Teaching Collaborative Multi-Robot Tasks through Demonstration

Sonia Chernova, Manuela Veloso

Computer Science Department,
Carnegie Mellon University,
Pittsburgh, PA, USA
soniac@cs.cmu.edu, veloso@cs.cmu.edu

Abstract— **Humanoid robots working alongside humans in everyday environments is a long standing goal of the robotics community. To achieve this goal, methods for developing new robot behaviors that are intuitive and accessible to non-programmers are required. In this paper, we present a demonstration-based method for teaching distributed autonomous robots to coordinate their actions and perform collaborative multi-robot tasks. Within the presented framework, each robot learns an individual policy from teacher demonstrations using a confidence-based algorithm. Based on this learning approach, we contribute three techniques for teaching multi-robot coordination using different information sharing strategies. We evaluate and compare these approaches by teaching two Sony QRIO humanoid robots to perform three collaborative ball sorting tasks.**

I. Introduction

One of the central goals of humanoid robot development is to create robots that are able to work alongside humans. Potential applications for humanoids include assistance in hospitals, offices and other work environments, as well as close interaction with families in homes. As robots increasingly become a part of our everyday lives, methods for developing new robot behaviors in a natural and intuitive way that is accessible to non-programmers are required. Inspired by the way humans and animals teach each other, we present a method for teaching humanoid robots to perform collaborative tasks through demonstration.

Teaching by demonstration is a learning approach based on human-robot interaction that provides an intuitive interface for robot programming. Using this approach, a robot learns to imitate the behavior of a teacher by observing a demonstration of the task. In the standard formalization of demonstration learning, a single robot is taught by a single teacher. However, solutions to complex tasks often require the coordination of multiple robots.

In this paper, we build upon our previous work with single-robot [1], [2] and multi-robot [3] demonstration learning and present a study of methods for teaching *emergent multi-robot coordination* [4] through demonstration. Specifically, we are interested in teaching distributed autonomous robots to coordinate their actions and perform collaborative multi-robot tasks. Using our approach, each robot uses the Confidence-Based Autonomy algorithm [2] to learn an individual policy that controls its behavior and the information the robot communicates to its teammates. The solution to the shared multi-robot task emerges from the complimentary actions performed by robots based on their independent policies.

In this paper, we formalize three approaches to teaching emergent collaborative behavior based on different information sharing strategies: *implicit coordination*, *coordination through active communication*, and *coordination through shared state*. We evaluate and compare these techniques by teaching two Sony QRIO humanoid robots to perform three collaborative ball sorting tasks utilizing a continuous and noisy state representation. Based on our evaluation, we conclude that the shared state approach is most suitable for teaching a wide variety of tasks due to two advantages it has over the other methods – a more compact representation requiring a smaller state and action space, and a smaller number of overall demonstrations required to learn the task.

In the following section, we briefly discuss other existing demonstration learning approaches. Section III presents the multi-robot learning framework, followed by the description of three techniques for teaching multi-robot coordination in Section IV. Experimental setup and evaluation are presented in Sections V and VI, respectively.

II. Related Work

Some of the earliest developments in learning from demonstration can be attributed to influential work by Kuniyoshi et al. [5], in which demonstration was used to teach a robotic arm coupled with a stereo vision system. Many additional demonstration learning algorithms have been proposed since that time, with a large number of projects focusing on humanoid robotics applications, including learning arm movement trajectories [6] and gestures [7], social learning methods [8], and teaching complex interactive games such as air hockey [9].

Multiple studies have also examined the role demonstration and imitation can play in multi-agent communities in which agents seek advice from other members of their group. Several projects show improvements over reinforcement learning performance by enabling agents to request and exchange advice with other agents in the environment [10], [11]. Other works enable novice agents to learn by passively observing other agents, without any explicit teaching or interaction [12]. Our approach differs from all of the above techniques in that it enables a single human to explicitly teach independent policies to multiple robots.

