Learning longer behavioral sequences in a modular robot team to better infer global formation

Taeyeong Choi, Sehyeok Kang, and Theodore P. Pavlic

***Abstract*— We propose a more powerful machine learning approach, as an extension of [1], to tackle the Remote Teammate Localization (RTL) problem where a robot member in a multi-robot team is to predict positions of all other teammates only using the observations on its nearest neighbor without any communication between robots. In the previous work, we showed feasibility of a scalable method by which a predictor robot was trained in a modular 3-robot team but could extend the prediction to a larger team without additional training, also suggesting an application of such an inference in caging mission. In this work, however, we focus mainly on 1) achieving better performance to improve the applicability and 2) conducting evaluation in more realistic environments. To be specific, we adopt a Long-Short Term Memory (LSTM) to learn possible evolution of behaviors in a modular team helping reduce the errors from regression outcomes. Furthermore, while the previous work relied only on computer simulations, all the experiments here are conducted on a physical two-wheeled robotic platform, *Thymio*, to demonstrate the performance gain under realistic constraints.**

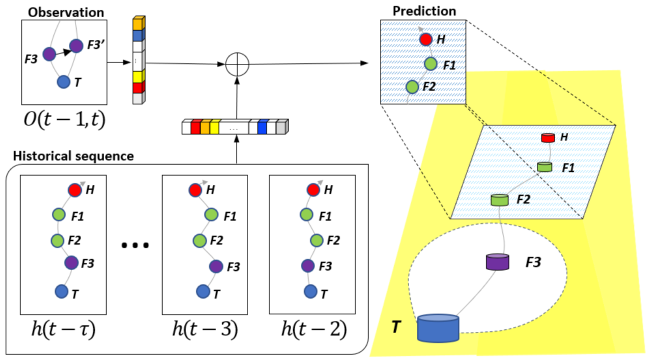

Fig. 1: Illustration of our proposed pipeline in a snapshot example of 5 robots at time t. Each robot has a limited view and a motion rule dependent on its neighbors except the *Head* robot leading the team at the front. *Tail* uses recent observations on its neighbor, *Follower 3*, which is denoted as $O(t-1,t)$. A sequence of historical poses, h, is also encoded for the model to make a final prediction on the unseen teammates.

I. Introduction

In multi-robot systems including swarms, every robot is usually allowed to observe only a subset of its team members and interact with them to determine the next action according to relatively simple motion rules. Such a property enables the entire system to perform in a distributed manner, and also the behavior of it can be driven easily by a few leader robots to eventually achieve the goal [1]–[3]. This implies that if a robot has an ability to reognize useful property, e.g.) formation, of the whole team in real time using its local sensors, it could present adjustive actions accordingly to better promote a collective behavior for the sake of the team.

In [1], we suggested a machine learning method to solve the RTL problem where a robot, called *Tail* at one end of a line formation of a multi-robot team, is to predict positions of all other teammates only using local observations about a single nearby teammate. Since each robot has a limited sensor radius and a relatively simple motion rule with dependency on the position of its nearest neighbors, the *Tail* has to be able to learn the regularity of the observed motions of neighbor to finally infer the poses of all other robots. We introduced a repetitive prediction scheme to use predictions on nearer teammates to make predictions on farther ones until the prediction reached the *Head* robot at the other end in line formation. Also, computer simulations showed the feasibility of the method especially with caging scenario in which the *Tail* could recognize a caging action of *Head* in early stage and promote a proactive maneuver to better assist in team cooperation.

RTL problem provides some unique characteristics compared to general localization problems. In RTL, the robot does not execute predictions on its own location but on its teammates using accessible information. Moreover, the robot is not allowed to communicate with other members during prediction phase to consider communication-free scenarios, which differs from cooperative localization problem. Hence, RTL usually assumes that robots behave with correlations with their neighbors so that the resulting behaviors contain cues about the state of their neighbors. In this sense, state observers in networked robotic system could be a more similar configuration to RTL, but RTL more emphasizes scalable applications to variable sizes of robot team as well as a general framework that would impose little constraint on dynamics of the system.

