ORIGINAL ARTICLE

Vision-Based Imitation Learning in Heterogeneous Multi-Robot Systems: Varying Physiology and Skill

Jeff Allen, John Anderson* and Jacky Baltes

Department of Computer Science University of Manitoba, Canada

(**Received** 15 October 2011; **Accepted** 10 Feburary 2012; **Published on line** 1 June 2012)

*Corresponding author: andersj@cs.umanitoba.ca

DOI: 10.5875/ausmt.v2i2.111

Abstract: Imitation learning enables a learner to improve its abilities by observing others. Most robotic imitation learning systems only learn from demonstrators that are similar physically and in terms of skill level. In order to employ imitation learning in a heterogeneous multi-agent environment, we must consider both differences in skill, and physical differences (physiology, size). This paper describes an approach to imitation learning from heterogeneous demonstrators, using global vision. It supports learning from physiologically different demonstrators (wheeled and legged, of various sizes), and self-adapts to demonstrators with varying levels of skill. The latter allows different parts of a task to be learned from different individuals (that is, worthwhile parts of a task can still be learned from a poorly-performing demonstrator). We assume the imitator has no initial knowledge of the observable effects of its own actions, and train a set of Hidden Markov Models to create an understanding of the imitator's own abilities. We then use a combination of tracking sequences of primitives and predicting future primitives from existing combinations of primitives, using forward models to learn abstract behaviors from demonstrations. This approach is evaluated using a group of heterogeneous robots that have been previously used in RoboCup soccer competitions.

Keywords: Imitation Learning; Multi-robot Systems; Heterogeneity; Hidden Markov Models; Robotic Soccer

1. Introduction

Imitation learning is a powerful mechanism for improving the abilities of an intelligent agent through observing demonstrations of behavior and producing functionally equivalent behavior using the agent's own (possibly distinct) abilities. Evidence of learning from the demonstrations of others can be seen in primates, birds, and humans [1-3]. From an AI perspective, this is attractive because of its potential for dealing with the general problem of knowledge acquisition: instead of programming a robot for each individual task, robots should ultimately be able to gather information from human demonstrations [4-6], or from one another [4, 7, 8] with the result that the robot's performance at that task improves over time. Additionally, demonstrations do not have to be active teaching exercises: the imitator can simply observe a demonstrator with no communication necessary.

To make imitation learning useful, an agent must first have an understanding of its own primitive motor skills (proprioception), observe demonstrations and their outcomes, and ultimately interpret these within the context of its own primitives. In doing so, the agent develops new motor skills by creating hierarchical combinations of primitives [3], providing a deeper understanding of the imitated behavior. In the vast majority of real world settings, an agent will not exist in isolation with a single demonstrator over its lifetime: multiple demonstrations will likely be performed by different agents. Arguably this *should* be the case, since seeing the full range of ways in which a task could be accomplished is faster than the learner discovering these itself, and different agents will likely perform a task in different ways.

Robots have been developed for many different purposes, and as a result, in a multi-robot setting, any robot may encounter others that differ in many ways. These differences range from subtle differences in internal control code, to major differences in physiology — the robots' physical components (sensors, effectors), their size, and the manner in which they function together. For similar reasons of generality, a robot learning by imitation should be exposed to tasks performed by this broad range of potential teammates.

Humans naturally deal with heterogeneous demonstrators in imitative learning: if a child's first exposure to the game of frisbee is through observing a dog catching a frisbee in its mouth, when the frisbee is thrown to the child they will likely attempt to catch it in their hand instead. Humans learn the task in terms of the skills that are natural and available to them, even if the demonstration displayed a different set of skills. In order to similarly increase the performance of a robotic imitation learner and allow it to learn from whatever demonstrators happen to be available in a multi-agent setting (ultimately, a mixture of humans and other robots), overcoming differences in physiology is absolutely essential [9]. Similarly, in working with a range of demonstrators, it will be unlikely that those demonstrators are all equally proficient at any task. An agent must create abstractions to deal with physical differences, but it must also consider the fact that demonstrations may vary widely in quality (in part because of differences in experience, but also as a result of physical differences — some tasks will present greater challenges to wheeled vs. legged robots, for example).

In this paper, we present a framework for imitation learning through global vision, which models multiple demonstrators by approximating the visual outcomes of their actions with those available to the imitator, with no prior knowledge of demonstrators' abilities or physiology. This framework is able to learn from a range of heterogeneous demonstrators (different physiologies, modes of locomotion, sizes, and behavioral control systems), as well as a different range of domain-specific skills. Individually modelling its teachers enables the robot to approximate differences in physiology by actions suited to its own abilities, and to leverage the power of heterogeneous demonstrators to learn portions of a task from one demonstrator that are difficult to approximate from others. It similarly allows an agent to be selective in learning from those who demonstrate better skills in the domain at hand (yet still learn useful portions of a task even from agents that are not skilled).

The experimental domain we use to ground this work is robotic soccer, a common domain in robotics because it presents most of the complex problems associated with intelligent mobile robotics, while remaining understandable to those outside the area. In our evaluation, an imitating robot learns to shoot the soccer ball into an open goal, from a range of demonstrators that differ in size and physiology (humanoid vs. wheeled), as well as in skill level. While this problem may seem trivial to a human adult, it is highly challenging to an individual that is learning about its own motion control. Manoeuvring behind a soccer ball and lining it up for a kick is a difficult task for an autonomous agent to perform, even without considering the ball's destination — just as it would be for a young child. It is also a task where it is easy to conceptualize a broad range of skills (demonstrators that have good versus poor motor control, for example), and one where heterogeneity matters (that is, there are visual differences in how physiologically-distinct robots move).

Jeff Allen received his M.Sc. in Computer Science from the University of Manitoba in 2009. He has competed in international robotics competitions such as FIRA and RoboCup. In 2010 he was a research assistant at the Igarashi Design Interface Project in Tokyo, Japan (part of ERATO: Exploratory Research for Advanced Technology, a research program funded by Japan Science and Technology Agency). While working at the Igarashi Design Interface Project, he co-developed an interactive semi-autonomous robotic puppet interface as part of a human robot interaction project. He is currently Head of Research at Cogmation Robotics, where he works on robotic simulation software and a universal robotic control API. He also works with the University of Manitoba and Cogmation Robotics on joint robotic research projects.

John Anderson is a Professor in the Department of Computer Science at the University of Manitoba. He received his Ph.D. from the University of Manitoba in 1995, in the area of artificial intelligence, and founded the Autonomous Agents Laboratory in the Department of Computer Science shortly afterward. His research interests lie in the areas of multi-agent systems and multi-robot systems, and include cooperative problem-solving, heterogeneous team formation, and social learning, and the application of these to challenging problems such as robotic soccer and urban search and rescue. He is also interested in improving Computer Science education through the use of artificial intelligence and robotics.

Jacky Baltes received his Ph.D. in 1996 from the University of Calgary in Artificial Intelligence. From 1996 to 2002, he worked as a senior lecturer for the University of Auckland in Auckland, New Zealand. Since 2002, he is a professor in the department of computer science at the University of Manitoba in Winnipeg, Manitoba. Prof. Baltes' research interests are intelligent robotics, artificial intelligence, machine learning, and computer vision. Dr. Baltes and his students have competed successfully at various international intelligent robot competitions. Dr. Baltes is also a vice president of the FIRA robotic soccer association, chair of the HuroCup competition, and a member of the RoboCup executive committee. He is also interested in robotics education and a member of the steering committee of the International Robot Olympiad robot competition for high school students.

