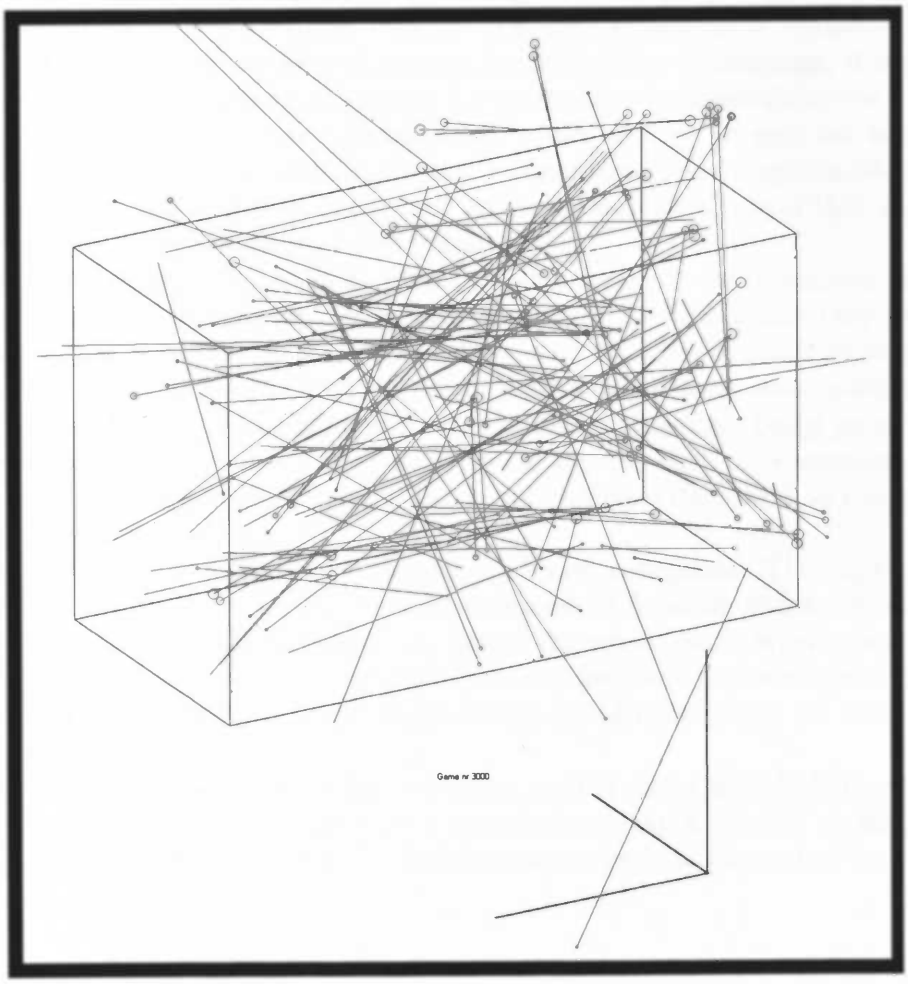
erging shared action categories in physical robotic agents through imitation

T. ten Thij (s0831611)
Artificial Intelligence
University of Groningen

September 20, 2004



Supervisors:

dr. Tony Belpaeme, Vrije Universiteit Brussel
dr. Bart de Boer, University of Groningen
drs. Gert Kootstra, University of Groningen

RuG

VRIJE UNIVERSITEIT BRUSSEL
SCIENTIA VINCERE TENEBRAS

Abstract

Artificial intelligence is concerned with studying and recreating human intelligence. Humans distinguish themselves from other animals and primates through their use of a rich and expressive communication system: natural language. The enterprise of studying human intelligence can not be complete without studying human natural language. When one is confronted with the efforts of linguists over the last decades to capture our knowledge of language, it becomes clear that this is not as easy as it may have seemed. Linguists have been searching for a set of rules that determine whether a sentence is grammatical or not. As yet no such set has been found that is capable of describing even one human natural language. Some linguists claim that the fact that such a ruleset has not been found by intelligent researches proves that children are born with a 'language module' (aka universal grammar).