Figure 1 illustrates our proposed pipeline in which a deep neural network is to learn to synthesize the observations on its neighbor with knowledge about historical positions of all the teammates. As shown in Fig. 2, a LSTM layer is deployed to encode the historical sequence input, which could learn probable evolution of the team shape over time under physical constraints. The sequence encoding could help filter out impossible solution candidates that the model might produce if it only utilized the recent observations on the neighbor, as in [1],

*This work was not supported by any organization

T. Choi, S. Kang, and T. P. Pavlic are with the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281, USA {`tchoi4, skang66, tpavlic`}`@asu.edu`

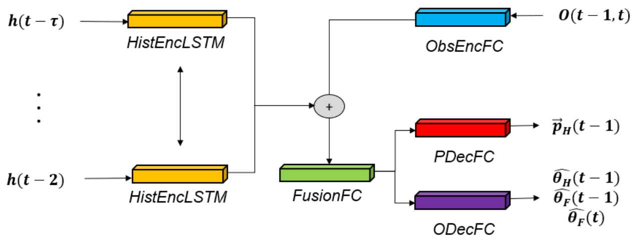

Fig. 2: Structure of our proposed deep neural network. This is an snapshot example when applied to a focused module of 3 robots, called *Tail, Follower, Head* within it, at time t. The Encoder-Decoder structure encodes 1) historical positions and orientations of *Follower* and *Head* until $t-2$ and 2) the observed positions of the *Follower* at $t-1$ and t. The decoder part learns to estimate 1) the position of *Head* at $t-1$ and 2) orientations of *Follower* at $t-1$ and t and *Head* at $t-1$.

Not only a more powerful model but also a more realistic experiment environment are prepared. Previously, we conducted all demonstrations on computer simulations, but in this work, a physical two-wheeled robotic platform, *Thymio* [4], is mainly used to set up more realistic environments.

This paper is organized as follow. In Section II, we explore related literature and the distinction of our work. Section **??** explains more details about our setting of RTL problem. Then, we introduce our method, *IPY-Net*, in Section IV, and Section V explains details about experiments performed on real robots, including data collection, hyperparameters used for learning, and the results. Lastly, we summarize our research and discuss future directions in Section VI.

II. Related work

Although the RTL problem is to localize robots, as mentioned in Section I, it has unique properties compared to conventional localization problems including cooperative localization or tracking [5]–[8]. It is rather about robot behavior as a container to represent useful team-level state and learning to decode it. We discuss related literature below that introduced scenarios to use and decrypt robot behaviors in multi-robot system, followed by a comparison with a work that exploited deep learning to learn dynamics of robot on presentation of force input. Lastly, we explain similarities to our previous work as well as main contributions as an extension.

A. Group state recognition

Our approach is basically an attempt to infer positional state of the entire multi-robot team, using only locally obtainable information in the aspect of a robot member. This is because the knowledge about global configuration could help make a better decision for the sake of whole team. In this spirit, the authors in [9], [10] sampled a subset of robot agents from a swarm to monitor their local interactions and classify the swarm-level behavior. In fact, they implemented relatively simple robots run on a virtual simulator and predefined selections of global behaviors such as *flock* and *torus*. In contrast, our work is to estimate pose of real robots, which would require a more powerful model to regress on any value in a wide range of arena size.

B. Behavioral cue interpretation

One of the main assumptions in RTL is the simple motion rule of each robot which has dependency on the state of its neighbors. This implies that the behavior of neighbors could be a method to encrypt the state change of more remote teammates. In [11], [12], Novitzky at el. and Das et al. employed a type of special action, which appeared like ”waggle dance” of honeybees [13], in robot team so that the behavior could convey a meaning between robots. Our work also encourages a robot to decode behavioral cues, but it occurs without entering an explicit phase of signaling. In other words, the robot has to learn to gain useful information from a sequence of continuous usual interactions.

