Through the Looking-Glass with ALICE — Trying to Imitate using Correspondences

Aris Alissandrakis, Chrystopher L. Nehaniv, and Kerstin Dautenhahn

Adaptive Systems Research Group
University of Hertfordshire
Hatfield, Hertfordshire
AL10 9AB, United Kingdom
{a.alissandrakis, c.l.nehaniv, k.dautenhahn}@herts.ac.uk
http://homepages.feis.herts.ac.uk/~nehaniv/ASRG

And Alice got the Red Queen off the table, and set it up before the kitten as a model for it to imitate: however, the thing didn't succeed, principally, Alice said, because the kitten wouldn't fold its arms properly. So, to punish it, she held it up to the Looking-glass, that it might see how sulky it was – "and if you're not good directly," she added, "I'll put you through into Looking-glass House. How would you like THAT? Now, if you'll only attend, Kitty, and not talk so much, I'll tell you all my ideas about Looking-glass House. First, there's the room you can see through the glass—that's just the same as our drawing room, only the things go the other way."

-Lewis Carroll, *Through the Looking Glass* (1871)

Abstract. Interactive behavior of biological agents represents an important area in life as we know it. Behavior matching and imitation may serve as fundamental mechanisms for the development of societies and individuals. Imitation and observational learning as means for acquiring new behaviors also represent a largely untapped resource for robotics and artificial life — both in the study of life as it could be and for applications of biological tricks to synthetic worlds. This paper describes a new general imitating mechanism called ALICE (Action Learning for Imitation via Correspondences between Embodiments) that addresses the important correspondence problem in imitation. The mechanism is implemented and illustrated on the chessworld test-bed that was used in previous work to address the effects of agent embodiment, metrics and granularity when learning how to imitate another. The performance of the imitating agent is shown to improve when ALICE is complementing its imitation behavior generating mechanism.

1. Introduction

In trying to get the kitten to imitate the red queen, Lewis Carroll's Alice found that things did not quite correspond. In the fantasy world she entered through the looking-glass, Alice often found that things did not work quite the same way as in her familiar world. She had to modify her natural behavior in order to get along in the similar, but not quite the same, looking-glass world. We share with Alice a fascination for worlds and artifacts that are similar but not quite the same.

A characteristic of many social animal species *as-we-know-them*, e.g. dolphins, chimpanzees, humans and other apes, is the ability to learn from others by imitation (see Zentall and Galef 1988, Heyes and Galef 1993, Nadel and Butterworth 1999). Inspired by nature, over the past decade many researchers have attempted to design life-like social artifacts *as-they-could-be*, i.e. software or robotic artifacts that are able to learn from each other or from human beings (Kuniyoshi et al. 1990, Hayes and Demiris 1994, Dautenhahn 1995, Gaussier et al. 1998, Billard and Dautenhahn 1998, Billard 2000). On the one hand, a robot or a software program that a human can teach simply by showing or demonstrating what needs to be done is an exciting new programming paradigm (Cypher 1993, Atkeson et al. 2000). On the other hand, imitation plays a crucial part in the development of animals as social beings. Robots or software systems that possess imitative skills are therefore an important step towards truly social artifacts (see discussions in Dautenhahn and Nehaniv, in press).

Imitation is a scientific challenge in many ways. In addition to fundamental problems of who, what, when and how to imitate, finding mappings between corresponding body parts and actions (e.g. lifting the right leg when imitating another human who is lifting the right leg) is a major challenge for artifacts to be solved. The work presented in this paper specifically addresses this *correspondence problem* (Nehaniv and Dautenhahn 2000, 2001; Alissandrakis et al. 2000, Dautenhahn and Nehaniv in press). While research on imitation usually takes the approach of studying *learning by imitation* (assuming that an artifact already possesses the skill to imitate successfully), this paper addresses the complementary approach of *trying to imitate* or *learning how to imitate* (Dautenhahn, 1994). We investigate how different such attempts at imitation can be evaluated, quantified and illustrate possible mechanisms for solving the correspondence problem between demonstrator and imitator. Differences in embodiment between animals, robotic and software systems make it more difficult but not necessarily impossible to acquire corresponding behaviors.