2. Related Work

A number of prior approaches to imitation learning have influenced this work. Demiris and Hayes [2] developed a computational model based on the phenomenon of *body babbling*, where babies practice movement through self-generated activity [10]. Demiris and Hayes [2] devised their system using *forward models* to predict the outcomes of the imitator's behaviors, in order to find the best match to an observed demonstrator's behavior. A forward model takes as input the state of the environment and a control command that is to be applied. Demiris and Hayes [2] use one forward model for each behavior, which is then refined based on how accurately the forward model predicts the behavior's outcome. By using many of these forward models, Demiris and Hayes construct a repertoire of behaviors with predictive capabilities. These were later extended [11], but used for essentially the same purpose. Similar forward models were used by Dearden and Demiris [12] to model the visual effects of actions, but these explored only a limited state space (one degree of freedom on a gripper) and were not used to deal with heterogeneity or to differentiate between demonstrators at all. In contrast to these prior works, the forward models in our framework model the repertoire of individual demonstrators performing a sophisticated task (instead of having an individual forward model for each behavior), and contain individual behaviors learned from specific demonstrators within them (the behaviors can still predict their effects on the environment, but these effects are not refined during execution). This provides the imitator with a model that can make predictions about what behaviors a specific demonstrator might use at a given time. Representing these separately also allows different (possibly complementary) behaviors to be learned from different demonstrators.

Prior work in imitation learning has often used a series of demonstrations from demonstrators that are similar in skill level and physiologies [6, 13]. The approach presented in this paper is designed from the bottom up to learn from multiple demonstrators that vary physically, as well as in underlying control programs and skill levels.

Some recent work in humanoid robots imitating humans has used many demonstrations, but not necessarily different demonstrators, and very few have modeled each demonstrator separately. Those that do employ different demonstrators, such as Calinon and Billard [13], often have demonstrators of similar skills and physiologies (in this work all humans performing simple drawing tasks) that also manipulate their environment using the same parts of their physiology as the imitator (in this case the imitator was a humanoid robot learning how to draw letters, the demonstrators and imitators used the same hands to draw). Inamura *et al.* [14, 15] use Hidden Markov Models in their mimesis architecture for imitation learning. They trained a humanoid robot to learn motions from human demonstrators, though they did not separately model or rank demonstrator skills relative to each other like we do in our work. They also only have humanoid demonstrators, unlike our work that focuses on multiple heterogeneous demonstrators.

Nicolescu and Mataric [6] motivate the desire to have robots with the ability to generalize over multiple teaching experiences. They explain that the quality of a teacher's demonstration and particularities of the environment can prevent the imitator from learning from a single trial. They also note that multiple trials help to identify important parts of a task, but point out that repeated observations of irrelevant steps can cause the imitator to learn undesirable behaviors. They do not implement any method of modelling individual demonstrators, or try to evaluate demonstrator skill levels as our work does.

3. Methodology

The robots used in this work are shown in Figure 1. The robot imitator, a two-wheeled differential-drive robot (built from a Lego Mindstorms kit, and previously used by us in the RoboCup Small-Size league), is on the far left. One of the three robot types used for demonstrators is physically identical (i.e. homogeneous) to the imitator, in order to provide a baseline to compare how well the imitator learns from heterogeneous demonstrators. Two demonstrators that are heterogeneous along different dimensions are also employed. The first is a humanoid robot based on a Bioloid kit, using a cell phone for vision and processing [16]. The choice of a humanoid was made because it provides an extremely different physiology from the imitator in terms of how motions made by the robot appear visually. The third demonstrator type is a two-wheeled Citizen Eco-Be (version I, shown in close-up in Figure 2) robot, which is about 1/10 the size of the imitator. This was chosen because the large size difference and difficulty in moving a ball due to light weight makes for a different dimension of heterogeneity.

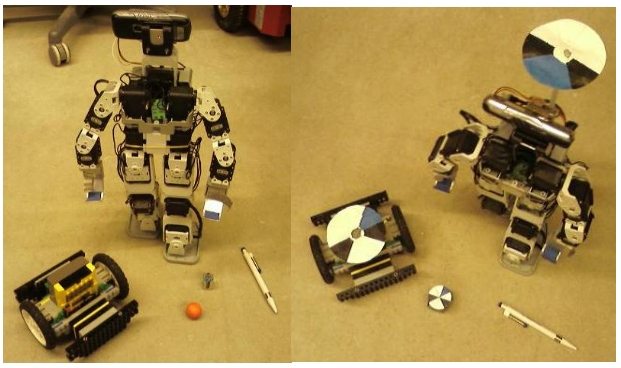


Figure 1. Left: heterogeneous robots used in this work (and a pen to indicate scale). Right: visual markers in place to allow motion to be tracked by a global vision system.

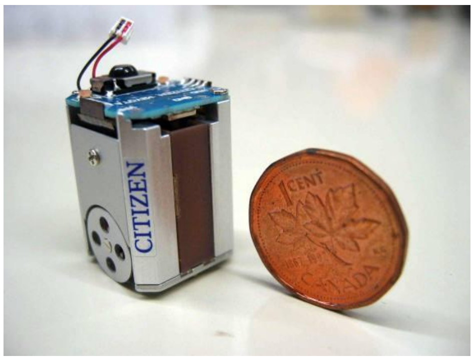


Figure 2. Closer view of the Citizen Eco-Be Microrobot (v.1).

The imitation learning robot observes one demonstrator at a time, with the demonstrated task being that of shooting a ball into an empty goal, similar to a penalty kick in soccer. This task should allow for enough variation between approaches for both different skill levels and different physiologies to have an impact. All knowledge of the task to be learned is gained by observing the demonstrators: no communication between the imitator and its demonstrators is allowed (or necessary).

Any learning robot must begin with a set of simple motion primitives from which it can base more sophisticated behaviors. In our implementation we define five primitive actions reflective of the atomic motor commands available to the wheeled imitator: (*forward*, *backward*, *left*, *right* and *stop*). We assume no initial knowledge of an imitator's own actions, and begin by having the imitator collect visual data of the outcomes of its own primitive actions using the Ergo vision system [17], to create its own model of what it can accomplish in the world. Once the robot has the ability to recognize the outcomes of its own action, primitive mimicry is possible. This must then be extended into abstractions to allow the robot to approximate outcomes for which it has no immediate explanation, and to deal with differences due to heterogeneity (physiology, size, skill) in demonstrations. We deal with these separately in the following subsections.

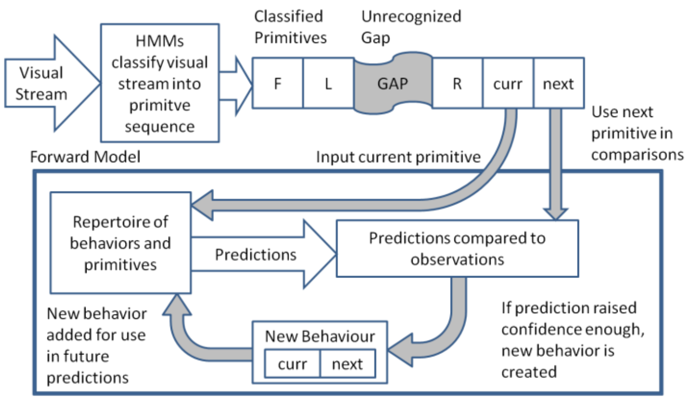
3.1. Converting Demonstration Visual Streams into Sequences of Primitive Symbols

Vision data from the execution of primitives is gathered as the imitator executes them in a random order, instead of gathering data from each primitive separately. This is done to ensure that the model of the imitator's own actions is robust enough to deal with residual motion between primitives. The outcome of each primitive execution ends up occupying 25-30 visual frames (one second) of video, with each frame recording the x and y position and orientation (θ) of the robot. 1000 visual sequences for each primitive type are randomly selected, for a total of 5000 training sequences. This data is then used to train a set of Hidden Markov Models (HMMs) [18], which can be used to match activity it views later to actions in the agent's repertoire.

We chose to use discrete HMMs as they are generally faster to train and use than continuous HMMs [14]. The observation sequences used to train the HMMs in our approach are the sequences of visual frames gathered from the imitator's primitives. Discrete HMMs require a set of discrete symbols to represent all possible observation symbols of a sequence, so the visual frames need to be processed into a codebook of discrete symbols [19]. Each symbol in the codebook represents one of the possible categories to which distinct visual elements of the primitives' visual outcomes can be classified.