III. Multi-Robot Demonstration Learning

Our multi-robot demonstration learning approach is based on the single-robot Confidence-Based Autonomy (CBA) algorithm [2]. The CBA algorithm enables a robot to learn an action policy by requesting demonstration examples from a teacher as it interacts with the environment. Each demonstration is represented by a state-action pair, (s, a), symbolizing the correct action to perform in a particular state, and the policy is learned using supervised learning using state vectors s_i as inputs, and discrete actions a_i as labels. At each decision timestep, CBA evaluates the robot's current state and selects between i) autonomously executing its policy-selected action, and ii) stopping to request an additional demonstration from the teacher. This decision is made based on a measure of confidence in selecting a policy action. Using this approach, the robot acts autonomously only in learned areas of the domain where it is confident about its actions, and stops to request help in new or uncertain situations. Complete details of CBA and its components are presented in [1], [2], [3].

Through its use of adjustable autonomy, the CBA algorithm addresses one of the greatest challenges that prevents most single-robot algorithms from generalizing to multi-robot domains, the problem of *limited human attention* – the fact that the teacher is not able to pay attention to, and interact with, all robots at the same time. The CBA algorithm prevents the autonomous execution of any actions in low-confidence states, making the learner robust to periods of teacher neglect.

Our multi-robot demonstration learning framework utilizes individual instances of CBA for each robot, such that each learner acquires a unique set of demonstrations and learns an individual task policy. Specifically, given a group of robots R, our goal is for each robot $r_i \in R$ to learn policy $\Pi_i : S_i \rightarrow A_i$ mapping from the robot's states to its actions. Note that each robot may have a unique state and action set, allowing distinct policies to be learned by possibly heterogeneous robots. Each robot selects actions based on its individual policy, without awareness of the global interaction taking place. In the resulting emergent coordination, the connection between individual robot actions and collaborative group behavior is not explicitly represented. Coordination between robots occurs when complimentary actions are selected independently by their respective policies.

The general procedure followed by the teacher in performing multi-robot demonstrations is outlined below.

```
Let $D$ be set of pending robot demonstration requests
loop
  if $D \neq \emptyset$ then
    Select request $r \in D$ according to $sel(D)$
    Perform demonstration for robot $r$
  else
    Observe autonomous execution of the robots
    if correction is required for robot $r$ then
      Perform correction for robot $r$
```

Using this approach, the teacher alternates between responding to demonstration requests when they are present, and correcting any mistakes in the autonomous behavior of the robots. The function $sel(D)$, which regulates the selection of demonstration requests, can be used to implement a variety of selection policies, such as a first-in-first-out or round-robin ordering. In the current implementation, a demonstration request is selected at random by the teacher.

IV. Teaching Multi-Robot Coordination

Using the above approach to teach multiple robots, we explore multi-robot coordination in the context of *loosely-coordinated* tasks, which we define as tasks that contain elements that can be independently performed by individual robots, but that require a degree of coordination to couple their execution.

Within this framework, each robot's action set is defined by $A = A_p \cup A_c$, where A_p is the set of physical robot actions and A_c is the set of communication actions. During training, the teacher selects actions for demonstration from the complete action set A.

Multi-robot coordination requires a rich state representation consisting of both local and communicated information. We categorize the robot's state features based on their source and purpose; for example, information locally observed by the robot's sensors may be private to the robot (e.g. current wheel angle), shared with its teammates at all times (e.g. robot position), or shared only under particular conditions (e.g. robot position, but only when it is known with high confidence). Specifically, we define the robot's state as $S = \{F_o \cup F_s \cup F_c \cup F_i \cup F_t\}$ where

- F_o = private, locally observed state features
- F_s = locally observed state features that are automatically communicated to teammates each time their value changes
- F_c = locally observed features communicated using communication actions A_c as defined by policy Π_i
- F_i = internal state features dependent upon other features or robot actions
- F_t = state features containing data either directly contained in, or calculated based on, information communicated from teammates

In this representation, local and communicated data is combined within the robot's state. Coordination between robots occurs when complimentary actions are selected by the policy based on this input. Using this representation, we now present three methods for teaching emergent multi-robot coordination using demonstration.

