Data-Efficient Multi-Robot, Multi-Task Transfer Learning for Trajectory Tracking

Karime Pereida, Mohamed K. Helwa and Angela P. Schoellig

***Abstract*— Learning can significantly improve the performance of robots in uncertain and changing environments; however, typical learning approaches need to start a new learning process for each new task or robot as transferring knowledge is cumbersome or not possible. In this work, we introduce a multi-robot, multi-task transfer learning framework that allows a system to complete a task by learning from a few demonstrations of another task executed on a different system. We focus on the trajectory tracking problem where each trajectory represents a different task. The proposed learning control architecture has two stages: (i) a *multi-robot* transfer learning framework that combines $\mathcal{L}_1$ adaptive control and iterative learning control, where the key idea is that the adaptive controller forces dynamically different systems to behave as a specified reference model; and (ii) a *multi-task* transfer learning framework that uses theoretical control results (e.g., the concept of vector relative degree) to learn a map from desired trajectories to the inputs that make the system track these trajectories with high accuracy. This map is used to calculate the inputs for a new, unseen trajectory. We conduct experiments on two different quadrotor platforms and six different trajectories where we show that using information from tracking a single trajectory learned by one quadrotor reduces, on average, the first-iteration tracking error on another quadrotor by 74%.**

I. INTRODUCTION

Robots are being deployed in unstructured environments where they face model uncertainties, unknown disturbances, and changing dynamics. Small changes in the environmental conditions may deteriorate the performance and cause instability in traditional controllers ([1] and [2]). Typical learning-based control methods can guarantee high overall performance; however, if conditions change a new learning process may be required. Training robots to operate in changing environments is complex and time-consuming. Transfer learning reduces training time and the unavoidable risks of the training phase. In contrast, adaptive controllers are able to adapt online and reject disturbances, but are not able to exploit prior knowledge. In this work we develop a multi-robot, multi-task transfer learning framework (see Fig. 1) that allows a system to complete a task by learning from a few demonstrations of another task executed on a different system, i.e. leveraging previous knowledge even when conditions change. We focus on trajectory tracking as many robotic tasks can be formulated as trajectory tracking problems. The proposed multi-robot, multi-task transfer framework achieves high-accuracy trajectory tracking from the first iteration for numerous robot dynamics and desired trajectories.

There have been few works on multi-robot transfer. On the theoretical side, the work in [3] proved that the optimal transfer learning map between two robots is, in general, a dynamic system. The properties of the optimal dynamic map include its order and relative degree among other variables. In [4], a data transfer mechanism based on manifold alignment of input-output data is proposed. The transferred data improves learning of a model of a robotic arm by using data from a different robotic arm.

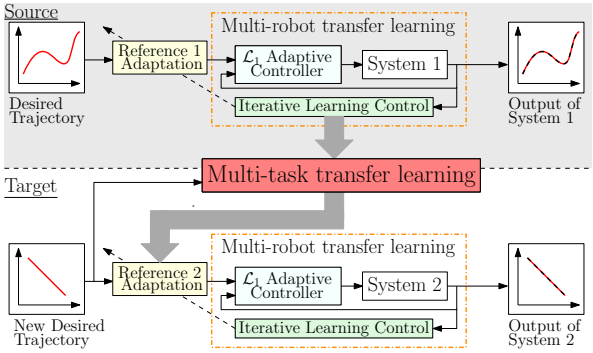

Fig. 1. The proposed multi-robot, multi-task architecture. The *multi-robot* transfer learning framework is composed of two methods *(i)* an iterative learning control (ILC) module to learn an input such that the output tracks a desired output signal and *(ii)* an $\mathcal{L}_1$ adaptive controller to force different systems to behave in the same repeatable, predefined way. Hence, trajectories learned on a source system can directly be transferred to a target system. The *multi-task* transfer learning framework learns a map from a desired trajectory to the inputs that make the system track it accurately. When a new trajectory is encountered, the learned map is used to calculate the inputs for the new trajectory.