Luc Steels' Artificial Intelligence Lab in Brussel has been trying to use computer models to investigate what aspects of language can be explained without assuming that they have to be innate. They try to show that language is a self-organizing complex dynamic system that emerges from human cultural interactions. The human brain must possess some mechanisms that other animals do not, but these does not have to be specific for language. Using populations of interacting agents, it has already been shown that some aspects that were assumed to be innate can be explained by cultural transmission (eg vowel systems (Bart de Boer) and color categories (Tony Belpaeme)).

My research focussed on the cultural transmission of action categories. The experiments have been performed on a robotic arm that uses four degrees of freedom and a stereoscopic camera. Using the arm and camera as embodiment, agents engage in games where they try to imitate the action they saw the other agent perform. The only feedback between agents during the experiment is given by the initiator of the game as one bit of information on whether he judges the game's outcome as successful.

In my experiment actions are not coupled to meaning, so it is not a model of the cultural learning of real gestures. But it turns out that a population using a scheme as simple as described is able to develop a shared repertoire of action categories in a very robust manner.

Contents

Chapter 0

Foreword

Before I decided what I wanted the subject of my graduation thesis to be, I have done some reflections about which topic in Artificial Intelligence I most wanted to contribute to. One of the events that helped me realize that language as a self-organizing system was 'it' was an excursion to the AI-Lab in Brussels organized by CoVer, the association of students of Artificial Intelligence in Groningen. Here we attended an interesting series of lectures and recieved a copy of a not officially published book about the Talking Heads. Reading this made me realize how good this experiment's approach to the origins and workings of language is and how it is possible to have an alternative view to the linguistic approaches I had been taught.

This made me want to do my research and thesis at the AI-Lab. I am very grateful that I have been given the opportunity to contribute to the lab's ongoing research in which I am so interested and for the fact that my results have been presented in an accepted paper of which I have been made second co-author.

Gert Kootstra I would like to thank for being a nice temporary supervisor and for a nice game of coin-soccer. Many thanks go out to all the people at the AI-Lab. Not only have they helped provide an even more stimulating environment than I had hoped for, but they also helped my time in Brussel be a lot of fun. Bart de Boer's 'klimaatontkenning' has saved us from many hours in thick smoke. Tony Belpaeme earns my respect not only for posessing a fluent English accent hard to live up to, but also for coming all the way to Groningen with me for my presentation. I owe Bart Jansen many thanks: not only has his Masters Thesis been the major guide for my thesis, but he always managed to stay cheerful every time I bothered him with questions and has saved me a lot of cubersome work on hardware because he had already done it. I had never before met such a kind mathematical genius as Bart de Vylder, who has also helped to make my dealings with the arm more pleasant and never hesitated to help me in any way he could. A special thank you goes out to him and Barbara for inviting me to their wedding which I very much enjoyed. Joris van Looveren was a Talking Heads hero but has now also become a friend after showing me around Brussels a bit: if possible I would still like to take you to Kill Bill 2. Joachim de Beule seems bloody good at everything he does, which is examplified by his role as a lisp oracle for me. If it had not for Peter Stuer I might still be looking for a segmentation fault. Dominique Osier has left an impression of coolness besides being the best 'administrative person' any institution could wish for. I would willingly sell my soul to be able to continue my own scientific career with a fraction of the passion that Professor Luc Steels brings to light. My thanks go out to him not only for providing such a stimulating environment to do my thesis work in, but also for giving me the opportunity to

attend the IK2004 in Günne which definitely is in the top-10 of most pleasant experiences in my life. 'Den' Joris Bleys was so kind to invite me to the garden party at his parent's house and has been a major cheer up factor at the lab. 'Den' Ruben Havermaet has been the other major cheer up factor, has provided a photo for Bart's present in Groningen and has allowed me to see the Stevie Ray Vaughan DVD I rented. Sandra de Jongh earns bonus points for cooking for our stressed lot even when we were not very sociable at that moment and bonus bonus point for an unexpected cup of chocolate mousse.