To convert the visual data of the primitives into discrete symbols, the vision data is clustered using the K-means algorithm [20]. The vectors used in the clustering algorithm are all three-dimensional, relating to the x, y, and θ (orientation) values of the robot. The clusters are calculated using all 5 of the primitive data sets (for a total of 5000 sequences, and approximately 125,000 individual data points). To split the centroids we used the algorithm described by Linde *et al*. [21], and through experimentation discovered that 512 clusters was an optimal number of discrete observation symbols for the HMMs in our approach.

In our approach to imitation learning, the data recorded in a demonstration (and observed during a trial of the imitator) are the x and y field coordinates of the demonstrator/imitator and the ball, as well as the orientations of the demonstrator/imitator. This data is sufficient for the imitator to learn the chosen task from the collection of demonstrators. During each observed demonstration, the imitator uses its knowledge of the visual effects of its own actions (i.e. the mapping represented by HMMs) to convert the visual stream of a demonstration into a sequence of primitive symbols (Figure 3, top).


Visual Stream
HMMs classify visual stream into primitve sequence
Classified Primitives
Unrecognized Gap
F
L
GAP
R
curr
next
Use next primitive in comparisons
Forward Model
Input current primitive
Repertoire of behaviors and primitives
Predictions
Predictions compared to observations
New behavior added for use in future predictions
New Behaviour
curr
next
If prediction raised confidence enough, new behavior is created
Figure 3. Imitation Learning Architecture

In our implementation, the initial step in converting a demonstration to a sequence of primitives is to segment the demonstration. A demonstration is segmented so that each segment is an atomic piece of a demonstration, in that it can be described by a single imitator primitive. This is achieved by creating a segment whenever the demonstrator has moved (or turned) too far for any of the imitator's primitives to possibly match the motion. We also used a minimum number of visual frames for segments, in order to give the HMMs performing the classification a reasonable amount of motion data to work with. The HMMs can try to classify a single frame as a primitive, though this is futile, as the motion data contained in one frame is so small that there is no way to differentiate between primitives. We use a minimum of 10 frames for each segment, which was determined during preliminary experimentation. We do not use a maximum value, as the segments are only cut off when the motion exceeds the amount of movement possible by any of the imitator's primitives. We determine stop primitives using a similar process: if the minimum number of frames has already occurred, but the motion contained in them is very low (the position of the robot has changed less than 1 millimetre, and the orientation less than 20 degrees), the segment is classified as a *stop* primitive.

If it is determined that a segment is not to be classified as a *stop* primitive, the segment is analyzed by the *forward*, *backward*, *left*, and *right* HMMs. The HMM that provides the highest probability of a match to the segment is chosen, and its primitive is used as the symbol to classify the segment. If two or more HMMs produce probabilities that are too close to each other (within 2%, a value obtained during experimentation) for a given segment, there is no clear frontrunner to classify that segment confidently. In this case no match is reliable, and the segment is instead classified as a *gap*. A gap in a demonstration represents a part of the demonstration where no primitives are capable of achieving the same state change as the segment, and so a higher level abstraction over individual primitives is needed to cover these gaps. For example, if a humanoid demonstrator performs a side-step, and a wheeled robot cannot achieve that same motion with any of its primitives, such a gap would occur. This is to be expected when using demonstrators that are heterogeneous.

3.2. Behaviors and Forward Models

To attempt to learn from portions of a demonstration where a match is poor or no match at all is possible, the imitator must construct a more meaningful abstraction of the demonstration, using behaviors. This meaningful abstraction is similarly useful in overcoming differences in physiology and skill. An implementation-level description of behavior creation and maintenance requires an understanding of all elements of this approach, and so the equations involved are presented following an abstract description.

Behaviors are learned by combining primitives to produce more complex actions based on observations [1, 6, 22]. A method must be used to determine which behaviors and primitives are combined to form new behaviors, to avoid a combinatorial explosion of behaviors that are likely useless. Our implementation is designed to determine the frequency with which primitives and behaviors recognized from the demonstrations follow each other in sequence. When two behaviors or primitives are found to be occurring sequentially frequently enough, a new behavior is created to encompass them both. To determine which behaviors or primitives occur sequentially, behaviors predict state change by applying the average state change (with a small random variation) from each of their primitives in sequence (the primitive state changes are gathered from the same primitive motion data used to train the HMMs). All behaviors' predictions about how they will affect the current state of the field are compared to the next state, and the behavior that has a prediction that best approximates the actual outcome of the demonstration is selected to occur next.

If behavior *A* follows behavior *B* in sequence, it means that behavior *A* was chosen as the best prediction, and then behavior *B* was chosen as the best prediction afterwards. When a behavior follows another in sequence, the element in the frequency matrix representing the frequency of the two behaviors occurring in sequence is increased. In Figure 4, the *L* primitive has just been predicted to follow the *L* primitive, and so the frequency is updated (to 0.3 in the figure). If that frequency surpasses the behavior creation threshold of 0.3, a new behavior is created that encompasses both of the primitives.

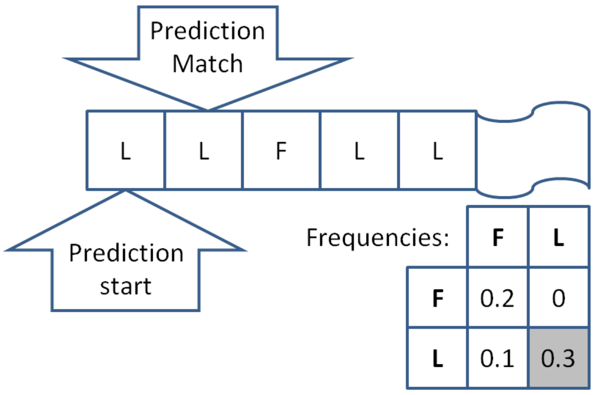


Figure 4. A left primitive is predicted to follow another left primitive, causing the frequency at row *L* and column *L* to be increased.

To keep track of how often behaviors follow each other in sequence, a matrix of behavior frequencies is maintained (the matrix labelled *Frequencies* in Figure 4). All behaviors have their own unique row and column in the frequency matrix. Each column in a given behavior's row represents the frequency that a behavior will follow another. For example, in Figure 4, the *L* behavior has not followed the *F* behavior, and as a result the frequency at row *F*, column *L* is 0.

Since behaviors can be created from other behaviors, the total number of behaviors needs to be kept under control. One of the ways this is done is to represent the behaviors as their component primitives. Instead of representing composite behaviors as links to other behaviors, which may in turn have links themselves, we represent all behaviors as the sequences of primitives that were used to create them, and copy primitives when behaviors are combined.

To keep the number of behaviors learned reasonable, each behavior has a *permanency* attribute, which is used in conjunction with predictive forward models (described below). As the ongoing actions of a demonstrator are observed, the primitive or behavior deemed most likely to occur next is predicted, and confirmed through future observations (which may involve a long sequence of primitives to be matched in the case of complex behaviors). A behavior's permanency is increased if the behavior is observed after being predicted (i.e. it is useful for modelling behavior), and slowly decays over time otherwise, to the point where the behavior is eventually deleted. If a behavior is created that already exists, a second copy is not made. Instead the existing behavior has its permanency increased, because it is useful enough to have been created more than once independently. If the behavior is predicted and then observed frequently enough, the decay rate will slow, and if the permanency attribute surpasses a threshold, the behavior will be marked undeletable.

Forward models are designed to take in the current state of the environment, and make predictions about future states [2, 12]. In our work, behaviors are built and stored using forward models (Figure 3, bottom) which represent frequencies of primitives and behaviors occurring in sequence. It is the behaviors within the forward models in our implementation that are used to explain and predict the behavior of demonstrators in terms of the imitator's repertoire. In our approach, each demonstrator that the imitator is exposed to is assigned a unique forward model. There is also a distinct forward model that represents the behaviors that the imitator itself has learned and acquired. The forward model representing the imitator is the final product of the entire imitation learning process in our approach, and once learning is complete, that forward model can be used to control an imitating robot to achieve the same tasks that it learned from the demonstrators. The forward model representing the imitator can learn its own behaviors like the demonstrator models, but it is also given frequently-used demonstrator behaviors to aid in its learning process. These additional demonstrator behaviors give the imitator a general model of all the useful activity obtained from the demonstrators. A demonstrator forward model learns the various behaviors exhibited by a specific demonstrator, and can be used to predict what that demonstrator might do in any given situation. Unlike other work with forward models in imitation learning, we use separate forward models for each demonstrator so that the relative skill levels of demonstrators can be modelled and compared, and differences in the manner that a task is performed due to physiology can be overcome.

