Autonomous Parsing of Behavior in a Multi-agent Setting

Dieter Vanderelst* and Emilia Barakova

Designed Intelligence Group, Eindhoven University of Technology, The Netherlands
dieter.vanderelst@emailengine.org

Abstract. Imitation learning is a promising route to instruct robotic multi-agent systems. However, imitating agents should be able to decide autonomously what behavior, observed in others, is interesting to copy. Here we investigate whether a simple recurrent network (Elman Net) can be used to extract meaningful chunks from a continuous sequence of observed actions. Results suggest that, even in spite of the high level of task specific noise, Elman nets can be used for isolating re-occurring action patterns in robots. Limitations and future directions are discussed.

1 Introduction

According to Thorndike [1] an organism is able to learn by imitation if it can acquire new behavioral skills by directly copying them from others. Imitation, like other forms of social learning [2,3], has, potentially, an enormous ecological advantage [4]. It allows animals to be flexible learners while avoiding the dangers associated with individual learning [5]. The behavior of others has often already been shaped by its consequences and can therefore be assumed to be save and rewarding to imitate [2]. Humans and some primates have been found to imitate [6,2,7,8,9].

Another property of imitation, together with its ecological value, is that it can support the spread of behavior through a population of individuals [10,11,12]. Several observational studies [6] have yielded evidence for this in groups of primates but recently also experimental evidence has been reported. For example, Bonnie [8] thought individual chimpanzees to deposit tokens in a box to receive a reward. Subsequently, these individuals were introduced into a population of naive animals. After some time the rewarding behavior was copied by the other animals and its frequency in the population increased. Similar findings have since then been reported by Whiten [13].

The ecological advantages and the capacity to support the spread of behavior make imitation learning a potentially interesting mechanism to support learning in robotic multi-agent systems [14]. In a multi-agent setting, agents could search simultaneously for a solution for a given problem (e.g. how to pick up food). Once a single agent has found a solution, this innovation could be imitated by others and could propagate through the population. In this way, learning by

* Corresponding author.

imitation could drastically reduce the total number of learning trials needed for a population of agents to solve a problem [14].

2 Problem Statement

Imitating agents in a multi-agent setting face a number of fundamental problems [15,16]. One of the most important questions in the multi-agent context is how agents can autonomously select the behavior that should be copied[1]. Great apes and humans seem to be very good at determining what behavior should be imitated when they observe a demonstrator [17]. However, for robots in the multi-agent scenario sketched above, determining what they should imitate is no simple task[2].

To see why this is a problem, imagine a number of agents exploring an artificial world in which different types of food and supplies are available (like in [11]). In this artificial world different resources need to be approached differently to use them. For example, nuts need to be gathered from the ground and smashed against a rock before they can be eaten while a banana must be picked from a tree and carefully peeled. It is assumed that originally all agents are naive concerning the rules governing the world. When the agents are released in the world they start off generating sequences of behavior in the hope of finding a sequence that gives access to some rewarding fruit. Agents that find such a sequence will remember it for future use. This means that after a while agents will alternate between generating new behavior (exploration) and exploiting gathered knowledge (when a fruit for which a known sequence is stored is encountered [19]. Exploitation behavior will consist of fairly fixed action sequences like picking up a nut, smashing it onto a rock and eating it.

An agent that tries to observe and imitate the action sequences of the others sees a continuous sequence of various kinds of actions. The actions which are, from the viewpoint of the observed agents, meaningful and can be parsed into exploring and exploiting parts, are unordered from the viewpoint of a learning agent. There is no a priori way for the learning agent to parse the actions of its colleagues. It can not know which sequences it should copy and which are to be ignored. It should copy the exploiting action sequences and ignore the exploring action sequences. However, it can not know where the exploiting sequences start or end. Even if it is assumed that the imitating agent can detect when an other agent is rewarded (i.e. the end of an exploiting sequence), it can not know where the sequence of events that lead to the reward started.