Multi-task transfer uses previously learned tasks to perform new, unseen tasks. Learning approaches, such as iterative learning control (ILC), are usually not able to transfer knowledge to new, unseen tasks. In [5], using knowledge from previously learned trajectories, a linear map is created (through trial-and-error) to calculate the inputs required to track unseen trajectories. Experimental results show that only one learned trajectory is needed to improve the performance on a new trajectory. However, experimentally creating the optimal map may be time consuming. In [6] and [7], a deep neural network (DNN) is trained to achieve a unity map between the desired and actual outputs. The DNN adapts the reference signal to a feedback control loop to enhance the tracking performance of unseen trajectories. In [8], neural networks allow generalization of a task based on a single instance of the given task. However, the architecture of the neural network must be tailored to the specific task.

Multi-robot, multi-task transfer transfers tasks learned on a robot to different tasks to be executed on a different robot. In [9] a neural network learns policies that can be decomposed into "task-specific" and "robot-specific" modules. When a new robot-task combination is encountered, the appropriate

robot and task modules are composed to solve the problem. This architecture enables zero-shot generalization with a variety of robots and tasks in simulation. However, neural network approaches require significant amounts of data and computational resources to train. In this work, we emphasize data efficiency to achieve successful transfer in experiments.

The contribution of this work is to design a learning architecture that is able to achieve high-accuracy tracking in the first iteration *(i)* despite the presence of changing dynamics which include switching the robot hardware altogether, and *(ii)* by using previously learned trajectories and generalizing knowledge to new, unseen trajectories. This work was published in RA-letters 2018 [10], but has not been presented at a conference. This workshop provides an opportunity to present our work in front of peers working in the field intersecting machine learning and adaptation, get feedback and discuss next steps.

II. METHODOLOGY

The objective of this work is to achieve high-accuracy trajectory tracking in the first iteration in a multi-robot, multi-task framework, in which *(i)* the training and testing robots are dynamically different, and *(ii)* the training and testing trajectories are different. We consider a control architecture as shown in Fig. 1. The proposed approach also allows the system to continue learning over iterations after transfer.

A. Multi-robot transfer

The multi-robot transfer framework is based on the combined $\mathcal{L}_1$ adaptive control and ILC approach introduced in [11]. The ILC improves tracking performance over iterations, while the $\mathcal{L}_1$ adaptive controller forces dynamically different nonlinear systems to behave close to a specified linear model. Hence, learned trajectories can be transferred among dynamically different systems (equipped with the same underlying $\mathcal{L}_1$ adaptive controller) to achieve high-accuracy tracking.

The extended $\mathcal{L}_1$ adaptive controller that we implemented in our experiments in Section III assumes that the uncertain and changing dynamics of the robotic system can be described by a MIMO system for output feedback:

$$y_1(s) = A(s)(u_{\mathcal{L}_1}(s) + d_{\mathcal{L}_1}(s)), \quad y_2(s) = \frac{1}{s} y_1(s), \quad (1)$$

where $y_1(s)$ and $y_2(s)$ are the Laplace transforms of the translational velocity $y_1(t) \in \mathbb{R}^p$ and the position $y_2(t) \in \mathbb{R}^p$, respectively, $A(s)$ is a transfer function matrix of strictly-proper *unknown* transfer functions that can be stabilized by a proportional-integral controller, $u_{\mathcal{L}_1}(s)$ is the Laplace transform of the input $u_{\mathcal{L}_1}(t) \in \mathbb{R}^p$, and $d_{\mathcal{L}_1}(s)$ is the Laplace transform of disturbance signals defined as: $d_{\mathcal{L}_1}(t) := f(t, y_1(t))$, where $f : \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}^p$ is an *unknown* map subject to the global Lipschitz continuity assumption with Lipschitz constant L (see Assumption 4.1.1 in [12]).