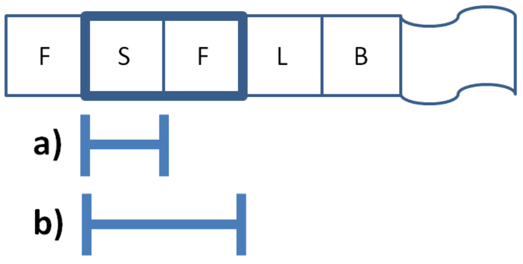


Figure 5. Predictions are compared incrementally down the demonstration's primitive segments. Stage a) represents a match up to one segment past the current behavior, while stage b) represents a match (the dark box) up to two segments past the current behavior.

The imitator should learn how to approximate the result of the demonstrations as efficiently as possible. We have designed the predictive process to ensure that all forward models attempt to span as many segments of the demonstration that they can with their available behaviors. As the predictions are made, some behaviors can make accurate predictions about segments that are further along the demonstration than others. For example, if the demonstration contains a sequence of repeating forward primitives (*forward*, *forward*, *forward*, *forward*, *forward*, ...), and a forward model has a behavior that is composed of three forward primitives in sequence (*forward*, *forward*, *forward*), that behavior can make a prediction that matches the third segment. In our system, as long as the next segment has at least one prediction that matches, the position of the next segment is incremented and all the predictions are compared to this segment. In Figure 5 at stage a) all behaviors have generated predictions about how they will change the state of the first primitive (*F*). All of these predictions are compared to the next segment, in this case a *stop* primitive (the *S*). Since at least one of the predictions is considered a match in stage a), all predictions are compared to the segment following, the second *F* in this example.

In Figure 5 at stage b), at least one prediction was a match, and so the segment being compared moves down another primitive. This process continues as long as at least one of the behavior's predictions matches the next segment. In the example given in Figure 5, no predictions match segments past those matched at stage b), and so stage b) is as far as the current round of predictions can reach in the demonstration. The primitives covered by the predictions in b) are shown outlined in gray. Once no more predictions can match the next segment, the maximum span that the forward model's existing behaviors can approximate has been reached. All behaviors that had a prediction that matched that segment are then evaluated and one is selected as the next behavior. The furthest segment that was matched becomes the new current segment, and in the next round of predictions, the behaviors will predict their effects on this new current segment. The selected behavior has the frequency with which it follows the current behavior increased. The selected behavior becomes the current behavior, and in the next round of predictions it will update the frequency of whatever behavior follows it. If no predictions match even the very first segment compared, then the current behavior is set to whatever primitive is in the next segment, and the prediction process starts over as if the next segment was the beginning of a demonstration.

The accuracy of a prediction is determined by how well the predicted position and rotation of the robot matches with the actual outcome (the next segment of the demonstration). Our implementation is designed to learn the shortest behaviors available to achieve desired environmental changes. To achieve this, the behavior with the smallest number of primitives is chosen from the accurately predicted behaviors. In the case of two or more behaviors tying for the fewest primitives, the behavior that has been in the forward model the longest (the oldest behavior) is taken as the best match. This is done to prevent newer behaviors from being preserved if an existing behavior achieves the same effects.

In our approach, each unique forward model (created by the imitator for each individual demonstrator) begins with only the imitator's primitives. The additional forward model for the imitator itself is used to model how the given task should be performed once imitation learning is complete. Training begins by viewing demonstrations for each demonstrator in turn, training only the forward model for that demonstrator: behaviors are proposed, promoted, and removed through decay as described above. Throughout the training of the demonstrator forward models, frequently occurring behaviors are passed on to the forward model representing the imitator, as suggestions for controlling the imitator's own actions (Figure 6). Following this, each forward model representing a demonstrator is then used to process each demonstration from all demonstrators (Figure 7). This step allows behaviors in one demonstrator model that may not have been the most frequently used, to be further stimulated by the demonstrations of others and passed along to the imitator forward model. That is, a particular movement combination may be useful but not be the best approach for demonstrator *X*, but might improve on some part of the technique demonstrated by demonstrator *Y*. This allows demonstrator *X* to make a partial contribution even if the technique ultimately followed by the imitator more closely resembles that of *Y* (for example, because of physiology differences). Finally, the imitator does the processing of all demonstrations using the candidate behaviors added by the forward models for the demonstrators, allowing the imitator to keep some demonstrator behaviors and discard others, while also learning new behaviors of its own.

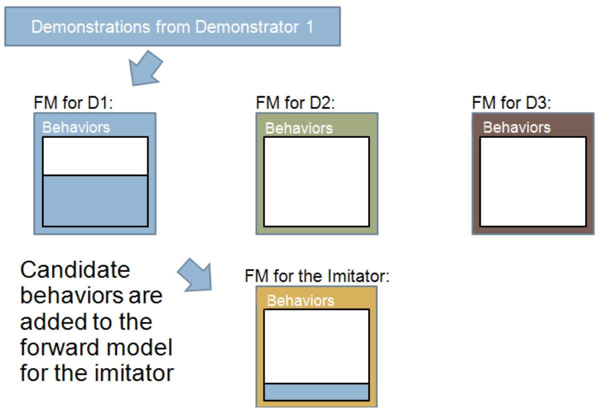

Demonstrations from Demonstrator 1
FM for D1:
Behaviors
FM for D2:
Behaviors
FM for D3:
Behaviors
Candidate behaviors are added to the forward model for the imitator
FM for the Imitator:
Behaviors

Figure 6. Demonstrations from each demonstrator are used to train a forward model representing that demonstrator. Frequently occurring behaviors in each session are moved to the forward model representing the imitator as potential behaviors to use in its own activities.

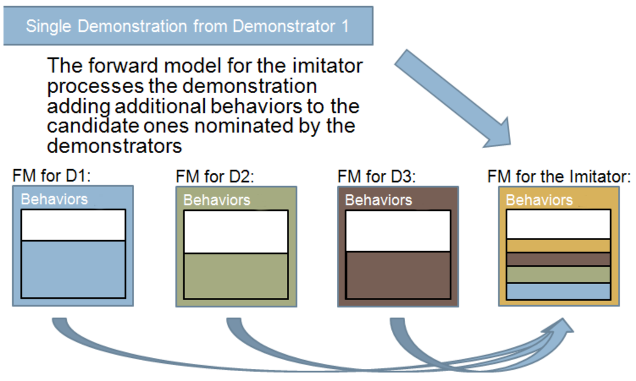

Single Demonstration from Demonstrator 1
The forward model for the imitator processes the demonstration adding additional behaviors to the candidate ones nominated by the demonstrators
FM for D1:
Behaviors
FM for D2:
Behaviors
FM for D3:
Behaviors
FM for the Imitator:
Behaviors

Figure 7. All demonstrations are passed to the demonstrator models to elicit any further candidate behavior nominations.

To model the relative skill levels of the demonstrators in our system, each of the demonstrator forward models maintain a demonstrator-specific learning rate: the *learning preference* (LP). A higher LP indicates that a demonstrator is more skilled than its peers, so behaviors should be learned from it at a faster rate. The LP is used as a weight when updating the frequency of two behaviors or primitives occurring in sequence. The LP of a demonstrator begins at the half way point between the minimum (0) and maximum (1) values.