In our approach, a correspondence is a recipe through which an imitator can map observed actions of the demonstrator to its own repertoire of actions as constrained by its embodiment and by context (Nehaniv and Dautenhahn 2000, 2001, in press). A correspondence thus serves as a 'looking-glass' through which an observed demonstrator's behavior is 'refracted' to yield similar, but possibly not quite the same, action sequences for the imitator. This allows it to get along in its environment, using the affordances of its own embodiment, while exploiting observations of the behavior of others. We address the role that imitation can play in the development of behavioral competencies in artificial agents. Imitation and interaction games between human infants and caretakers, e.g. turn-taking, play an important role in the development of social cognition and communication in the young human animal (see articles in Uzgiris et al. 1989, Nadel and Butterworth 1999). The fact that an imitative action -- even an accidental one-- may receive positive feedback could increase the animal's motivation and tendency to imitate (cf. Dautenhahn and Nehaniv, in press b). Moreover, this can serve to draw attention toward salient aspects of the environment and reveal affordances of actions and objects useful for survival in the course of ontogeny.

2. Chessworld Revisited

For purposes of this research, we use the chessworld test-bed that was first introduced and described in (Alissandrakis et al. 2000). It consists of a chessboard on which agents as chess pieces travel according to the movement rules defined by their type e.g. the Bishop can only move diagonally on squares of the same color. Note that our intention is only to borrow elements like the simple discrete nature and the different piece embodiments. We do not to consider the actual game of chess. Each move results in a displacement of the piece on the chessboard, and these actions are to be imitated. The success of the imitation can be measured by using many different metrics. In this case three simple standard distance metrics are used, namely Hamming norm, Euclidean distance and infinity norm (e.g. Rudin, 1976).

In the work described in our previous paper (Alissandrakis et al. 2000), as a generating mechanism the imitating agent uses only a simple greedy type algorithm to match the actions of the demonstrator. This *greedy algorithm* tries to match the behavior by choosing a valid action for the imitator minimizing the distance (measured using a metric) between the square that was visited as a result of the demonstrator move and the square reached by the imitator. By ‘valid’ we mean that this action can be performed by the imitator chess piece type and does not result in the piece moving beyond the edges of the board. This can be repeated until the algorithm produces an attempted matching behavior i.e. a sequence of actions that can then be used by the imitator to move as close as possible to that target location.

For example let us consider a Queen as the demonstrator that performs the action E3 (move three squares to the east). If the imitator is another Queen, the algorithm will simply produce the sequence [E3]. The same sequence will be produced if the imitator is a Rook. If the imitator is instead a King, the algorithm will produce the sequence [E, E, E] (three sequential moves of a single square to the east). If the imitator is a Bishop, the algorithm will produce [NE, SE] or [SE, NE]. Note that due to embodiment limitations (the Bishop cannot occupy the target square as it is of different color) moving according to either action sequence, the imitator cannot reach the desired square exactly, but only an adjacent one. Similar embodiment issues occur for an imitator Knight using the sequences [N1E2] or [S1E2].[1]

[1] To avoid confusion interpreting the action names, the entire Knight action set is {E1N2, E1S2, W1N2, W1S2, N1E2, N1W2, S1E2, S1W2}. We try to avoid multiple names for actions such as E2N1 and N1E2, which both would correspond to hopping two squares east and one square north, or, equivalently, one square north and two squares east.

Table 1. Possible correspondence relations for different imitator types found using the greedy generating algorithm (Euclidean distance metric used), together with performance *P* of the attempted match (see section 4). *Disp.* is the relative displacement effect of the action sequence. Given the demonstrator type, action and effect, the possible sequences with their effects for each imitator type are shown below.

Demonstrator	*Action*	*Disp*
Queen	E3	(+3,0)

Imitator	*Sequence*	*Disp.*	*P*
Queen	[E3]	(+3,0)	100%
Rook	[E3]	(+3,0)	100%
King	[E,E,E]	(+3,0)	100%
Bishop	[NE,SE] or [SE,NE]	(+2,0)	67%
Knight	[N1E2]	(+2,+1)	53%
	[S1E2]	(+2,-1)	53%

In order to study different types of imitation, the squares visited as a result of the demonstrator’s actions were presented to the algorithm in three different ways to emulate different sub-goal granularity. For *end-goal level granularity* the imitating agent was using as input to the algorithm only the final square visited as a result of an entire sequence of demonstrator moves instead of using sequentially each of the squares visited at each move. At *path level granularity* the algorithm must also go through the intermediate squares between the visited squares during the demonstrator sequence; e.g. for a Queen starting at square (1,1) and reaching square (4,4) by a diagonal move, the imitator would have to also sequentially consider the squares (2,2) and (3,3) along the path of the demonstrator move. For the current work only *trajectory level granularity* – matching the end result of actions on a move-by-move basis - will be used. This takes advantage of the natural segmentation of actions (moves) in the chessworld, but also conceals deep issues of perception and action that must be addressed in physical applications (cf. Heyes and Ray 2000, Nehaniv and Dautenhahn 2001).

