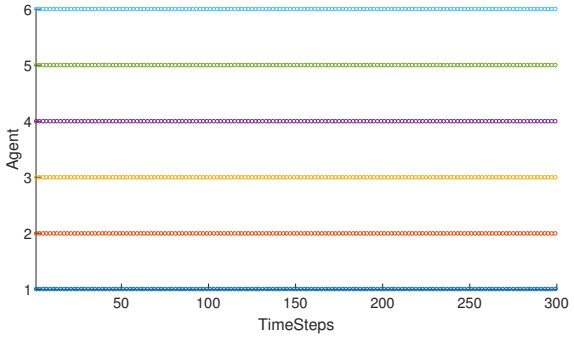

Figure 6: Communication events of agents for a periodic communication.

that the number of events stops declining at $h_p \approx 10$ for the linear extrapolation and at $h_p \approx 25$ for the ANN approach. This can be explained by the trajectories of the agents which can be linearly approximated just for short intervals.

A second simulation considered event-triggered communication. Figure 11 shows the formation flight for an event-trigger with $\delta = 0.1$. The trajectories are noisy in comparison to the ones in Figure 5. This is a result of the reduced communication. Since the ANN from the first simulation was trained with smooth velocity signals, it fails to make predictions on noisy signals. Thus, it had to be retrained with a noisy dataset. Also, linear extrapolations from noisy signals fail. Therefore, a moving average filter is smoothing the signals in advance. The trajectories of the formation flight using an event-triggered communication with state prediction look similar to Figure 11 and are therefore not plotted.

Figure 12 shows how the depth of the prediction horizon influences the total number of communication events. All triggers were tested separately. It can be seen that the total number of events for all three methods is significantly higher than in the passive trigger test. This is due to the fact that the triggers now effect the formation flight. Also, the number of events stagnates at a lower prediction horizon. Qualitatively, the ANN approach still performs best.

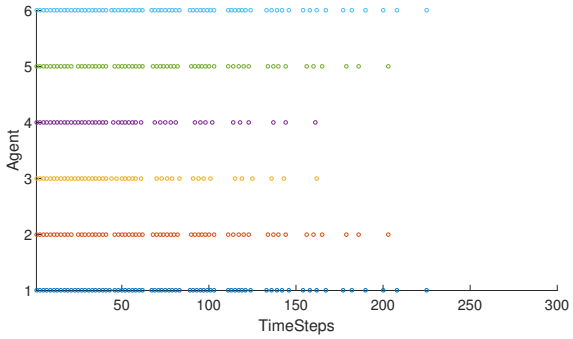

Figure 7: Communication events of agents for an event-triggered communication.

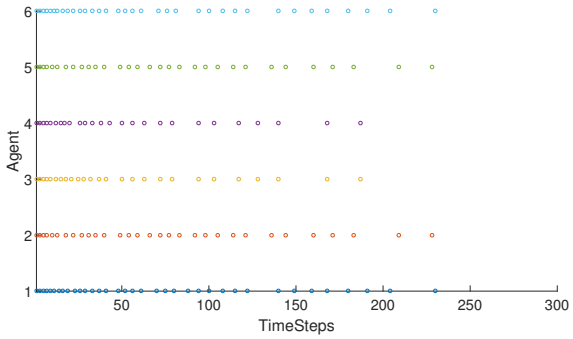

Figure 8: Communication events of agents for an event-triggered communication with linear extrapolation and $h_p = 25$.

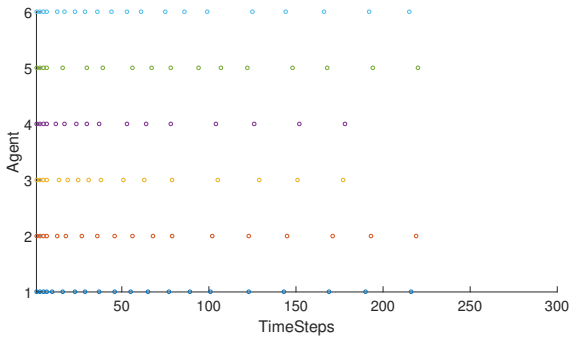

Figure 9: Communication events of agents for an event-triggered communication with an ANN and $h_p = 25$.

4 CONCLUSION

In this paper, a novel distributed event-triggered communication is presented. Artificial neural networks allow agents to predict future states solely on own past states. This approach is independent of other agent's states. Thus, the method is scalable with the number of agents. The results show that a distributed state prediction

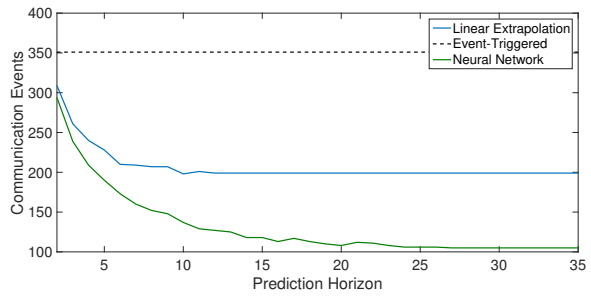

Figure 10: Number of communication events for different passively tested triggers, varying prediction horizons and a constant δ.

