On the Evolution of Behaviours Through Embodied Imitation

Mehmet D Erbas
Faculty of Engineering and Architecture
Istanbul Kemerburgaz University
Istanbul, Turkey
Email: mehmet.erbas@kemerburgaz.edu.tr
Phone: +90 212 6040100 - 4022
Corresponding author

Larry Bull
Faculty of Environment and Technology
University of the West of England
Bristol, United Kingdom
Email: larry.bull@uwe.ac.uk
Phone: +44 117 3283161

Alan F T Winfield
Faculty of Environment and Technology
University of the West of England
Bristol, United Kingdom
Email: alan.winfield@uwe.ac.uk
Phone: +44 117 328 2644

Abstract

This paper describes research in which embodied imitation and behavioural adaptation are investigated in collective robotics. We model social learning in artificial agents with real robots. The robots are able to observe and learn each others' movement patterns using their on-board sensors only; so that imitation is embodied. We show that the variations that arise from embodiment allow certain behaviours, that are better adapted to the process of imitation, to emerge and evolve during multiple cycles of imitation. As these behaviours are more robust to uncertainties in the real robots' sensors and actuators, they can be learned by other members of the collective with higher fidelity. Three different types of learned-behaviour memory have been experimentally tested to investigate the effect of memory capacity on the evolution of movement patterns, and results show that as the movement patterns evolve through multiple cycles of imitation, selection and variation, the robots are able to, in a sense, agree on the structure of the behaviours that are imitated.

Keywords: Social learning, behavioural adaptation, embodied imitation, multi-robot systems, swarm intelligence.

I. Introduction

This paper presents research on social learning in a group of robots. The work reported here was undertaken within a research project called "the emergence of artificial culture in robot societies" whose overall aim was to investigate the processes and mechanisms by which proto-cultural behaviours, better described as traditions, might emerge in a free running collective robot system. In a previous paper we described how novel behavioural forms do indeed emerge in a robot collective in which robots have been programmed to learn socially, from each other, by imitation [24]. We showed that behaviours are subject to variation as they are copied from one robot to another, multiple cycles of imitation give rise to behavioural heredity and, when robots are also able to select which learned behaviours to enact, then we have a process of embodied behavioural evolution.

This paper extends that work by focussing on two research questions. First, how do behaviours evolve and adapt as they undergo multiple cycles of embodied imitation and, in particular, do behaviours adapt to be better fitted to the 'environment' of the robot collective and the robots themselves? Second, we seek to understand how clusters of related behaviours arise and persist within the collective memory of the robot group; in particular we explore several approaches to the robots' learned-behaviour memory. We believe these questions to be important in advancing our understanding of the role of embodiment and environment in social learning and - by extension - cultural evolution.

This paper proceeds as follows: In section II, we present research that are particularly related to the paper. In section III, we describe the experimental setup. An embodied movement imitation algorithm is presented in section IV. In section V, we describe a method to quantitatively asses the fidelity of imitation between robots. In section VI, we describe a series of experiments to examine the evolution of movement patterns in a robot group. Finally, section VII concludes the paper.

II. Related Work

A. Research on Imitation in Robotics

In recent years, the study of imitation in robotics has received cross-disciplinary attention [19], [20] as it offers many benefits for increasing the performance of robotic agents. Demiris and Hayes [10] claimed that learning by imitation has certain desirable effects on robots. For instance, it can reduce the solution space for the tasks that the robot is trying to achieve. By using a suitable reinforcement function, the robot can learn the solution to any problem through reinforcement learning; however the presence of an expert can be used to increase the learning speed of the robot. The expert can demonstrate the solution and the learning robot can achieve the solution more quickly by imitation. In addition, learning by imitation may provide the robot with novel solutions which it may not be able to achieve. Learning by imitation does not require the expert to spend additional time or energy in teaching others, as it can continue to perform its task as the learners observe it. No explicit communication is needed, which is beneficial where communication is costly. Bakker and Kunisyoshi [4] claimed that an agent that has the ability to imitate has an increased level of adaptation to its environment. The observed actions are likely to be useful, as they were executed by an agent sharing the same environment. Dautenhahn et al. [8] asserted that the study of imitation in robotics holds the promise of overcoming the need to program every behaviour a robot may need to perform. A robot that is able to imitate can learn new actions by observing demonstrations of those actions.

For an agent to be able to imitate it has to match the observed behaviour of the demonstrator with a behaviour of its own. The problem of the imitating robot finding those matching behaviours has been characterised as the correspondence problem. Nehaniv and Dautenhahn [19] defined the correspondence problem thus:

'Given an observed behaviour of the model, which from a given starting state leads the model through a sequence (or hierarchy) of subgoals - in states, action, and/or effects, while possibly responding to sensory stimuli and external events, find and execute a sequence of actions using one's own (possibly dissimilar) embodiment, which from a corresponding starting state, lead through corresponding subgoals - in corresponding states actions, and/or effects, while possibly responding to corresponding events'

Throughout this paper, when one robot imitates another it performs the following operations:

- observe: The imitating robot watches a demonstrator robot with its on-board camera. Therefore observation is vision-based.
- learn: The imitating robot uses the observed movement patterns of the demonstrator robot to infer a set of moves and turns, i.e. a trajectory, using the algorithm described in section IV. The imitating robot then saves the newly learned movement pattern to its memory so that it can be enacted later.
- enact: The learned movement pattern is converted into a sequence of motor commands which are executed.

The use of real robots instead of simulated agents or biological social entities is motivated as follows:

- Real robots, with their less than perfect perception and actuation - and small differences between robots, provide natural variation in the imitation process which allow new behaviours to emerge and evolve. Using simulated agents in a simulated environment, it would be possible to control the degree and types of heterogeneities and noise, but this may preclude or predetermine any emergent processes that are part of imitation; the level and characteristics of emergence in a simulated environment would be limited to the level of variance that is artificially introduced.
- Data about the imitative activity, including the internal data and calculations of the robots, can easily be extracted and examined. This would not be the case if biological social entities (for example, people or monkeys) were used.
- The implementation of imitation on real hardware makes clear how theoretical assumptions and hypotheses regarding imitation can be operationalised.

In addition, we do not allow robots to transmit behaviours (i.e. sequences of motor actions) directly from one to another. This means that the robots have to overcome essentially the same problems of inferring each other's behaviours from possibly unreliable first-person perceptions as any embodied agents (robots, animals or humans), yet at the same time the robots implement a rather minimal model of social learning by imitation. This embodied yet abstract model of social learning provides both a degree of biological plausibility and opportunities for unexpected emergence that would not be present in an agent-based simulation.

B. Research on Cultural Evolution in Artificial Agents

In work of particular relevance to this paper Acerbi and Parisi [2] examined the cultural transmission between and within generations in a population of embodied agents controlled by neural networks. It was shown that intra-generational cultural

transmission adds new variability so that successful behaviours, that increase the adaptation ability of agents, can evolve in the population. Acerbi and Nolfi [1] presented an adaptive algorithm based on a combination of selective reproduction, individual learning and social learning. They claimed that social learning provides an adaptive advantage when individuals are allowed to learn socially from experienced individuals and individually. Their results show that agents that learn on the basis of both social and individual learning outperform agents that learn on the basis of social learning only or individual learning only. Parisi [22] presented a method in which a neural network is trained so that it demonstrates the same behaviour as another neural network. The two networks were exposed to the same input and the connection weights of the learner network were changed so that the learner network progressively learn to behave like the teacher network. He claimed that if random noise is added to this training process, some students may have higher performance than their teachers. This random noise allows the evolution of behaviours in a group of social agents as it is sufficient to trigger the evolution of useful behaviours. The work presented in this paper is also relevant to the research on the evolution of language. In a work particularly relevant to this paper, Kirby [14] argued that the language must be transmitted between generations through a repeated cycle of use and learning. In the evolution of language, compositional syntax may have emerged not because of its utility to us, but rather because it ensures that the language can be transmitted successfully. The process of linguistic transmission is itself an adaptive system which operates on a time scale between individual learning and biological evolution. In another research, Kirby et al. [15] claimed that languages, as they are culturally transmitted, evolve so that they can be transmitted with high fidelity. In an experimental scenario, they showed how a basic artificial language became easier to learn and structured as it was transmitted in a group of human participants. In this research, it is shown that behaviours that are copied from one robot to another evolve and adapt during multiple cycles of iterated learning and these evolved behaviours are better fitted to the environment of the robot collective and the robots themselves. Limitations and heterogeneities in the real robots' sensors and actuators give rise to variations in imitated behaviours and these variations allow better adapted behaviours to emerge and evolve during multiple cycles of imitation.

Apart from its functionality in skill transmission between individuals, imitation has a social dimension as it allows individuals to become part of a social community. In related research on imitation in robotics, Steels and Kaplan [23] argued that social learning can play a crucial role in initiating a humanoid robot into a linguistic culture. They used methods such as open-ended dialogue among humans and robots in which social leaning can be embedded. Beals and Steels [5] claimed that grammatical agreement systems have an important functionality in the emergence of language in groups of social agents. They presented agent-based models to explain how and why grammatical agreement systems emerge and get culturally transmitted by social interactions. They presented a set of language games in which two agents agree on the semantic and syntax of the vocabulary that identify the objects in their shared environment. Billard [6] claimed that imitation can be used to enhance autonomous robots' learning of communication skills. The sharing of a similar perceptual context between the imitator and the demonstrator can create the necessary social context where language can develop in. Billard devised experiments in which robots were able to learn a proto-language by using imitation to match their environmental perceptions with the observed actions. Alissandrakis et al. [3] developed the ALICE architecture to address the problems of imitation between agents with dissimilar embodiments.