3. Go Ask ALICE — Introducing Correspondences

Building on the earlier work (Alissandrakis et al. 2000) in a discrete chessworld environment, we introduce ALICE (**A**ction **L**earning for **I**mitation via **C**orrespondences between **E**mbodiments) to illustrate how building up correspondences can help solve the problem of how to perform similar behavior with a possibly different body. ALICE is a generic mechanism for building up a correspondence based on any generating method for attempts at imitation by examining the history of such attempts (cf. Byrne’s string parsing approach to imitation (Byrne 1999)).

The correspondence library that ALICE builds up functions as a kind of mirror through which to refract a demonstrator's behavior into the repertoire of the imitator's own actions as constrained by its embodiment. Such a library of action correspondences can be employed when imitating (cf. the natural imitation of humans by dolphins (Herman, in press) or robotic imitation (Nehaniv and Dautenhahn 2000, 2001)). Mechanisms and correspondences of this type are also relevant to the imitation of perceptually opaque behaviors[2] and to sensory motor correspondences (Heyes and Galef, 1993), to the extraction of the structure of demonstrated behavior (Byrne 1999, Whiten in press,), and to neural mechanisms for the perception of actions and affordances and its direct mapping to motor actions via 'mirror neurons' (Gallese et al. 1996, Rizzolatti and Arbib 1998, Arbib in press).

ALICE is comprised of two major components on top of the generating mechanism. First, when the imitator observes a new demonstrator action not seen before, the imitator can relate the result of the generating mechanism used (in this case the greedy algorithm) to that action. This relation is then placed in the library of correspondences[3].

Using the entries in the library instead of performing the algorithm for actions already observed is less computationally expensive; especially when the complexity of the algorithm that produces the matching behavior increases. In this case the generating algorithm used is a simple one, but potentially in another setting that may no longer be true, and the cost of recalculating instead of using an already found solution may be considerable - for example a ten degrees of freedom robot arm in the real world having to solve again the inverse kinematics equations for moving the manipulator to a point in the workspace that has been visited before from the same initial configuration.

If correspondences were built up making use only of the sequences generated by the generating algorithm, the imitator could not perform any better, although it could perform faster. Some of these imitator sequences related to the demonstrator actions may even be invalid in certain contexts. For example the Bishop cannot use the sequence [NE, SE] to imitate the Queen action E3 (see Table 1) if the piece is currently located along the northern edge of the board. So the need to discover and consider also the alternative sequence [SE, NE] becomes apparent if the agent is to mostly rely on using the correspondence library instead of using the generating algorithm every time. For the same demonstrator action it is impossible for the Bishop to achieve perfect imitation, as the piece cannot occupy the target location due to its movement rules. But for the Knight there do exist possible imitating sequences that can achieve this required displacement e.g. [W1S2, S1E2, S1E2]. These sequences cannot be found using the greedy generating algorithm because the metric measured distance not only decreases but also increases as a result of certain actions.

[2] Perceptually opaque behaviors (Heyes and Galef, 1993) are perceived very differently when observed than when being performed e.g. tongue protrusion or winking, but not singing.

[3] At each stage in its growth, a library of correspondences is an example of a (partial) relational homomorphism between the abstract automata associated to the demonstrator and the imitator (Nehaniv and Dautenhahn 2001, in press).

The second component of ALICE overcomes this difficulty: To discover such sequences the imitator agent can examine its own history without having to modify or improve the generating algorithm used. By history we mean the list of actions that were performed so far by the agent while imitating the demonstrator together with these actions' relative displacements and possible effects on the environment, ignoring the imitation context. This history provides helpful experience data that ALICE uses to extract useful mappings to improve and add to the correspondence relation library created up to that point. The methods for actually extracting this information can vary and also managing the sequences that are found can depend on additional metrics e.g. keep only the shortest sequence that can achieve that displacement, or keep only the top five sequences according to performance. For the current work any number of sequences (of length up to five actions) can be related to an observed demonstrator action, while keeping track of their performance.