C. Robot dynamics learning

Byranvan and Fox in [14] proposed a deep learning approach to predict the next visual frame given a current frame and the information of force that would affect one of the shown objects. One of their examples was to understand the dynamics of robotic arms and the relationship with control commands possibly executed. In the RTL problem also, the *Tail* robot may have known a global shape of robot team, and it has to be able to predict the future formation as a new observation on its neighbor is provided, which would encode the information of an unseen force driving the team. Such a similarity inspired the architecture of our neural network model, but they focused on learning motions of rigid objects, while a chain of robots in our work can present much flexibility in team shape. In addition, the force is inferred by movements of the neighbor, which would likely contain noise to some level, while *SE3-Nets* is provided with clear information about the applied force.

In our previous work [1], the detailed configurations for RTL problem were first introduced with realistic assumptions. To solve the problem particularly with little robot-to-robot communication, the *Tail* robot is designed to use a repetitive strategy of inference with which the predictions on a closer robot are used as input to prediction for more distant team members. Such an approach enables to scale up the estimation capability without further training even though more robots are added. Albeit we use the same prediction scheme with repetition, here we focus more on developing a more powerful regressor to broaden applicability and also on performing realistic evaluations in a multi-robot system realized on an off-the-shelf robotic platform.

III. RTL Problem

Basically, we follow the problem description for the RTL first introduced in [1]. A robot team moves in a line formation in which each robot has two adjacent neighbors except for the ones at the ends, each of which has only one neighbor. The robot on one end is called *Head*, which is the only robot that drives the team toward a destination predetermined in an independent manner. The robot on the opposite end is named *Tail*, and all other robots in-between are *Follower* i where $i \in \{1, 2, ..., n-2\}$ starting from the robot closest to the *Tail*. Every robot has the same capability of sensor and motion under the same kinematic constraints, although they may take different actions according to motion rules depending on their roles, as formulated in [1]. This catalyzes the *Follower* robots and the *Tail* to behave based on the current positions of their close neighbors. For simplicity, we assume that every robot has an ability to accelerate fast enough to always keep the neighbors within their sensory range.

The RTL is to localize all the teammates from the view of *Tail* only using locally visible information, which is the position of *Follower* 1 at every time step. We assume that until a specific time instant t, all the information about positions and orientations of all robots have been shared especially with the *Tail* robot, possibly via global communication, but from $t+1$, such information becomes no longer available causing a need of the localization technique only utilizing local observations. This may simulate some realistic scenarios where an unexpected technical issue occurs at a time point disabling any robot-to-robot data transfer, or the robot team intentionally stops the information sharing when entering some specific regions either for security or for saving on the communication cost.

IV. Method

In this section, we introduce more details about our method. First, we briefly revisit the prediction approach, shown in [1], to train the model on a small team and to be able to produce predictions even on a larger team without additional training. Then, we present the regressor built for learning not only with observations on the neighbor but also with a sequence of past team formations.

A. Scalable prediction

We refer to the inference scheme of our previous work [1] in which a regressor is trained in a modular subteam of 3 robots, and when employed for a larger team, repetitive predictions are performed along with the chain of 3-robot modular teams until reaching the *Head*. More specifically, given a robot team of size n, the *Tail* robot is allowed

IPY-Net is designed to produce probabilistic prediction outcomes that contain uncertainty to some level. Specifically, the training process is to encourage to learn the probability distribution on predicted position or orientation. Furthermore, during relaying prediction, the model has to be able to understand the probability distribution predicted at a previous 3-robot module to make prediction for the current one.

To be specific, *IPY-Net* adopts a $m \times m$ matrix, which is an image, and a $1 \times k$ matrix to represent position and orientation, respectively. Because each matrix is a probability distribution, the summation of all elements must be 1. For example, an observed x coordinate is first converted by Equation 1.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \times m - \frac{m}{2} \tag{1}$$

where x_{min} and x_{max} are the minimum and maximum of x coordinates empirically observed. y' can be calculated from a y coordinate in a similar way. Then, the position matrix M_p initialized with zeros is updated

	Duration	Num. of Samples	Num. of Instances
3 robots	100.6 minutes	$6,975$	465
5 robots	45.0 minutes	$3,120$	208

TABLE I: Description of data collected from executions of 3-robot and 5-robot teams.

$$\begin{aligned} M_p(\lfloor x' \rfloor, \lfloor y' \rfloor) \\ M_p(\lfloor x' \rfloor + 1, \lfloor y' \rfloor) \\ M_p(\lfloor x' \rfloor, \lfloor y' \rfloor + 1) \\ M_p(\lfloor x' \rfloor + 1, \lfloor y' \rfloor + 1) \end{aligned} \tag{2}$$

We use the image representation, since it allows to learn an arbitrary probability distribution instead of a known distribution constrained on a set of parameters.