They examined the rules of synchronisation, looseness of perceptual matching and proprioceptive matching in a series of experiments in which simulated robotic arms with variably sized and numbered joints tried to imitate each other. Alissandrakis et al. showed that patterns can be transmitted between robotic arms and variations occur during imitation because of the heterogeneities between the arms. They proposed that these variations might provide the evolutionary substrate for an artificial culture, as new behavioural patterns may emerge and be transferred between robots. This paper presents experiments in which real robots are used to model imitation between artificial agents. The robots are able to observe and imitate each other's movement patterns using their on-board sensors only, hence the imitation is embodied. We show that, as the robots' sensors and actuators are not perfect, even with a homogeneous group of real robots, variations occur during the imitation process that allow certain behavioural patterns to emerge and evolve during multiple cycles of imitation. These evolved behaviours can then be imitated with higher fidelity as they are more robust to uncertainties in the real robots' sensors and actuators; the behaviours have adapted to the robots, and the environment of the robot collective. The paper then investigates the effect of different learned-behaviour memory sizes on the relatedness of the population of evolved behaviours across the whole collective.

III. Hardware Setup

The artificial system that is used to model imitation consists of e-puck miniature robots [17]. E-puck robots are 7 cm in diameter and 5 cm in height. They are equipped with 2 stepper motors driven wheels, 8 proximity sensors, a CMOS image sensor, an accelerometer, a microphone, a speaker, a ring of LEDs, a bluetooth adapter and - with the LiION battery - they have up to 3 hours of autonomy. Importantly, the e-puck robots can sense and track the movement of other robots nearby (albeit imperfectly because of their limited sensors); thus they have the physical apparatus for imitation. Robots can signal to each other with movement and light, one to one or one to many, providing alternative methods for robot to robot interaction.

Despite these benefits, the original e-puck, with its default micro-processor dsPIC30F6014A, lacks the computational power needed for image processing. To overcome this limitation, the robots are enhanced with a Linux extension board [16] based on the 32-bit ARM9 microcontroller with Debian/Linux system installed. The board has a USB extension port which is used for connecting a wireless network card and it is equipped with a MicroSD card slot. These additions to the standard e-puck robot have a number of benefits including increased processing power and increased memory capacity. In addition, Player [13], a cross platform robot device interface and server, is installed on the Linux extension board. Running on the robot, it provides an interface to the robot's sensors and actuators over the IP network. The high-level control software runs on the Linux extension board and sends necessary commands to the dsPIC in which a low-level server collects these commands, activates the actuators of the robot if necessary, and sends sensor readings back to the linux extension board.

Since the e-puck bodies are transparent, in order to make it possible for the robots to 'see' each other, they are fitted with coloured skirts (figure 1). Thus robots observe each other's movements by visually sensing their skirts. Experiments are performed in a 3 m by 3 m robot arena (figure 2). Figure 3 shows the infrastructure of the experimental setup.

A vision tracking system from VICON^{TM} (*http://www.vicon.com*) provides high precision position tracking. Each robot is fitted with a tracking hat which provides a matrix of pins for the reflective spheres that allow the tracking system to identify

Fig. 1. An e-puck with Linux board fitted between the e-puck motherboard (lower) and the e-puck speaker board (upper). Note both the red skirt and the yellow hat that provides a matrix of pins for the reflective spheres which allow the tracking system to identify and track each robot.

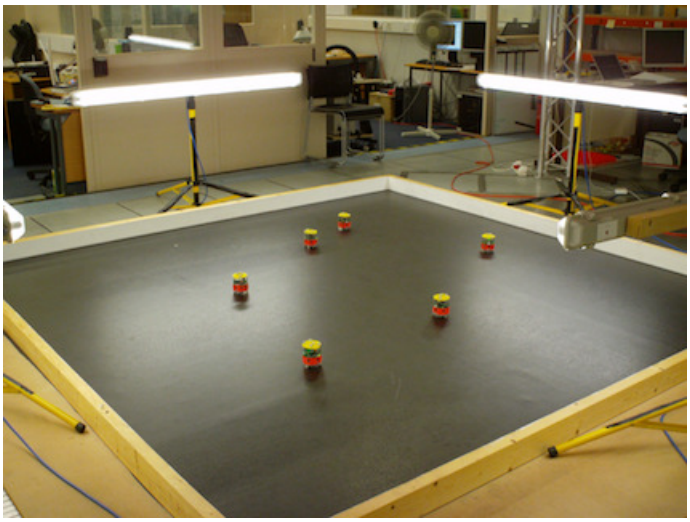

Fig. 2. Robot arena with 6 robots.

and track robots (figure 1). The tracking system is connected to the local network in which it broadcasts the real-time position of each robot over TCP/IP during experiments. Tracking data, together with robot status, is logged and used for offline analysis of experimental runs.

IV. Imitation Algorithm

Imitation is embodied in the sense that it completely depends on the robot's on-board (image) sensor. The imitation algorithm has two phases: *frame processing* and *data processing*. The frame processing phase corresponds to the *observe* step, as defined in section II. During this phase, the imitating robot observes the movement of the demonstrator robot by processing frames captured by its image sensor. The following operations are applied to each frame:

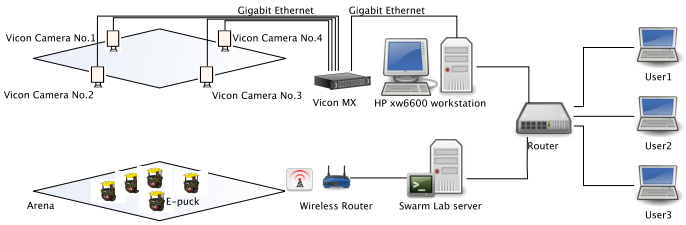

Fig. 3. Infrastructure of the experimental setup. Each robot is identified with a static IP. The dedicated swarm lab server works both as a data logging pool for the experiments and as a router to bridge the robots and the local network. Diagram from [16].

1) *The blobfinder* [7] tool of Player determines the position and size of the observed robot's skirt. It takes a colour value as input and fits a rectangular blob around the regions with that colour.
2) The exact size and position of the blob is determined by approaching from 4 sides to the approximate blob position and checking pixel values.
3) In each frame, the new position of the observed robot is determined by comparing it with its previous position and considering the speed of the robot. Every 0.2 sec, a new image frame is captured. The robot's speed is set to $5.2\ cm/sec$ during experiments so the displacement of the robot between two consecutive image frames should be equal to (if the robot was moving) or smaller than (if the robot was turning) $1.04\ cm$. Thus the observed robot must be within a circle of radius $1.04\ cm$, centred on its previous position, in the new frame. Any observed position outside this radius is ignored.
4) The relative position of the observed robot to the observing robot is calculated and stored in a linked list. Since the robots have only one camera and hence monoscopic vision, the observing robot determines the relative distance of the observed robot by previously calculated blob position and size.

In this way, up to 5 frames per second are processed and the relative position information for each frame, $(d^x_{robot}, d^y_{robot})$, is stored in a linked list. The pseudo-code for the frame processing phase is given in algorithm 1. During this phase, the observing robot rotates, if necessary, to keep the demonstrator robot within its field of vision.

Algorithm 1 Pseudocode for frame processing phase of imitation algorithm

Input:
Robot being observed, $Rob_{observed}$
Linked list of relative positions of the observed robot, $PositionList$, image frame f

$BlobList \leftarrow GetBlobs(f)$
$RobotList \leftarrow GetRobots(BlobList)$
for each $rob \in RobotList$ **do**
  if $rob = Rob_{observed}$ **then**
    $[d^x_{robot}, d^y_{robot}] \leftarrow GetRelativePosition(rob)$
    $PositionList \leftarrow AddNewPosition([d^x_{robot}, d^y_{robot}])$
  end if
end for

On completion of the observed robot's movement sequence[1], the linked list of relative positions is processed during the data processing phase. The data processing phase corresponds to the *learn* step, as defined in section II. The objective of this phase

[1]The robot-robot signalling protocol that determines the start and end of a movement sequence will be explained in section VI.

is to reconstruct the observed robot's movement pattern. The following operations are applied during this phase (figure 4):

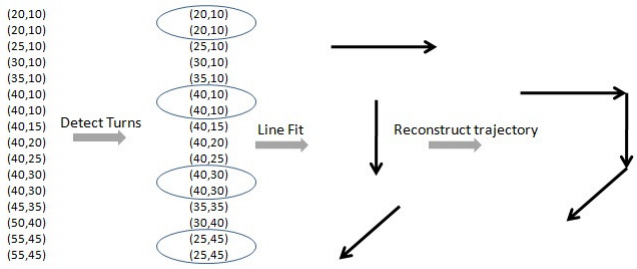

Fig. 4. Data processing phase: Given a linked list of positions as input, the first three steps of the data processing phase are detecting turns, line-fitting to determine straight line segments between turns and reconstructing the (estimated) observed trajectory. These three data transformations are shown as light grey arrows. The inputs to detect turns and line fit are shown numerically with example values where each data pair is a relative position of the demonstrator robot to the imitating robot; the outputs for line fit and reconstructed trajectory are shown as vectors.