Parsing a continuous stream of actions is reminiscent of another problem in cognition: that of segmenting a continuous input stream of sounds into words. In a set of seminal papers Elman [20,21] presented a simple recurrent neural network (figure 1a) that was able to segment a sequence of letters into words. His network was constructed to take an input letter n and predict the next

[1] This question has been termed as "What To Imitate?" by Dautenhahn [15].

[2] Even in a setting where humans act as explicit teachers, determining what to imitate is not easy for a robot [18].

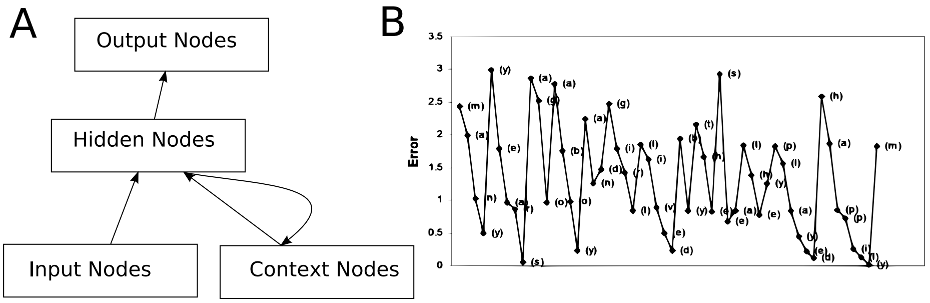

Fig. 1. (a) Schematic representation of an Elman net. On each feedforward sweep, the activation of the Hidden Nodes is copied to the Context Nodes. The Context Nodes are used as an additional set of input nodes on the next feedforward sweep.(b)The error curve produced by an Elman network that was trained to predict the next letter in a sequence from the previous one. Here the sequence "many-years-ago-a-boy-and-a-girl-lived-by-the-sea-they-played-happily" is given to the network.

letter $n+1$ in the sequence from n. After some training the network was capable of making good predictions. The network made its predictions based on the co-occurrence statistics available in the data. Furthermore, segmenting the sequence into words was possible using the error signal produced by the network while executing this task. Figure 1b shows an error curve that was produced while the network processed an input stream of letters. Inspecting the error curve, it can be seen that the error is high at the boundaries between words while it drops over the course of a word. This is caused by the fact that while a word unfolds, the next letter becomes more and more predictable with each new letter. On the other hand, at the boundaries between words, the next letter is very hard to predict since it is not determined by the previous one (in the dataset provided to the net). Therefore, the error provides a good clue as to what are recurring sequences in the input, and these correlate highly with words [20]. The network learned which parts of the input should be regarded as meaningful chunks.

A similar solution might be used to let agents autonomously decide what to imitate (see [22] for a related suggestion). An imitating agent could notice that some sequences of actions are consistently executed in the same order. Rewarding action sequences will be repeated often by the demonstrating robots to reap the fruits of their explorations. Using an Elman-net, an observing agent could try to predict the actions of its fellow agents. After a given amount of training the observer could use the error curve to isolate the segments of the input that are interesting to imitate (or a least to evaluate before attempting imitation). Exploiting sequences will be characterized by being predicable (low predicting error).

The current study focuses on a simplified version of the multi-agent setting sketched in the previous paragraphs. In this study it will be assumed that the demonstrating agent has already discovered several rewarding sequences at the start of the experiment and that it does not learn any new behavior in the course of the experiment. A second simplification is that only a single demonstrating robot will be considered. This setting mimics the situation in the cited experiments of Bonnie and Whiten [8,13] where a single, well-trained, animal is observed by others.

The number of studies related to the current one is very limited. While imitation in multi-agent settings seems to be a promising learning mechanism, until now little or no research has been done in this area[3]. Most research on imitation in artificial agents focuses on human-machine imitation (see [18] for an overview) where the human is a teacher that clearly marks the boundaries of the to-be-imitated behavior. In such a setting there is no need for an agent to detect the boundaries between meaningful chunks of actions since they are marked by the teacher [16]. To the best of our knowledge only [16] has investigated the use of imitation in the context of embodied autonomous multi-agent systems where no explicit teacher is present.