Over the course of ten years of study many friends helped to make it take so long and enjoyable. I can not hope to mention all of them but I would like to especially thank Wiebe Baron for moving my stuff, letting me crash at his place and for just being a friend.

I also would like to thank Niels Taatgen, Petra Hendriks and Tjeerd Andringa for keeping the Cognitive Science and Engineering curriculum running and for giving the student fraction of the 'Educational Board' such a hard time because everything was already arranged as good as humanly possible.

But I owe most to my father Sjaak ten Thij and aunt Ria ten Thij, because without their emotional and financial support I would not have been able to pursue the dream that is now coming true for me.

Words printed with a circle° as superscript can be found as an entry in the glossary.

Chapter 1

Introduction

1.1 Aim

The research at the AI-Lab focusses on developing artificial natural language in embodied agents. This is a daunting task and is tackled by concentrating on different aspects of language. The central research question of the AI-Lab is: *What are the minimal requirements for a language to evolve in a population of agents?*

The research question of my project is a subquestion of this and can (conform Jansen et al. [2003]) be stated as: *Is it possible to make shared action categories emerge in robotic agents through imitation*? Experiments in which the robot were simulated had already been performed and had shown that this is possible (as reported in Jansen [2003]). In order to be able to make the claim that this can be done with a real robot arm, the experiments done in simulation had to be implemented in a physical system. The goal of my project was to perform this implementation and obtain similar results using a physical robot arm and stereo vision system.

1.2 Reading this thesis

Chapter 2 puts the research I have performed into a scientific context. I explain how the experiments work in chapter 3 and the issues when performing this on a physical setup in chapter 4. In chapter 5 the measures that indicate the performance are presented and discussed. Results are presented in chapter 6 and discussed in the final chapter: 7.

Appendices contain a sensitivity study, tips on viewing figure 4.6, a list of things that I have done that do not fit elsewhere, learned lessons and a glossary.

Chapter 2

Background

Artificial intelligence is concerned with studying and recreating intelligent behavior in general human intelligence specifically. AI researchers want to develop precise explicit models of the structures and processes that give rise to intelligent behavior, and apply their insights to the construction of artifacts that are useful to people.

What makes Artificial Intelligence unique, compared to other sciences that also study intelligence like psychology or neurophysiology, is the strong emphasis on the construction of artificial systems as a way to test theories.

2.1 Two approaches

The fascinating mystery of what intelligence is, and how it comes about in physical systems, remains far from being resolved. It is an intriguing phenomonon with many facets, only partially understood. It is therefore not surprising that there are many ways to approach the phenomenon, even if one accepts the methodology of AI. Two major approaches stand out: the symbolic approach and the dynamics approach.

The *symbolic approach* makes the assumption that symbols are an atomic component of thought and that it is possible to represent the world in a symbolic system. Cognition is regarded as the manipulation of symbols. This approach goes a long way and has been applied succesfully in practical applications such as expert systems. But it is quite hard to use this approach in an embodied architecture because it remains unclear how perception of the real world is to be mapped to symbols and how the result of the manipulations can be actions in the world.

The *dynamical approach* uses a bottom up approach to intelligent behavior. It views agents as complex dynamical systems which are in continuous interaction with the dynamical processes in the real world. The dynamics paradigm directly addresses the question how a (physical) agent could operate in real time in a dynamically changing environment. Amongst the many disciplines that have contributed to this approach are: behavior-oriented architectures, neural networks, genetic algorithms and self-organizing systems. This approach is still relatively young and therefore not as advanced as the symbolic approach. One of its disadvantage is that it is very hard to transfer world-knowledge to a dynamical system.

One of the important insights of Luc Steels is that language can be seen as the bridge between the symbolical and dynamical paradigms. In order to be able to communicate, there has to be some form of symbolic representation of the world. But this representation must also be grounded at a perceptual level.

2.2 Language is ...

The hypothesis at the AI-Lab about language is that it is **an adaptive self-organizing open system emerging from human cultural interaction** [Steels, 1999, 2000; de Boer, 2001; Steels and Belpaeme, 2004].

The components of this hypothesis are further explained in this section.