1) The robots' movement patterns consist of turns (rotations) and straight line segments. While the demonstrator robot is rotating itself to turn to a new direction, it would appear to be static to the observing robot as its relative distance and position stays unchanged. The Euclidean distance between each consecutive relative position in the link list is calculated to determine the intervals that the observed robot is rotating itself in its current position. These intervals are marked as 'turns'.
2) During each period between consecutive turns, the robot is moving in a straight line. To better estimate the direction of the line, a regression line fitting algorithm [11] is utilised by considering all the relative positions that are recorded during that period. The regression line associated with n relative positions $(x_1, y_1), (x_2, y_2)...(x_n, y_n)$ is calculated by the regression (best fit) line algorithm with the form of:

$$y = mx + b \tag{1}$$

in which

$$m = \frac{n(\sum xy) - (\sum x)(\sum y)}{n\sum(x^2) - (\sum x^2)} \tag{2}$$

$$b = \frac{\sum y - m(\sum x)}{n} \tag{3}$$

Assume there are 10 positions stored in the linked list from the frame processing phase, p_1, p_2, p_3,..., p_{10}. If p_3 and p_4 correspond to the same location on the arena, it means that the observed robot was turning during the period p_3 and p_4 was recorded. If the next such occasion is at p_8 and p_9, the observed robot was moving on a straight line during the period that p_4 to p_8 was recorded. So the positions p_4 to p_8 are utilised for the line regression calculation to determine the length and direction of the movement (figure 5).

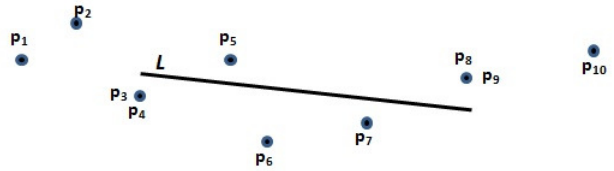

Fig. 5. Regression line fitting: p_3 and p_4 are located on the same location of the arena which means that the robot was turning when these positions were recorded. The next such occasion is at p_8 and p_9. So the robot was moving on a straight line between these two turns. Line L is the regression line fitted by using points p_4, p_5, p_6, p_7 and p_8.

3) By combining the straight line segments and the turns linking them, the observed trajectory is reconstructed. The pseudo-code for the data processing phase is given in algorithm 2.

Algorithm 2 Pseudocode for data processing phase of imitation algorithm

Input:
Linked list of relative positions of the observed robot, $PositionList$
List of moves, $MoveList \leftarrow \emptyset$

$Turns \leftarrow DetectTurns(PositionList)$
for $i = 1 \rightarrow size(Turns) - 1$ **do**
  $LinePositionList \leftarrow GetPointsBetweenTurns(Turns[i], Turns[i+1])$
  $move \leftarrow LineFitBetweenTurns(LinePositionList)$
  $MoveList \leftarrow AddNewMove(move \rightarrow angle, move \rightarrow displacement)$
end for

At the end of the data processing phase, the imitating robot saves the newly observed set of moves and turns, i.e. the estimated trajectory of the observed robot, in its memory. A sample entry in the memory of the imitating robot might look like:

- (271 15) (21 15) (142 19)

which would mean that the observed robot, as estimated by the imitating robot, moved 15 cm in a 271° relative angle to the imitating robot, then turned 110° counter clockwise[2] and moved 15 cm, then turned 121° counter clockwise and moved 19 cm.

V. On the Quality of Imitation

To quantitatively assess the fidelity of imitation (that is, the similarity between the original movement pattern and its copy), a quality of imitation function needs to be defined. Since each movement pattern consists of straight moves and turns, there are 3 components to each pattern that can be copied: the number of segments (straight moves), the length of each move and the angle (turn) between each consecutive move. Therefore, the overall quality of a copy can be calculated by separately estimating 3 quality indicators. The quality of move length, Q_l, between the original path O and its copy C is calculated as follows:

$$Q_l = 1 - \frac{\sum_m |l_m^O - l_m^C|}{\sum_m l_m^O} \tag{4}$$

[2] 110° is calculated by $|271 - 360| + 21 = 110$

where l_m is the length of move m that is to be compared. Here, the ratio of move length differences between the original pattern and its copy, to the total move length of the original pattern is calculated. If the original movement pattern and its copy have different numbers of segments, N^O and N^C respectively, the sum is calculated only over the number of segments in the smaller: $min(N^O, N^C)$. The quality of angle (turn) imitation similarly calculated as:

$$Q_a = 1 - \frac{\sum_m |a_m^O - a_m^C|}{\sum_m a_m^O} \tag{5}$$

where a_m is the turn angle following the move m. The quality of segment imitation simply compares the difference between the number of segments of the original pattern and its copy. It is calculated as:

$$Q_s = 1 - \frac{|N^C - N^O|}{N^O} \tag{6}$$

where N^O and N^C are the number of segments of the original path and its copy[3]. The overall quality of imitation, Q_i, is a combination of 3 quality indicators:

$$Q_i = \frac{LQ_l + AQ_a + SQ_s}{L + A + S} \tag{7}$$

where L, A and S are weighting coefficients.

To test the performance of the algorithm, a demonstrator robot is programmed to follow sequences of moves and turns that describe different geometrical shapes while an imitator robot watches and tries to learn the movement pattern. Then the imitator robot performs its copy of the demonstrator's pattern (figure 6). By comparing these two movement patterns, the quality of imitation is determined. The same movement pattern is repeated and copied multiple times at different distances between robots. Movement patterns are classified according to their structure: if it has none or only one turn, it is defined as a *low-complexity* movement pattern. If it has 2 or 3 turns (a triangle or a square), it is defined as a *medium-complexity* movement pattern and if it has 4 or more turns it is defined as a *high-complexity* movement pattern. Following this classification, 3 shapes that have different levels of complexity are used (figure 7) to test the algorithm. In the first set of experiments the demonstrator robot is programmed to move in a line, forward and backward, which is a low-complexity movement pattern, while the imitator robot watches it from different distances. In the second set of experiments the demonstrator robot has an equilateral triangular trajectory (medium-complexity) movement pattern. In the third set of experiments, the demonstrator robot has a high-complexity movement pattern which consists of a complex trajectory with 4 turns. Once the experiments are completed, the quality of imitation is assessed by an external program used for the offline analysis of the experiments.

Figures 8 to 10 show results for these experiments. We see that, for all 3 movement patterns, the highest quality is achieved when the distance between robots is 1 m. When the distance is increased (1.5 m or more), the mean quality of imitation starts to fall. The reason is that the relative positional changes are calculated based on the size and the location of the demonstrator

[3] N^O is always greater than 0 as there should be a movement in the original movement pattern so that it could be copied.

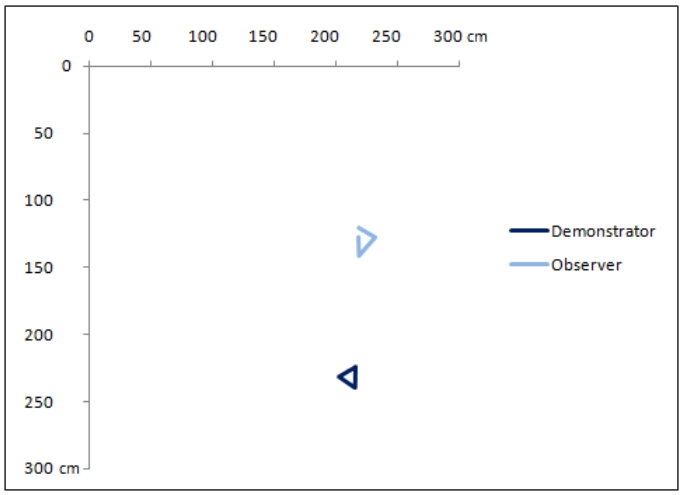

Fig. 6. Plot of the trajectory of robots during an imitation run. The demonstrator robot moved in an equilateral triangular trajectory (each side of length 20 cm) which was then copied by the imitator robot.