A summary of the ALICE mechanism is given by the following pseudocode.

```
ALICE MECHANISM PSEUDOCODE

Consider the demonstrator behavior as a
sequence of actions.

For each of these demonstrator actions:

    •  Create new entry in the
       correspondence library and add
       sequence of imitator actions
       found by the generating
       mechanism, if the demonstrator
       action has not been observed
       before.

    •  Use appropriate imitator actions
       sequence from the correspondence
       library if entry already exists.

Examine history by considering
sequences of past actions performed so
far.

For each of these sequences:

    •  If sequence of imitator actions
       produces same effects to known
       demonstrator action, add
       sequence to that entry in the
       correspondence library.
```

In figure 1, four different possible solutions to the correspondence problem for a Knight imitating a Queen (or Bishop) performing a particular diagonal move are shown. All result in the imitator achieving the same displacement as the imitator, although each follows a different trajectory. Note that the two sequences in the bottom can become invalid if the imitator is too close to the upper or lower edges of the board, making again apparent that having several options in the correspondence library is helpful.

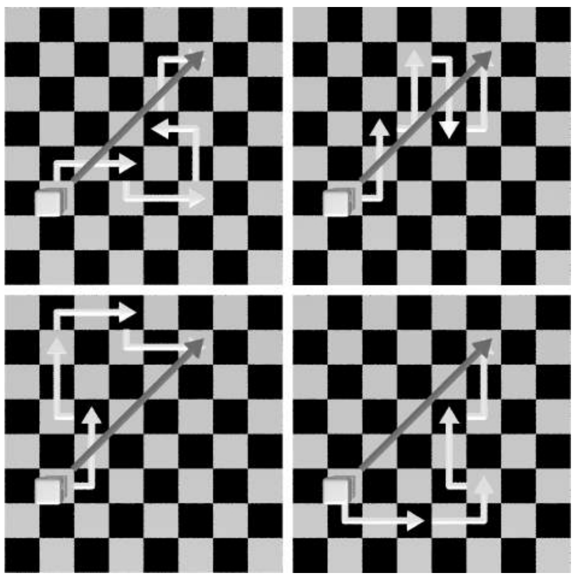

Fig. 1. This figure shows four different possible corresponding sequences ([N1E2, S1E2, W1N2, E1N2], [E1N2, E1N2, E1S2, E1N2], [E1N2, W1N2, N1E2, S1E2] and [S1E2, N1E2, W1N2, E1N2]) that can be used by the Knight to imitate the action NE4 by the demonstrator Queen (or Bishop). These can be found with ALICE but cannot with only the greedy algorithm.

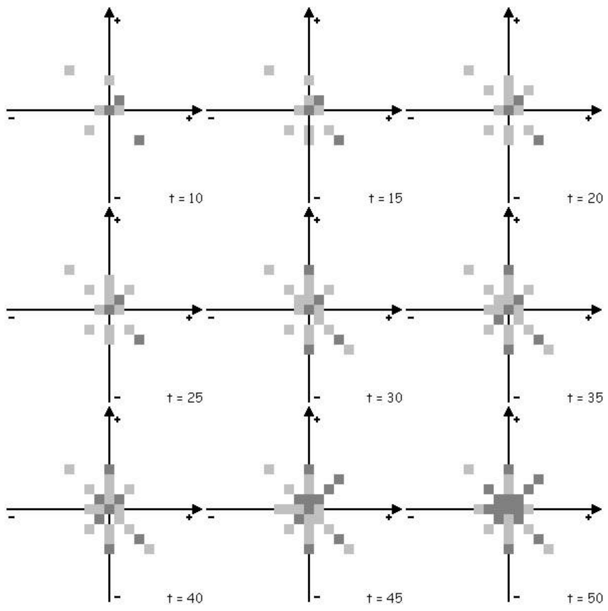

Fig. 2. A possible development for a demonstrator Queen imitator Knight correspondence library. The correspondence build-up with ALICE is shown in intervals of five simulation steps, left to right, top to bottom. At the displacement coordinates for each observed demonstrator action dark or light color tones indicate whether at least one of the corresponding sequences satisfy the imitation criteria perfectly or not, respectively. Given that the demonstrator to be imitated is a Queen, the shape that slowly emerges is composed of the vertical, horizontal and diagonal directions that characterize its movement rules.

The figure 2 above shows the development of such a correspondence library for a Knight imitating a Queen. At each of the time instances shown, every observed demonstrator action is noted as a point at the appropriate vertical and horizontal co-ordinates of its resulting displacement. These can be both negative and positive, relative to the current location of the chess piece. If at least one of the correspondence sequences found so far accomplishes that exact displacement a dark color tone is used, otherwise a light one. The shape that slowly emerges relates to the set of demonstrator actions observed so far.

