In Situ Interactive Teaching of Trustworthy Robotic Assistants

Sekou Remy
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: sekou@ece.gatech.edu

Ayanna M. Howard
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: ayanna@ece.gatech.edu

Abstract—In this paper we discuss a method for transferring human knowledge to a robotic platform via teleoperation. The method combines unsupervised clustering and classification with interactive instruction to enable behavior capture in a transferable form. We discuss the approach in both simulation and robotic hardware platform to show the capability of the learning system. In this work we also present a definition and associated metric for trustworthiness, and relate this quantity to system performance. Improved performance and trustworthiness are motivations for our application of interactive learning, and we present results that indicate that these were indeed attained.

I. MOTIVATION

It is envisioned that many future robotic applications will feature robots and humans collaborating closely to accomplish their task. To that end, humans will need to be able to trust that their robotic associates will perform the proper task or have the ability to learn how to perform these tasks in the field (in situ). This specific definition of trustworthiness is linked to a robot’s ability to demonstrate that it will perform the correct task with high probability, whether it is being monitored or not. The definition relates to task specific trustworthiness, where the task’s evaluation can only be measured in terms of what was taught or presented.

One of the key elements of trustworthiness is providing the robot with the ability to learn. Several learning methods exist to transfer knowledge from human teacher to a robotic student. Among them are observation, demonstration, exploration and self observation. Some examples of these can be seen in [1], [2], [3]. Scripting [4] is also a valid approach for this transfer, especially when considering static environments, or those where the system is completely understood and able to be modeled. These learning methods have not been widely applied to situations where knowledge can be adapted over time or to situations where it can be transferred from one system to another.

As such, in this paper we apply unsupervised clustering and classification to implement an interactive method of teaching which permits these situations to be addressed. The teaching occurs onsite (in situ) whenever needed.

II. RELATED WORKS

There are several past and current research efforts that seek to address the need of enabling robotic learning so that the robots can perform properly in the real world environments.

There are some that map the input to the output based on derived a priori relationships between these data. Examples of such include [5] which uses a neurocontroller developed through the use of evolutionary programming to control a robotic bulldozer’s actions and [6] which uses a variant of the Markov Decision Process (MDP) to enable robot navigation through an active academic engineering building. Others, like [7], provide flexible representations of systems and models and are tuned prior to use. Still others rely on abstract representations of knowledge stored in the robot and some sort of communications protocol to extract the desired behavior [8].

These approaches, as varied as they come, do not tend to allow active human interaction once the robot has learned the characteristics of a desired task. For example, once the robot has learned the characteristics of navigation in a particular environment, a new environment in many cases requires reprogramming. Also, if the robot has been trained or programmed with one sensor suite, the addition of additional sensors also requires reprogramming as well. Modification of the task, the environment or the robot thus requires the attention of someone intimately familiar with the inner workings of the robot.

There are some works that display some of the important traits to overcome such challenges. One work, [9], features a humanoid robot that uses real time learning to modify an existing behavior. Human interaction does not feature prominently in this work that uses Linear Weighted Projection Regression (LWPR) to enable learning of sufficiently accurate models of the robot in high dimensional spaces.

There are two works [11], [12] that do use human interaction to facilitate robotic learning. These parallel approaches in effect model the operation of the human, thereby endowing the robot to learn how the human would perform the task. The approach in [11] is composed of two methods, scaffolding and moulding, which when combined provide a mechanism that enables new tasks to be learned and incorporated into the robot’s capabilities. [12] presents behavior cloning, a method quite similar to moulding, which also enabled similar learning capabilities. Both approaches feature a three part sequence of training, model development, and model execution. These stages were required for each new behavior that is added.
to the set of behaviors and they were also required for behaviors that needed to be modified. Preprocessing was also required for both approaches, with [12] even generating a grid representation of the world. This method ([12]) was devised for a robotic entrant in the Robocup Rescue Robot League (RRRL) and it like [11] is closely tied to the application and the platform for which it was developed.

To loosen the ties between learning approach and the application/platform to which it is applied, the work presented in [10] seems to be of use. [10] demonstrates a method that permits a robot to learn how to categorize relevant or important classes directly from raw data by itself. Such capability proved quite useful in a situation where information was incrementally presented to the robot, and the performance is cited as better than performing the task using Principal Component Analysis (PCA).