2.2.1 Human

The human species is - as far as we know - the only one that is capable of transferring information through compositional language. It is yet undetermined what it is exactly that humans possess that other animals do not. Two important aspects that seem to be required are the ability to manipulate (and 'stack') symbols [Deacon, 1997] and imitation.

2.2.2 Language

Language can be defined as "a systematic means of communicating by the use of sounds or conventional symbols" [Wordnet].

The principle of innateness is based on the remarkable ability of human children to acquire or "invent" language. If it can be shown that they can exhibit some skill by other means (like cultural interaction), this alleviates the need for an innate grammar or language instinct for it.

2.2.3 Adaptive

Language is often seen by linguists as something that is relatively "fixed" in the sense that it is possible to study "the english language". And in some way it is, but it exhibits far more plasticity and flexibility than that. There is no one language, but every user has his own version of it. "The language" is what happens when users interact with their respective version of it. It is quite natural for a part of the population of English speakers to use a slightly different grammar or lexicon. This can not be regarded as wrong, it is just the nature of language. This is also the reason why languages change slightly over time. This change of language is less likely to happen in these times of mass media, but there is also a counterexample. A lot of young people use sms or internet chat to communicate. Because typing is awkward or there is only room for a limited number of characters, a new language is being created with it's own grammar and spelling and shorthands for some of the more common expressions. Some examples of this are: [*FTBOMH*: *from the bottom of my heart*] and [*GOOML*: *Get out of my life*].

2.2.4 Open system

A system is a group of independent but interrelated elements comprising a unified whole [Wordnet]. A closed system can be described completely within itself, meaning that there is no interaction with elements without the system influencing either the system or it's outside world. For an open system such interactions are assumed to take place.

It is a common and good practice for scientists that are trying to understand and model something that is not yet totally understood to break the subject under study up into more simple systems that are worth being studied in their own right. To keep things tractable it is also assumed that the simpler systems are closed, meaning that they do not interact with other

systems. Often this is a necessary simplification but increasingly more often it turns out that interactions with the 'outside world' of systems that are assumed to be closed are relevant to explain desired phenomena.

The agents in language games are intentionally designed as an open system. This means that they must be able to adopt the 'language' of any other agent in the population. The number or forms of words should not be predefined. All agents should start out with no knowledge about the language they should learn.

2.2.5 Culture

Language can be regarded as a meme-complex. The term meme has first been coined in [Dawkins, 1976], meaning a unit of knowledge that can be transmitted culturally. The cultural spreading of memes is analogous to 'biological evolution'. All human brains have the capability to 'inherit' a certain meme, while every indivual human performs the 'natural selection' by choosing (intentionally or not) which memes to pass on to other people, as well as the 'mutation' because not every meme can be remembered perfectly. Memes are also Lamarckian where genetic evolution is not, meaning that a meme can change it's 'genetic' make-up during it's lifetime, which is then spread further in stead of the original one.

2.2.6 Self-organization, interaction and emergence

In the way these terms are used in the context presented here, they are very much inter-related.

A system is self-organizing if the number of states that the system takes on is significantly less than the number of all possible states for the system and if this is caused by the system itself and not by external influences. Such a system has the ability to increase it's order. Usually in nature, there is a tendency for chaos to increase. This can be illustrated with an aquarium filled with white balls on one side and black ones on the other. If you then start shaking it, the balls will mix to the effect that for every position the chances to find a black ball or white ball there will become the same. The state where the balls with the same color are on one side is so unlikely that it will never naturally occur. The claim that language is self-organizing can be made because it is not fully determined by external or internal (innate) factors. There is a multitude of possible languages and grammars, of which usually only one becomes stable in an isolated population.

A classic and illustrative example of self-organisation is the honeycomb: a honeycomb is an emergent property of a population of bees. A single isolated bee will not produce cells in a hexagonal pattern but it results from the local interactions between individual bees working near each other.

If a system has a property that is not a property of any of it's elements, this property is said to be emergent.