When every demonstrator action has been encountered at least once, we can say that the set is complete with at least one corresponding sequence for every possible action. But such a complete set of correspondence relations between a demonstrator and an imitator cannot necessary guarantee a consistently satisfying performance of imitation, even in simple environments like the chessworld. The corresponding sequence may be invalid in a different context than the one it was observed in, for example by requiring movement outside the edges of the chessboard. It becomes apparent that as the world resolution and complexity increases, the context becomes more relevant and therefore the variety and quality of the correspondence relations becomes more important. Using the mechanism that extracts sequences from the history as an ongoing feature can address this, as it will continue to enrich the individual mappings with more alternatives that possibly provide better solutions.

4. Experimental Set-up and Results

Using the Swarm simulation system, a series of exhaustive experiments were done, implementing ALICE on the chessworld test-bed. At each run, a demonstrator agent is performing a random walk on the chessboard using any (valid) actions from the set defined by its chess piece type. An imitating agent starts from the same initial square (at the beginning of the run) and tries to imitate each of these actions using sequences consisting of actions (of its own type) returned from the generating mechanism (in this case the greedy algorithm explained in section 2 above). For each combination of demonstrator and imitator types two experiments are carried out, one using the correspondences built up with ALICE enabled and one without.

Due to space limitations, only results for imitator Knight with demonstrator Queen are shown. In figures 3 and 4, for a random walk of 2100 demonstrator actions, the imitation performance is shown as the percentage of the distance from the target location achieved by the demonstrator action that the imitator is able to cover by performing the imitating sequence. In each figure the three subplots correspond to the three different metrics used. More precisely, given a fixed metric, performance P is measured as $P = (a\text{-}b)/a$, where the distances from the imitator's position to the target position before and after performing the imitating sequence are a and b respectively. For clarity, a running average trendline of P with period of 15 demonstrator actions (time steps) is shown in each case. The plots in both figures are from representative model runs and also show that the choice of metric used does not affect the imitation success as calculated by this performance measure.