B. Neural network

V. Experiments

To demonstrate the effectiveness of our method, we employ a physical robotic platform, *Thymio* [4], which allows to execute a team of small two-wheeled mobile robots. We use a central computer connected with a overhead camera to simulate better proximity sensors, a more powerful computing power, and a GPS system that would easily run on each robot in real scenarios. The central system is set up to detect the locations of robots in real time and communicate with a *Raspberry Pi* board [15] on each robot to send the next command relying on its neighbors. Although the experiment configuration involves such external computations and sensors due to limited capability of *Thymio*, most of realistic assumptions hold, and the robots are still influenced by physical constraints and disturbance in motion. For better understanding of readers, we submit a supplementary video.

The location detection is performed at 4 frames per second at each of which a new command is received by each robot. Also, all coordinates and orientations at 2 frames per second are collected, which is not necessarily synchronized with the command timing.

Two different sizes of robot team are deployed throughout following experiments: 3 robots and 5 robots in which the *Head* robot continues to receive a command to move to an arbitrary destination point nearby in a *OO* × *OO* meters arena. Table **??** shows details about the data collected from each configuration where a sample refers to a set of coordinates and orientations recorded at an instant time step, and an instance is a sequence of samples for 13 seconds. The instances are clustered to have at least 7-second time gap to another so that the motions between separate ones have only little dependency.

70% data from the 3-robot team is used to train *IPY-Net* by feeding each sample, and the rest 20% is used for test. The trained model is also tested with the 5-robot team, in which the relay prediction is initialized at the beginning of each instance. The length of history is set to 5 seconds to accept

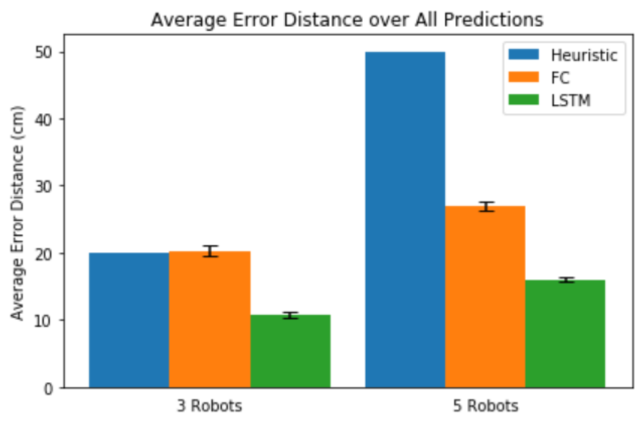

Fig. 3: Prediction errors of *IPY-Net* and *I-LSTM-FC* for different robots at each time step.

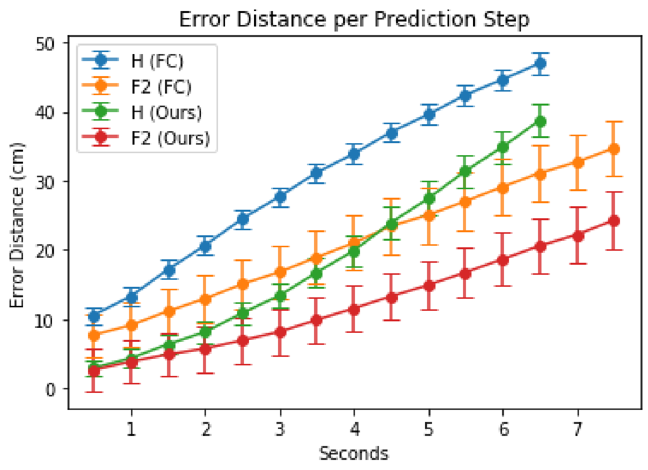

Fig. 4: Prediction errors of *IPY-Net* and *I-LSTM-FC* for different robots at each time step.

inputs for past 10 time steps, and the predictions occur for the next 8 seconds.