An approach that combines the principles applied in [9], [10], [11], [12] and expands upon them, even in a small measure, would be useful and would likely overcome a broad range of challenges. As individual works, most of these are closely tied to the robots they were developed with, and none of them apply the same care to action spaces that they do to the sensing. In these approaches unsupervised adaptation in thinking and acting has not yet been demonstrated nor has knowledge transfer been shown which would hint at the ability to capture the essence of the learned behavior.

### III. LEARNING ALGORITHM

Teleoperation of a robot is the process of controlling the device at some distance. In effect the thinking portion of the sensing-thinking-acting cycle of robotics is provided by a remote human operator. Interactive instruction is a process through which a teacher provides instruction to a student and gauges the student’s progress over time and adjusts the level of instruction in response to the student’s performance. By combining teleoperation and interactive learning we seek to transfer human knowledge to a robotic platform.

The approach that we present features a human teacher and a robotic student. The teacher provides interactive instruction to the student via teleoperation. When combined with unsupervised clustering and classification, and incorporation of exposure and experience, interactive instruction enables reactive behaviors to be captured in concise and transferable form. Interactive instruction in this case is a process where the student is allowed to demonstrate its capabilities and the teacher provides correction or new capabilities when needed.

The approach is composed of three major components: encoding the sensing and acting spaces, knowledge base population, and the recall of data stored within the knowledge base. Encoding the sensing and acting spaces is a process that accomplishes two tasks: it converts raw sensor data to a sensor state and it converts actuator values to actuator states. Populating the knowledge base is the process that links sensor and actuator states in addition to also storing supporting data. Recall of data from the knowledge base is the process that permits the raw data to generate the necessary actuator values that should be executed. The components, each of which will be discussed in further detail, are implemented by the various blocks shown in Fig. 1.

#### A. Encoding Sensing and Acting Spaces

Encoding the sensing and acting spaces is a process that accomplishes the twin tasks of converting raw sensor data and actuator values to sensor state and to actuator state respectively. A Self Organizing Map (SOM) [13], [14], a type of artificial neural network that was initially designed to implement a data visualization technique, was the key component used for the conversion. The map reduces the dimensionality of data by grouping similar data points together and implicitly clusters the data points.

For this approach, two separate SOMs are used, one to treat the sensor values \( s_i \), and the other for the actuator values \( a_j \). Where \( i \) ranges from 1 to \( N_s \), the number of sensors, and \( j \) from one to \( N_a \), the number of actuators.

The SOM used for sensing was populated with 100 uniquely labeled neurons, each with weights of dimension \( i \) (the same dimension as the sensor values). The weights of these neurons were initially randomly generated. When a set of sensor values \( S = \{ s_i \} \) is presented to the map, the neuron \( x \) with weights closest to \( S \) is rewarded and permitted to become more like \( S \). The neurons in the immediate neighborhood of \( x \), are also rewarded to a lesser degree. All other neurons remain unchanged. After being presented with multiple sensor values, the map through this process of adaptation organizes itself to reflect the similarities in the sensor space. For this work the neurons were dispersed over a 10 by 10 space, somewhat like that displayed in Fig. 2.

In the case where the sensor values \( S \) were presented to the map, the label of neuron \( x \) was used to generate the state that represents this sensor input. This is the process through which sets of sensor values were encoded to sensor states. The same process was applied to perform the same for the actuator values and associated action state. In the latter case the map used was populated with neurons of dimension \( j \), the dimension of the actuator values \( A \).

#### B. Knowledge Base Population

Knowledge base population, a process that over time links sensor states with associated action states, is the key mechanism to capture observed behaviors. The target behaviors were reactive ones so the actions executed during the process of demonstrating the behavior were due only to the sensory data present. For this work a single behavior was learned in an incremental fashion to demonstrate the viability of this approach. When the student encounters sensory input that it recognizes from its knowledge base, it performs the associated action. If there are multiple actions associated with that sensory state, the action is selected based on the related probability distribution. If however, the input is unfamiliar, the robot does nothing.

So with this purpose in mind, the knowledge base itself was represented by capturing the pair wise combination of sensing
and acting states that the teacher demonstrated. The number of times that a given pair wise combination was presented was also associated with this information. This three dimensional representation of the behavior is generated every time new training information is presented, since the maps change with new data as explained in the encoding process.

Note that the behaviors captured in the knowledge base were never explicitly written; whatever was shown was recorded and will be performed. Whether the command was “follow wall” or “pick up can”, the knowledge base captures what was done and when it was done without concern for the high level concept or for the equipment utilized. The expected limitation of local minima is not a tripping point since this method is not to be applied in isolation. Among other things there will always be a human in the loop.