3 Methods

3.1 Experimental Scenario: Selecting Action Sequences

The experimental setting is designed to test whether an Elman network can be used by an observing agent to select interesting sequences in a continuous stream of actions. Therefore, an autonomous robot will be observed while it alternates between exploring (random action sequences) and exploiting (predetermined action sequences). It will be tested whether the network can be used to isolate the predetermined action sequences.

3.2 Data Collection and Preprocessing

All experiments reported in this paper were conducted using the e-puck robot platform (`http://www.e-puck.org`). A single robot was used during the experiments. The e-puck is a small mobile robot measuring 70 mm in diameter and 55 mm in height. The robot is equipped with infrared distance sensors that are located around the body at 10°, 45°, 90°, 270°, 315° and 350° with respect to the heading direction of the robot. Two sensors located at the back of the robot were not used in the reported experiments. The robot was controlled by a personal computer through a Bluetooth interface. A rectangular arena was constructed for the robot which measured about 100 cm $\times$ 70 cm. The arena was fenced by cardboard walls which were about 10 cm high. The robots movements were filmed by a Logitech QuickCam camera (`http://www.logitech.com`) suspended about 160 cm above the floor of the arena. The camera captured the entire arena using 320 $\times$ 240 pixels at 10 Hz. The floor of the arena was white. The robot was fitted with a black cap for maximal contrast so that tracking the robot was easy. All image processing and tracking of the robot was done using RoboRealm

[3] See the recent ECAL workshop on social learning in robot for some explorative papers: `http://laral.istc.cnr.it/slea/`.

software (http://roborealm.com). Processing the images of the camera included correcting for radial distortion. The tracking software provided the approximate location of the center of the robot in each camera frame and its speed.

In this study the robot could only drive straight on and to turn in place. The robot was allowed to drive 4 fixed driving distances: 32 mm, 64 mm, 96 mm and 128 mm. Also 10 turning angles were fixed: -120°, -90°, -60°, -30°, 0°, 30°, 60°, 90°, 120° and 180°. Negative values denote counterclockwise turns. Given the constraints imposed on the movements, all movement of the robot was an alternation between turning in place conform to one of the fixed angles, followed by driving one of the fixed distances. The robot iteratively selected a turning angle and a traveling distance to execute.

The e-puck executed two different kinds of behavior. First, the robot could execute exploration behavior. In this mode a turning angle and a traveling distance were selected at random on each iteration. Second, after each turn and drive action the robot could, with a probability of 0.3, select at random one of 3 patterns to execute. The patterns are depicted in figure 2. Detailed information on the patterns can be found in table 1. While the robot was driving, the distance sensors were probed each 200 ms to determine whether it was about to hit the walls of the arena. If the robot detected a wall, it aborted its current action and moved away from the wall. In case the robot was executing one of the predetermined patterns of action, the pattern was aborted and a random move was initiated after the avoidance maneuver.

In the experiment reported here, the robot executed 1500 moves consisting of turning and driving. This amounted to about 120 minutes and 74702 image frames.

The robot path captured by the camera was preprocessed using R-software [23]. Preprocessing aimed at reconstructing the actions of the robot from the camera images as an observing agent could do. Figure 3 illustrates the preprocessing steps. Note that only a small subset of data have been used for these plots. Plotting an entire dataset would result in graphs that are too cluttered.