A. Implicit Coordination without Communication

The most basic level of multi-robot coordination is **implicit coordination**, in which physical actions and observed state allow complementary behaviors to occur without communication or shared intent [13]. Using implicit coordination, robots make decisions based only on locally observed information and are often not aware of the coordination or even of each other's presence. Teaching implicit coordination through demonstration can therefore be reduced to the problem of teaching multiple robots to perform independent tasks at the same time. Within our framework, implicit coordination is represented by the policy

$$\Pi_i : \{F_o, \emptyset, \emptyset, \emptyset, \emptyset\} \rightarrow \{A_p, \emptyset\}$$

which maps the robot's locally observed state directly to the physical actions. Coordination occurs through the environmental changes resulting from the executed actions.

B. Coordination through Active Communication

Domains in which information required for coordination can not be obtained through the robot's own sensors require explicit communication, such as wireless messages. **Coordination through active communication** enables the teacher to use demonstration to explicitly teach *when* communication is required. Based on demonstrations of communication actions A_c obtained from the teacher, communication is incorporated directly into a robot's policy along with the physical actions A_p. This technique enables the teacher to specify the conditions under which communication should take place. The resulting policy is defined by:

$$\Pi_i : \{F_o, \emptyset, F_c, F_i, F_t\} \rightarrow \{A_p, A_c\}$$

While most physical actions have an observable effect that changes the robot's state (i.e. moving an object changes its location), the immediate effect of communication actions is not observable. To prevent the robot from remaining in the same state following a communication action, and from repeating it indefinitely, we utilize internal state features F_i to keep track of the last communicated value of each element of F_c ($\forall f \in F_c \rightarrow f \in F_i$). A mismatch between the value of a particular feature in F_i and F_c indicates that the local state no longer matches the teammates' knowledge. All state information received from other robots through communication is stored within the set F_t.

C. Coordination through Shared State

Robot coordination frequently relies on shared state information that must be maintained up to date at *all* times, not just under specific conditions. For example, a robot performing a navigation task with its teammates may always need to know their locations. **Coordination through shared state** automates the communication process for this common case, enabling robot coordination based on automatically updated state features. Specifically, we define F_s as the set of local features that are automatically communicated to teammates each time their value changes. Coordination through shared state is therefore defined by the policy:

$$\Pi_i : \{F_o, F_s, \emptyset, \emptyset, F_t\} \rightarrow \{A_p, \emptyset\}$$

Since communication occurs automatically, this approach does not require communication actions to be demonstrated or incorporated into the robot policy. Using this technique, the teacher is able to focus on demonstrating only the physical actions to be performed based on state information shared between robots. Note that this approach assumes that shared features do not change very rapidly; attempting to share a sensor value which changes at a high frequency would quickly cause network congestion.

V. Experimental Setup

Experimental evaluation of multi-robot coordination was conducted using Sony QRIO humanoid robots, Figure 1. The QRIO robot is a fully autonomous system enabled with 38 degrees of freedom, onboard processing, stereo vision and speech [14]. The robot's anthropomorphic design and ability to express emotions through human-like motion and speech make it highly suitable for interactive task learning.

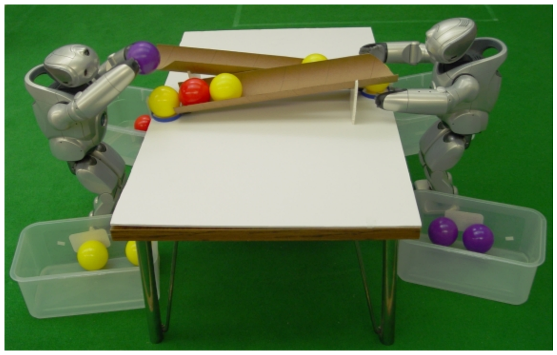

Fig. 1. QRIO robots performing ball sorting task.

A. Human-Robot Interaction

Learning from demonstration is an interactive process that requires two-way communication between the robot and the teacher. When requesting a demonstration, the QRIO robot attempts to audibly attract the teacher's attention by speaking the phrase "What should I do now?". Visually, uncertainty is indicated by an open arm gesture and the lighting of LEDs.

The teacher interacts with each robot via a GUI interface, which allows him or her to select the action to demonstrate from among the available action primitives. In a collaborative setting, a robot's state often depends on information communicated from other robots. Since communicated information is not always easily observable by the teacher, the GUI also displays the robot's current state vector. Both QRIOs utilize the same learning architecture, algorithm and interface, however, the policies learned by each robot are independent as they are based on individual task training received from the teacher.