When updating the frequencies (*freq*) of sequentially occurring behaviors (Equation 1), a minimum increase in frequency (*minFreq* — 0.05 in our implementation) is preserved, to ensure that a forward model for a demonstrator that has an LP of 0 does not stagnate. The forward model for a given demonstrator would still update frequencies, albeit more slowly than if its LP were above 0. Equation 2 shows the decay step, which happens every time a prediction is made, and is how the permanency of all behaviors is slowly decreased. The *decayRate* is equal to 1 − LP and the *decayStep* is a constant (0.007 was used in our experiments). To overcome this constant decay, the permanency of a behavior is increased when it is successfully predicted. The increase in permanency is given in Equation 3, which shows that a correctly predicted behavior has its permanency increased by a constant *permUpdate* (0.09 in our experiments).

$$freq = freq + minFreq + minFreq \times \mathrm{LP} \tag{1}$$

$$perm = perm - decayRate \times decayStep \tag{2}$$

$$perm = perm + permUpdate \tag{3}$$

$$\mathrm{LP} = \mathrm{LP} \pm lpShapeAmount \tag{4}$$

The LP of a demonstrator is increased if one of its behaviors results in the demonstrator (ordered from highest LP increase to lowest): scoring a goal, moving the ball closer to the goal, or moving closer to the ball. The LP of a demonstrator is decreased if the opposite of these criteria results from one of the demonstrator's behaviors. Equation 4 shows the update step, where *lpShapeAmount* is either a constant (0.001) if the LP is adjusted by the non-criteria factors, or plus or minus 0.01 for a behavior that results in scoring a correct/incorrect goal, 0.005 for moving the ball closer to the goal, or 0.002 for moving the robot closer to the ball. These criteria are obviously domain-specific, and are used to shape the learning (a technique that has been shown to be effective in other domains [23]) in our system to speed up the imitator's learning. Though this may seem like pure reinforcement learning, these criteria do not directly influence which behaviors are saved, and which behaviors are deleted. The criteria merely influence the LP of a demonstrator, affecting how much the imitator will learn from that particular demonstrator. Dependence on these criteria was minimized so that future work (such as learning the criteria from demonstrators) can remove them entirely.

When the learning process is complete, the imitator is left with a final forward model that it can use as a basis for performing the tasks it has learned from the demonstrators.

4. Experimental Results

To evaluate this approach in a heterogeneous setting, we employed the robots previously shown in Figure 1 to gather demonstrations. Each of the robots used in these experiments was controlled using its own behavior-based control system that was developed for robotic soccer competitions, and all would be considered expert demonstrations. The Bioloid and Lego Mindstorms robots were demonstrated on a 1020 x 810 cm field, while the Citizen was demonstrated on a 56 x 34.5 cm field (the small size of this robot made for significant battery power issues given the distances covered on the large size field). The ball used by the Bioloid and Lego Mindstorms robots was 10 centimetres in diameter, while a smaller (2.5 cm) ball was needed for the Citizen robot.

We limited the positions to the two field configurations shown in Figure 8. In the configuration on the left, the demonstrator is positioned for a direct approach to the ball. As a more challenging scenario, we also used a more degenerate configuration (on the left, which puts the robot in a position to more easily score on its own goal while manoeuvring).

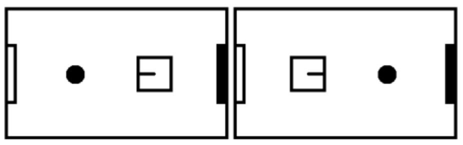


Figure 8. Field configurations. The demonstrator is represented by a square with an orientation marker. The target goal is indicated by a black rectangle.

The individual demonstrators were recorded by the Ergo global vision system [17] while they performed 25 goal kicks for each of the two field configurations. The global vision system continually captures the x and y motion and orientation of the demonstrating robot and the ball. The demonstrations were filtered manually for simple vision problems such as when the vision server was unable to track the robot, or when the robot broke down (falls/loses power). The individual demonstrations were considered complete when the ball or robot left the field. A demonstration could result in a goal on the opposing net (*goal*), a goal on the robot's own net (*wrong goal*), or no goal at all.

One learning trial consists of each forward model representing a given demonstrator training on the full set of kick demonstrations for that particular demonstrator, presented in random order. Once the forward models representing each demonstrator are trained, the forward model representing the imitator begins training. At this point all the forward models for the demonstrators have been trained for their own data, and have provided the forward model representing the imitator with candidate behaviors. The forward model for the imitator then processes all the demonstrations for each of the two field configurations (a total of 150 attempted goal kicks) in random order. All of the forward models for each demonstrator predict and update their models at this time, one step ahead of the forward model for the imitator. (This is done to allow each forward model a chance to nominate additional candidate behaviors relevant to the current demonstration instance, to the forward model for the imitator.)

The total number of goals and wrong goals each demonstrator scored during all 50 of their individual demonstrations is given in Table 1.

Table 1. Demonstrator performance, Goals and Wrong Goals.

Demonstrator	Goals Scored	Wrong Goals Scored
RC2004	27	4
Citizen	15	3
Bioloid	12	1

To determine if the order in which an imitator is exposed to the various demonstrators had any impact on its learning, we ordered demonstrators in two ways. The first is in order of homogeneity to the imitator. In this ordering, the MindStorms robot demonstrator (labelled *RC2004* here because its expert-level control code was from our small-sized team at RoboCup-2004) is first, then the Citizen demonstrator (which is much smaller than the imitator, but still a differential-drive robot), and finally the Bioloid demonstrator. The shorthand we have adopted for this ordering is *RCB*. The second ordering is the reverse of the first, that is, in order of greatest heterogeneity to the imitator. The second ordering is thus Bioloid, Citizen, RC2004, or *BCR* for short.

For each of the two orderings, we ran 100 trials. The results of the forward model training processes using the RCB and BCR demonstrator orderings are presented here. All the following data has been averaged over 100 trials.

Figures 9 and 10 show results for the number of behaviors created and deleted for each of the forward models representing the given demonstrators, with the two orderings for comparison purposes and standard deviations given above each bar. It can be seen that the RCB and BCR demonstration orderings do not affect the number of behaviors created or deleted from any of the forward models. The forward models representing the Bioloid demonstrator can be seen to create many more behaviors than the other forward models (and have a higher standard deviation), but they also end up deleting many more than the others. The vast difference in physiology from the other two-wheeled robots cause the forward models representing the humanoid to build many behaviors in an attempt to match the visual outcome of the Bioloid's demonstrations. When trying to use those behaviors to predict the outcome of the other two-wheeled robot demonstrators, they do not match frequently enough (i.e. they are not a useful basis for imitation), and are eventually deleted as a result.

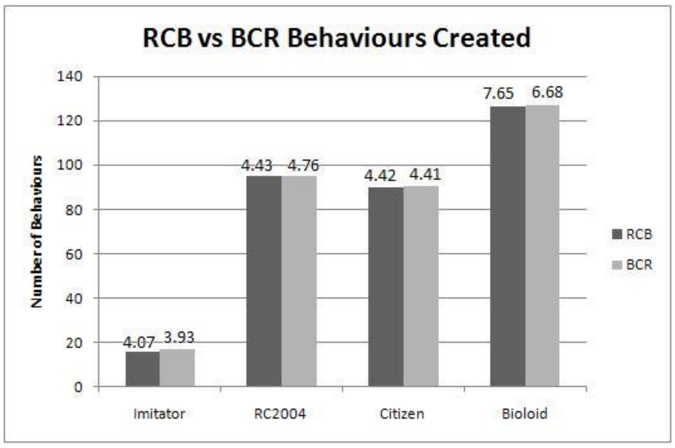


Figure 9. The number of behaviors created, comparing RCB and BCR demonstrator orderings. Corresponding standard deviations are given at the top of each bar.