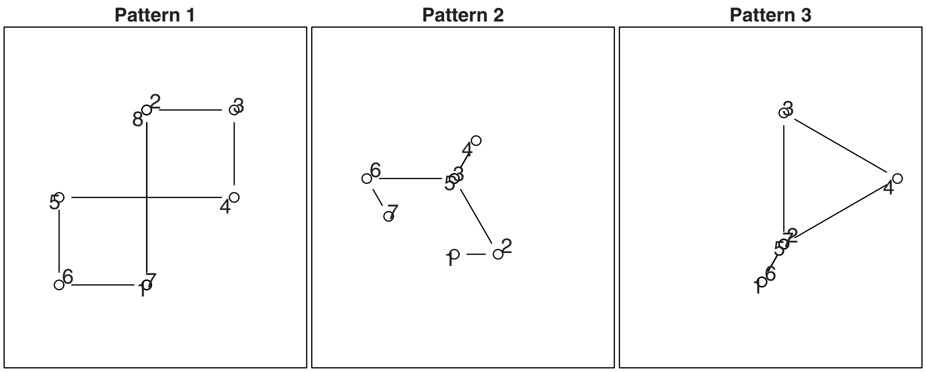

Fig. 2. These are the three predefined patterns that could be driven by the robot. Numbers signify the order of execution. See table 1 for details about these patterns.

Table 1. This table lists the turning angles and the driving distances that made up the 3 predefined patterns plotted in figure 2

	Pattern 1		Pattern 2		Pattern 3	
Step	Angle (Degrees)	Distance (mm)	Angle (Degrees)	Distance (mm)	Angle (Degrees)	Distance (mm)
1	0	128	90	32	30	32
2	90	64	-120	64	-30	96
3	90	64	60	32	120	96
4	90	128	180	32	120	96
5	-90	64	60	64	-30	32
6	-90	64	-120	32	180	32
7	-90	128				

As can be seen in plot 3a-c, the path of the robot consists of random sequences interwoven with a number of predefined patterns. A cubic smoothing spline was fitted to the raw robot path. The smoothed path is plotted in figure 3b. The smoothed track was segmented in order to reconstruct the moves the robot executed. Segmenting the track was done based on the detected speed of the robot. Because the top of the e-puck robot is perfectly round, it looks as if it stands completely still, from the viewpoint of the overhead camera, when it turns in place. Therefore, local minima in the speed curve signify points in time when the robot was (probably) executing a turn. The smoothed robot path was segmented at these points in time. Figure 3c depicts the segmented version of the robot path. Next, the segmentation points were connected by straight lines because the robot could only drive straight on between two turning points. From the segmented path it was trivial to calculate the sequence of the approximate angles the robot turned and the distances it drove.

The final result of the preprocessing step is an approximate reconstruction of the program that has been executed by the robot. This is a record of the actions of the robot as perceived by the camera (or any other onlooker).

Figure 4 depicts the distribution of the angles and distances that were detected by the camera. As can be seen in this graph, the execution and the perception of the moves of the robots was liable to noise. The distribution of the detected distances shows five clusters labeled as *a*-*e*. Cluster *b*-*e* correspond to the 4 distances the robot could drive in the experiment. Cluster *a* contains very short traveling distances. These are caused by over-segmentation of the robot path and by instances in which the driving of the robot was aborted due to the detection of the walls. The distribution of the turning angles also shows overlapping clusters. The fact that both distributions are characterized as a number of overlapping clusters indicates that the level of noise was relatively high.

Because of the noise, some further processing of the perceived moves was necessary before the data could be fed to an Elman network. The turning angles and the traveling distances were discretized. Turning angles were mapped onto the nearest 30°. After this, the data contained 12 different turning angles