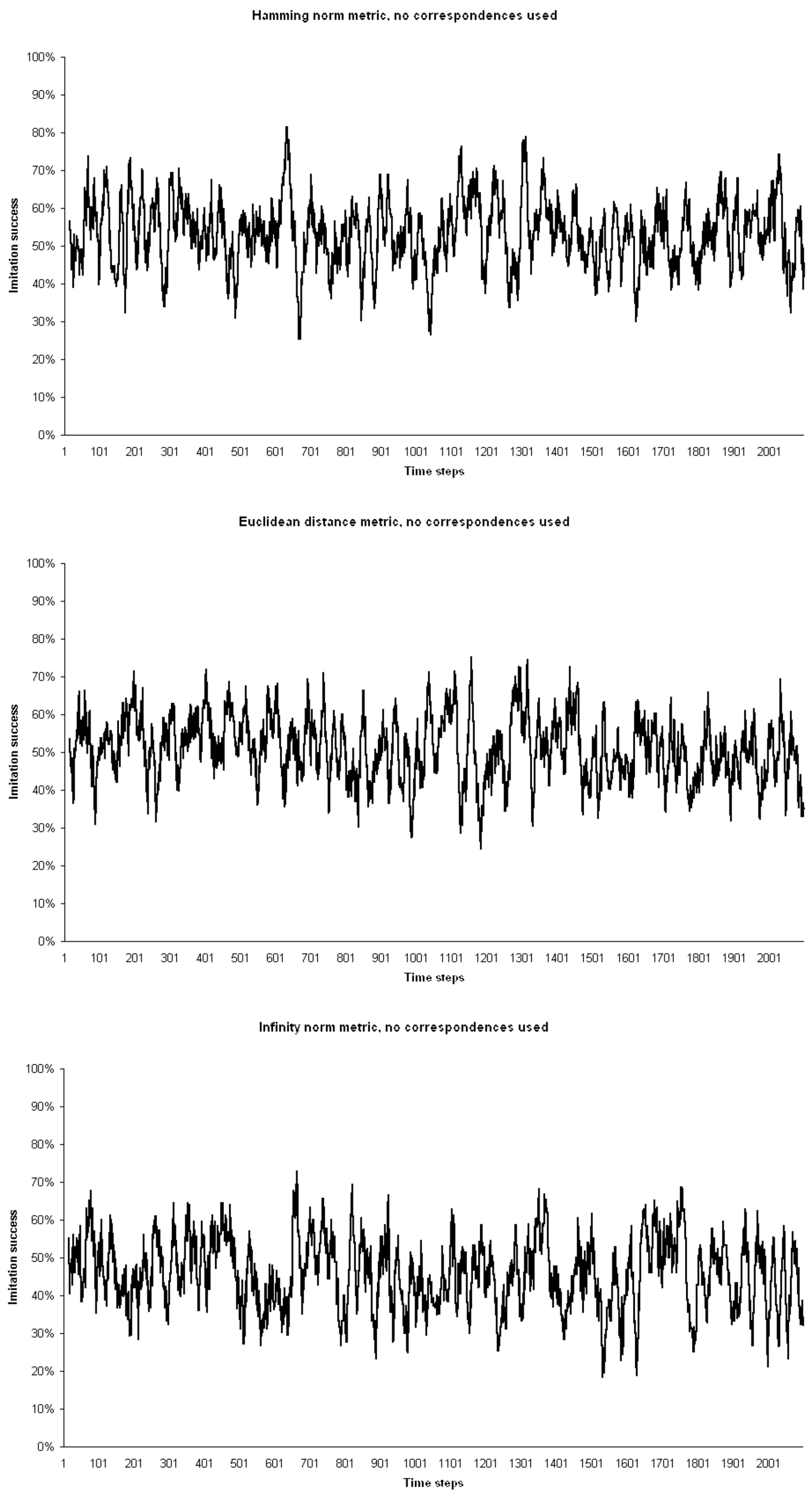

Fig. 3. Imitation performance without ALICE for the Knight imitating the Queen using Hamming norm (top), Euclidean (middle) and infinity norm (bottom) metrics, for a random walk comprised of 2100 demonstrator actions. In each plot, a moving average performance trendline with a period of 15 time steps is shown. Imitation success is measured as performance *P* described in the text.