C. Recall of Data from Knowledge base

Recall of data from the knowledge base is the process that permits the raw data to generate the necessary actuator values that should be executed. As previously indicated, the data in the knowledge base was encoded using two SOMs. A decoding process was required so that the raw sensor data and the actuator values were useful. Because of the method used to encode the sensing and acting data, decoding the representation is a straightforward process. When new sensor values are encountered, the sensor state is generated in the same manner previously discussed. The combinations of sensing and action states with the count are searched for items that match the current sensory input state. From the items that match the sensor state, a single triplet is selected randomly with preference given to each triplet based on its count. The action state of the selected triplet is the label of the neuron that has weights which are desired. These weights are the actuator values that are to be executed by the robot.

This approach is not set up to perform interpolation between known points in the sensing or acting spaces. If an exact match does not exist within the confines of the generalization provided by the SOM, no interpolation is performed to generate one. Some will suggest that good results are likely if the robot performs actions associated with the closest match, but since the experience does not exist for a state that is far from the previously demonstrated states, the knowledge base cannot know what action should be generated, so to maintain predictability and trustworthiness, this approach will not be adopted.

D. Summary of Learning Algorithm

While this learning algorithm learns from what it has been shown, it contains an effective mechanism to filter incorrect or incoherent data. The dominant properties of the behaviors are captured with the nuances of the differences teased out through the use of SOMs. There is however the underlying assumption that the sensory-action spaces is rich enough that a human would have been able to perform the same task given the same information.

This approach is special in that it incorporates learned data as it is presented during operation, not afterwards. Further, it incorporates all presented information, and is not concerned with how that information is formatted, or what it represents; if a human could use it, the robot will be able to use it. Finally,
IV. EXPERIMENTAL SETUP

The experimental goal was to assess the algorithm’s performance in simulation and on hardware platforms in both simple and complex environments. The behavior selected was wall following, a standard element in many mobile robotic applications. In each of the experiments, the teacher, after some practice performing the task with the robotic platform, used a joystick to teach the robot.

The experiments performed utilized both hardware and simulated robotic platforms. The Amigobot (Activmedia Robotics, Amherst, NH) pictured in Fig 6 was used as the hardware platform. KiKS [15], a MATLAB based simulator was the primary simulation environment, provided access to a simulated Khepera robot. SRIsim (Activmedia), a precursor to the more popular Stage simulator [16], was also used to provide the simulated Amigobot platform.

A. Arenas

1) Simulated arenas: The arenas shown in Figs 3 and 4 depict the environments that the simulated Khepera robots were exposed to during these experiments. The environment shown in Fig. 4 provided the robot with more complex sensory stimulus, but the same reactive behavior was demonstrated by the teacher in both environments. Fig. 5 depicts the environment in which the simulated Amigobot was operated.

2) Real arenas: There were three arenas in which the Amigobot was operated. The first, shown in Fig. 6 was where the robot was trained, while Figs 7 and 8 show the arenas where the performance was evaluated. These evaluation arenas were complex in that they challenged the system by presenting the robot with surfaces and obstacle configurations with which it had not been trained.

B. Transfer simulated performance from Khepera to Amigobot

The Amigobot, a robot with an almost circular chassis used eight sonar sensors that provided distance measurements. Each sensor operated in the range of 15cm to 5m and the responses of these sonar sensors varied linearly in the range 0 to 5000. The sensors were distributed around the circumference with highest concentration at the front of the robot. The Khepera also possessed eight sensors on its circular chassis. Its sensors were infra red and provided distance measurements in the range 0 mm to 30 mm. These sensors were also dispersed in higher concentration towards the front of the chassis and they produced responses that varied from 1024 to 0. Figs. 9 and 10 show the sensor responses of the two types of sensors. For this work a linear relationship was assumed between the two sensor types and Eq. 1 shows the relationship implemented, where $x$ was an Amigobot sonar sensor reading and $y$ was the calculated equivalent Khepera infrared sensor reading.$\text{Eq. 1}$

$$y = \frac{1024}{5000} \times x + 1024$$

C. Performance measurement

1) Simulation with Khepera robot: The approach taken to measure performance was a parametric one that was applied to both the simple and the complex environments. The approach can be tailored to any environment. There were two components of the metric, distance from desired path, $d$, and number of time steps to complete the given path, $t$. The
desired path was parameterized to \( N \) points, and the minimum distance from the actual path taken by the robot and each of the \( N \) points was accumulated to generate \( d \). The values of \( N \) used were 36 and 14 respectively for the simple and complex environments. The performance was determined by the following equation

\[ \text{performance} = \alpha_1 d + \alpha_2 t \]  

