Emerging shared action categories in robotic agents through imitation

Bart Jansen, Bart De Vylder, Bart de Boer and Tony Belpaeme[⋆]
[⋆] Vrije Universiteit Brussel
Artificial Intelligence Laboratory
Pleinlaan 2, 1050 Brussels, Belgium

{bartj,bartdv,bartb,tony}@arti.vub.ac.be

Abstract

In this paper we present our work on developing a shared repertoire of action categories through imitation. A population of robotic agents invents and shares a repertoire of action categories by engaging in imitative interactions. We present an experimental set-up which enables us to investigate what properties agents should have in order to achieve this. Among these properties are: being able to determine the other's actions from visual observation and doing incremental unsupervised categorisation of actions.

1 Introduction

Most of the work that has been published on imitation in robots focuses on the learning of action categories in a teacher - student context (Vogt, 2000; Billard and Hayes, 1997; Alissandrakis et al., 2002) . In such a set-up one agent acting as teacher already has action categories. By observing the teacher who is executing actions, the action categories can be passed on to the student: by imitating the teacher's action using for instance inverse kinematics and evaluating that action, the learner can know whether he correctly reconstructed the observed action. However, such a set-up does not explain how action categories emerge. How does the teacher acquire its categories when they are not preprogrammed by a human operator?

We propose a set-up in which new action categories can emerge when imitation of actions fails. This is done in a population of agents engaging in imitative interactions, called *imitation games*. Action categories are only learned if they can be successfully imitated. If an action is hard to observe or to imitate, it will not be learned by other agents. The experiments are conducted both in simulation and on real robots, however results presented in this article were only obtained in simulation. Our concept of imitation games strongly resembles the concept of imitation games used in (de Boer, 2000) in the context of vowel systems. The imitation game presented in this paper is work in progress.

In section two, our experimental set-up is proposed. Section three presents the actual imitation game, section four proposes objective measures for determining how successful the imitation game is and results are presented in section six. Future work is discussed in section seven.

Figure 1: The experimental set-up, consisting of a stereo head and a robot arm.

2 Experimental set-up

An agent consists of a stereo camera head and a robot arm, see figure 1. With the arm it can make different kinds of gestures which it can observe through a vision system. Gestures are restricted to motion trajectories from one point to another. Gestures do not involve manipulations of other objects and do not carry any meaning, yet.

Using this set-up, we are investigating how a repertoire of grounded action categories can emerge. For investigating how this repertoire can be shared through a population of agents, we use several agents which interact. However, at this stage of the project we do not have multiple physical installations. One robot arm and one vision system are used by all agents of the population. Agents take turns in using the arm and stereo head[1].

[1]A solution also used in a previous experiment investigating the emergence of word-meaning lexicons on embodied agents (Steels and Kaplan, 1999).

2.1 The robot arm

We use a readily available commercial robot arm, called Teach-robot[2]. It has six degrees of freedom and is equipped with a gripper. The forward and inverse kinematics are known (De Vylder, 2002). The robot arm is position-driven: one can send goal motor positions to it and query its current position but one cannot control the velocity of the movement nor query the robot while it is moving. At the moment the gestures the robot makes are only interpreted on the basis of the gripper's trajectory. No attention is paid to movements of other joints. The gripper is clearly marked with a bright colour to facilitate image processing.

2.2 The vision system

The observation of an action results in a series of gripper coordinates. These three dimensional time series will be categorised during the imitation game. The vision system focuses on finding the 3D gripper coordinates in the captured images. We use a MEGA-D stereo head for acquiring both left and right images and the Small Vision System (SVS) from Konolige [3] for obtaining a depth map. Using colour templates, the left image is segmented into gripper and non-gripper regions, the right image is left unprocessed. Using simple tracking mechanisms, the most probable gripper regions are extracted. From the gripper segment, a set of features is extracted, such as colour and size. From all gripper pixels in the left image, the 3D coordinates are obtained using the SVS. Finally, the position of the gripper is defined as the average of all coordinates obtained. If the stereo matching in the SVS fails, the right image is segmented as well. The centre of the left image gripper region and the centre of the right image gripper region are then assumed to correspond. Using this correspondence, the 3D coordinates of the gripper can be approximated. The entire process is repeated approximately 25 times per second. We assume an agent knows when it needs to start and stop observing its own actions and we explicitly tell the agent when it needs to observe another agent which is executing an action. In principle, the second agent could determine on its own when a new action begins, for instance by splitting sequences at the frames in which no change is observed.