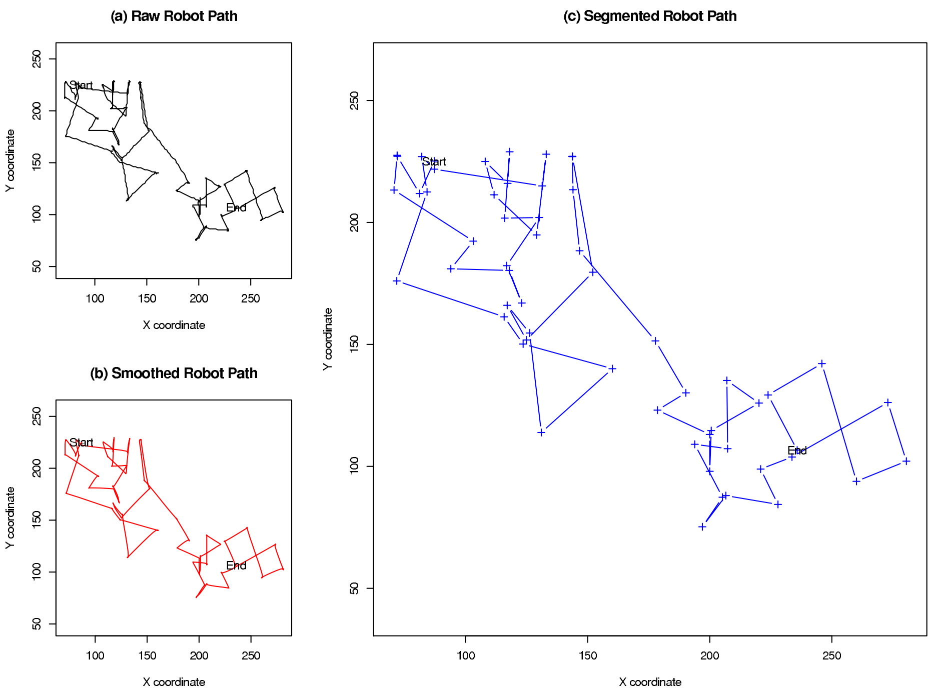

Fig. 3. These plots depict the raw (a), smoothed (b) and segmented (c) path the robot drove in experiment 1. for reasons of clarity only a small subset moves of the robot have been plotted.

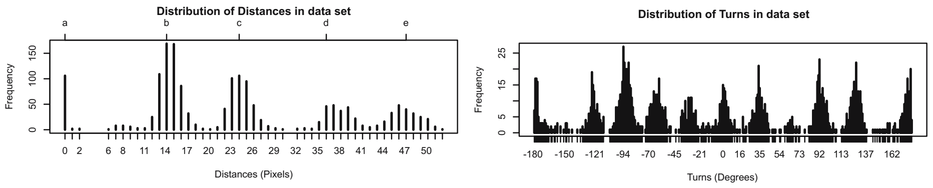

Fig. 4. (a) The distribution of traveling distances detected by the camera in the dataset reported in the results section. Letters $a - e$ refer to five clusters present in this distribution. (b) The distribution of the turning angles in the same data.

(360/12) instead of the 10 used by the robot. Traveling distances were mapped onto the nearest cluster center (0, 14, 24, 36 or 47). Moves that were classified as belonging to the first cluster (i.e. 0), were discarded from further processing. These final data cleaning steps can be considered as reflecting categorical perception.

3.3 Training the Elman Network

A generic Elman network was implemented (see [20,21] for more details about the structure of Elman networks). The network had 12 input nodes coding the turning angles and 4 inputs that coded each of the 4 traveling distances present in the data. The hidden layer of the network consisted of 30 nodes. All neurons had a sigmoid activation function. The network was trained by presenting it with a turning angle and a traveling distance by setting the corresponding input nodes to 1 (other nodes were assigned an activation value of 0). So, an input vector consisted of 16 values of which 2 were set to 1 to signify the current angle and driving distance. Importantly, each turning angle and traveling distance was assigned a coding input neuron at random. In this way, the coding of the moves was completely abstract. Thus, although the predefined patterns were visually symmetrical (see figure 2), from the viewpoint of the network they were not. The visual symmetry could not be exploited by the network to learn to recognize the patterns. After the presentation of an angle and distance, the network predicted the next turning angle and driving distance the robot would execute. Simple gradient descent (Error Backpropagation) was used to adapt the connection weights of the network after each presented input. After updating the network connections, the next turning angle and traveling distance was presented to the network. In this way the whole data set was presented 10 times to the network (i.e. 10 epochs of training). This amounted to about 15000 training trials.

