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.**

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 [x]. This implies that if a robot has an ability to recognize 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 embedded in a multi-robot team is to predict positions of all other teammates using only 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 in line formation, the predicting robot has to be able to learn the regularity of the observed motions. Through simulated experiments, we showed the feasibility of the idea especially in caging scenario in which the predicting robot could recognize the team behavior and use a proactive maneuver accordingly.

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

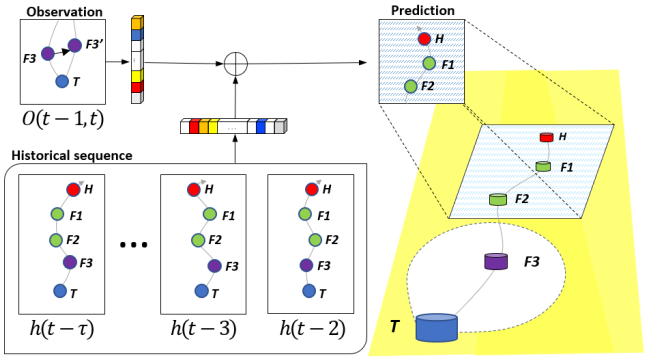

Fig. 1: Example of *IPY-Net* performing RTL on 6-robot team. While the *Head* (red) is driving the team in line formation, the *Tail* can observe the motions of *Follower 1* (purple) within its sensor radius (yellow). The observation, O, and the historical positions of all robots, h, that the *Tail* has known so far become an input to *IPY-Net* to pro All positions of robots in the local view of *Tail* are visualized in image as probabilistic representation. Note here that since *Follower 4* can be observed directly, the image representation does not illustrate any uncertainty for it.

In this work, we propose *IPY-Net* to significantly reduce prediction errors solving the RTL problem, which would allow to gain stable performance even with more robots added. *IPY-Net* runs a deep neural network as a backbone, first to consider physical constraints on the sequential change of team formation and, second to use a probabilistic representation in the form of image for obtained observations and new predictions. Specifically, every observed or predicted position is encoded as an image in which the summation of all values must be 1, which can be viewed as a probability distribution of position of the robot. Hence, *IPY-Net* learns how to synthesize inputs with some uncertainty to produce a probabilistic prediction outcome in end-to-end fashion. Furthermore, a recurrent layer is deployed in *IPY-Net*, which enables the model to take into account possible evolution

*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

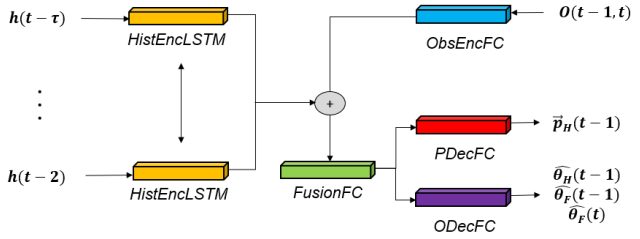

Fig. 2: Structure of *IPY-Net*. This is an snapshot example when applied to a focused module of 3 robots, called *Tail, Follower, Head* within it, at time t. The Encoder-Decoder structure encodes 1) historical positions and orientations of *Follower* and *Head* until $t-2$ and 2) the observed positions 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 $t-1$.

of team formation based on recent history and reduce the searching space for predicted solution. With all combined, historical probabilistic (soft) formations are used as input per prediction providing more flexibility to the predictor in selecting salient features and finally leading to more robust performance even when the input is noisy to some level.

We conduct all experiments on a physical robotic platform, *Thymio* [2], to demonstrate the improvement in more realistic environments in contrast to the previous work that relied only on computer simulations. All codes are available online[1], and supplementary videos are submitted as well.

This paper is organized as follow. In Section II, we explore related literature and the distinction of our work. Section III 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

Byranvan et al. in [3] proposed a deep neural network to predict the next visual frame given a current frame and the information of force that affects one of the shown objects. The suggested *SE3-Nets* is to understand the dynamics of objects depending on the force executed. Because *IPY-Net* also learns mapping behaviors of a robot to the evolution of team formation, it has a similar structure of deep neural network. *IPY-Net*, however, deals with a multi robotic scenario where it has to be able to utilize the observed information to infer a dependent change of a far robot.

The most motivating work is in [1] is the most similar to

III. Problem Description

Basically, we follow the problem configurations for remote teammate localization introduced in [1]. A robot team moves in a line formation in which each robot has two adjacent neighbors except for the ones at both ends, each of which has only one neighbor. The robot on one end is the only robot that tries to move to a predetermined destination independent from its neighbor. The robot is called *Head*, and the robot on the opposite end is named *Tail*. All the robots in-between are *Follower* i where $i \in \{1, 2, ..., n-1\}$ starting from the robot closest to the *Tail*.

All the robots are set to take actions as formulated in [1] leading the *Follower* robots and the *Tail* to behave based on the current positions of their close neighbors. As will be explained in Section V, we utilize the computational, sensory capacity of a central computer in experiments. Therefore, technically the global positions are used to calculate the next spatial destination of each robot, and then the formula in [] helps generate an actual motion command to operate two wheels to change its pose according to nonholonomic kinematics.

The remote teammate localization is to localize all the teammates in 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 were shared with the *Tail* robot, but from $t+1$, such information is no longer available causing a need of the localization approach utilizing local observations. This may simulate some realistic scenarios where an external technical issue occurs at some time point disabling the information transfer, or the robot team intentionally stops the communication at some regions for security or for saving on the cost.