Using the methods described above, the observation of an action results in a sequence of 3D coordinates representing the position of the gripper at different time steps during execution of an action. There is no restriction on the length of such sequences. As conditions of our experimental set-up are not tightly controlled, the calculated gripper positions are not accurate and subject to large amounts of noise, caused by changing light conditions, people passing by or the presence of other disturbing factors. Filtering could improve the quality of the time series.

[2]Microelectronic Kalms, http://www.teach-robot.de/

[3]http://www.ai.sri.com/konolige/svs/Papers

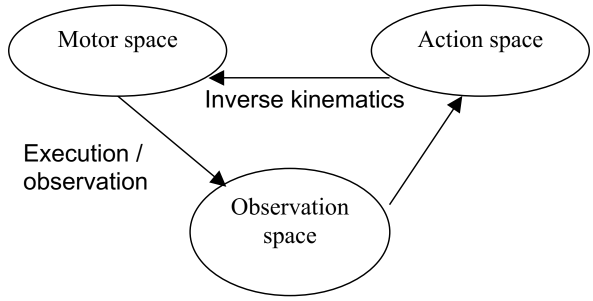

Figure 2: The motor space, action space and observation space and their relations

2.3 The agent architecture

An *action category* is a tuple containing an *action* and an *observation*. The action is a sequence of 3D points, consisting of consecutive target coordinates of the gripper. In our experiments, we use only two points, i.e. a start point and an end point. The action executed by the robot is the movement of the gripper from start point to end point. The target coordinates are defined in an *action space*, defined by the coordinate system of the robot arm. The different spaces and the mappings between them are shown in figure 2. The robot arm is position driven, meaning that for each of the six motors, a target position needs to be specified. So, the robot is controlled by sending it 6-tuples of target positions, called *motor commands*. The motor commands define a *motor space*. The mapping from action space to motor space *(1)* is defined by the *inverse kinematics* which is assumed to be known to the agents. Note that action categories contain actions defined in the action space and not motor commands. Both approaches are equivalent, but it is easier to define actions in the action space, as the Euclidean distance holds in the action space.

Using its vision system, the robot observes the execution of actions as sequences of 3D gripper coordinates, called observations. The gripper coordinates are defined in the camera coordinate system, defining an *observation space*. There is a mapping from points in the observation space to points in the action space *(2)*, defined by a ca-libration matrix which is known to the agents. Note that there is also a direct mapping from motor space to observation space *(3)*. This mapping does not need not to be preprogrammed, as the agent can execute a motor command and observe the result of its own execution. The agent can obtain the inverse mappings of (1),(2) and (3) by concatenating its known mappings. Using the mappings explained above, an agent can always know which observation corresponds to a given action, by using inverse kinematics and executing the appropriate motor commands. In the pseudo-code used in this document, this synthesis step is called *S*. From an observation, the corresponding action can be derived by the agent using the calibration. This step is called S_{inv} in the pseudo-code. It is our goal to show that action categories can even be learnt without inverse kinematics being known to the agents and without inverse kinematics. At the beginning of the imitation games, all agents start empty, they have no preprogrammed action categories. During the imitation games, agents learn action categories for the actions they observe. Those learnt action categories are stored in an *action category memory*.

3 Building a repertoire of basic action categories

The goal of the imitation game presented in this section is to develop a shared repertoire of action categories. As we do not investigate how imitation itself could emerge, we assume that all agents have the same learning mechanism and engage in preprogrammed imitation games.