IPY-Net is implemented in *Tensorflow Python* library [1] to realize the entire pipeline. The evaluation is reported in Euclidean distance between the predicted position and the truth, using the model that achieves the best performance during 100 epochs, by which the loss mostly converges to zero.

We compare our proposed model to four different approaches such as:

- 2X *Heuristic*: The prediction on *Head* within a modular team is performed by doubling the vector $\vec{p}_F - \vec{p}_T$.
- *FC*: Two fully connected layers run in the predictor without considering historical information, which is based on [1].
- *LSTM-FC*: Model combining the historical LSTM and FC layers similar to *IPY-Net* except that all inputs and outputs are represented as numerical values instead of images.
- *I-LSTM-FC*: Imaginative model using the same structure as *IPY-Net* with difference that only best predictions on coordinates or orientation are preserved from each predicted image. As needed, the preserved information is processed to generate a new image input during relay prediction process.

A. Overall accuracy

We show overall performance of each model in Table **??**. The error distance is averaged for all samples of all robots.

B. Microscopic analysis

Here, we further analyze the behaviors of top two models in overall performance by focusing more on errors for different robots at each time step. The robot-wise error is averaged at each step across instances. The result could help compare the upper bounds of errors of the models and understand the feasibility of them during the instance interval.

[1]https://www.tensorflow.org/

C. Qualitative results

This section visualizes the prediction outcomes of *IPY-Net*, while the robot team evolves its shape over time. Figure **??** displays three separate instances in which the generated heatmaps are overlaid on the corresponding original video frames illustrating the predicted probability distribution of the team-level formation.

The results imply that *IPY-Net* can properly track the emerging behaviors of the team triggered by arbitrary motion sequences. Moreover, the visualized output helps human users better interpret the outcome taking into consideration the confidence levels offered by the predictive model.

VI. Discussion & Future work

We have shown *IPY-Net*, a powerful approach to tackle the remote localization problem where a robot with limited sensors is to predict positions of all others only using observations about its nearest neighbor. *IPY-Net* follow the fundamental spirit of [1] to apply a predictor trained for a small robot team to a chain of them in a larger team, since it can provide a scalable feature without re-training. The distinction in *IPY-Net* is , however, to use a sequence of historical knowledge and make predictions based on imagination of robot states including position and orientation. We expected learning with history would help regulate the model on physical constraints on the team formation, and also the imaginative representation could contain certainty about the prediction, which would bring about more robust performance.

The experiment results were obtained using a realistic robotic platform to prove that our method outperforms not only the regressor introduced in [1] but also a general state-of-the-art neural network model that is able to use historical data. Especially, our model can present stable performance even when the input contains some errors which have occurred in previous prediction.