4 Results

The experiment and the training of the network were replicated several times using slightly different parameter settings. However, qualitatively the results were always similar to the ones reported in this section.

Data about the moves executed by the robot can be found in table 2. These data show that a large proportion of the patterns were not completed. This introduced additional noise into the training data.

Figure 5 depicts the most important training results. Plot 5a shows the change in the prediction error by the network in the form of a density plot. One can see that after some initial training, there is a bifurcation in the error. After about 2000 trials, most moves of the robot are well predicted (low error) while others are not (high error). This binomial distribution of the error is also clearly visible in plot 5b. A Gaussian Mixture Model [24] with 2 components was fitted to the error distribution across all trianing trials to obtain an objective threshold to separate trials for which the prediction error was low and trials for the error was high. At about a value of 0.9, an error value had an equal probability of belonging to either of the two clusters (assuming equal priors for both components). This value is indicated by an arrow in figure 5b. This value was used as a cut-off to identify trials in which the network had made a good prediction. Trials in which the network predicted the next step with an error lower than 0.9, were considered as trials with a low error. Subsequently, sequences of trials longer than 2 steps, in which the prediction was better than the cut-off, were identified

Table 2. This table lists the number of patterns the robot drove while collecting the data reported in the results section. Also the number of random moves is listed. The number of patterns and random moves that were terminated because the robot detected a wall are listed separately.

	Termination		
	Normal	Stopped	Total
Pattern 1	39	19	58
Pattern 2	47	21	68
Pattern 3	37	17	54
Total	123	57	180
Random	274	46	320

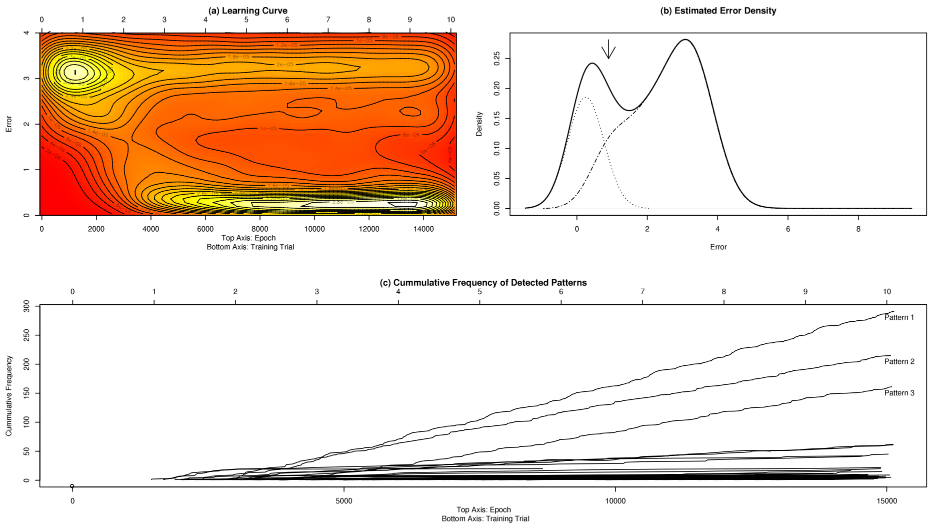

Fig. 5. These plots depict the training results of the experiment. (a) A density estimation of the prediction error of the Elman network in individual learning trials in function of the number of trials. (b) A density estimation of the prediction error collapsed across all learning trials. (c) The cumulative frequency of all different sequences of actions that were well predicted by the network (see text for details).

in the data. The cumulative frequency of all identified sequences is plotted in figure 5c as a function of the number of training trials. Many different sequences were identified ($n = 141$). However, most of these are identified only a few times. Only 10 patterns were recognized 20 times or more while 54 were encountered only once. As can be seen in plot 5c, three patterns are clearly encountered more often than all others. These are the patterns the network was supposed to learn (see labels in plot 5c). So, by analyzing the error curve produced by the Elman