The imitation game is a simple interaction between two agents comprising three essential elements of imitation: (1) the observation of an action, (2) the classification of the observed action and (3) the imitation of the observed action.

How an agent observes actions was described in section two. The observation results in a time series of 3D points. In the memory of action categories, the agent finds the action category with an associated observation closest to the new observation. This action category is the representation of the observed action. Imitation is performed by executing the action associated with the recognized action category.

For finding the action category with an associated observation closest to the observation just made, the agent needs a method to evaluate the distance between observations. We use Dynamic Time Warping (DTW)(Myers and Rabiner, 1981) as distance metric on observations of actions. DTW was for instance used in (Corradini, 2001).

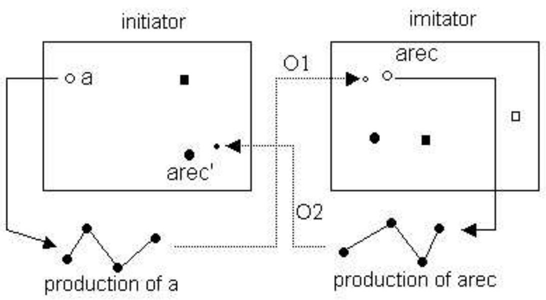

Figure 3: The imitation game. In this example it fails, as the imitation is categorised as a'_{rec} instead of a.

For every game two agents are randomly selected from the population. One agent will take the role of *initiator*, the other agent will be the *imitator*. Both agents use the vision system and the manipulator. In figure 3 the observation space for both the initiator and the imitator is depicted. The larger dots indicate the observations associated with the action categories, the small dots indicate the actual observations. The game starts when the initiator selects a random action category *a* from its memory. If its memory is still empty, a random action category is first added. The initiator executes the action that is associated with the action category *a*. The imitator observes this action and calls its observation O_1. Then it tries to categorise the observation O_1. If it finds a category, this category will be called a_{rec}. If the imitator has no categories yet, a new category a_{rec} is added for the observation O_1. The imitator now executes the action associated with the category a_{rec}. The initiator observes this action and calls its observation O_2. The initiator categorizes this observation and calls the category a'_{rec}. If the initial category and the category of the imitated action are equal ($a = a'_{rec}$), then the initiator decides that the game succeeds, otherwise it fails. The initiator sends non-verbal feedback about the outcome of the game to the imitator. If the game succeeded, the imitator shifts the category it used closer to the observation O_1 of the initiators action. If the game fails, two different update strategies are considered. If the success-ratio of the category the imitator used is above a certain threshold (*e.g. 0.5*), it means that the action category itself has been successful in the past. So, the failure in this game is probably caused because the initiator executed an action the imitator does not know yet. In this case a new category is constructed for the observation. If the success-ratio of the category used is below the threshold, the category itself is probably not very good. In that case, the category is shifted towards the observation. As a last step of the game, both the initiator and the imitator update their repertoire of action categories. They remove action categories that have proven not to be successful in the past and with a small probability they add new action categories to their repertoire. This forces the agents to optimise their growing repertoire. The general outline of the game can be seen in figure 4 in pseudo-code.

The addition of a new action category to the action category memory is done by generating a random action. The new action category consists of this random action and its observation.

Shifting an action category *c* towards an observation *O* means that the action category *c* is slightly changed such that the associated observation O_c resembles the observation *O* more closely. First the action *A* corresponding to the observation *O* and the difference between the actions *A* and A_c are calculated. The action associated to the shifted action category A_s will be the original action A_c added to a small portion of this difference, such that the actions *A* and A_c are now closer to each other. By executing this shifted action A_s the shifted observation O_s can be obtained.