The extended $\mathcal{L}_1$ adaptive output feedback controller aims to design a control input $u_{\mathcal{L}_1}(t)$ such that the output $y_2(t) \in \mathbb{R}^p$ tracks a bounded piecewise continuous reference input $u_2(t) \in \mathbb{R}^p$. We aim to achieve a desired closed-loop behavior by nesting the output of the $\mathcal{L}_1$ adaptive controller with output $y_1(t) \in \mathbb{R}^p$ which tracks $u_1(t) \in \mathbb{R}^p$ within a proportional feedback loop. Through the use of an output predictor, adaptation law, control law and closed-loop feedback, the extended $\mathcal{L}_1$ adaptive controller makes the system behave close to a linear, MIMO system described by:

$$\begin{gathered} y_2(s) = \text{diag}(D_1(s), \ldots, D_p(s)) u_2(s), \text{ where} \\ D_i(s) = \frac{K_i m_i}{s^2 + m_i s + K_i m_i}, \end{gathered} \quad (2)$$

and K_i, $m_i > 0$. The details of the extended $\mathcal{L}_1$ adaptive control implementation can be found in [10].

The multi-task transfer framework, discussed in the next subsection, requires a desired trajectory and the correct input that makes the system track said trajectory. To construct this pair of desired trajectory and corresponding correct input, we use an **optimization-based ILC** [13] to modify the input and improve the tracking performance of the system, which now behaves close to (2), in a small number of iterations $1, \ldots, j$. After each iteration, we use an iteration-domain Kalman filter to obtain an estimate of the disturbance in the system, based on measurements from iterations $1, \ldots, j$. Finally, based on the disturbance and system (2), the next input sequence is calculated by minimizing a quadratic cost function that includes the estimated output error.

Remark 1. *If the source and target systems have underlying $\mathcal{L}_1$ adaptive controllers with different reference models, then it is still possible to implement the multi-robot framework by using the reference models to build a map from the source system to the target system [3]. Using this map, trajectories learned on the source system can be transferred to the target system, which has a different reference model.*

B. Multi-task transfer

The multi-task transfer scheme learns a map between a single desired trajectory and the inputs that make the system track the desired trajectory accurately. This map is used to calculate the inputs needed to track a new, unseen trajectory with high accuracy. Suppose that we are given a smooth desired trajectory $\mathbf{y}_2^*$ and using a learning approach, such as ILC, we are able to obtain for this particular desired trajectory an input sequence vector $\mathbf{u}_2$ that achieves high-accuracy tracking performance. Our proposed multi-task transfer learning framework uses insights from control systems theory to identify the components of the map.

Lemma 1. *Consider a minimum phase, discrete-time, MIMO, LTI system (e.g. the one obtained by discretizing* (2)*), and a smooth desired trajectory $\mathbf{y}_2^*$. Then, there exists a control input sequence $\mathbf{u}_2$ that achieves perfect tracking of $\mathbf{y}_2^*$. Moreover, at time instant k, the control input $u_2(k)$ can be represented as a linear combination of the state $x(k)$ and the values $y_{2,1}^*(k + r_1), \ldots, y_{2,p}^*(k + r_p)$, where $y_{2,i}^*(j)$ is the value of the i^{th} component of the desired output at time index j, and $(r_1, \ldots, r_p)$ is the vector relative degree of the system.*

The proof of this lemma can be found in [10]. We know

from Lemma 1 that to achieve perfect tracking, $u_{2,i}(k)$, $i = 1, \ldots, p$, should be a linear combination of $x(k)$ and $y^*_{2,1}(k+r_1), \ldots, y^*_{2,p}(k+r_p)$, where $(r_1, \ldots, r_p)$ is the vector relative degree of the system. We assume, for now, that the state $x(k)$ can be measured or estimated, and stored. Hence, we propose to build, with the available information, the following windowing function:

$$W(\mathbf{x}, \mathbf{y}^*_2) = \begin{bmatrix} x^T(0) & \bar{y}^*_2(0) \\ \vdots & \vdots \\ x^T(N_r) & \bar{y}^*_2(N_r) \end{bmatrix}, \quad (3)$$