IV. Method

We refer to the inference scheme of our previous work [1] in which a regressor is trained in a short line of 3 robots, and when employed for a larger team, relaying pose estimations are performed along with the chain of 3-robot modular teams.

delay,

A. Imaginative representation

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

[1]http://www.github.com/ctyeong

	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.

	3 robots	5 robots
Avg Distance	20.0	30.0

TABLE II: 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. IPY-Net pipeline

V. Experiments

To demonstrate the effectiveness of our method, we employ a physical robotic platform, *Thymio* [2], 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 [4] 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 $OO \times OO$ 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

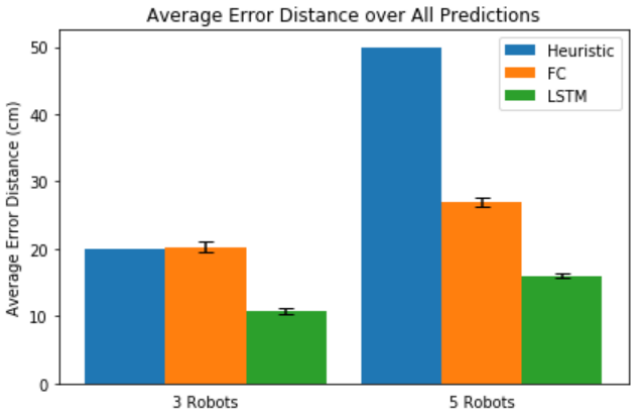

Fig. 3: Prediction errors of *IPY-Net* and *I-LSTM-FC* for different robots at each time step.

the relay prediction is initialized at the beginning of each instance. The length of history is set to 5 seconds to accept inputs for past 10 time steps, and the predictions occur for the next 8 seconds.

IPY-Net is implemented in *Tensorflow Python* library [2] 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:

- 2*X 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

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

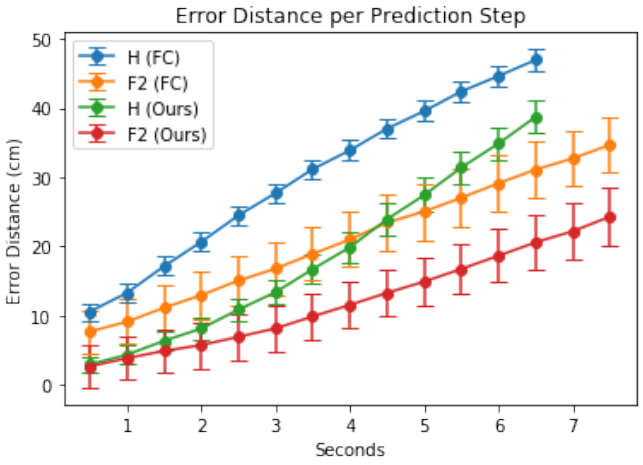

Fig. 4: Prediction errors of *IPY-Net* and *I-LSTM-FC* for different robots at each time step.

and understand the feasibility of them during the instance interval.

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,

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. 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.

[3] 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.

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

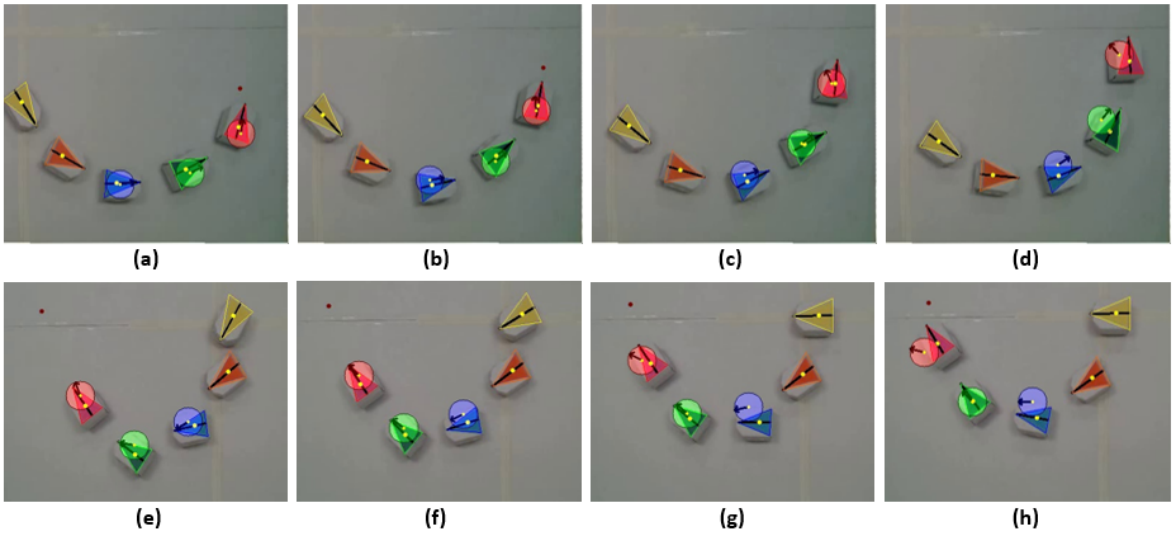

Fig. 5: Visualized predicted heatmap examples generated from *IPY-Net*. Each row contains snapshots of three instant time steps during a unique instance. In each instance, the frames are sorted in increasing order in time.