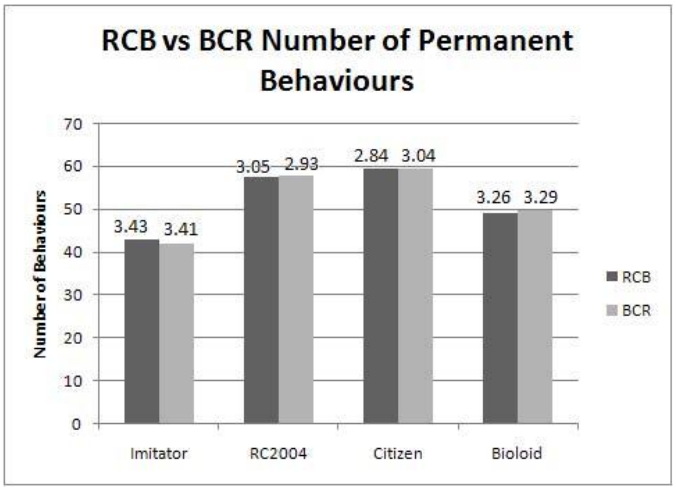


Figure 10. The number of behaviors deleted, comparing RCB and BCR demonstrator orderings. Corresponding standard deviations are given at the top of each bar.

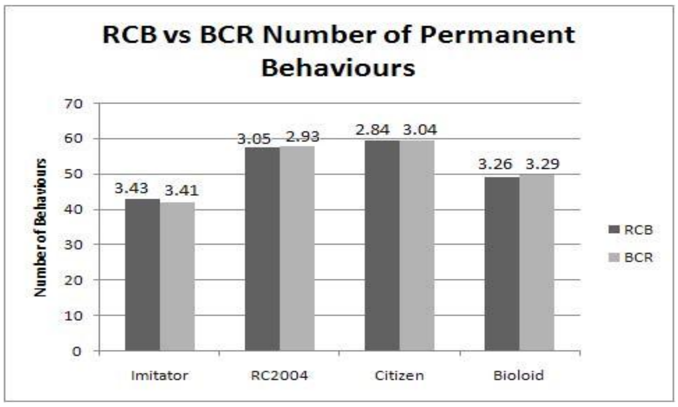


Figure 11. The number of permanent behaviors in each forward model, comparing RCB and BCR demonstrator orderings. Corresponding standard deviations are given at the top of each bar.

In Figure 11, the number of permanent behaviors for each of the forward models are shown along with standard deviations above each bar, grouped by RCB and BCR to see any effect on demonstrator orderings. It can be seen that the orderings do not affect the number of behaviors made permanent to any of the forward models, indicating that ordering does not affect the number of useful behaviors acquired by the forward models representing the demonstrators, or the imitator itself. Even though the Bioloid has a very different physiology, the forward models representing its actions still learn a relatively similar number of behaviors as the other two forward models for the other demonstrators. The forward models representing the imitator have fewer permanent behaviors, partly because the forward model for an imitator filters the candidate behaviors given to it by the forward models representing the demonstrators, but it could also be due to the fact that the imitator is only exposed to each set of demonstrations once, while the other forward models see all demonstrations once, but the demonstrations for their particular demonstrator twice.

To evaluate the performance of the imitators trained using this approach, we selected two imitators from the learning trials evaluated in this section at random (one from the RCB training order, and one from the BCR order). We used the forward models to control the Lego Mindstorms robots and recorded them in exactly the same way that we recorded the demonstrators, for 25 shots on goal in each of the two field configurations (Figure 8) for a total of 50 trials. Table 2 shows the results of these penalty kick attempts by the two imitators trained using our framework. We believe the poor performance is related to the rough statistics used when a forward model is controlling the imitator. The LP shaping criteria are used during the control process for selecting a behavior to execute. The statistical methods used to calculate preconditions were not robust enough given the task at hand, and had small sample sizes to work with. This resulted in the criteria of the robot driving closer to the ball overriding the other LP criteria in most cases. This could be avoided if future work explored methods of gathering more precondition statistics, possibly in simulation for initial training, and then moving to physical robots later.

Table 2. Goals and wrong goals scored by imitators trained with different demonstrator orderings.

Demonstrator Ordering	Goals Scored	Wrong Goals Scored
RCB	11	9
BCR	7	13

Table 3. The number of goals and wrong goals scored for each demonstrator.

Demonstrator	Goals Scored	Wrong Goals Scored
PoorDemonstrator	13	23
AverageDemonstrator	11	9
ExpertDemonstrator	27	4

We also examined the ability of this approach to train an imitator through the observation of demonstrators of varying skill but identical physiology. The physiology chosen was the differential-drive MindStorms robot. Three demonstrators were employed. The *ExpertDemonstrator* runs international competition-level code previously used at RoboCup, while the *PoorDemonstrator* simply turns until it has a minimum angle threshold to the ball and then moves on that heading. Since it will normally take more than one bump with the robot to get the ball to the goal, the latter approach will cause significant wandering over the field and a greater likelihood of scoring on its own net even from the favourable configuration. Finally, there is also an *AverageDemonstrator*, chosen randomly from the imitators trained in the heterogeneity experiments described above. This was done because their performance fell between the two extremes of the other demonstrators, and to illustrate the potential for generational learning using this approach. The actual performance of these demonstrators (in terms of the number of goals and wrong goals scored by each) is shown in Table 3. To avoid any influence of demonstrator ordering on these experiments, during the phase where the forward models representing the demonstrators are trained, each demonstration is chosen randomly.

Figures 12 and 13 show the number of behaviors created and deleted for the various forward models. The forward models for the ExpertDemonstrator have fewer behaviors created than the others, though they also have far fewer of them deleted. This indicates that the behaviors learned by the forward models for the

ExpertDemonstrator are more useful than those learned by the other models. There is not a large difference between the models representing the PoorDemonstrator or AverageDemonstrator. We believe this is due to the control system of the imitator (the AverageDemonstrator) relying too heavily on the LP criteria of its behaviors, which cause it to favour driving toward the ball. As mentioned previously, a larger set of training data would aid in proper pruning of behaviors based on preconditions.

Figure 14 shows that the forward models for the ExpertDemonstrator retain (i.e. make permanent) more of the behaviors they create than the other forward models. This validates our approach to behavior permanencies that decay over time. The less skilled demonstrators have lower LPs, and therefore higher decay rates. Since the forward models representing the ExpertDemonstrator have a higher LP than the others (shown in Figures 15:17), the forward models learn behaviors more quickly, and have their behaviors decay more slowly. The number of behaviors retained by each model is thus strongly related to the LP, which was our intention when employing demonstrator specific learning rates. These results show that our imitation learning architecture adaptively weights its learning toward demonstrators that are highly skilled. At the same time, our approach still allows less-preferred demonstrators to supply behaviors that support portions of behavior that preferred demonstrators cannot (for reasons of physiology difference, for example).

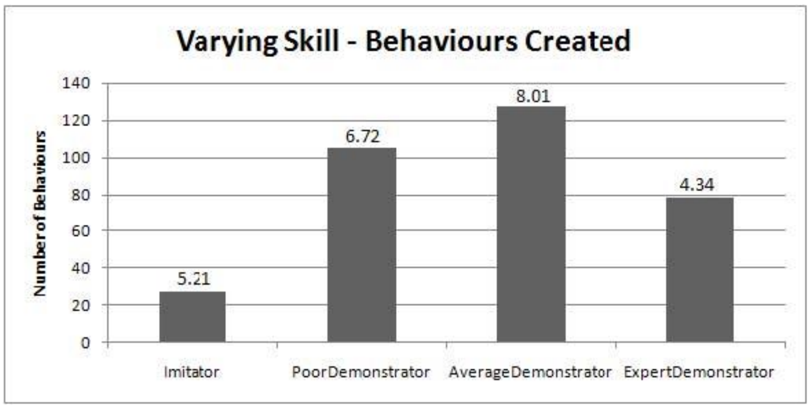


Figure 12. Number of behaviors created. Corresponding standard deviations are given at the top of each bar.