B. Ball Sorting Domain

Evaluation was performed in a ball sorting domain. Figure 1 shows the robots operating in the domain, which consists of two sorting stations connected by ramps. Each station has an individual queue of colored balls (red, yellow or blue) that arrive via a sloped ramp for sorting. The robots' task is to sort the balls by color into four bins.

The following set of physical actions is available to each robot: A_p = {*SortLeft, SortRight, PassRamp, Wait, Leave*}. Actions *SortLeft* and *SortRight* enable the robot to pick up a ball and place it into a bin on either side. The *PassRamp* action causes the ball to be placed into the teammate's ramp, where it rolls down and takes position at the tail end of the other robot's queue. The *Wait* and *Leave* actions enable the robot to wait for a short duration or walk away from the table, respectively. Each robot determines the color and location of the balls using its onboard vision system. Using these abilities, the robot are taught to perform the following three tasks:

Task 1: Each robot begins with multiple balls of various colors in its queue. QRIO A sorts red and yellow balls into the left and right bins, respectively, and passes blue balls to QRIO B. QRIO B sorts blue and yellow balls into the left and right bins, respectively, and passes red balls to QRIO A. If its queue is empty, the robot waits until additional balls arrive.

Task 2: Extend Task 1 such that each robot communicates to its teammate the status of its queue, empty or full. When its teammate's queue is empty, a robot in possession of a ball should pass the ball to the teammate's queue. However, only balls that can be sorted by the other robot should be passed. For example, QRIO A should pass only the blue and yellow balls, and QRIO B should pass only the red and yellow balls. If both queues are empty, the robots should wait.

Task 3: Each robot begins with multiple balls of various colors in its queue. QRIO A first sorts all of the red balls on the table into its left bin, while QRIO B passes balls of all colors, thereby helping to rotate the queue. Once all the red balls are sorted, QRIO B sorts all the blue balls into its left bin while QRIO A passes. Once all the blue balls are sorted, both robots sort the remaining yellow balls into their right bins. Whenever both queues are empty, the task is complete and the robots leave the table.

Each of the above tasks is designed to test different aspects of multi-robot teaching and coordination. In the first task, coordination between robots emerges naturally based solely on the physical actions of the robots and communication is not required. Task 2 requires coordination through communication to ensure that the sorted balls are distributed more evenly between the robots, while Task 3 adds ordering constraints and additional coordination requirements.

VI. Evaluation

In this section, we present an evaluation of the three approaches to teaching multi-robot coordination presented in this paper. We begin by evaluating each coordination approach independently by applying it to one of the ball sorting tasks. We then evaluate the performance of the communication-based coordination approaches in greater detail using Task 3.

For all evaluations, the robots observe ball color as the average RGB values of the pixels in the detected ball region. Results are reported as the total number of demonstrations required to learn the task, averaged over ten trials.

A. Implicit Coordination without Communication

We evaluate implicit coordination learning by training Task 1, in which the teacher's demonstrations are limited to the robot's physical actions A_p (note that the *Leave* action is not used for tasks 1 and 2). The following state representation, consisting only of locally observed information, is used:

- $F_o = \{R, G, B\}$
- $F_s = F_c = F_i = F_t = \emptyset$

Using this representation, both robots successfully learned their individual policies, which enabled them to collaboratively sort the balls by coordinating through their actions. Training the entire task required an average of 20 demonstrations (10 per robot), with a standard deviation of 1.1.

Note that during the execution of the task, the robots encounter both noisy and noiseless states. The number of demonstrations required to learn each state-action mapping is proportional to the level of noise in the sensor readings. For example, an empty queue contains no ball color information and consistently results in the state vector S ={0,0,0}. Since this value is not affected by noise, only a single demonstration is required to teach each robot to *Wait* when the queue is empty. Ball color readings, on the other hand, vary due to both sensor noise and slight variations in the robot's view angle and ball distance between actions. An average of 3.1 demonstrations are therefore required for the model to learn to generalize over each ball color class.