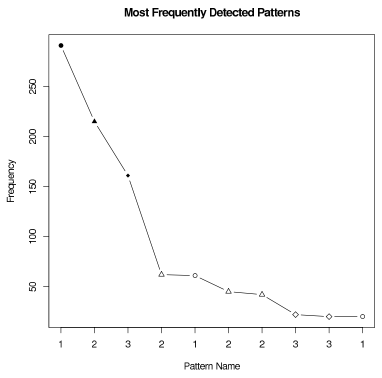

Fig. 6. Total frequency of detection of the most frequently detected patterns of actions. ●; Pattern 1, ▲; pattern 2, ◆; Pattern 3. Opened markers signify subpatterns of the patterns denoted with a similar shape.

network during training, the three predefined patterns could be isolated as being high frequent sequences the network could predict very well.

The fact that the network reliably isolated the three predefined patterns is further demonstrated in figure 6. This plot depicts the frequency of the patterns that were detected more than 19 times in the course of the training. As stated before, the most frequently observed patterns were the three target patterns. Of equal interest is the fact that the other detected patterns were parts of the goal patterns.

5 Discussion

The reported experiment investigated whether an Elman network could be used to reliably isolate re-occurring action sequences. The results show that Elman nets can indeed be used for this task. In contrast to the original simulation studies of Elman [25,20,21], the data collected by observing the robot contained substantial amounts of task-specific noise. Nevertheless, given enough learning trials, the error curve could be used to reconstruct the three goal patterns. Therefore it is possible for an observer to extract the interesting parts of the behavior of a demonstrator by using a simple recurrent network.

However, while the proposition of the study was confirmed by the data, their main valor might be to suggest new lines of research and improvements of the current approach. To this end some issues of the study will be discussed.

A first issue is the learning speed demonstrated in the current study. The Elman network needs many trials before a reliable extraction of the patterns is possible (something that was also experienced by Elman in his original studies). This makes the mechanism, exactly as it is implemented in this paper, an unlikely candidate to be used in a multi-agent setting. Adaptations to speed up learning will be necessary.

The current study is still far away from having a collection of robots learning from each other through imitation. The current experiment modeled one robot (the camera) watching another one continuously (from a favorable perspective). When implementing the current setup with different demonstrators and observers, the noise levels in the perception of each and every robot will rise. Even if robots are allowed to watch each other from the viewpoint used in this paper. A robot will have to choose between a number of agents to observe. Some of these will perform very well while others might be still learning themselves. If the robot chooses to imitate an under trained co-agent, it will not be able to learn the task. Instead, it will extract any re-occurring sequences that are coincidently demonstrated. Furthermore, in absence of any re-occurring pattern in the demonstrators behavior, nothing will be learned. In short, because any learner has access to the behavior of trained as well as untrained individuals, the noise in the perception will increase and the chance of mastering the task decrease. To avoid such a scenario, more flexible learning mechanisms, adapted to multi-agent settings, must be researched. One obvious way to extend the current mechanism is to add reinforcement (or a similar mechanism) - a source of information also used by animals and humans.

Adding reinforcement to the learning mechanism will also reduce the number of learning trials needed. A robot could try out each (partial) sequence of actions it discovers in the behavior of others. If the action sequence is successful according to some measure, the behavior should be consolidated. If not, it should be discarded until further training changes it in some respect. Such a mechanism will speed up learning drastically since a target sequence has to be isolated only once. Furthermore, this way the number of demonstrations of unadaptive behavior in a set of robots is kept at bay which reduces the noise level [10].

We are currently carrying out simulation studies to investigate different options for extending the current research.