The term 'emergent' can be quite slippery if one is not careful about it [Brunner, 2002]. Originally (in the 1920s) emergentism was a philosophical movement that stated that when you put simple things together, it is possible that the resulting system exhibits properties that can not be explained from the properties of the simple things. This meaning was used to defend a philosophical position in which one is unwilling to accept either reductionism (everything that happens in our universe can ultimately be explained from the fundamental laws of nature) or vitalism (life is a vital principle distinct from physics and chemistry). This position is called 'weak emergentism' (additional assumptions are required for the strong version), but it already

makes the rather strong claim that it is *impossible* to infer the emergent property from the simple entities because otherwise the term would serve no purpose in the philosophical debate.

Few scientists still hold the position of vitalism, but there are still some reductionists left (of which I am one). I believe that although we still do not understand all the intricate workings of nature sufficiently to explain everything from it, in principle it could be possible. If one holds this view, the original meaning of emergentism can no longer be used.

The meaning we use for the term emergent is: 'resulting from self-organization' or more specifically for our case: 'resulting from the interaction of agents'.

2.3 Computer models

If one makes claims about aspects of how language works, it is desired to have a way to test those claims. Some can be tested by looking at actual human language, but not all. If one claims for instance that the cultural transmission of vowel sounds is sufficient to explain universal tendencies in human languages, there is no way to test this claim in the real world. In such a case it is an option to try to build a model that can support this claim. It is important that such a model is as simple as possible to be able to see exactly the processes that are going on and to find what is minimally required for the effect you are looking at to occur. An important aspect of using computer models is that every assumption of your model must be made explicit, because otherwise it is not possible to implement the theory as a computer model. There can be no shortcuts or aspects left vague. Computer models are not yet capable of capturing the complexity of human language and if something can be shown in a model, this does not imply that it works in the same way in humans. But if universal tendencies of vowel systems can be explained with a computer model that uses cultural interaction and no innateness of vowel categories, it can be assumed that innateness of them is also not required for humans.

2.4 Language games and Talking Heads

The term 'language game' has been inspired by the work of Wittgenstein [Wittgenstein, 1953] but is based only loosely on it. The term as it is used here can be defined as the interaction that (possibly robotic) agents engage in in a specific environment that models some aspect of the learning of language Steels [2000]. Two important aspects of language games as defined by Steels is that there are no predefined roles for the agent (like teacher and student) and that transfer of knowledge is horizontal (between peers) and not (or at least not only) vertical (from parents to children). Some similar approaches to language that are more focussed on how language can have evolved include [Christiansen and Kirby, 2003; Nowak et al., 2001; Oliphant and Batali, 1997; Zuidema, 2004]. The line of research involving language games originates from around 1995 when Steels was working on ecosystems of behavioral based robots and realized the importance of the 'artificial-life' approach to language. Work in this line resulted in experiments on lexical grounding on robots [Steels and Vogt, 1997; Vogt, 2000] and a first prototype of the Talking Heads. In the Talking Heads experiments agents could internet-hop between several sites across the world that had a similar setup. The setup consisted of a white-board with geometrical shapes of different shapes, colors and sizes at which two pan-tilt cameras were directed. Agents could use the camera as a body to play language games with. This implemented a global population of agents that all started out with no prior categorization of the world and without any predefined utterances. It has been shown that a stable categorization

and lexicon was able to emerge in the population using only language games. Unfortunately there has never been a good publication on the results of the experiment. There is however an excellent reference in the form of an unpublished book which explains the experiment into detail and that can be obtained by direct request to the AI-Lab in Brussels [Steels, 1999].

2.5 Emerging vowel systems through imitation

A similar approach using the 'imitation game' has been used to address the issue of universal tendencies of human vowel systems [de Boer, 2000a]. One of the most astonishing accomplishments of this work is the fact that human-like vowel systems have been shown to be able to emerge with no predetermined partition of the acoustic space and only constrained by human-like limitations in utterance and perception capabilities.

The experiment described in this thesis is basically De Boer's work adapted for emerging 'movement categories' using a robot. A movement category in this work is analogous to a vowel. The arm and camera are the equivalent instruments for utterance and perception.

Since the imitation game is the motor behind my experiments as well, chapter 3 is devoted to explaining how it works.