A new action category for an observation is calculated

model	imitator
if *A* *a* random from *A* *a*.usage *a*.usage + 1 **execute** S(*a*.production)	
	observe O_1 **if** *A* = *A* new random action **else** a_{rec} action from *A* such that a_{rec}.observation closest to O_1 **execute** S(a_{rec}.production)
observe O2 a'_{rec} action from *A* such that a'_{rec}.observation closest to O_2 **if** $a_{rec} = a'_{rec}$ *a*.success *a*.success + 1 send non-verbal feedback : success	
	update-feedback(a_{rec}, O_1, success, A)
else send non-verbal feedback : failure	
	update-feedback(a_{rec}, O_1, failure, A)
do-other-updates() **else** *A* *A* a new random action	**do-other-updates**()

Figure 4: The outline of the imitation game

```
do-other-updates(A)
∀ actioncategory ∈ A do
   if actioncategory.success / actioncategory.usage < *throwawaytreshold* and
      actioncategory.usage > *minuses*
      A ← A \ actioncategory
with probability *addprobability* do  A ← A ∪ newactioncategory
----------------------------------------------------------------
update-feedback(a_rec,O1,signal,A)
a_rec.usage ← a_rec.usage + 1
if signal = positive
   a_rec.success ← a_rec.success + 1
   shiftcloser(a_rec,O1)
else
   if  a_rec.success / a_rec.usage > *threshold*
      A ← A ∪ findaction(O1)
   else
      shiftcloser(a_rec, O1)
----------------------------------------------------------------
shiftcloser(a, O)
a.action = a.action + *shifttreshold* x [ Sinv(O) -  a.action ]
a.observation = S(a.action)
----------------------------------------------------------------
findaction(O)
a.action = Sinv(O)
a.observation = O
return a
```

Figure 5: Pseudo-code for the different update procedures

using a similar technique. The pseudo code for these different update procedures is given in figure 5.

The result of a single imitation game is that the imitator adapts its action category memory such that it resembles the action categories of the initiator more closely. The imitator continues to adapt until it would perfectly imitate the initiator. However, all agents continuously add new random action categories, which forces them to keep on adapting. It is obvious that only a limited number of action categories can be learnt, as the execution and observation of action categories are bounded by the range of the robot arm and as they are both prone to noise.

As all agents engage many times in imitation games—both in the role of initiator and imitator—action categories are adapted as long as agents do not successfully imitate all actions. The imitation of an action can only be successful if the initiator's categorization and the imitator's categorization of the same action are close to each other. In the long run, this leads to a repertoire of action categories, shared by all agents.

4 Measures

In order to evaluate whether a population of agents can indeed develop a shared repertoire of action categories, three measures have been defined : *Imitative success*, *number of categories* and *category variance*. None of those measures operates on a single agent, they are all defined over the entire population.

The imitative success is simply the fraction of successful imitation games, averaged per 100 imitation games. As such, this measure shows how good the agents are capable of imitating each other and therefore it is a good measure of the quality of the action categories developed by all agents. However, if all agents only develop a single action category, imitative success reaches its maximum. Therefore, the imitative success must be compared to the *number of categories* the agents develop on average. After every imitation game, the number of categories of all agents is averaged. Those averages are averaged per 100 imitation games.

Using both the imitative success and the number of categories as measures, one can investigate whether all agents are capable of developing a repertoire of action categories which is suited for playing successful imitation games. Although it is easy to understand that successful imitation is not possible if the agents have very different action categories, it must be shown that the categories are shared throughout the population. Therefore we need to define how similar the categories of two agents are, or—in general—how similar two sets of points are in an N-dimensional space. Note that two agents can develop a different number of categories, so the sets can contain different numbers of points. The metric used here is taken from Belpaeme (2002). The similarity measure of the categories of two agents *Category Distance (CD)* is based on the *weighted sum of minimum distances* metric and is given in equation 1.

$$CD(A,B) = \frac{\sum_{a\in A} \min_{b\in B} d(a,b) + \sum_{b\in B} \min_{a\in A} d(a,b)}{|A|.|B|} \quad (1)$$

We can now calculate the *Category Variance (CV)* (see equation 2) of the population of agents, indicating how much the categories of all agents deviate from each other.

$$CV(population) = \frac{1}{2N(N-1)} \sum_{i=2}^{N} \sum_{j=i-1}^{N} CD(A_i, A_j) \quad (2)$$

As opposed to the imitation success and the number of categories, the category variance is not averaged over 100 games. In order to reduce computation time in the simulations, it was only calculated every 100 games.