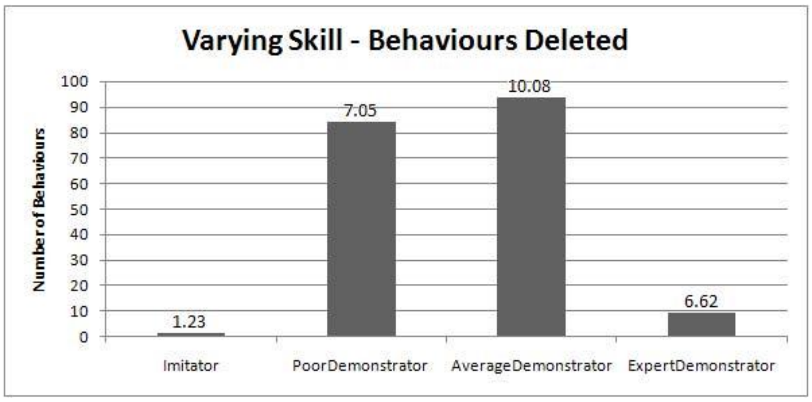


Figure 13. Number of behaviors deleted. Corresponding standard deviations are given at the top of each bar.

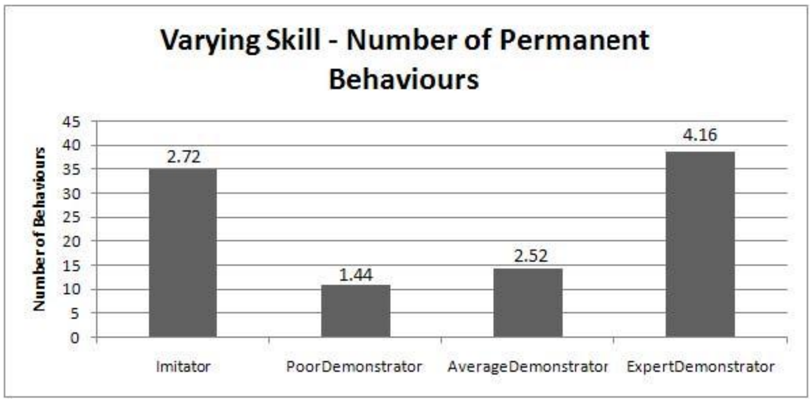


Figure 14. Number of permanent behaviors. Corresponding standard deviations are given at the top of each bar.

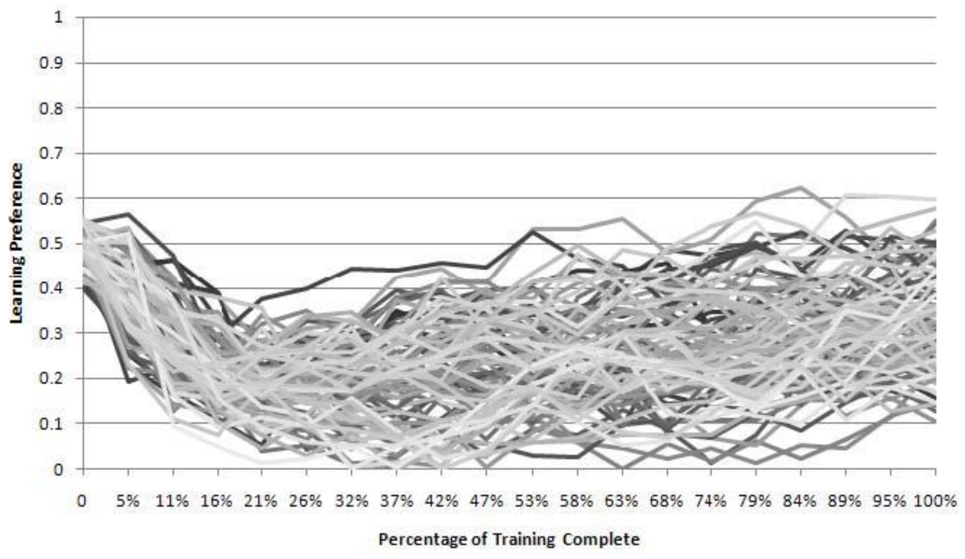


Figure 15. The change in LP over time for the PoorDemonstrator.

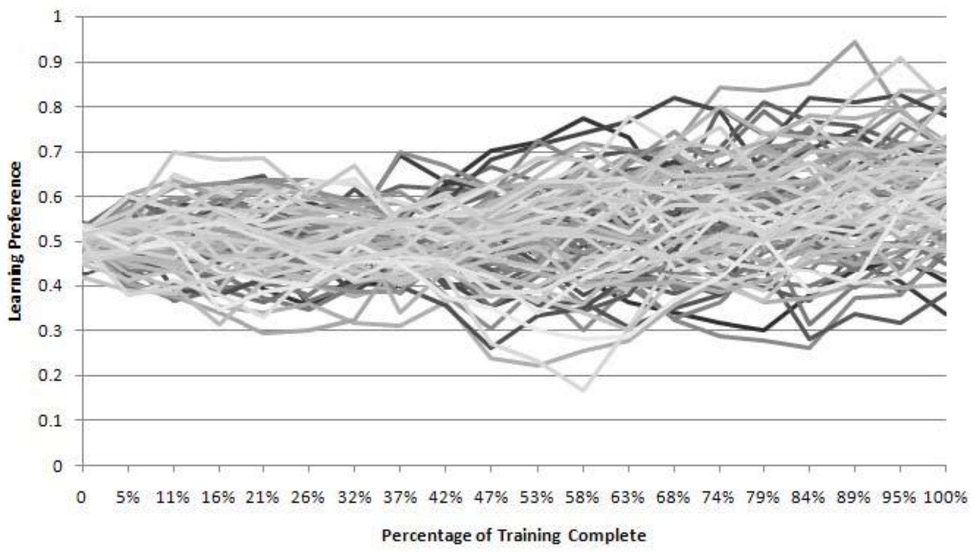


Figure 16. The change in LP over time for the AverageDemonstrator.

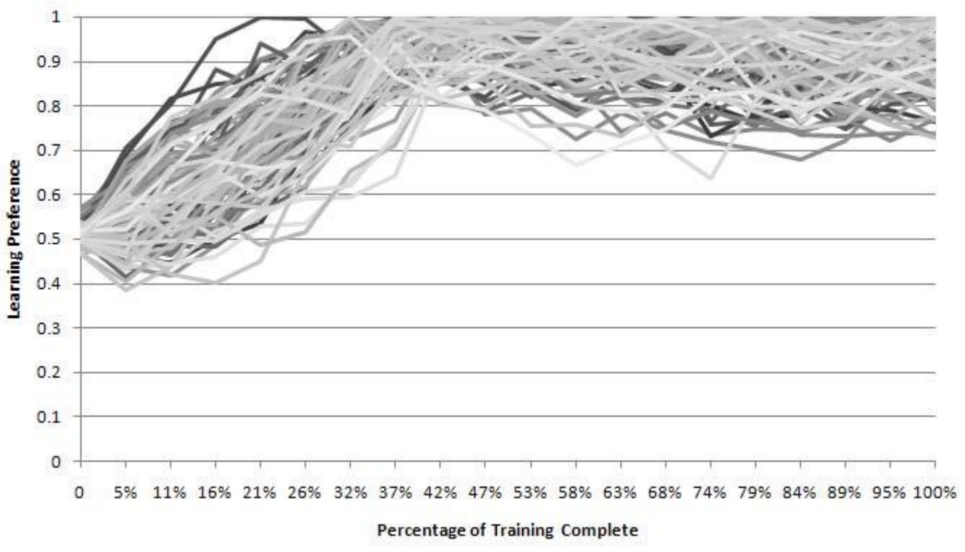


Figure 17. The change in LP over time for the ExpertDemonstrator.

The PoorDemonstrator in these trials is recognized as poorly skilled by the imitation learning architecture fairly quickly, as the forward models representing it have their LP decrease below the average LP value (0.5), and then fluctuate around 0.3. The trend is downwards for most of the PoorDemonstrator’s LP over time, but trends slightly upward as training progresses. We believe that the few behaviors the PoorDemonstrator acquires later in the training phase aid in generating predictions that match the demonstration, which in turn increases the LP of the PoorDemonstrator.

To evaluate the performance of the imitators trained using this approach, we selected an imitator from these trials at random. We used the forward model to control the Lego Mindstorms robot and recorded it in exactly the same way that we recorded the demonstrators, for 25 shots on goal in each of the two field configurations (Figure 8) for a total of 50 trials. Table 4 shows the results of these penalty kick attempts by the imitator trained from demonstrators of varying skill. Though somewhat disappointing in an absolute sense, the performance of a robot using the imitator as a control program still showed that the imitator can learn behaviors from demonstrators and perform the same tasks as the demonstrators. Moreover, this imitator achieves roughly the same results as that trained only with expert demonstrators in the previous experiment, despite having average and poor demonstrators working with it.