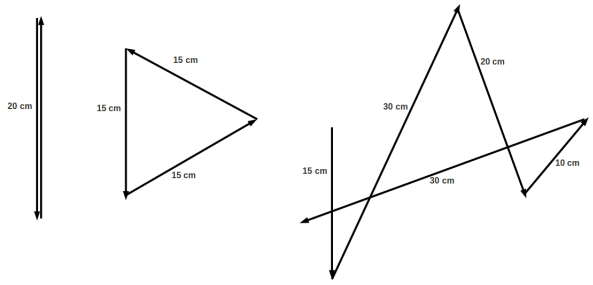

Fig. 7. Movement patterns that the demonstrator robot followed during experiments. The first movement pattern is a low-complexity pattern which consists of two moves of length 20 cm and a 180° turn between them. The second pattern is a medium-complexity pattern which consists of an equilateral triangular trajectory. The third is a high-complexity pattern which consists of a trajectory with 4 turns.

robot in the imitator robot's field of view. When the distance is high, the positional changes are harder to detect as they cause smaller variations in the perceived size of the demonstrator robot. As a result, the second best mean quality achieved when the distance between robots is 1.5 m (figure 8 to 10). The lowest mean quality of imitation is observed when the distance between robots is 0.5 m. This is due to the fact that when the distance between robots is low (i.e. 0.5 m or less), the demonstrator robot leaves the field of view of the imitator robot many times, which forces the imitator robot to rotate itself each time. The imitator robot may then miss some moves and especially turns of the demonstrator robot while it is itself busy turning. This effect can be clearly seen when the observed robot has a high-complexity trajectory. As many of the turns are missed, the mean quality of imitation is very low when the distance is 0.5 m, compared to other cases. For imitation of low-complexity and medium-complexity patterns, the highest standard deviations are seen when the distance between robots is 0.5 m, which is a result of some low quality imitations.

We see in figure 9 that when the demonstrator robot describes a medium-complexity movement pattern, the mean quality of imitation is similar at all tested distances, compared to the imitations of low-complexity and high-complexity movement patterns. It appears that imitation of medium-complexity movement patterns is relatively less affected by the distance between robots. Furthermore, when a low-complexity pattern is copied, although sensor errors may occur, its copy is typically another

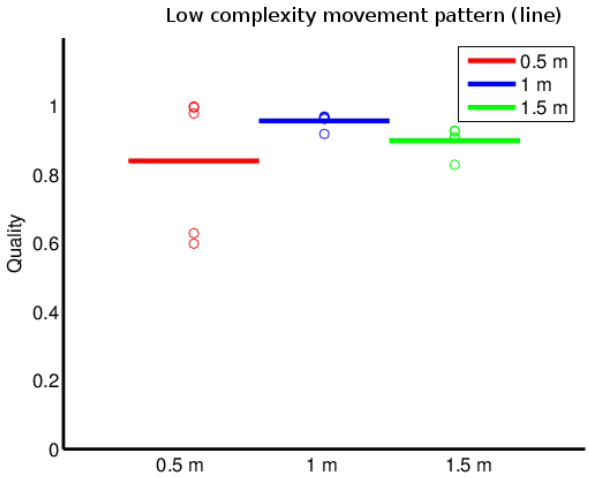

Fig. 8. Mean quality value calculated at different distances between robots. The demonstrator robot has a low-complexity movement pattern which consists of a line. The quality of each imitation is shown by a circle and the horizontal line shows the mean over 5 imitations. Each quality indicator was given equal weight: $L = A = S = 1$.

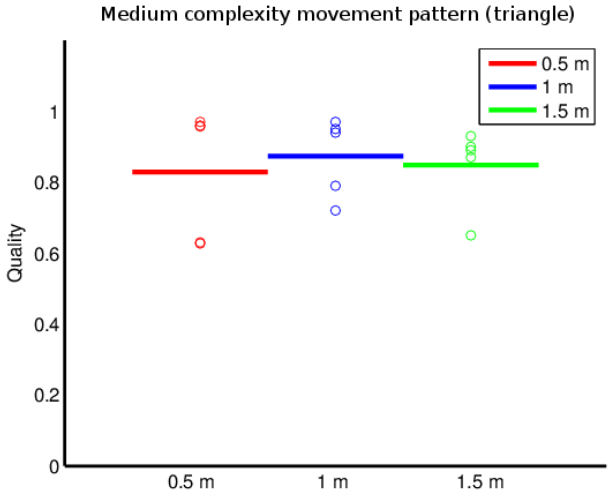

Fig. 9. Mean quality value calculated at different distances between robots. The demonstrator robot has a medium-complexity movement pattern which consists of an equilateral triangle. The quality of each imitation is shown by a circle and the horizontal line shows the mean over 5 imitations. Each quality indicator was given equal weight: $L = A = S = 1$.

low-complexity pattern. When a hand-sketched high-complexity movement pattern is demonstrated, the copy typically has a lower complexity. However, when a medium-complexity pattern is copied, we see that it may result in patterns with different levels of complexity at all distances. For these reasons, when investigating the evolution of movement patterns during multiple cycles of imitation - described below, we initialise robots with medium-complexity patterns. As stated above, when a hand-sketched high-complexity movement pattern is demonstrated, the copy typically has a lower complexity. Therefore the results should generalize if the robots are initialised with more complex patterns. When the distance between robots is 1 m, the quality of imitation was greater and the difference in the quality of imitation for imitation of patterns with different complexity is smaller. For these reasons the distance between robots is initialised to 1 m in the experiments described in the next section.

An imitation with $Q_i \geq 0.85$ is defined as *high quality*. This value is determined as follows: in order to accept that a copy

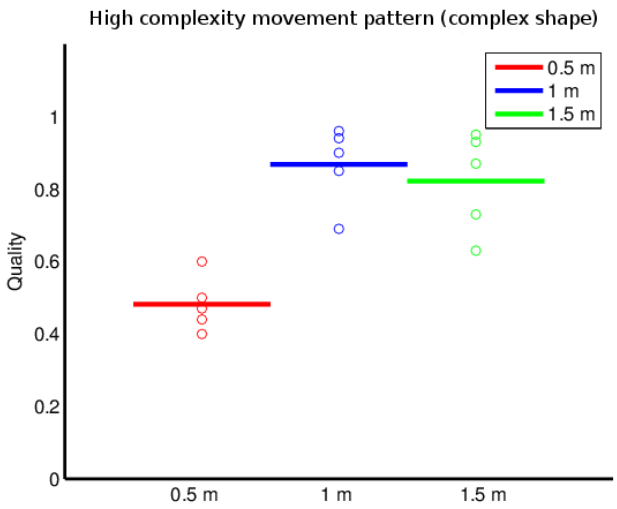

Fig. 10. Mean quality value calculated at different distances between robots. The demonstrator robot has a high-complexity movement pattern which consists of complex trajectory. The quality of each imitation is shown by a circle and the horizontal line shows the mean over 5 imitations. Each quality indicator was given equal weight: $L = A = S = 1$.

is of high quality, it should have a relatively high Q_i value and all of its 3 quality indicators should be higher than 0.5. To guarantee that the second condition is always true, Q_i should be higher than 83.33. On the basis of results observed during the experiments presented in this section, a slightly higher value, 0.85, is selected as the threshold for high quality.

VI. Emergence of Structure in Behaviours Evolved Through Embodied Imitation

In the previous section we saw that variations that arise from embodiment cause copied patterns to differ from their originals. This section examines the effects of these variations on the structure of copied movement patterns during multiple cycles of imitation. We show that movement patterns that appear to be more robust to noise and uncertainties in the robot's sensors emerge and these adapted behaviours can be copied with higher fidelity by the group members.

A. Experimental Setup

As we are interested in the effects of variations on the structure of the imitated patterns, an experimental setup in which 4 robots copy each other's movement patterns is introduced. The 4 robots are placed 1 m apart from each other in the arena, as shown in figure 11. They interact by copying each others' movement patterns using the imitation algorithm outlined in section IV. Figure 12 shows the finite state machine (FSM) for the controller of the robots. Each robot runs the same FSM. Robots can be in one of two modes during experiments: demonstrator or observer. When a robot enters demonstrator mode, it turns its LEDS on for 35 seconds[4] to signal that it will start to demonstrate (enact) a movement pattern. During this period the demonstrator signals to get the attention of an observer robot. Then the demonstrator robot turns its LEDs off and executes a movement pattern that consists of straight-line moves and turns. When execution is complete, the demonstrator robot blinks its LEDs for 1.6 seconds[5] to signal 'finish'. Then the demonstrator robot returns to its original start position and enters observer

[4]35 seconds is the approximate time needed for an observer robot to complete two complete scans of the arena, searching for a demonstrator robot.

[5]In order to discriminate between a *start* and a *finish* signal, a finish signal is shorter than a start signal. A finish signal takes 1.6 seconds while a start signal should be observed for at least 2 seconds.

mode. When a robot enters observer mode it searches for a start signal by scanning the arena while rotating. When it detects a start signal it waits for the demonstration to start. After completion of the demonstration the observer robot learns what it has observed and enters demonstrator mode. At the start of an experimental run, two of the 4 robots start in demonstrator mode while the other 2 start in observer mode. The experiment then free-runs as the robots change roles while imitating each other. The pseudo-code for the robots' controller is listed in algorithm 3. Since the robots need to have a movement trajectory in their memory to be able to act as demonstrators they are each initialised with one medium-complexity movement pattern: 2 with an equilateral triangle trajectory and 2 with a square trajectory.

No fitness value is attached to the movement patterns as we are interested in the evolution of movement patterns during multiple cycles of imitation, regardless of their utility or context. Thus we choose the simplest possible selection strategy as follows: each time a robot needs to enact a movement pattern it chooses, at random with equal probability, one of the trajectories in its learned-pattern memory. Once selected, the pattern is converted to motor commands that can be executed by the demonstrator robot, which corresponds to the *enact* step, as defined in section II (figure 13). In a previous paper we explored different selection strategies [24].