where $\bar{y}^*_2(a) = [y^*_{2,1}(a + r_1), \ldots, y^*_{2,p}(a + r_p)]$, and $N_r = N - \max_{j\in\{1,\ldots,p\}}(r_j)$. Using the windowing function $W(\mathbf{x}, \mathbf{y}^*_2)$, we define the following learning process:

$$\mathbf{u}_{2,i} = W(\mathbf{x}, \mathbf{y}^*_2)\theta_i \,, \quad (4)$$

where $\mathbf{u}_{2,i} = [u_{2,i}(0), \ldots, u_{2,i}(N_r)]^T$ is the collection of the i^{th} elements of $\mathbf{u}_2$, obtained from the ILC algorithm. This is a linear regression problem for the parameter vector $\theta_i \in \mathbb{R}^{n+p}$.

Remark 2. *The vectors of unknowns θ_i, $i \in \{1, \ldots, p\}$, are all functions of the system matrices.*

Therefore, we can reuse the calculated vectors θ_i, which build an invariant map, to calculate for new, unseen, desired trajectories correct input vectors that achieve perfect tracking. In particular, we use the vectors θ_i to calculate the control input that achieves perfect tracking of a new desired trajectory $\mathbf{y}^{*,new}_2$ as follows:

$$u^{new}_{2,i}(k) = \begin{bmatrix} x^T(k) & \bar{y}^{*,new}_2(k) \end{bmatrix} \theta_i \quad \forall i \in \{1, \ldots, p\}, \quad (5)$$

where θ_i, $i \in \{1, \ldots, p\}$ are calculated by (4) and $\bar{y}^{*,new}_2(k) = [y^{*,new}_{2,1}(k + r_1), \ldots, y^{*,new}_{2,p}(k + r_p)]$.

Remark 3. *Our proposed control law* (5) *only assumes the knowledge of the vector relative degree of the system, which can be obtained through experiments, see details in [10].*

Notice that the construction of the windowing function $W(\mathbf{x}, \mathbf{y}^*_2)$ requires the knowledge of the system states or estimated values of the states. We extend our approach using Lemma 1 from [14] to use past inputs and outputs of the system instead of state measurements.

It should be noted that the proposed multi-task transfer framework can be also extended to **nonlinear systems** with well-defined vector relative degrees and stable inverse dynamics. Analogous to Lemma 1, it can be shown that there exists a control input satisfying perfect tracking of an arbitrary, smooth trajectory $\mathbf{y}^*_2$, and this input is a nonlinear function of the state $x(k)$ and the values $y^*_{2,1}(k + r_1), \cdots, y^*_{2,p}(k + r_p)$, where $(r_1, \ldots, r_p)$ is the system's vector relative degree. The nonlinear function can be approximated for the whole state space using a nonlinear regression model, or by partitioning the state space and using local, affine/linear models in each region.

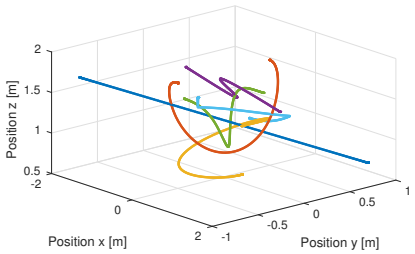

Fig. 2. The six different trajectories that are used to test the multi-robot, multi-task transfer framework. The source and target trajectories used to assess repeatability and different reference models are depicted in dark blue and orange, respectively.

III. EXPERIMENTAL RESULTS

We assess three aspects to verify the effectiveness of the proposed framework: *(i)* capability to transfer different trajectories between different quadrotor platforms, *(ii)* repeatability of results, and *(iii)* use of different reference models for the $\mathcal{L}_1$ adaptive controllers of the source and target systems.