2.6 Imitation

When I joined the research effort of the AI-Lab, one of the subjects that was being worked on was robot-imitation, which has been a 'hot topic' since about 1990. The Lab's view deviates from most other work in the field however. Our goal is to have a repertoire of shared action categories emerge in population of robotic agents through imitation. The nature of this approach causes it to differ from others on robot imitation:

1. We let robots imitating each other in stead of letting robots imitate humans. The robot-human imitation projects are interesting in their own right because this can be a relatively easy way to teach a robot the complex movements required for human-like movement or learn them to perform a task that humans can perform.

2. There is no pre-defined teacher or student role for the robots. Every game, every agent has an equal chance to be either the initiator° or the imitator°.

At the start of my project two papers had just been published by four researchers at the AI-Lab. One which discussed issues surrounding imitation in populations and the use of robots and simulations for investigating them [Belpaeme et al., 2003]. The other that reported on the successfulness of the intended experiment in simulation [Jansen et al., 2003].

2.7 Embodiment

Embodiment is a paradigm in Artificial Intelligence that originated in the 1980s. It is based on the idea that in order to implement an artificial intelligence, it must be able to sense and act upon it's environment by means of a (physical) body. Because it is hard to give an exact definition of embodied AI, some recent examples of other work in this field might serve to illustrate it. Barbara Webb and others have built a robot that performs phonotaxis using a

biologically plausible neural network [Webb, 2004; Webb and Consi, 2000]. Dario Floreano and others have used an evolutionary design approach to learning robots to the effect that the resulting robots can optimally perform a defined task, also making use of non-linear properties that could not have been used had they been engineered [Floreano et al., 2004; Nolfi and Floreano, 2001]. What the projects using this approach have in common is that a lot of extra attention has to be spent on the design of a function body (usually robot) and that design methods that currently are conventional can not be used to obtain similar results.

2.8 Ultimate goal

The ultimate goal of the research such as carried out at the AI-Lab would be to build a robotic agent that has all the linguistic capabilities of a two year old child. There is still a long way to go to reach this point. There are other projects that use a developmental approach to robots like BabyBot [Metta, 2000] and Lucy the robot [Grand, 2004]. Other very complex but promising research at the lab focusses on the development of grammar in agents engaging in language games [Steels et al., 2004].

2.9 My mission

When I started my research I was given a clear mission: 'validate the results obtained in simulation using a robotic experimental set-up'. All the hardware was provided and was to varying degrees of success already interfaced with a pc on which the experiment was to be performed. From there I ventured to 'make it work'.

More information about the experiment is found in chapter 3, the robotic setup is covered in chapter 4.

Chapter 3

Imitation game

The imitation game is the core of the experiment described in this thesis. It was first introduced in the context of language for the emerging of shared vowel systems [de Boer, 2000b]. My experiments rely heavily on the techniques used in his work.

In this chapter I explain all aspects of the imitation game and how it is used in my experiments. Some of the information provided here might be understood better after reading chapter 4 about the physical setup as well. Although I have attempted to seperate that and this chapter, some concepts are rather intertwined.

3.1 Agent architecture

In computer simulations used we make use of a population of agents that can interact, all with the same architecture. It's architecture defines what an agent 'knows'.

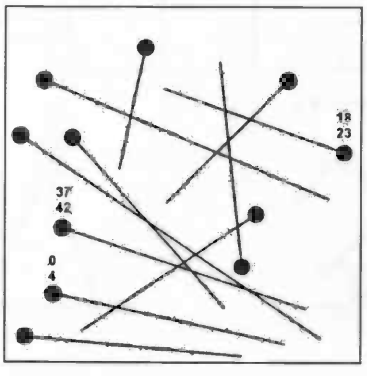


Figure 3.1: An agent's mind

Figure 3.1 gives a schematic representation of the 'mind' of an agent. Every category has a start-point (represented by a circle) and an end-point.

Every agent a has a unique identity number and a (possibly empty) set of categories called it's inventory ($\mathcal{I}_a$).