References

1. Thorndike, E.: Animal intelligence: an experimental study of the associative process in animals. Psychological Review, Monograph Supplements 2, 551–553 (1889)
2. Zentall, T.R.: Imitation: definitions, evidence, and mechanisms. Animal Cognition 9(4), 335–353 (2006)
3. Noble, J., Todd, P.M.: Imitation or something simpler? modelling simple mechanisms for social information processing. In: Imitation in Animals and Artifacts, MIT Press, Cambridge (2002)
4. Leadbeater, E., Chittka, L.: Social learning in insects–from miniature brains to consensus building. Current Biology 17(16), R703–R713 (2007)
5. Boyd, R., Richardson, P.J.: An evolutionary model of social learning: the effect of spatial and temporal variation. In: Zentall, R.R., Galef, B.j. (eds.) Social Learning: Psychological and Biological Perspectives, pp. 29–48. Erlbaum, Hillsdale (1988)
6. Whiten, A.: The second inheritance system of chimpanzees and humans. Nature 437(7055), 52–55 (2005)

7. Stoinski, T.S., Whiten, A.: Social learning by orangutans (pongo abelii and pongo pygmaeus) in a simulated food-processing task. Journal of Comparative Psychology 117(3), 272–282 (2003)
8. Bonnie, K.E., Horner, V., Whiten, A., de Waal, F.B.M.: Spread of arbitrary conventions among chimpanzees: a controlled experiment. Proceedings in Biological Science 274(1608), 367–372 (2006)
9. Price, E., Caldwell, C.A.: Artificially generated cultural variation between two groups of captive monkeys, colobus guereza kikuyenis. Behavioral processes 27(1), 13–20 (2007)
10. Parisi, D.: Cultural evolution in neural networks. IEEE Expert: Intelligent Systems and Their Applications 12(4), 9–11 (1997)
11. Noble, J., Franks, D.W.: Social learning mechanisms compared in a simple environment. In: Proceedings of the Eighth International Conference on Artificial Life (2002)
12. Nakamaru, M., Levin, S.A.: Spread of two linked social norms on complex interaction networks. Journal of Theoretical Biology 230(1), 57–64 (2004)
13. Whiten, A., Spiteri, A., Horner, V., Bonnie, K.E., Lambeth, S.P., Schapiro, S.J., de Waal, F.B.M.: Transmission of multiple traditions within and between chimpanzee groups. Current Biology 17(12), 1038–1043 (2007)
14. Pini, G., Tuci, E., Dorigo, M.: Evolution of social and individual learning in autonomous robots. In: Ecal Workshop: Social Learning in Embodied Agents (2007)
15. Dautenhahn, K., Nehaniv, C.: An agent-based per- spective on imitation. In: Dautenhahn, K., Nehaniv, C.L. (eds.) Imitation in Animals and Artifacts, MIT Press (2007)
16. Belpaeme, T., de Boer, B., Jansen, B.: The dynamic emergence of categories trough imitation. In: Dautenhahn, K., Nehaniv, C.L. (eds.) Imitation in Animals and Artifacts, MIT Press (2007)
17. Carpenter, M., Call, J.: The question of what to imitate: inferring goals and intentions from demonstrations. In: Dautenhahn, K., Nehaniv, C.L. (eds.) Imitation in Animals and Artifacts, MIT Press (2007)
18. Breazeal, C., Scassellati, B.: Robots that imitate humans. Trends in Cognitive Science 6(11), 481–487 (2002)
19. Thrun, S.B.: Efficient exploration in reinforcement learning. Technical report, Pittsburgh, PA, USA (1992)
20. Elman, J.L.: Finding structure in time. Cognitive Science 14(2), 179–211 (1990)
21. Elman, J.L.: Learning and development in neural networks: the importance of starting small. Cognition 48(1), 71–99 (1993)
22. Moga, S., Gaussier, P.: Artificial neural network for sequence learning. In: International Joint Conference on Artificial Intelligence (2003)
23. Team, R.D.C.: R: A language and environment for statistical computing (2007) ISBN 3-900051-07-0
24. Fraley, C., Raftery, A.E.: MCLUST version 3 for R: Normal mixture modeling and model-based clustering. Technical Report 504, University of Washington, Department of Statistics (September 2006)
25. Elman, J.L.: Distributed representations, simple recurrent networks, and grammatical structure. Machine Learning 7, 195–225 (1991)