The vehicles used in the experiments are the Parrot AR.Drone 2.0 and the Parrot Bebop 2. Each quadrotor has an underlying $\mathcal{L}_1$ adaptive controller that makes both vehicles behave close to a reference system. In what follows, the signals $u_{2,i}(t)$ are the desired translational positions and $y_{2,i}(t)$ are the quadrotor translational positions in the $i = x, y, z$ directions, respectively. A central overhead motion capture camera system provides position, roll-pitch-yaw Euler angles and rotational velocity measurements. We propose six different trajectories to test our approach, as shown in Fig. 2. To quantify the tracking performance, an average position error along the trajectory is defined by:

$$e = \frac{1}{N}\sum_{i=1}^{N} \sqrt{e^2_x(i) + e^2_y(i) + e^2_z(i)} \,, \quad (6)$$

where $e_j(i) = u_{2,j}(i) - y_{2,j}(i)$ and $j = x, y, z$.

A. Multi-robot, multi-task transfer: capabilities

We generate six different trajectories (see Fig. 2) to be transferred from the AR.Drone 2.0 to the Bebop 2. We learn each of the six trajectories on the AR.Drone 2.0 using the multi-robot framework (Section II-A). We devise a *one-to-all* transfer scheme, in which we use one of the six trajectories as a source trajectory to transfer to each of the six trajectories, now called target trajectories (including the source trajectory, which is the the multi-robot transfer learning case). Using the multi-task framework (Section II-B), we apply the one-to-all scheme six times such that each of the six trajectories is the source trajectory once (6 source $\times$ 6 target trajectories). Fig. 3 shows, for each of the six target trajectories, the tracking error on a Bebop 2 during a learning process at iteration 1 (open circles) and at iteration 10 (filled circles) when no transfer information is used. The tracking errors in the first iteration after applying the one-to-all, multi-robot, multi-task transfer are summarized in six boxplots, each of which corresponds to one target trajectory. The red mark on each box indicates the median, while the bottom and

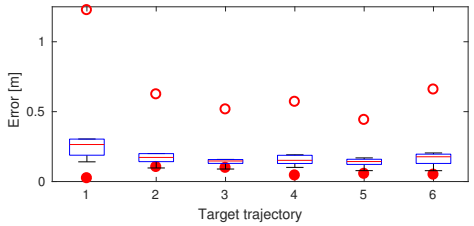

Fig. 3. Tracking errors of six learning processes without transfer information on the Bebop 2 at iteration 1 (open circles) and iteration 10 (filled circles). Boxplots summarize the six one-to-all experiments of the multi-robot, multi-task framework, which significantly decreases the error in the first iteration after transfer. On each box, the red mark indicates the median, while the bottom and top edges indicate the 25^{th} and 75^{th} percentiles, respectively. The whiskers represent the most extreme data points.

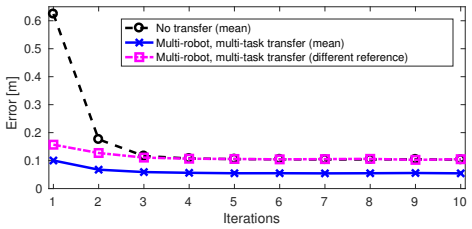

Fig. 4. For a single source and target trajectory pair, the average position errors over ten 10-iteration ILC processes when the Bebop 2 is tracking the target trajectory are shown *(i)* in blue, when the proposed multi-robot (AR.Drone 2.0 to Bebop 2), multi-task (source to target trajectory) framework is used, and *(ii)* in black, when no transfer information is used. The errors after the proposed multi-robot, multi-task transfer when the source and target systems have $\mathcal{L}_1$ adaptive controllers with different reference models are shown in magenta.

top edges of the box indicate the 25^{th} and 75^{th} percentiles, respectively. The whiskers represent the most extreme data points. On average, the percentage of error reduction for the 36 experiments is 74.12%, when using *only 6 seconds* of training data (length of each trajectory).

B. Multi-robot, multi-task transfer: repeatability

To test the repeatability, we choose a single pair of source and target trajectories (in Fig. 2, dark blue and orange trajectory, respectively). Fig. 4 shows the mean error when we repeat ten times a 10-iteration learning process on the Bebop 2 with underlying $\mathcal{L}_1$ adaptive controller and ILC when *(i)* no transfer is used (shown in black), and *(ii)* transfer information from the source trajectory learned on the AR.Drone 2.0 is used to initialize the learning process (shown in blue). The proposed framework significantly decreases the tracking error in the first iteration.