5 Results

The results presented in this section were obtained from simulation and will be validated using the experimental set-up described in section 2, however the results presented here serve as a proof of concept.

In the simulation, all generated action trajectories are taken within the reachable area of the robot arm. New action categories were added with a small probability of 0.02. Action categories that were used more than 5 times but had a success-ratio below 0.7 were removed. Only categories with a success-ratio below 0.5 can be shifted in case of a failing imitation game. These parameter settings are the same as those used in the context of vowel systems in (de Boer, 2000).

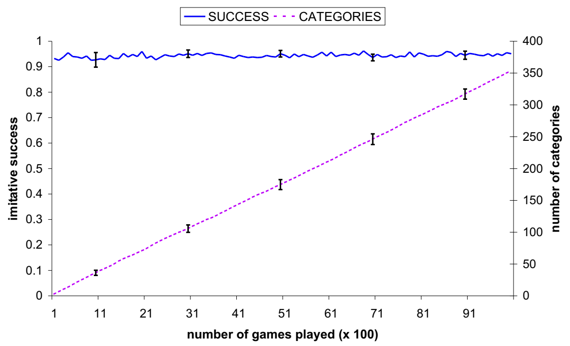

Figure 6: Imitative success and action categories developed over 10000 imitation games played by two agents.

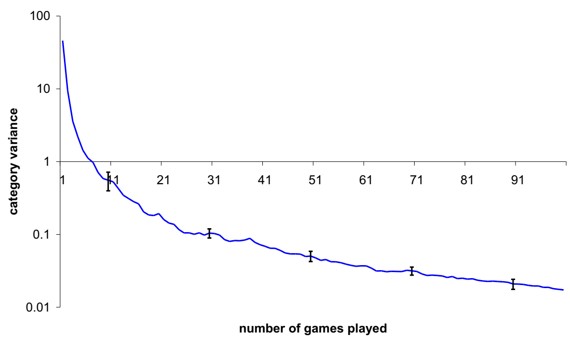

Figure 7: Category variance for the same 10000 imitation games played by two agents.

In figures 6 the imitative success and the number of categories for a population of two agents are shown. The experiments are averaged over 10 runs. In figure 7 the category variance is shown for the same 10000 imitation games. These results show that a shared repertoire of action categories can indeed emerge from a population of two agents.

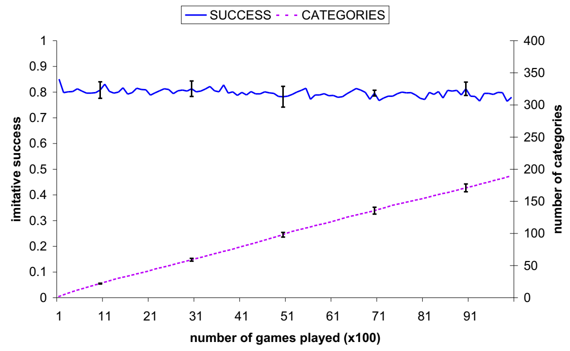

Figure 8: Imitative success and action categories developed over 10000 imitation games played by five agents.

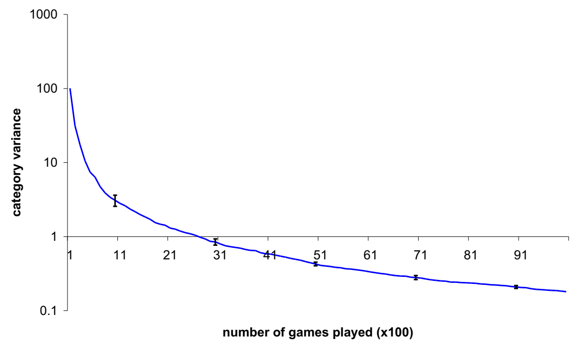

Figure 9: Category variance for the same 10000 imitation games played by five agents.