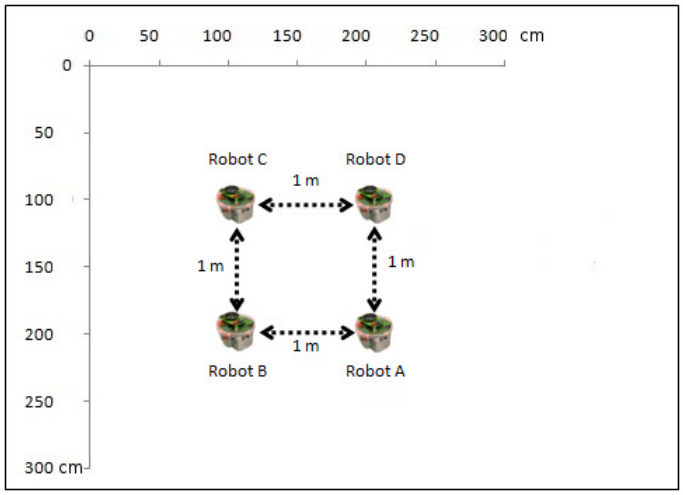

Fig. 11. Each experiment presented in this section is performed in a 3 m by 3 m arena with 4 robots, placed 1 m apart and arranged as shown here.

Consider the learned-pattern memory. This is a long-term memory that persists between cycles of observation and demonstration. In order to test the effect of this memory on the structure of adapted behaviours we test three cases. The first is *no memory*, in which a robot saves only the most recently learned pattern, overwriting the previously learned pattern. Our second case is *unlimited memory*, in which a robot saves all learned patterns - extending the size of the memory each time a newly learned pattern is appended. In our third case, a robot has a *limited memory* which can store only a fixed number of movement patterns. Once the memory is full when a new pattern is learned it overwrites the oldest pattern. The evolution of movement patterns is examined, for all three memory cases in experiments presented in the next section.

Algorithm 3 Pseudocode for robots' controller

Input:
$PositionList \leftarrow \emptyset$, $Rob_{Observed} \leftarrow \emptyset$, $MoveList \leftarrow \emptyset$, $f \leftarrow \emptyset$
$RobState \leftarrow searchForStartSignal$
while *experiment continues* **do**
  if $RobState = searchForStartSignal$ **then**
    $GetRobotList()$
    for each $rob \in RobotList$ **do**
      if $RobotSignalOn(rob) =$ **true then**
        $Rob_{Observed} \leftarrow rob$
        $RobState \leftarrow waitForDemonstration$
      end if
    end for
  else if $RobState = waitForDemonstration$ **then**
    $GetRobotList()$
    $Rob_{Observed} \leftarrow FindObservedRobot(Rob_{Observed})$
    if $RobotSignalOff(Rob_{Observed}) =$ **true then**
      $RobState \leftarrow trackDemonstrator$
    end if
  else if $RobState = trackDemonstrator$ **then**
    $GetRobotList()$
    $Rob_{Observed} \leftarrow FindObservedRobot(Rob_{Observed})$
    if $RobotSignalOff(Rob_{Observed}) =$ **true then**
      $FrameProcess(Rob_{Observed}, PositionList, f)$
    else
      $RobState \leftarrow recordObservedPattern$
    end if
  else if $RobState = recordObservedPattern$ **then**
    $DataProcess(PositionList)$
    $SaveNewPattern()$
    $PositionList \leftarrow \emptyset$, $Rob_{Observed} \leftarrow \emptyset$, $MoveList \leftarrow \emptyset$
    $RobState \leftarrow signalStart$
  else if $RobState = signalStart$ **then**
    $SignalFor(35)$
    $RobState \leftarrow demonstrate$
  else if $RobState = demostrate$ **then**
    $selectedPattern \leftarrow SelectMovementPatternFromMemory()$
    $DemonstratePattern(selectedPattern)$
    $SignalFor(1.6)$
    $RobState \leftarrow returnToOrigin$
  else if $RobotState = returnToOrigin$ **then**
    $ReturnOriginalPosition()$
    $RobState \leftarrow searchForStartSignal$
  end if
end while

function $GetRobotList()$
    $f = GetImageFrame()$
    $BlobList = GetBlobs(f)$
    $RobotList = GetRobots(BlobList)$
end function

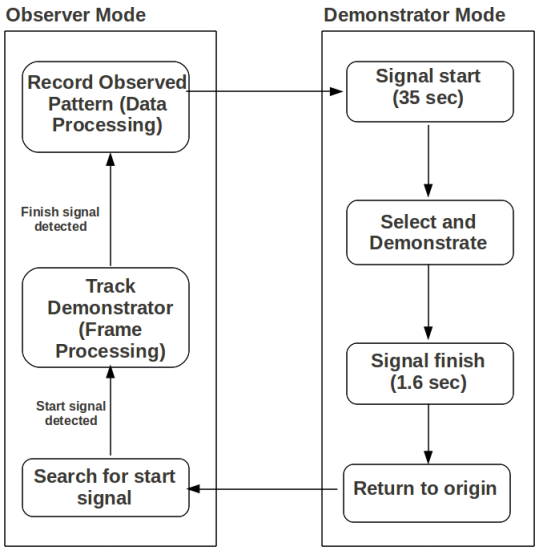

Fig. 12. Finite state machine of the controller of the robots. Once started the FSM loops indefinitely, alternating between demonstrator and observer modes. The states within each mode are shown in the two large boxes here.

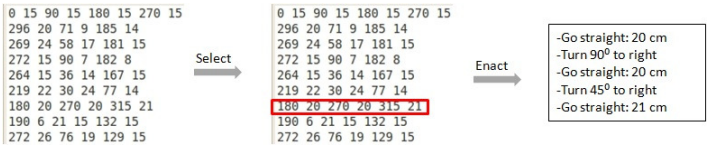

Fig. 13. The demonstrator robot randomly selects one of the patterns in its learned-pattern memory and then the selected pattern is converted into a set of motor commands that can be executed by the demonstrator robot (*enact*).

B. Experiments

1) Imitation with No Memory: In the first set of experiments robots are able to remember only the most recently learned movement pattern; a newly-learned pattern overwrites the previous one. Figure 14 shows the pattern evolution tree of a typical experimental run with these settings. After the experiment is complete an external program calculates the quality of imitation for each copied movement pattern and generates the pattern evolution tree. In the figure each node represents a pattern. If an arrow exists at a node this means one of the robots demonstrated that pattern and it was learned by another robot. The new (child) copy is at the end of the arrow. If the copy is high-quality ($Q_i \geq 0.85$), then the node is shown with dark shading.

In this experiment we observe that the original patterns change very quickly. At the start of the run the robots that started in observer mode by chance both copied the square trajectory, and the triangular trajectory vanished from the experiment. The square trajectory also deteriorated rapidly. In this run some poor copies, in which the observer robot missed some turns, caused the robots to eventually end up with a low-complexity movement pattern consisting of a single forward move. These low-quality copies do not occur often, but with this no-memory setting just one is sufficient to disrupt the evolutionary process. As explained in section V, these low-complexity patterns can be copied with high quality though we still observe some poor quality copies. In this experimental run, all patterns after number 22 are low-complexity patterns with only one forward move without turns. All runs with these settings have the same characteristics. As variations that arise during imitation strongly impact the evolution of movement patterns their adaptation is highly sensitive to errors. In all runs the original patterns vary

quickly and most runs result in a low-complexity movement pattern after relatively few imitation cycles.

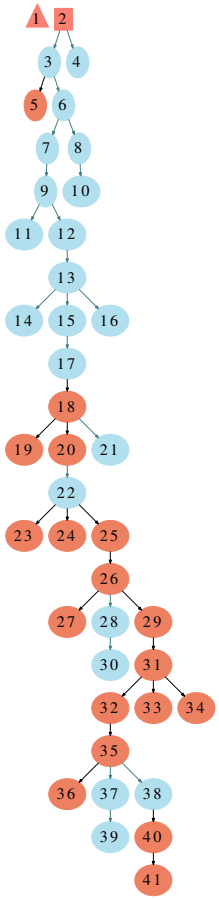

Fig. 14. Pattern evolution tree for a 4 robot experiment with no memory. Each node in the figure represents the demonstration of a movement pattern. If a pattern is demonstrated and imitated, the new copy of that pattern is linked to it by an arrow. For instance, pattern 2, the original square, was demonstrated by Robot A and was learned by two robots. The new (child) copies of pattern 2 are patterns 3 and 4. If the copy is of high quality (i.e. $Q_i \geq 0.85$), then the node has a darker shading. Initial movement patterns are a triangle (1) and a square (2). The nodes are numbered according to the time that they were copied: if $x < y$, pattern x was copied before pattern y.