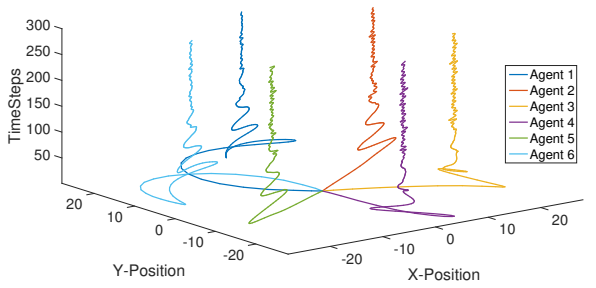

Figure 11: Formation control of 6 agents with a line topology and an event-triggered communication with $\delta = 0.1$.

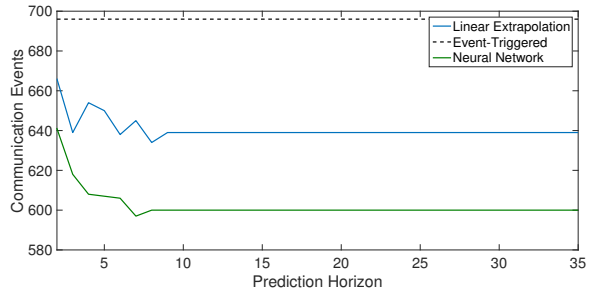

Figure 12: Number of communication events for different triggers, varying prediction horizons and a constant δ.

can reduce the communication effort. Trade-off is an error in the consensus. Although the communication frequency reduces, the data-packages sent are due to the prediction h_p-times larger. Currently, methods to compress those data-packages are investigated. Future research will address this issue and compare the state of the art prediction methods with the presented novel approach.

REFERENCES

[1] Space Studies Board, Engineering National Academies of Sciences, Medicine, et al. 2016. *Achieving Science with CubeSats: Thinking Inside the Box.* National Academies Press.

[2] Haibin Cheng, Pang-Ning Tan, Jing Gao, and Jerry Scripps. 2006. Multistep-ahead time series prediction. (2006), 765–774.

[3] Guido Dartmann, Elham Almodaresi, Mahdi Barhoush, Naim Bajcinca, Gunes Karabulut Kurt, Volker Lücken, E Zandi, and Gerd Ascheid. 2017. Adaptive Control in Cyber-Physical Systems: Distributed Consensus Control for Wireless Cyber-Physical Systems. (2017), 15–30.

[4] Ozan Demir and Jan Lunze. 2012. Event-based synchronisation of multi-agent systems. *IFAC Proceedings Volumes* 45, 9 (2012), 1–6.

[5] Zhongkui Li and Zhisheng Duan. 2017. *Cooperative control of multi-agent systems: a consensus region approach.* CRC Press.

[6] Zhongkui Li, Zhisheng Duan, and Guanrong Chen. 2011. Consensus of discrete-time linear multi-agent systems with observer-type protocols. *arXiv preprint arXiv:1102.5599* (2011).

[7] Manuel Mazo and Paulo Tabuada. 2011. Decentralized event-triggered control over wireless sensor/actuator networks. *IEEE Trans. Automat. Control* 56, 10 (2011), 2456–2461.

[8] Xiangyu Meng and Tongwen Chen. 2013. Event based agreement protocols for multi-agent networks. *Automatica* 49, 7 (2013), 2125–2132.

[9] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. 2007. Consensus and cooperation in networked multi-agent systems. *Proc. IEEE* 95, 1 (2007), 215–233.

[10] Georg S Seyboth, Dimos V Dimarogonas, and Karl H Johansson. 2013. Event-based broadcasting for multi-agent average consensus. *Automatica* 49, 1 (2013), 245–252.

[11] Antti Sorjamaa, Jin Hao, Nima Reyhani, Yongnan Ji, and Amaury Lendasse. 2007. Methodology for long-term prediction of time series. *Neurocomputing* 70, 16-18 (2007), 2861–2869.

[12] Christophe Viel, Sylvain Bertrand, Michel Kieffer, and Hélène Piet-Lahanier. 2017. Distributed event-triggered control for multi-agent formation stabilization and tracking. *arXiv preprint arXiv:1709.06652* (2017).

[13] Christophe Viel, Sylvain Bertrand, Kieffer Michel, and Piet-Lahanier Helene. 2017. New state estimators and communication protocol for distributed event-triggered consensus of linear multi-agent systems with bounded perturbations. *IET Control Theory & Applications* 11, 11 (2017), 1736–1748.

[14] Bo-Suk Yang, Andy Chit Chiow Tan, et al. 2009. Multi-step ahead direct prediction for the machine condition prognosis using regression trees and neuro-fuzzy systems. *Expert Systems with Applications* 36, 5 (2009), 9378–9387.

[15] H-T Zhang, Michael ZhiQiang Chen, and Tao Zhou. 2009. Improve consensus via decentralized predictive mechanisms. *EPL (Europhysics Letters)* 86, 4 (2009), 40011.

[16] Hai-Tao Zhang, Michael ZQ Chen, and Tao Zhou. 2009. Predictive protocol of flocks with small-world connection pattern. Vol. 79. APS, 016113.

[17] Hai-Tao Zhang, Michael ZhiQiang Chen, Tao Zhou, and Guy-Bart Stan. 2008. Ultrafast consensus via predictive mechanisms. *EPL (Europhysics Letters)* 83, 4 (2008), 40003.