More interesting is that similar results can be obtained for larger populations, as can be seen in the figures 8 and 9 for a population consisting of five agents. These results show that agents are capable of transmitting action categories from one agent to another using imitation. We have shown that this is possible without the roles of initiator and imitator being predefined and without predefined action categories. All agents take turns in teaching and learning, which causes the action categories to spread into the entire population. Due to the pressure of adding random action categories and adapting those categories depending on the outcome of the game, the action categories can emerge while the agents are interacting. In this game it is assumed that the agents know the inverse kinematics of their robot arm and the relation between the observation space and action space. In the next section, it is investigated whether imitatable action categories could be learnt successfully without those two assumptions.

model	imitator
if $A \neq \varnothing$ $a \leftarrow$ random from A a.usage $\leftarrow$ a.usage + 1 **execute** a.production	
	observe $O1$ **if** $A = \varnothing$ $A \leftarrow$ findaction(O_1) **else** $a_{rec} \leftarrow$ action from A such that a_{rec}.observation closest to O_1 **execute** a_{rec}.production
observe O_2 $a'_{rec} \leftarrow$ actioncategory from A such that a'_{rec}.observation closest to O_2 **if** $a = a'_{rec}$ a.success $\leftarrow$ a.success + 1 **update-feedback**(a, a'_{rec}, success, A) **else** **update-feedback**(a, a'_{rec}, failure, A)	
do-other-updates() **else** $A \leftarrow A \cup$ random actioncategory	**do-other-updates**()

Figure 10: Overview of the game in which the initiator is learning.

```
do-other-updates(A)
∀ actioncategory ∈ A do
   if actioncategory.success / actioncategory.usage < *throwawaytreshold* and
      actioncategory.usage > *minuses*
       A ← A \ actioncategory
with probability *addprobability* do  A ← A ∪ newactioncategory
---
update-feedback(a, a_rec, signal, A)
a.usage ← a.usage + 1
if signal = positive
   a.success ← a.success + 1
   shiftcloser(a, a_rec)
else
   if  a.success / a.usage < *threshold*
      shiftcloser(a, a_rec)
---
shiftcloser(a, a_rec)
a.action = a.action + *shifttreshold* x [ a.action -  a_rec.action ]
a.observation = observe execute a.action
```

Figure 11: Update procedures for the game in which the initiator is learning.

6 A variation on the imitation game

In the game described in the previous section the imitator adapts its repertoire of action categories by shifting its action categories towards those of the initiators, see figures 4 and 5. This shifting is performed by calculating the action that resulted in the actual observation. The action associated with the action category the imitator used is then modified to resemble this calculated action more closely. Finally, a new observation is obtained for the shifted action. The shifted action and the shifted observation together constitute the shifted action category. This operation requires calibration between action and observation space (for obtaining the action corresponding to the observation) and requires inverse kinematics (for executing the shifted action).

In the modified imitation game we describe here, action categories are not shifted towards observations. A new shift operation is introduced, which shifts an action category towards another. This means that the shift could be performed in the action space directly, which eliminates the requirement of calibration between action and observation space. This works on condition that if an action *a* is closer to an action *b* than to an action *c*, the same relation holds for the associated observations. This is the case, as the action space and the observation space are related to each other by a rotation and a translation. Thus, by introducing a modified shift operation, the calibration is no longer required. The modified shift operation is not applicable to the case of the learning imitator in the game described above, as the imitator can never know the category of the action observed by the initiator.

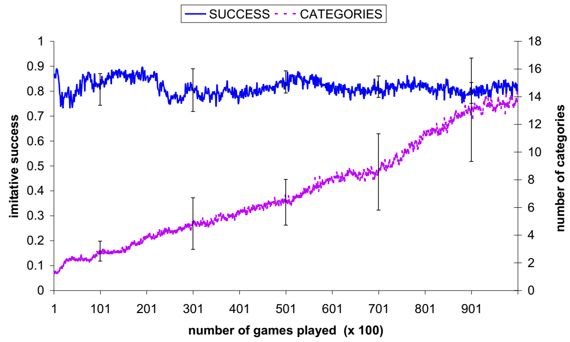

Figure 12: Imitative success and number of categories developed over 100000 imitation games in which the initiator learns.