C. Multi-robot, multi-task transfer: different reference models for the $\mathcal{L}_1$ adaptive controllers

We include experimental results for Remark 1, when the source and target systems have $\mathcal{L}_1$ adaptive controllers with different reference models. We modified the reference model of the Bebop 2, and used a mapping between the reference models in addition to the proposed multi-robot, multi-task transfer framework. The tracking error of a 10-iteration ILC process using transfer information is shown in magenta in Fig. 4. The proposed framework reduces the tracking error in the first iteration after transfer by 74.86%, even when the $\mathcal{L}_1$ adaptive controller of the target system has a different reference model.

IV. CONCLUSIONS

We introduced a multi-robot, multi-task transfer learning framework for MIMO systems. We focused on the trajectory tracking problem. The multi-robot transfer learning framework is based on a combined $\mathcal{L}_1$ adaptive controller and ILC. The $\mathcal{L}_1$ adaptive controller makes systems to behave like a specified linear reference model, even in the presence of disturbances, allowing learned tasks to be directly transferred to other systems. The multi-task transfer learning framework uses control theory results to build a time- and state-invariant map from the desired trajectory to the input that accurately tracks this trajectory. This map can be used to generate inputs for new desired trajectories. Experimental results on two different quadrotors and six different trajectories show that the proposed framework reduces the first-iteration tracking error by 74% on average, when information from tracking a different, single trajectory on a different quadrotor is utilized.

REFERENCES

[1] R. Skelton, "Model error concepts in control design," *International Journal of Control*, vol. 49, no. 5, pp. 1725–1753, 1989.

[2] S. Skogestad and I. Postlethwaite, *Multivariable feedback control: analysis and design*. Wiley New York, 2007, vol. 2.

[3] M. K. Helwa and A. P. Schoellig, "Multi-robot transfer learning: A dynamical system perspective," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 4702–4708.

[4] B. Bocsi, L. Csató, and J. Peters, "Alignment-based transfer learning for robot models," in *Proc. of the International Joint Conference on Neural Networks (IJCNN)*, 2013, pp. 1–7.

[5] M. Hamer, M. Waibel, and R. D'Andrea, "Knowledge transfer for high-performance quadrocopter maneuvers," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 1714–1719.

[6] S. Zhou, M. K. Helwa, and A. P. Schoellig, "Design of deep neural networks as add-on blocks for improving impromptu trajectory tracking," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5201–5207.

[7] Q. Li, J. Qian, Z. Zhu, X. Bao, M. K. Helwa, and A. P. Schoellig, "Deep neural networks for improved, impromptu trajectory tracking of quadrotors," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5183–5189.

[8] Y. Duan, M. Andrychowicz, B. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot imitation learning," *arXiv preprint arXiv:1703.07326*, 2017.

[9] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, "Learning modular neural network policies for multi-task and multi-robot transfer," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2169–2176.

[10] K. Pereida, M. K. Helwa, and A. P. Schoellig, "Data-efficient multirobot, multitask transfer learning for trajectory tracking," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1260–1267, 2018.

[11] K. Pereida, R. R. Duivenvoorden, and A. P. Schoellig, "High-precision trajectory tracking in changing environments through $\mathcal{L}_1$ adaptive feedback and iterative learning," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 344–350.

[12] N. Hovakimyan and C. Cao, *$\mathcal{L}_1$ Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2010.

[13] A. P. Schoellig and R. D'Andrea, "Optimization-based iterative learning control for trajectory tracking," in *Proc. of the European Control Conference (ECC)*, 2009, pp. 1505–1510.

[14] F. L. Lewis and K. G. Vamvoudakis, "Reinforcement learning for partially observable dynamic processes: adaptive dynamic programming using measured output data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 1, pp. 14–25, 2011.