In future work,

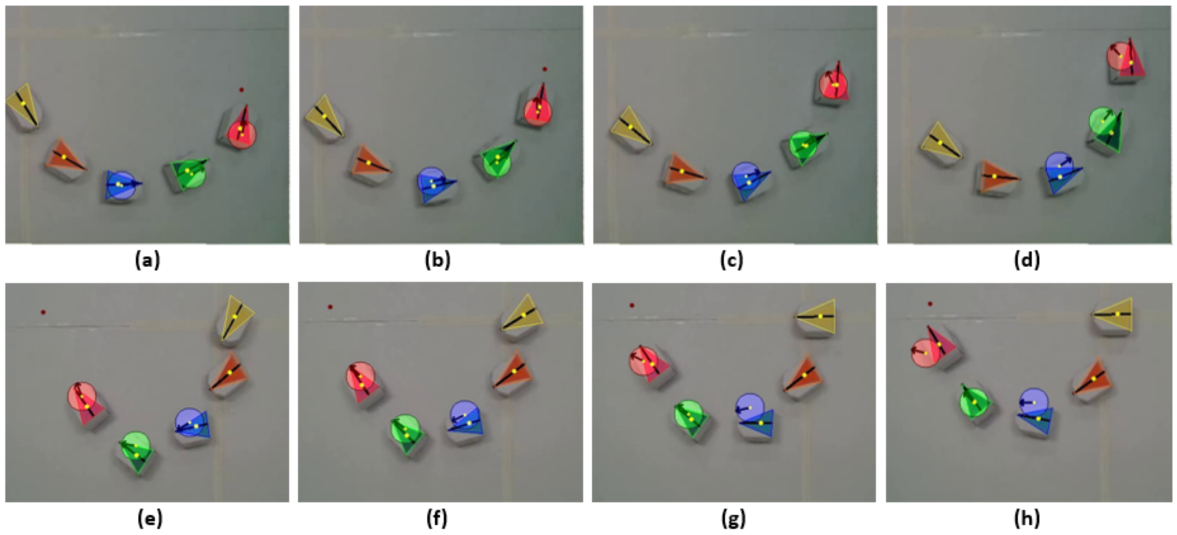

Fig. 5: Visualized predicted heatmap examples generated from *IPY-Net*. Each row contains snapshots of three instant time steps during a unique instance. In each instance, the frames are sorted in increasing order in time.

References

[1] T. Choi, T. P. Pavlic, and A. W. Richa, "Automated synthesis of scalable algorithms for inferring non-local properties to assist in multi-robot teaming," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*. IEEE, 2017, pp. 1522–1527.

[2] J. J. Daymude, R. Gmyr, A. W. Richa, C. Scheideler, and T. Strothmann, "Improved leader election for self-organizing programmable matter," in *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*. Springer, 2017, pp. 127–140.

[3] K. Elamvazhuthi and S. Berman, "Scalable formation control of multi-robot chain networks using a PDE abstraction," in *Proc. of the 12th International Symposium on Distributed Autonomous Robotic Systems*, 2016, pp. 357–369.

[4] J. Shin, R. Siegwart, and S. Magnenat, "Visual programming language for thymio ii robot," in *Conference on Interaction Design and Children (IDC'14)*. ETH Zürich, 2014.

[5] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, "Recursive decentralized collaborative localization for sparsely communicating robots." in *Robotics: Science and Systems*. New York, NY, USA, 2016.

[6] A. Franchi, P. Stegagno, M. Di Rocco, and G. Oriolo, "Distributed target localization and encirclement with a multi-robot system," in *Proc. of the 7th IFAC Symposium on Intelligent Autonomous Vehicles*, 2010, pp. 151–156.

[7] Y. Cheng and D. Xie, "Distributed observer design for bounded tracking control of leader–follower multi-agent systems in a sampled-data setting," *International Journal of Control*, vol. 87, no. 1, pp. 41–51, 2014.

[8] O. De Silva, G. K. I. Mann, and R. G. Gosine, "Efficient distributed multi-robot localization: A target tracking inspired design," in *Proc. of the 2015 IEEE International Conference on Robotics and Automation*, 2015, pp. 434–439.

[9] D. S. Brown and M. A. Goodrich, "Limited bandwidth recognition of collective behaviors in bio-inspired swarms," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 405–412.

[10] M. Berger, L. M. Seversky, and D. S. Brown, "Classifying swarm behavior via compressive subspace learning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5328–5335.

[11] M. Novitzky, C. Pippin, T. R. Collins, T. R. Balch, and M. E. West, "Bio-inspired multi-robot communication through behavior recognition," in *Proc. of the 2012 IEEE Conference on Robotics and Biomimetics (ROBIO)*, 2012, pp. 771–776.

[12] B. Das, M. S. Couceiro, and P. A. Vargas, "MRoCS: A new multi-robot communication system based on passive action recognition," *Robotics and Autonomous Systems*, vol. 82, pp. 46–60, 2016.

[13] K. Von Frisch, *The Dance Language and Orientation of Bees*. Harvard University Press, 1967.

[14] A. Byravan and D. Fox, "Se3-nets: Learning rigid body motion using deep neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 173–180.

[15] E. Upton and G. Halfacree, *Raspberry Pi user guide*. John Wiley & Sons, 2014.