Therefore, a modified game is introduced in which the initiator and not the imitator adapts its categories. As will be seen, this enables us to perform the shift operation directly in the action space. The pseudo-code for this game is given in figure 6. Again, two agents are randomly selected from a population of agents and start an imitation game. The initiator randomly selects an action category *a* from its repertoire and executes the associated action. The imitator observes this and categorizes its observation as a_{rec}. The imitator executes the action associated with this category. This is observed and categorized by the initiator as a'_{rec}. If the initial action category chosen by the initiator equals the category of the imitated action ($a = a'_{rec}$), the game succeeds, otherwise it fails. In this game, the initiator and not the imitator updates its repertoire of actions, so no feedback about the outcome of the game is required. In case of success, the initiator shifts the action category *a* towards the action category a'_{rec}. In case of failure, the update depends again on the success-ratio of the action category *a*. If *a* was not successful in the past, *a* will be shifted towards a'_{rec}. If *a* was successful, nothing is done. In the first game, a new action category is created in that case. This can not be done in this game, because the newly created category would be equal to a'_{rec}. This would confuse the agent.

Now one could ask whether it is possible to adapt the game even further such that no inverse kinematics is re-

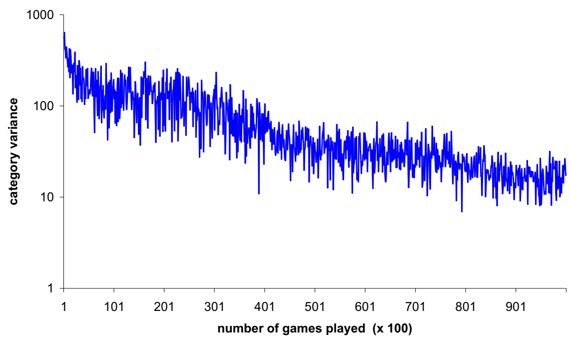

Figure 13: Category variance for the same 100000 imitation games.

quired. It is possible to modify the action space such that it contains direct motor commands instead of target positions. In that case, the shift operation must be performed on the motor space. However, if one motor command *a* is closer to a motor command *b* than to a motor command *c*, the same relation is not guaranteed to hold for the associated observations. So, it is not sure whether this would work. However, as can be seen in figures 12 and 13, two agents are actually capable of developing a shared repertoire of action categories playing imitation games. The initiator uses the imitation of its own action by the imitator to update its action categories to resemble more closely to those of the imitator. This game appears not to work for populations of more than two agents. It needs to be investigated how this game could be modified in order to overcome this.

7 Parallel and Future work

In parallel with the work on the experimental set-up, including the vision system, and the work on unsupervised incremental clustering of action categories, we are also studying batch unsupervised clustering of action categories. We used DTW as a distance metric in the observation space. Hierarchical agglomerative clustering (Everitt, 1993) was used to extract clusters in the agents' observation space. Both a stop criterion based on the t-test (Oates et al., 1999) and one based on the Hartman (Tibshirani et al., 2000) criterion were found to give fairly good results. This clustering is however not incremental and the categories developed by each individual agent do not resemble each other. This means that this method cannot be used directly and must be adapted before it can be useful in unsupervised action categorization.

Three important research issues should be addressed in the near future using the experimental set-up described in this paper.

1. What are the exact conditions required for imitation to emerge? As argued in this paper, we believe that agents can develop shared action categories through imitation without predefined action categories.
2. How can more complex actions be learned? At the moment, actions are movements from one coordinate to another. One of the most natural extensions is that actions are trajectories defined by a sequence of coordinates. This will lead to more complex observations, requiring advanced distance metrics and clustering methods.
3. In the long run, the most challenging research issue is the question on how actions that carry meaning could emerge. We believe this question can be answered if actions involving object manipulations could emerge.