2) Imitation with Unlimited Memory: In the second set of experiments robots have unlimited pattern memory, so they save all learned patterns. As explained in section VI, when they enter demonstrator mode robots randomly select, with equal probability, one of the patterns in their memory and demonstrate (enact) it. Figure 15 shows the pattern evolution tree of one particular run with these settings. Compared to the case with no memory, since each newly learned pattern is stored in memory, the original movement patterns are more likely to be inherited, with variation. Low quality copies do occasionally occur but as these do not replace previously observed patterns, they are less likely to disrupt the evolution of movement patterns. We see that, as patterns vary during multiple cycles of imitation, some patterns that are able to be copied with high quality emerge and propagate between robots. In this particular run pattern 27 has this property. Figure 16 shows the evolution of pattern 27. In this experiment, Robot A watched Robot C enact pattern 1 (the original equilateral triangle) and attempted to learn the movement sequence; the result is pattern 5. Then Robot D watched pattern 5, enacted by robot A, and attempted to learn it; thus pattern 11 is an imitation of pattern 5. Pattern 27 is a descendant of the original equilateral triangle trajectory and there

are 5 intermediate copies between the original triangle and pattern 27: pattern 1 $\rightarrow$ pattern 5[6] $\rightarrow$ pattern 11 $\rightarrow$ pattern 18 $\rightarrow$ pattern 20 $\rightarrow$ pattern 26 $\rightarrow$ pattern 27. As can be seen, pattern 1 $\rightarrow$ pattern 5, pattern 5 $\rightarrow$ pattern 11 and pattern 11 $\rightarrow$ pattern 18 are high fidelity imitations while pattern 18 $\rightarrow$ pattern 20, pattern 20 $\rightarrow$ pattern 26 and pattern 26 $\rightarrow$ pattern 27 are low fidelity imitations. Finally pattern 27 emerges and a sharp increase in quality of imitation can be observed after this point ($Q_i > 0.94$ for all of its descendants).

What makes this pattern and its descendants easily copiable? First, short moves are more prone to error, as a small mistake in perception can cause them to vanish; a pattern that can be copied with high quality typically does not include short moves. Second, the length of each move varies at each subsequent copy. Although estimating the relative size and position of the demonstrator robot is straightforward image processing, it is error-prone because of the relatively low resolution of each robot's image sensor. A move directed towards or away from the observing robot can only be detected if it causes a perceptible change in the size of the demonstrator robot, i.e. a detectable change in number of pixels in the image of the demonstrator. At each copy, the observing robot stores what it infers from the demonstration as perceived from its relative position and perspective. Thus the patterns tend to evolve into ones that can be more easily imitated. Figure 17 shows pattern 27 and its descendants. As can be seen there is a high level of similarity between these. At the end of the run, pattern 27 and its descendants form a cluster of similarly-shaped patterns in the robots' memories. A cluster is defined here as a group of movement patterns, with 4 or more members, that are related to each other by a series of high quality imitations (i.e. $Q_i \geq 0.85$). Figure 18 shows the average Q_i value for this experiment in comparison with the average Q_i value for the cluster formed by pattern 27's cluster. We see a sharp increase in Q_i value after a pattern emerges that is more robust to uncertainties in the robot's sensors and the imitation process: the average Q_i value for the cluster that is formed by the descendants of pattern 27 is around 0.96, while the average quality of imitation for this experiment run is around 0.82.

[6]We use notation A $\rightarrow$ B as shorthand for B is a learned copy of A.

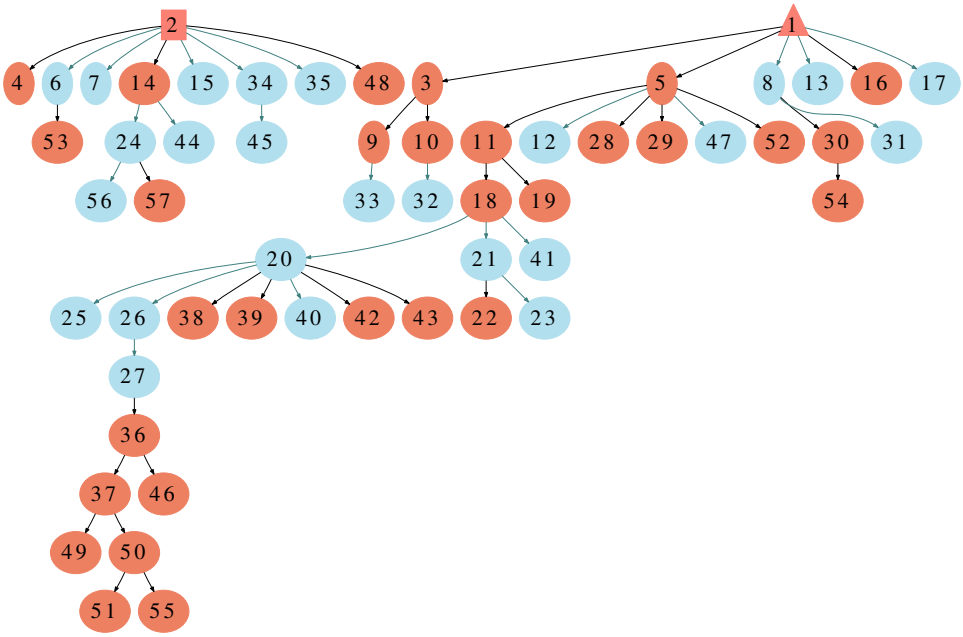

Fig. 15. Pattern evolution tree for a 4 robot experiment with unlimited memory. Initial movement patterns are a triangle (1) and a square (2).

Table I shows results from 10 experimental runs with unlimited memory. The table is created by determining the clusters of movement patterns that are related to each other by high quality imitations. We see that in all runs such clusters of highly similar movement patterns emerge in the robots' memory. The average similarity between the members of these clusters is very high, 0.927. Note that members of these clusters may have low quality copies. For instance, in figure 15, patterns 20, 38, 39, 42 and 43 form a cluster of size 5. Patterns 25, 26 and 40 are copies of pattern 20 but since they are low quality copies they are not counted as members of this cluster. Column 4 in table I shows the average quality of imitation for the members of clusters (including low quality copies of the members of the clusters) while column 5 shows the average quality of imitation for all copies of each run. A pair-wise t-test reveals that, considering all runs, there is a statistically significant difference between these two values. These results suggest that the emerging movement patterns that constitute the clusters are more robust to the process of embodied imitation.

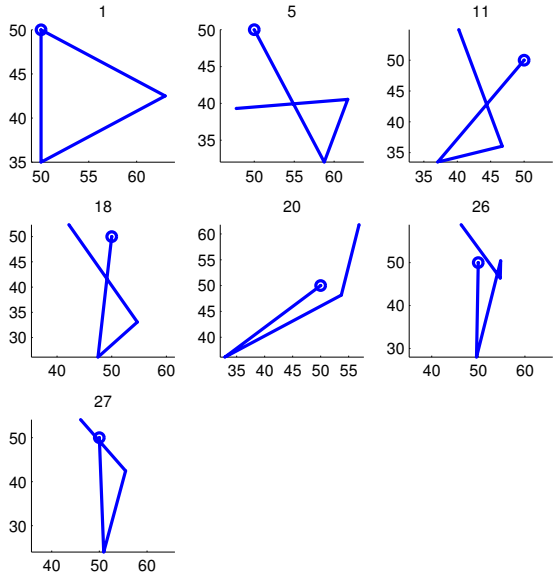

Fig. 16. Evolution of pattern 27 in Figure 15. Pattern 27 is a descendant of the original equilateral triangle pattern. By following the imitation links on the pattern progress map for this experiment, we can see that there are 5 intermediate copies between the original triangle and pattern 27: the patterns numbered 5, 11, 18, 20, 26. All of these patterns, starting with the original triangle and ending with pattern 27, are shown here in order. All axes are marked in cm. Beginning of each movement pattern is marked with a circle.

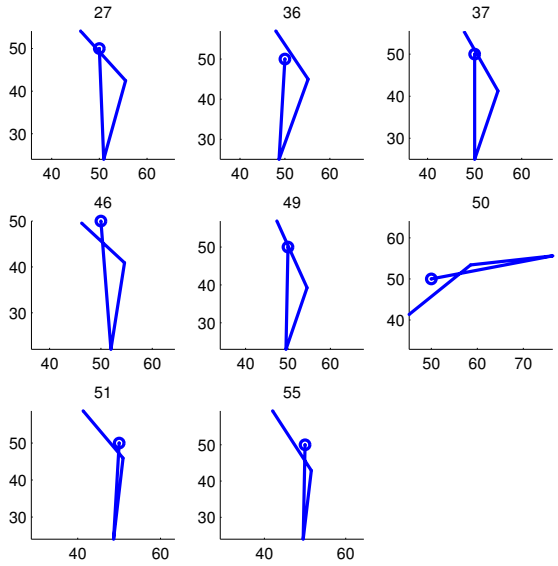

Fig. 17. The descendants of pattern 27 in Figure 15. Starting with pattern 27, its descendants (patterns 27, 36, 37, 46, 49, 50, 51, 55) are shown in order. All axes are marked in cm.