B. Coordination through Active Communication

Coordination through active communication enables communication to be represented directly within each robot's policy. In this section, we evaluate this approach using Task 2 and the following state representation:

- $F_o = \{R, G, B\}$
- $F_s = \emptyset$
- $F_c = F_i = F_t = \{RampStatus\}$

We define a new boolean feature, F_c ={*RampStatus*}, to represent the status of the robot's ball queue, empty or full. This feature is locally observed and communicated to each robot's teammate using communication action $A_c = \{SendRampStatus\}$. For each robot, feature sets F_i and F_t contain the robot's own last communicated value of *RampStatus* and the most recent value of *RampStatus* received from its teammate, respectively.

The complete state vector consists of six features. Note that all of the information available to a robot, whether continuous or discrete, locally observed or communicated, is combined to form the robot's state vector, allowing the algorithm to generalize over all of these variations. Each feature value is typically normalized by the classifier prior to analysis to give each feature equal weight.

Using the above setup, the teacher required an average of 35 demonstrations to teach Task 2. While each ball color still required multiple demonstrations, the algorithm was able to rapidly generalize over the boolean *RampStatus* features. For example, learning that communication updates must be performed each time the value of the feature *RampStatus* in F_c and F_i does not match, regardless of the ball color being observed, required only two demonstrations.

The challenge of utilizing the active communication approach, however, is the extra responsibility in places on the teacher. In addition to keeping track of fast, unobservable communication messages, the teacher must perform bookkeeping operations in order to teach the robot to maintain a match between its own state and the teammate's knowledge. In the following section we evaluate the way coordination through shared state addresses this drawback.

C. Coordination through Shared State

Coordination through shared state automates the communication process, leaving the teacher to demonstrate the robot's physical behavior based on shared information. In the place of explicit communication actions, this technique utilizes a set of *shared state features*. Any of the robot's locally observed state features may be selected by the teacher to be shared with a teammate. The status of each shared feature is then tracked by the system, and updates are communicated to the teammate each time the value changes.

Coordination through shared state was similarly evaluated using Task 2. For this task, state information shared between robots consists of the state feature $RampStatus$. The complete complete state is represented by:

- $F_o = \{R, G, B\}$
- $F_s = F_t = \{RampStatus\}$
- $F_c = F_i = \emptyset$

where F_s ={*RampStatus*} represents a robot's local shared ramp status, and F_t ={*RampStatus*} represents the ramp status of its teammate. The complete state contains five features, and the entire task required an average of only 27 demonstrations to learn.

D. Combined Approach and Comparison

In the above evaluation, we showed that each of the presented coordination approaches can be successfully used to learn a variation of the ball sorting task. Of the two communication-based approaches, both successfully learned the required policy for Task 2, however, the active communication approach required a greater number of demonstrations. More training data is required by this approach because of its added representation overhead. Compared to the shared state approach, active communication requires an additional state feature ($F_i = RampStatus$) and an additional action class ($SendRampStatus$) for the classifier to learn. This overhead is expensive for communicating a single boolean feature, resulting in better performance by the shared state method.

But what about communication of non-boolean state features, such as a robot's global position? For example, consider a robot that only cares about the location of its teammate under some specific conditions (e.g. once the teammate has found something of interest). In this case, the robot does not need to know its teammates location at any other time, instead, communicating this uninformative information could complicate the learning process, leading to more demonstrations. We hypothesize that the active communication approach may improve learning performance in such cases.

In this section, we use Task 3 as an example of such a domain and compare the performance of three coordination learning methods: active communication, shared state, and a combined approach utilizing a combination of these techniques. In addition to providing further evaluation for the already presented techniques, this example also highlights the flexibility of the presented learning framework in utilizing different data representations.

To represent Task 3, we introduce two additional state features. The feature $SortColor$ is used to represent the current color being sorted (red, blue or yellow), and the feature $PassCount$ is used to represent the number of consecutive $PassRamp$ actions that have been executed by a particular robot. Since the robots do not have a global view of the world, the $PassCount$ feature is required to determine when all balls of a particular color have been removed. For example, when sorting red balls, both robots pass any blue or yellow balls they encounter into their teammate's ramp. The resulting effect is that the entire queue of balls rotates, enabling the robots to examine all balls one at a time. Once the value of the $PassCount$ feature for both robots passes some threshold, in our case 10, it is safe to assume that no additional balls of the current $SortColor$ remain on the table.