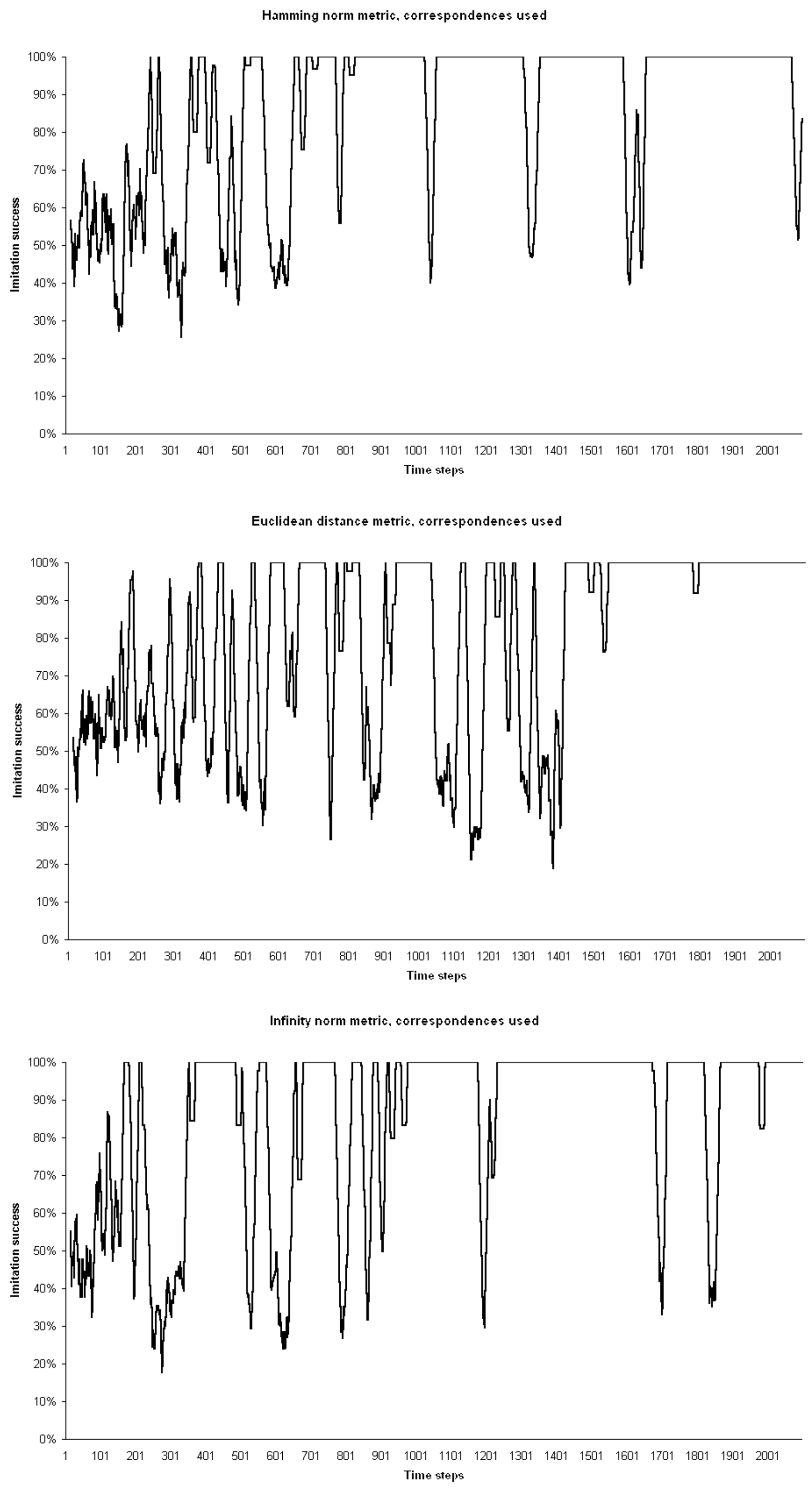

Fig. 4. Imitation performance with ALICE for the Knight imitating the Queen using Hamming norm (top), Euclidean (middle) and infinity norm (bottom) metrics, for a random walk comprised of 2100 demonstrator actions. In each plot, a moving average performance trendline with a period of 15 time steps is shown. Imitation success is measured as performance *P* described in the text.

In figure 3 we can see that the generating mechanism alone is unable to find all but a very small portion of perfectly matching sequences for the demonstrator's set of actions, resulting in an overall mediocre performance of imitation, exhibiting no improvement over time. In figure 4 the use of the ALICE mechanism to build up the correspondence relation library results in an overall improvement and extended periods of P = 100% performance.

The glitches observed between the long periods of such performance are due to the context of situations that can make the sequences learned so far corresponding to an action invalid (as discussed in section 3), although alternatives may later be discovered by the sequence extracting component of ALICE that looks for matches of the effects of sequences in the imitator's history to previously observed actions of the demonstrator.

5. Discussion

Imitation and behavioral matching can serve as a fundamental component for behavior acquisition and life-like social interaction in life as it could be. The work presented in (Alissandrakis et al. 2000) showed that agent embodiment, together with the use of different metrics and sub-goal granularities can affect the success and character of the imitation observed. In this work we introduce ALICE (Action Learning for Imitation via Correspondences between Embodiments). A correspondence serves as a 'looking-glass' through which an observed demonstrator's behavior is 'refracted' to yield similar, but possibly not quite the same, action sequences for the imitator. This allows it to get along, using the affordances of its own embodiment, while exploiting observations of the behavior of others in its environment. We show how the imitator agent with ALICE exposed to the demonstrator behavior in the chessworld, can build up useful partial solutions to the correspondence problem, i.e. mapping the demonstrator actions to those it can perform in it own particular embodiment to achieve similar effects, exhibiting highly successful imitating performance. Effectively ALICE provides a combination of learning and memory to help solve the correspondence problem. Due to its generic nature, it can be implemented in a variety of ways, not depending on a specific generating mechanism. Future work will bring ALICE to more complex real world test-beds, studying the requirements for the different mechanism components (the imitating sequence generating algorithm and the algorithm that extracts alternative corresponding sequences from the history of the imitator) in more complex settings, addressing issues of perception, segmentation, context, self repair (via self-imitation of previous optimal behavior), development and interaction.

References

A. Alissandrakis, C. L. Nehaniv and K. Dautenhahn (2000), Learning How to Do Things with Imitation, *AAAI Fall Symposium on Learning How to Do Things*, 3-5 November 2000, American Association for Artificial Intelligence, pp. 1-6.

M. Arbib (in press), The mirror system, imitation, and the evolution of language, in (Dautenhahn and Nehaniv, in press).

C. G. Atkeson, J. G. Hale, F. Pollick, M. Riley, S. Kotosaka, S. Schaal, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, E. Kawato and M. Kawato (2000), Using Humanoid Robots to Study Human Behavior. *IEEE Intelligent Systems*, 15(4): 46 -56.