(2)

Where \( \alpha_1 \) and \( \alpha_2 \) are weight values associated with the two components of the metric. In our case, \( \alpha_1 \) can represent the reciprocal of the accumulated distance between the actual path and the waypoints when an optimal controller was used. Likewise, \( \alpha_2 \) can represent the reciprocal of the number of time steps used to traverse the path when an optimal controller was applied to the robot.

2) Hardware and simulated Amigobot: Since the approach was applied in all cases and the performance was already explicitly measured for one, the process was not repeated in the same manner. For the case where a real robot’s performance was evaluated and the case where the performance of the simulated robot that underwent the knowledge transfer process, a binary test was applied that indicated whether or not the behavior was demonstrated, and not how well the behavior was performed.

D. On Trust

Re-visiting this notion of quantifying trustworthiness, we define that there is a probability \( \alpha \) that the robot will perform the proper task where \( \alpha \in (0, 1) \). In this case, \( \alpha = 0 \) implies that the robot will perform the proper task with probability zero (will never perform the proper task) and \( \alpha = 1 \) implies that the robot will perform the proper task with probability one (will always perform the proper task). The structure of the performance metric applied is such that if the robot demonstrates the correct task, it scores lower on the metric. This means that the lower the performance metric value, the closer the robot’s behavior is to the correct behavior, thus the lower it scores on the metric, then \( \alpha \) approaches to one. For this work the performance metric will be substituted for the measure of trustworthiness.

V. RESULTS

Figs. 3 and 4, which were introduced earlier, show the paths of simulated Khepera robots in simple and complex environments. These images confirm that the robot was indeed able to learn how to wall follow in each of these environments. The performance was displayed after a single interactive learning session with the teacher. Figures 11 and 12 show how the number of interactions vary on average during a training bout. Since length of the bout was fixed and the level of interaction was only related to when the human perceived the robot needed assistance, these graphs clearly capture that there is significantly less human interaction at the end of the bout than at the beginning.

Figs. 13 and 14 confirm why the human interacted less; the robot is performing the shown task better after the initial interaction occurred during the training bout. The robot displays that it will perform the appropriate tasks, hence it is displaying that it is trustworthy.

The simulated and hardware versions of the Amigobot did indeed perform the prescribed wall following task. The hardware version at times had to be rescued since it ran into walls that it was not able to detect. These occasional occurrences were confirmed by verifying the state of the sensor space, and were due in part to the sensor type and location. There were many times that a single sensor was able to detect the presence of a nearby wall. The combination of this single sensor phenomenon with the challenges of sonar sensors such as unwanted or multiple reflections, outlined in [19], served to make demonstrating of reactive wall following difficult, but it was accomplished. After the transfer of the behavior from a simulated Khepera to an Amigobot, the performance of the task was also visually confirmed.

A final point that should be noted is that with interaction, the performance of the robot appears to approach a specific performance value. This phenomenon occurred in both simple and complex environments and future work will have to determine how close this value was to optimal for this behavior.

VI. DISCUSSION AND FUTURE WORK

The results depict that the approach presented works in both simulation and hardware. It was shown that interactive teaching allowed the algorithm to improve its performance after the training episode, thereby being able to demonstrate trustworthiness. This quality was captured in the robot’s consistent demonstration of executing the proper task hence scoring well on the performance metric. Wall following, the reactive behavior implemented for this paper, was not flawless when applied to the hardware platform, however it performed commendably given the limitations of the sensor suite.
Unsupervised clustering and classification, a key component to this approach not only enabled the robot to learn via interactive teaching, but it also enabled transfer of the reactive behavior from a Khepera robot to an Amigobot, robots that differ in size, sensor suite, as well as chassis shape.

Future work will include investigation of alternative approaches to performing classification and clustering especially seeking to apply approaches that grow with the complexity of the spaces used. Attention will also be placed on investigating the performance value that is approached by the algorithm. Comparative analysis will be performed with this and other methods of implementing reactive behaviors with special interest on the upper limits on performance. Finally multiple behaviors will be implemented along with an implementation of interactive teaching via remote teleoperation.

REFERENCES