Table 4. Goals and wrong goals scored by an imitator trained by demonstrators of varying skill levels.

Imitator	**Goals Scored**	**Wrong Goals Scored**
VaryingSkillTrained	11	13

5. Discussion

We have presented the results and analysis of the experiments used to evaluate our approach to developing an imitation learning architecture that can learn from multiple demonstrators of varying physiologies and skill levels. The complete set of experiments and all results are found in [24]. The results for the performance of our forward models when used as control systems did not perform as well as the expert demonstrators, but they still were able to control the imitator adequately. The main focus on our research was in developing an imitation learning architecture that could learn from multiple demonstrators of varying physiologies and skill levels. The results in Section 4 indicate that the learning architecture we have devised is capable of properly modeling relative demonstrator skill levels and can learn from physiologically distinct demonstrators.

The demonstrators in the varying skill experiments were ranked appropriately by the imitator: the expert had the highest LP and the poor demonstrator had the lowest. More importantly, the imitator trained in the varying skill experiment was as skilled as the imitator trained in the varying physiologies experiment where all demonstrators were highly skilled. This shows that our learning architecture can learn as well when the demonstrators are skilled as when some of the demonstrators are quite poor. A stronger focus on the refinement of behavior preconditions and control (possibly through simulation) similar to the work of Demiris and Hayes [2] would likely make our system even more robust.

Because our focus in this work was on heterogeneity in imitation, we concentrated on developing a complete imitation learning architecture that could be deployed on robots, as opposed to dealing with the many other robotics problems that must be taken into account for robots to operate in the real world. One assumption in this work was the accuracy provided by global vision, and future work will involve applying the architecture presented in this paper to a local vision setting (i.e., individual cameras on each robot). Moving to local vision makes this problem more challenging in a number of ways. First, since global vision provides a common global perspective, the real world coordinates of all objects are known. When using local vision, all coordinates are relative to the viewer, and understanding visual information is thus dependent on an accurate localization of the imitator. The differences in physiology that are a cornerstone of this work will also make the use of local vision significantly more challenging. On a humanoid robot, for example, there is significant side-to-side motion when walking that is not normally present on a wheeled robot, and a locally mounted camera would thus show continual shifting in its view of the world. Unless the imitator compensates for these effects, as humans do when moving, perception will be significantly noisier.

References

[1] A. Billard and M. J. Matarić, "A biologically inspired robotic model for learning by imitation," in *International conference on Autonomous agents*, New York, USA, 2000, pp. 373-380. doi: 10.1145/336595.337544

[2] J. Demiris and G. M. Hayes, "Imitation as a dual-route process featuring predictive and learning components: A biologically plausible computational model," in *Imitation in animals and artifacts*, K. Dautenhahn and C. L. Nehaniv, Eds. Cambridge, MA, USA: MIT Press, 2002, pp. 327-361.

[3] M. J. Mataric, "Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics," in *Imitation in animals and artifacts*, K. Dautenhahn and C. L. Nehaniv, Eds. Cambridge, MA, USA: MIT Press, 2002, pp. 391-422.

[4] C. Breazeal and B. Scassellati, "Challenges in building robots that imitate people," in *Imitation in animals and artifacts*, K. Dautenhahn and C. L. Nehaniv, Eds. Cambridge, MA, USA: MIT Press, 2002, pp. 363-390.

[5] M. J. Mataric, "Getting humanoids to move and imitate," *IEEE Intelligent Systems,* vol. 15, no. 4, pp. 18-24, 2000.
doi: 10.1109/5254.867908

[6] M. N. Nicolescu and M. J. Mataric, "Natural methods for robot task learning: Instructive demonstrations, generalization and practice," in *International joint conference on Autonomous agents and multiagent systems*, New York, USA, 2003, pp. 241-248.
doi: 10.1145/860575.860614.

[7] P. Riley and M. Veloso, "Coaching a simulated soccer team by opponent model recognition," in *International conference on Autonomous agents*, Montreal, Quebec, Canada, 2001, pp. 155-156.
doi: 10.1145/375735.376034

[8] J. Anderson, B. Tanner, and J. Baltes, "Reinforcement learning from teammates of varying skill in robotic soccer," in *FIRA Robot World Congress*, Busan, Korea, 2004.
doi: 10.1023/A:1008819414322

[9] C. L. Nehaniv and K. Dautenhahn, "Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications," in *Interdisciplinary approaches to robot learning*, D. John and B. Andreas, Eds. Singapore; River Edge, NJ: World Scientific, 2000.

[10] A. N. Meltzoff and M. K. Moore, "Explaining facial imitation: A theoretical model," *Early development and parenting,* vol. 6, no. 34, pp. 179-192, 1997.

[11] Y. Demiris and M. Johnson, "Distributed, predictive perception of actions: A biologically inspired robotics architecture for imitation and learning," *Connection Science,* vol. 15, no. 4, pp. 231-243, 2003.
doi: 10.1080/09540090310001655129

[12] A. Dearden and Y. Demiris, "Learning forward models for robots," in *International joint conference on Artificial intelligence*, San Francisco, CA, USA, 2005, pp. 1440-1445.
Available:
http://dl.acm.org/citation.cfm?id=1642293.1642521

[13] S. Calinon and A. Billard, "Learning of gestures by imitation in a humanoid robot," in *Imitation and social learning in robots, humans, and animals : Behavioural, social and communicative dimensions*, C. L. Nehaniv and K. Dautenhahn, Eds. Cambridge, UK; New York: Cambridge University Press, 2007, pp. 153-177.

[14] T. Inamura, I. Toshima, and Y. Nakamura, "Acquiring motion elements for bidirectional computation of motion recognition and generation," in *Experimental robotics VIII*. vol. 5, B. Siciliano and P. Dario, Eds. Berlin; Heidelberg: Springer, 2003, pp. 372-381.

[15] T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *The International Journal of Robotics Research,* vol. 23, no. 4-5, pp. 363-377, 2004.
doi: 10.1177/0278364904042199

[16] J. Baltes and J. E. Anderson, "Complex ai on small embedded systems: Humanoid robotics using mobile phones," in *AAAI Spring Symposium*, Palo Alto, California, 2010.
Available:
http://www.aaai.org/ocs/index.php/SSS/SSS10/paper/viewPaper/1136

[17] J. Baltes and J. Anderson, "Intelligent global vision for teams of mobile robots," in *Mobile robots : Perception & navigation*, S. Kolski, Ed. Mammendorf; Vienna: Advanced Robotic Systems International, 2007, pp. 165-186.

[18] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *ASSP Magazine, IEEE,* vol. 3, no. 1, pp. 4-16, 1986.
doi: 10.1109/MASSP.1986.1165342

[19] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE,* vol. 77, no. 2, pp. 257-286, 1989.
doi: 10.1109/5.18626

[20] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A K-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics),* vol. 28, no. 1, pp. 100-108, 1979.
doi: 10.2307/2346830

[21] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications,* vol. 28, no. 1, pp. 84-95, 1980.
doi: 10.1109/TCOM.1980.1094577

[22] C. A. A. Calderon and H. Hu, "Goals and actions: Learning by imitation," in *International Symposium on Imitation in Animals and Artifacts*, Aberystwyth, United Kingdom, 2003, pp. 179-182. Available: http://cswww.essex.ac.uk/staff/hhu/Papers/AISB2003.PDF

[23] M. J. Matarić, "Reinforcement learning in the multi-robot domain," *Autonomous Robots,* vol. 4, no. 1, pp. 73-83, 1997. doi: 10.1023/A:1008819414322

[24] J. Allen, "Imitation learning from multiple demonstrators using global vision," M.S. Thesis, The Department of Computer Science, The University of Manitoba, Winnipeg, Manitoba, 2009.