A. Billard (2001), Imitation: A means to enhance learning of a synthetic proto-language in autonomous robots. In (Dautenhahn and Nehaniv, in press).

A. Billard and K. Dautenhahn (1998), Grounding communication in autonomous robots: an experimental study. *Robotics and Autonomous Systems*, 24(1-2):71-81.

R. W. Byrne (1999), Imitation without intentionality. Using string parsing to copy the organization of behaviour, *Animal Cognition*, 2:63-72.

A. Cypher, Ed. (1993), *Watch What I Do: Programming by Demonstration*, MIT Press.

K. Dautenhahn (1994), Trying to Imitate — a step towards releasing robots from social isolation. In P. Gaussier and J.-D. Nicoud (Eds.), *Proc. From Perception to Actions Conference (Lausanne, Switzerland)*, IEEE Computer Society Press, pp. 290-301.

K. Dautenhahn (1995), Getting to know each other - artificial social intelligence for autonomous robots, *Robotics and Autonomous Systems*, 16:333-256.

K. Dautenhahn and C. L. Nehaniv, Eds. (in press), *Imitation in Animals and Artifacts*, MIT Press.

K. Dautenhahn and C. L. Nehaniv (in press b), The Agent-Based Perspective on Imitation. In (Dautenhahn and Nehaniv, in press).

V. Gallese, L. Fadiga, L. Fogassi and G. Rizzolatti (1996), Action recognition in the premotor cortex, *Brain* 119:593-609.

P. Gaussier, S. Moga, J. P. Banquet, and M. Quoy (1998), From perception-action loops to imitation processes: A bottom-up approach to learning by imitation. *Applied Artificial Intelligence Journal*, special issue on "Socially Intelligent Agents", 12(7-8):701-729.

G. Hayes and J. Demiris (1994), A robot controller using learning by imitation. In A. Borkowski and J. L. Crowley (Eds.), *SIRS-94, Proceedings of the 2nd International Symposium on Intelligent Robotic Systems*, LIFIA-IMAG, Grenoble, France, pp. 198-204, July 1994.

L. M. Herman (in press), Vocal, social, and self imitation by bottlenosed dolphins. In (Dautenhahn and Nehaniv, in press).

C. M. Heyes and B. G. Galef, Jr., Eds. (1993), *Learning in Animals: The Roots of Culture*, Academic Press.

C. M. Heyes and E. D. Ray (2000), What is the significance of imitation in animals? *Advances in the Study of Behavior*, 29:215-245.

Y. Kuniyoshi, H. Inoue and M. Inaba (1990), Design and implementation of a system that generates assembly programs from visual recognition of human action sequences. In *Proc. IEEE International Workshop on Intelligent Robots and Systems IROS '90*, pp. 567-574.

J. Nadel and G. Butterworth, Eds. (1999), *Imitation in Infancy*, Cambridge University Press.

C. L. Nehaniv and K. Dautenhahn (2000), Of Hummingbirds and Helicopters: An Algebraic Framework for Interdisciplinary Studies of Imitation and Its Applications. In: J. Demiris and A. Birk, eds., *Interdisciplinary Approaches to Robot Learning*, World Scientific Series in Robotics and Intelligent Systems - Vol. 24.

C. L. Nehaniv and K. Dautenhahn (2001), Like Me? - Measures of Correspondence and Imitation, *Cybernetics & Systems: An International Journal*, 32(1-2): 11-51.

C. L. Nehaniv and K. Dautenhahn (in press), The Correspondence Problem, in (Dautenhahn and Nehaniv, in press).

G. Rizzolatti and M. A. Arbib (1998), Language within our grasp, *Trends in Neurosciences*, 21(5):188-194.

W. Rudin (1976), *Principles of Mathematical Analysis*, third edition. New York: McGraw-Hill.

I. C. Uzgiris, J. Benson, J. Kruper & M. Vasek (1989), Establishing action-environment correspondences: Contextual influences on imitative interactions between mothers and infants. In J. Lockman & N. Hazen (Eds.), Action in Social Context, Plenum, pp. 103-127.

T. R. Zentall and B. G. Galef, Jr., Eds. (1988), *Social Learning: Psychological and Biological Perspectives*, Lawrence Erlbaum Associates, Hillsdale, NJ.

A. Whiten (in press), Imitation of sequential and hierarchical structure in action: Experimental studies with children and chimpanzee. In (Dautenhahn and Nehaniv, in press).

Swarm Development Group Homepage, http://www.swarm.org