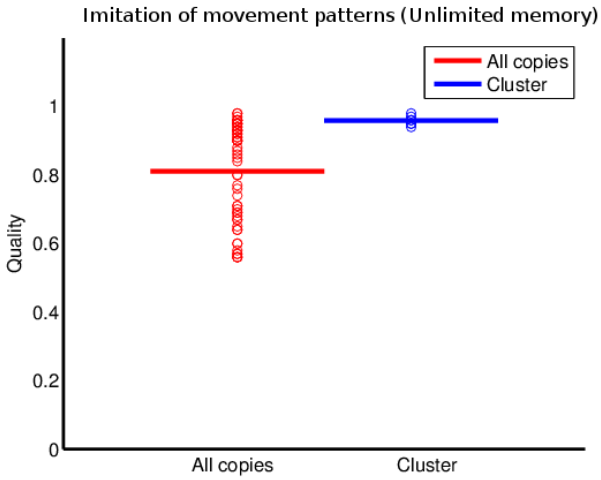

Fig. 18. Mean quality value calculated for all imitations in the experiment shown in Figure 15 (All copies) and mean quality value for the cluster formed by pattern 27 (Cluster). The quality of each imitation is shown by a circle and the horizontal line shows the mean.

Number of clus-ters	Average size of clus-ters	Average similarity between the members of the clusters	Average quality of imitation for the members of the clusters	Average quality of imitation for all copies
4	5.25	0.94	0.85	0.81
4	6	0.94	0.86	0.8
8	4.88	0.93	0.83	0.75
2	7	0.93	0.85	0.72
4	9.5	0.92	0.82	0.78
4	6.25	0.93	0.82	0.78
5	6	0.92	0.82	0.8
3	5.33	0.93	0.79	0.72
2	12	0.92	0.81	0.79
3	6.6	0.91	0.78	0.73
3.9	6.88	0.927	0.823	0.768

TABLE I
RESULTS FOR 10 EXPERIMENTAL RUNS WITH UNLIMITED MEMORY.

3) Imitation with Limited Memory: In the previous set of experiments we saw that certain patterns, those that are more robust to uncertainties in the real robots' sensors and the estimation process of imitation, can emerge during multiple cycles of imitation. As these emergent patterns can be copied with high quality their descendants have similar, inherited characteristics. As a result, clusters of highly copiable patterns are formed in the robots' memories. These clusters may grow larger with subsequent cycles of imitation if, by chance, cluster members are selected for demonstration. In our third set of experiments robots have a limited memory in which they store only the most recent 5 patterns observed. When the memory is full and a new pattern is learned, the oldest pattern in their memory is overwritten. A typical run with these settings is shown here in detail. Figure 19 shows the pattern evolution tree for a particular run with limited memory. In this experiment, Robot C

watched Robot B enact pattern 1 (the original equilateral triangle) and attempted to learn it; the result is pattern 10. Then robot D watched pattern 10, enacted by Robot C, and attempted to learn it; thus pattern 12 is an imitation of pattern 10. As can be seen, pattern 1 $\rightarrow$ pattern 10 is a high fidelity imitation and pattern 10 $\rightarrow$ pattern 12 is a low fidelity imitation. Once the V-shaped pattern, pattern 12, emerged, there is a sharp increase in the quality of imitation as all descendants of pattern 12 are high quality imitations. Figure 20 shows the evolution of this path and Figure 21 shows some of its high-quality descendants. At the end of this run, 12 of the 20 patterns in the memory of all 4 robots are descendants of this pattern. Since the robots randomly choose which pattern to enact, there is now a 60% chance that one of the descendants of pattern 12 selected. Once selected and copied, the new copy is itself likely to be a high quality copy, and so similar to pattern 12. This process will then increase the percentage of patterns in the memory that are similar to pattern 12. Thus, with a limited memory, the emergent patterns and high quality copies that are adapted through multiple cycles of imitation can become dominant in the robots' collective memory. Figure 22 shows the average Q_i value for this experiment in comparison with the average Q_i value for the cluster formed by pattern 12's descendants. There is a sharp increase in the average Q_i value for the pattern 12 cluster.

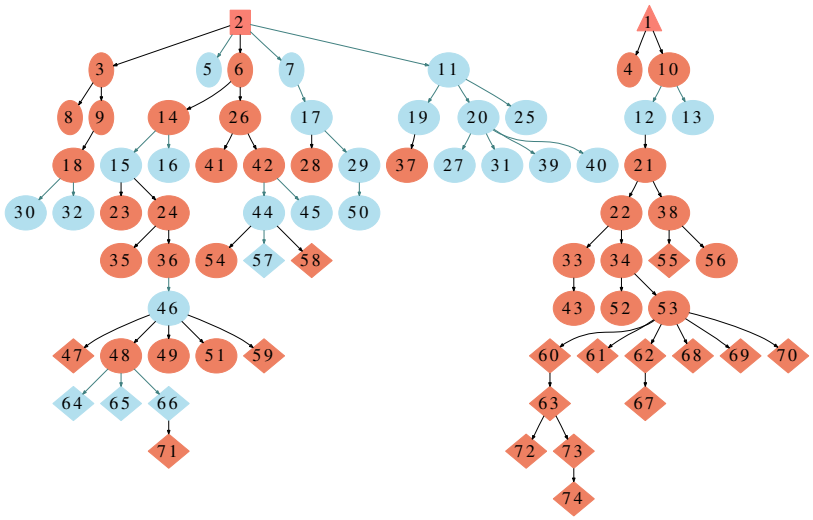

Fig. 19. Pattern evolution tree for a 4-robot experiment with limited memory. Initial movement patterns are a triangle (1) and a square (2). The 20 patterns in the memory of all 4 robots at the end of the experiment are highlighted as diamonds.

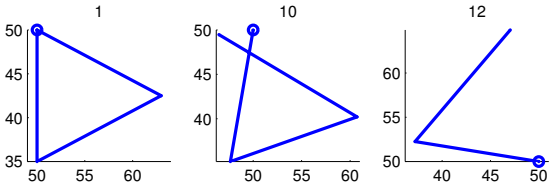

Fig. 20. Evolution of pattern 12 in Figure 19. There is an intermediate copy (10) between the original triangle and pattern 12. All axes are marked in cm.

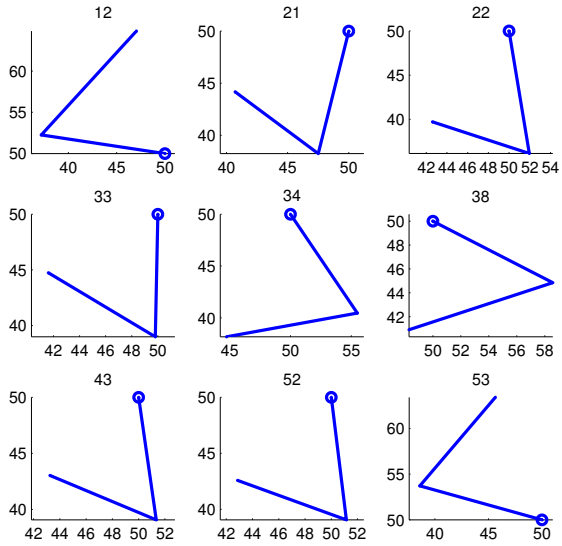

Fig. 21. The descendants of pattern 12 in Figure 19. Starting with pattern 12, some of its high-quality copy descendants (patterns 12, 21, 22, 33, 34, 38, 43, 52, 53) are shown in order. All axes are marked in cm.

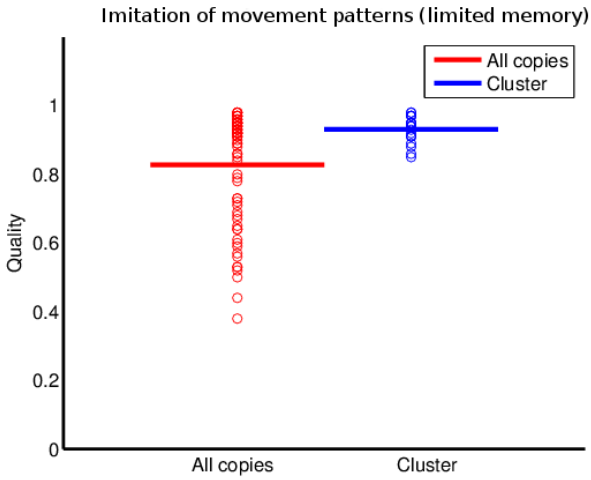

Fig. 22. Mean quality value calculated for all imitations in the experiment shown in figure 19 (All copies) and mean quality value for the cluster formed by pattern 12 (Cluster). The quality of each imitation is shown by a circle and the horizontal line shows the mean.

Table II shows results from 10 experimental runs with limited memory. We see that in all runs clusters of highly similar movement patterns emerged in the robots' collective memory. The average similarity between members of these clusters is very high, 0.93. A pair-wise t-test shows that, considering all runs, there is a statistically significant difference between the average quality of imitation for the members of clusters (column 4) and the average quality of imitation for all copies (column 5). Therefore, as was the case with unlimited memory, clusters of movement patterns that are more robust to uncertainties emerge during multiple cycles of imitation.

When we compare experimental results for unlimited and limited memory, in the limited memory case we see a smaller

number of larger clusters; compare number of clusters, col. 1 in tables I and II: 3.9 and 2.8, and average size of clusters, col. 2: 6.88 and 8.33. This is because in some limited-memory runs the collective memory of the robots is dominated by the patterns that formed the clusters; each time a pattern that can be imitated with high fidelity emerges, it starts to compete for domination of the robots' memory. If the members of their clusters are, by chance, enacted the new copies are likely to be similar, thus increasing the size of the clusters. As a result of this process we see fewer, larger clusters with limited than with umlimited memory.