	Active Communication	**Shared State**	**Combined**
F_o	$\{R, G, B\}$	$\{R, G, B\}$	$\{R, G, B\}$
F_s	$\emptyset$	$\{PassCount, RampStatus\}$	$\{RampStatus\}$
F_c	$\{PassCount, RampStatus\}$	$\emptyset$	$\{PassCount\}$
F_i	$\{PassCount, RampStatus\}$	$\emptyset$	$\{PassCount\}$
F_t	$\{PassCount, RampStatus, SortColor\}$	$\{PassCount, RampStatus, SortColor\}$	$\{PassCount, RampStatus, SortColor\}$
A_c	$\{IncSortColor\ SendPassCount, SendRampStatus\}$	$\{IncSortColor\}$	$\{IncSortColor, SendPassCount\}$

TABLE I
REPRESENTATION OF TASK 3 USED FOR EACH LEARNING METHOD.

In total, three pieces of information are to be communicated between robots: 1) the current color being sorted ($SortColor$), 2) the number of consecutive passes each robot has performed ($PassCount$), and 3) the status of each robot's ball queue ($RampStatus$). Actions available to the robots for performing this task include the previously defined physical actions A_p, and the communication actions $SendRampStatus$, $SendPassCount$, and $IncrementSortColor$.

Before discussing each coordination approach, we define the $SortColor$ state feature in detail. Unlike previously encountered communicated features, the sorting color is a value that is common to both robots; to achieve accurate performance, the robots must maintain the same value for this feature at all times. As a result, instead of representing this value both as a local and teammate copy of the information, we use a single value for each robot. The value of this feature can be updated both locally and through communication from the teammate using the $IncrementSortColor$ action. Specifically, the execution of this action by a robot first increments its local copy of the variable, and then communicates the new value to its teammate where it immediately updates the other robot's state. Additionally, this action resets the value of the $PassCount$ feature to 0. In summary, this approach achieves state synchrony between robots through the use of a single feature and a multi-function communication action.

Table I presents a summary of the complete state and action sets used for Task 3 by each learning method. Note that for each approach, the information locally observed by each robot (F_o) and received from its teammate (F_t) remains the same. The differences consist of the local representation of the state features to be communicated. Below we summarize the distinguishing features of each approach.

Active Communication – In the active communication approach, both the $PassCount$ and $RampStatus$ features are communicated explicitly using communication actions. The teacher selects the $PassCount$ feature to be communicated each time its value reaches 10. This representation requires a total of 10 state features and 8 action classes.

Shared State – In the shared state approach, the values of

	Implicit	Active Comm.	Shared State	Combined
Task 1	10.1 ± 1	10.1 ± 1	10.1 ± 1	-
Task 2	-	17.3 ± 2	13.3 ± 2	-
Task 3	-	135.0 ± 6	52.6 ± 7	92.2 ± 5

TABLE II
EVALUATION RESULT SUMMARY: AVERAGE NUMBER OF DEMONSTRATIONS PER ROBOT FOR EACH TASK.

all communicated features are shared each time their value changes. The main difference between this approach and the active communication method is that each robot always knows the exact number of passes that its teammate has performed. This approach utilizes a more compact representation, requiring 8 state features and 6 action classes.

Combined Approach – In the combined approach, active communication is used for some state features, and shared state for others. Specifically, shared state is used for the $RampStatus$ feature, the value of which must be communicated each time it changes. Active communication is used for the $PassCount$ feature, the value of which is communicated only once it passes a set threshold[1]. This representation uses a total of 9 state features and 7 action classes.

Table II presents the results of this experiment, along with a summary of previous evaluations. Somewhat surprisingly, we find that the shared state approach significantly outperforms both active communication and combined approaches, requiring far fewer demonstrations. This important result indicates that even in the presence of variables spanning a range of values that contain no useful information, the shared state approach requires less training data. This suggests that learning a threshold for a particular input feature (in our case $PassCount$) is easier for the underlying classifier than dealing with an additional state feature ($F_i = PassCount$) and output class ($SendPassCount$). We note, however, that this analysis is far from exhaustive and does not eliminate the possibility of a domain for which active communication, or a combined approach, will result in better performance. Nevertheless, we conclude that for a large range of applications, especially domains with a large number of boolean or discrete features, shared state provides the most efficient solution.