8 Conclusion

In this paper we propose an experimental set-up, based on imitation, suited for conducting experiments on the exact conditions required for shared action categories to emerge in a population of real-world agents. We have shown that these categories can be shared in the population without a fixed teacher - student pattern of imitation where the teacher already has fully developed categories. Instead, the action categories can emerge and become shared through multiple imitation games in a population of agents where all agents can act as a teacher or student. The learnt actions can be observed, categorized and imitated by other agents, which is not guaranteed in set-ups where a teacher starts with built-in action categories and transfers those categories to the student(s).

Preliminary simulation results show that using imitation games, shared repertoires of action categories can be obtained. These repertoires are non-trivial, as they consist of multiple action categories. They are also very successful: for populations of only two agents imitation success is always in the 90% range even though noise is present. In larger populations the imitation success is lower, but still better than random repertoires would achieve. Imitation games in this setup are based on the imitator of the imitation game updating its action categories. In this case calibration, inverse kinematics and non-verbal feedback are required.

If the population is restricted to only two agents, action categories can emerge and be shared without the action space and the observation space are calibrated (1), without feedback between the agents about the outcome of the game (2) and without the agents have built-in notion of the inverse kinematics of their manipulator (3).

Acknowledgments

Bart Jansen is sponsored by a grant from the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT). Tony Belpaeme is a postdoctoral fellow of the Flemish Fund for Scientific Research (FWO - Vlaanderen).

References

Alissandrakis, A., Nehaniv, C. L., and Dautenhahn, K. (2002). Do as I do: Correspondences across different robotic embodiments. In Polani, D., Kim, J., and Martinetz, T., editors, *Proceedings of the Fifth German Workshop on Artificial Life (GWAL5)*, pages 143–152. 18-20 March 2002, Lübeck, Germany.

Belpaeme, T. (2002). *Factors influencing the origins of colour categories*. PhD thesis, Artificial Intelligence Lab, Vrije Universiteit Brussel.

Billard, A. and Hayes, G. (1997). Transmitting communication skills through imitation in autonomous robots. In Birk, A. and Demiris, Y., editors, *Proceedings of EWLR97, Sixth European Workshop on Learning Robots*, *Lecture Notes on Artificial Intelligence*, volume 1545, pages 79–95. Springer. Brighton, UK, July 1997.

Corradini, A. (2001). Dynamic time warping for off-line recognition of a small gesture vocabulary. In *IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, pages 82 – 90. IEEE Computer Society. Vancouver, B.C., Canada, July 13 - August 13, 2001.

de Boer, B. (2000). Self organization in vowel systems. *Journal of Phonetics*, 28:441–465.

De Vylder, B. (2002). Forward and inverse kinematics of the teach-robot. Technical Report AI MEMO 02-02, Artificial Intelligence Lab, Vrije Universiteit Brussel, Brussels.

Everitt, B. (1993). *Cluster Analysis*. Edward Arnold, London.

Myers, C. S. and Rabiner, L. R. (1981). A comparative study of several dynamic time-warping algorithms for connected word recognition. *The Bell System Technical Journal*, 60:1389–1409.

Oates, T., Firoiu, L., and Cohen, P. (1999). Clustering time series with hidden markov models and dynamic time warping. In *Proceedings of the IJCAI-99 Workshop on Neural, Symbolic and Reinforcement Learning Methods for Sequence Learning*, pages 17–21.

Steels, L. and Kaplan, F. (1999). Bootstrapping grounded word semantics. In Briscoe, T., editor, *Linguistic evolution through language acquisition: formal and computational models*, pages 53–73. Cambridge University Press, Cambridge, UK.

Tibshirani, R., Walther, G., and Hastie, T. (2000). Estimating the number of clusters in a dataset via the gap statistic. Technical Report 208, Dept. of Statistics, Stanford University.

Vogt, P. (2000). Grounding language about actions: Mobile robots playing follow me games. In Meyer, Bertholz, Floreano, Roitblat, and Wilson, editors, *SAB2000 Proceedings Supplement Book*. International Society for Adaptive Behavior.