Number of clus-ters	Average size of clus-ters	Average similarity between the members of the clusters	Average quality of imitation for the members of the clusters	Average quality of imitation for all copies
5	6.4	0.94	0.86	0.82
4	5	0.93	0.81	0.77
4	6.75	0.92	0.81	0.78
1	20	0.92	0.84	0.73
3	6	0.93	0.81	0.74
2	5.5	0.92	0.85	0.77
3	6.6	0.91	0.81	0.76
3	5.6	0.92	0.77	0.72
1	14	0.96	0.92	0.73
2	7.5	0.95	0.81	0.75
2.8	8.33	0.93	0.829	0.757

TABLE II
RESULTS FOR 10 EXPERIMENTAL RUNS WITH LIMITED MEMORY

VII. CONCLUSION

In this research, real robots are used to model imitation between artificial agents. We have shown that limitations and heterogeneities in the real robots' sensors and actuators give rise naturally to variation in imitated behaviours. Despite the fact that we have no fitness function and the behaviours themselves have no utility, we see that these variations allow better adapted behaviours to emerge and evolve during multiple cycles of imitation. As the robots share a similar perceptual context, the imitated behaviours adapt to the limitations and uncertainties inherent in interactions between real physical robots, so that these evolved behaviours can then be imitated with higher fidelity.

Three different types of learned-behaviour memory have been experimentally tested: no memory, unlimited memory and limited memory. In the no memory case, the evolution of movement patterns is extremely sensitive to any instance of poor quality imitation, which means that the original movement patterns very quickly change. In the unlimited memory case, patterns emerge that can be easily copied but were less likely to then become dominant as the number of patterns in the robots' collective memory grow larger with each new imitation cycle. However, in the case with limited memory, these adapted patterns can become dominant if they and their descendants are, by chance, chosen for demonstration. In all experimental scenarios, robots randomly select one of the learned movement patterns for enaction. Thus, as in biological evolution, we have the three

evolutionary operators: variation (due to embodied imitation), selection and inheritance. As the movement patterns evolve through multiple cycles of imitation, selection and variation, the robots are able to, in a sense, agree on the structure of the behaviours that are imitated. This process can be observed more clearly in experiments with limited memory as the number of clusters of related patterns is smaller and the average size of those clusters is larger.

We believe this work is interesting for the following reasons. We have - for the first time - demonstrated behavioural evolution of socially learned behaviours in real robot collectives, and shown that variation arises as a natural consequence of the process of embodied imitation, especially the limitations and heterogeneities of real physical robots. Our embodied approach has highlighted the importance of imperfect imitation, or noisy social learning, in providing behavioural evolution with a larger behavioural landscape to explore than might be apparent from the experimental setup. If we think of the robots and the robot collective as the environment for behavioural evolution, noisy social learning is the principle mechanism by which behaviours can adapt to be better fitted to that environment. We agree with Alissandrakis et al. [3] that variations might provide the evolutionary substrate for an artificial culture, and the work of this paper provides further exploration in this direction. In particular our experiments with different robot memory sizes have, we argue, provided new insights into how coherence or diversity in the population of behaviours in a collective is affected by behavioural memory size. If an artificial culture is characterised by persistent shared behavioural traditions then mechanisms are needed that balance discovery of new behaviours with persistence and coherence (relatedness) of existing behaviours. This work suggests such mechanisms.

There are a number of research questions that can be further explored following the work presented in this paper. The robot - robot movement imitation algorithm can be extended to provide feedback to the demonstrator robot and thus allow selection mechanisms based on the success of imitation. The algorithm can be extended to include the imitation of responses to sensed inputs; this would allow the imitation of interaction, so that interactions between robots could be propagated across the robot collective. Furthermore, in the work of this paper the imitated patterns are not linked to a task or environmental context; it should be possible and testable that using the embodied imitation approach presented, associating imitated behaviours with tasks that have utility can increase the efficiency of the robot group. In a research related to the paper, Erbas et al. [12] described an imitation-enhanced reinforcement learning algorithm in which real robots learn the task of reaching a target location. It is shown that imitation of purely observed behaviours enhances the learning speed of robots and the variations that result from copying errors may allow novel solutions to emerge.

Acknowledgements

This work was supported by EPSRC research grant: EP/E062083/1.

REFERENCES

[1] Acerbi, A. and Nolfi, S. (2007). Social Learning and Cultural Evolution in Embodied and Situated Agents. In Proceedings of the IEEE Symposium on Artificial Life, pages: 333-340, IEEE Press.

[2] Acerbi, A. and Parisi, D. (2006). Cultural Transmission between and within Generations, Journal of Artificial Societies and Social Simulations, 9(1).

[3] Alissandrakis, A., Nehaniv, C. L., and Dautenhahn, K. (2004). Towards Robot Cultures? Learning to Imitate in a Robotic Arm Test-bed with Dissimilar Embodied Agents. Interaction Studies: Social Behaviour and Communication in Biological and Artificial Systems, 5(1):344.

[4] Bakker, P. and Kuniyoshi, Y. (1996). Robot See, Robot Do: An Overview of Robot Imitation. In Proceedings of AISB96 Workshop on Learning in Robots and Animals, pages: 3-11.

[5] Beuls, k. and Steels, L. (2013). Agent-Based Models of Strategies for the Emergence and Evolution of Grammatical Agreement, PLoS ONE, 8(3), e58960. doi:10.1371/journal.pone.0058960.

[6] Billard, A. (1999). Imitation: A Means to Enhance Learning of a Synthetic Proto-Language in an Autonomous Robot. In Imitation in Animals and Artifacs, pages 281311. MIT Press.

[7] Bruce, J., Balch, T., and Veloso, M. (2000). Fast and Inexpensive Color Image Segmentation for Interactive Robots. In Proceedings of IROS 2000, pages 20612066.

[8] Dautenhahn, K., Nehaniv, C. L., and Alissandrakis, A. (2003). Learning by Experience from Others - Social Learning and Imitation in Animals and Robots. In Khn, R., Menzel, R., Menzel, W., Ratsch, U., Richter, M. M., and Stamatescu, I. O., editors, Adaptivity and Learning: An Interdisciplinary Debate, pages 217241. Springer Verlag.

[9] Demiris, J. (1999). Movement Imitation Mechanisms in Robots and Humans. PhD thesis, University of Edinburgh, Scotland, UK.

[10] Demiris, J. and Hayes, G. (1996). Imitative Learning Mechanisms in Robots and Humans. In Proceedings of 5th European Workshop on Learning Robots, pages: 9-16.

[11] Edwards, A. L. (1976). An Introduction to Linear Regression and Correlation. W. H. Freeman Company.

[12] Erbas, M. D., Winfield, A. F. T. and Bull, L. (2014). Embodied Imitation-Enhanced Reinforcement Learning in Multi-Agent Systems, Adaptive Behavior, 22(1), Sage Publications.

[13] Gerkey, B. P., Vaughan, R. T., and Howard, A. (2003). The Player/Stage Project: Tools for Multi-robot and Distributed Sensor Systems. In Proceedings of the 11th International Conference on Advanced Robotics, pages 317323. Los Alamitos: IEEE Computer Society Press.

[14] Kirby, S. (2007). The evolution of meaning-space structure through iterated learning. In Lyon, C., Nehaniv, C., and Cangelosi, A., editors, Emergence of Communication and Language, pages: 253-268. Springer Verlag.

[15] Kirby, S., Cornish, H., and Smith, K. (2008). Cumulative Cultural Evolution in the Laboratory: an experimental approach to the origins of structure in human language. Proceedings of the National Academy of Sciences, 105(31), pages: 10681-10686.

[16] Liu, W. and Winfield, A. F. T. (2011). Open-hardware e-puck Linux Extension Board for Experimental Swarm Robotics Research. Microprocessors and Microsystems, 35(1).

[17] Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J. C., Floreano, D., and Martinoli, A. (2009). The e-puck, a Robot Designed for Education in Engineering. In 9th Conference on Autonomous Robot Systems and Competitions, page 59:65.

[18] Nehaniv, C. L. (2007). Nine Billion Correspondence Problems. In Nehaniv, C. L. and Dautenhahn, K. (eds.), Imitation and Social Learning in Robots, Humans and Animals, pages 3546. Cambridge University Press.

[19] Nehaniv, C. L. and Dautenhahn, K. (eds.) (2002). Imitation in Animals and Artefacts. MIT Press.

[20] Nehaniv, C. L. and Dautenhahn, K. (eds.) (2007). Imitation and Social Learning in Robots, Humans and Animals. Cambridge University Press.

[21] Nolfi, S. and Floreano, D. (2000). Evolutionary Robotics, MIT Press.

[22] Parisi, D. (1997). Cultural Evolution in Neural Networks. IEEE Experts, 12(4):9-11.

[23] Steels, L. and Kaplan, F. (2001). Aibos First Words: The social Learning of Language and Meaning. Evolution of Communication, 4(1):332.

[24] Winfield A. F. T. and Erbas M. D. (2011). On Embodied Memetic Evolution and the Emergence of Behavioural Traditions in Robots. Memetic Computing. 3(4), pages: 261-270.