VII. CONCLUSION

Natural and intuitive methods for developing novel robot behaviors are required to achieve the goal of humanoid robots working alongside humans. In this paper, we presented a method for teaching robots through demonstration, focusing on the specific problem of teaching multiple robots to coordinate their actions. We contributed three approaches for teaching multi-robot coordination and evaluated each technique using two Sony QRIO humanoid robots performing variations of a ball sorting task. In our detailed evaluation, we showed the generality and flexibility of the proposed teaching method, highlighting the algorithm's ability to support coordination based on many factors: action effects, environmental changes, state features which must be communicated to teammates each time they change, state features which are communicated only under specific conditions, and state features that are common to and must be maintained in sync among all robots.

Of the proposed approaches to teaching multi-robot coordination, we find implicit coordination to be the most general, as it is present in almost any application, but also the most limiting, as it does not support explicit communication between robots. Among the proposed communication-based methods, the shared state approach consistently required fewer demonstrations than both the active communication and combined approaches. We conclude that shared state is the most suitable approach for teaching a wide variety of tasks due to its two advantages over other methods – a more compact representation and the resulting smaller number of overall demonstrations required to learn the task. Additionally, our experience shows that teaching using shared state is easier for the teacher, who has less information to keep track of and can focus on the physical elements of the task. However, we remain open to the possibility that a type of domain exists for which the active communication approach may be more suitable. Identifying these types of domains remains an interesting area for future work.

[1]Alternatively, *PassCount* could also be defined as a boolean feature representing whether enough passes have occurred or not. We do not use this representation here for evaluation purposes.

REFERENCES

[1] S. Chernova and M. Veloso, "Confidence-based policy learning from demonstration using gaussian mixture models," in *Proceedings of AAMAS'07*, May 2007.

[2] ——, "Interactive policy learning through confidence-based autonomy," *Journal of Artificial Intelligence Research (to appear)*.

[3] ——, "Teaching multi-robot coordination using demonstration of communication and state sharing (short paper)," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AMMAS '08)*, May 2008.

[4] S. Ossowski and R. Menezes, "On coordination and its significance to distributed and multi-agent systems: Research articles," *Concurr. Comput. : Pract. Exper.*, vol. 18, no. 4, pp. 359–370, 2006.

[5] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," in *IEEE Transactions on Robotics and Automation*, vol. 10, 1994, pp. 799–822.

[6] O. C. Jenkins, M. J. Mataric, and S. Weber, "Primitive-based movement classification for humanoid imitation," in *First IEEE-RAS International Conference on Humanoid Robotics (Humanoids-2000)*, 2000.

[7] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *HRI '07: Proceedings of the ACM/IEEE international conference on Human-robot interaction*. New York, NY, USA: ACM, 2007, pp. 255–262.

[8] A. Lockerd and C. Breazeal, "Tutelage and socially guided robot learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[9] D. C. Bentivegna, A. Ude, C. G. Atkeson, and G. Cheng, "Learning to act from observation and practice," *International Journal of Humanoid Robotics*, vol. 1, no. 4, 2004.

[10] J. A. Clouse, "On integrating apprentice learning and reinforcement learning," Ph.D. dissertation, University of Massachusetts, Dept. of Computer Science, 1996.

[11] E. Oliveira and L. Nunes, *Learning by exchanging Advice*. Springer, 2004.

[12] B. Price and C. Boutilier, "Accelerating reinforcement learning through implicit imitation." *J. Artificial Intelligence Research (JAIR)*, vol. 19, pp. 569–629, 2003.

[13] C. Jones, D. Shell, M. Matarić, and B. Gerkey, "Principled approaches to the design of multi-robot systems," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, Workshop on Networked Robotics*, 2004.

[14] T. Ishida, "Development of a small biped entertainment robot QRIO," in *Micro-Nanomechatronics and Human Science, 2004*, 2004, pp. 23–28.