The information contained in an action category c is a unique id $c.id$, two points representing the start and end of a movement, a counter $c.use$ of how many times this category has been used and a counter $c.success$ of how many times it has been used in an imitation game that was successful.

In the current experiment all agents in the population are created at the beginning of the experiment with an empty inventory. During the experiment no agents are added to or removed from the population.

3.2 General overview of the imitation game

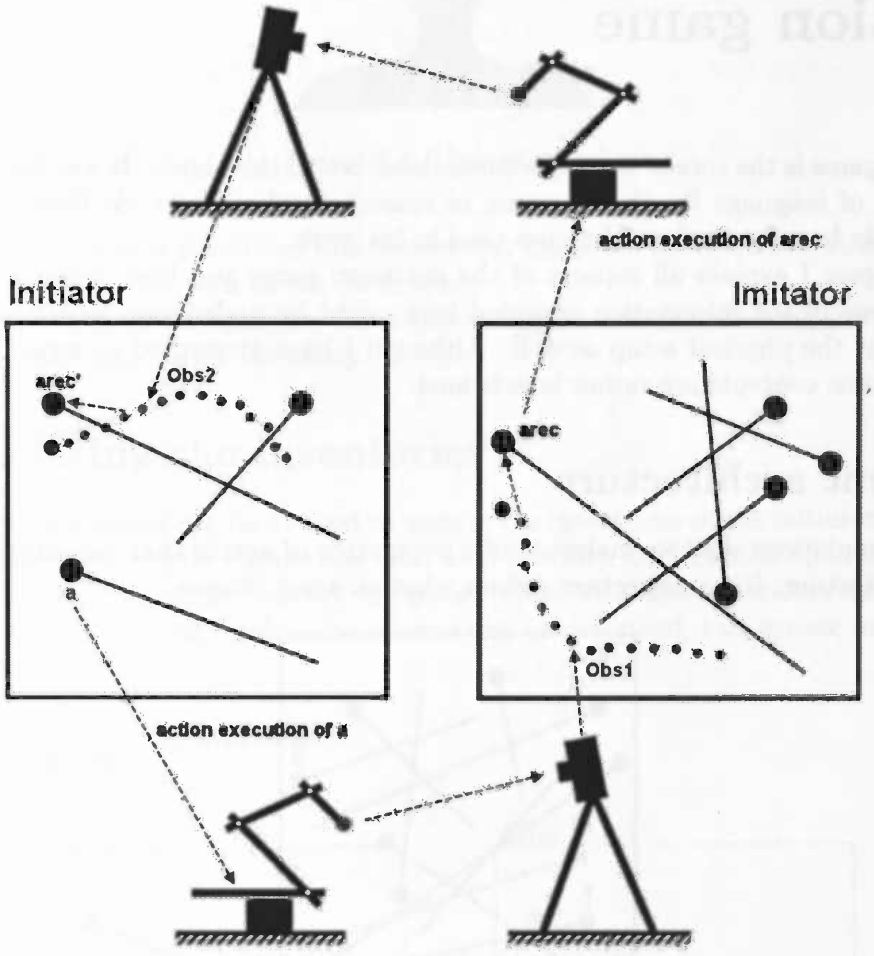


Figure 3.2: Schematic representation of the imitation game

First two agents are randomly chosen from the population: one as initiator and the other as imitator. Every agent has a set of categories which is called it's inventory. Assuming neither agent's inventory is empty, the initiator chooses one category at random from his inventory. This category is called a in figure 3.2. The initiator then computes how he has to perform the action associated with this category on the robotic arm and executes it. While the initiator is performing this action, the imitator is observing the movement of the arm. This observation results in a series of positions where the gripper was found during the movement. This series is called $obs1$ The imitator then looks in its set of categories for the category that best matches $obs1$. The found category is labeled $arec$. This category is then performed by the imitator in

much the same way as the initiator performed a while now the initiator is the one who observes the arm, resulting in an observation $obs2$. The best match for $obs2$ is the category $arec'$. At this point, the initiator decides the outcome of the imitation game. If $a = arec'$, the game is successful, otherwise a failure. The initiator signals the game-result to the imitator. Both the initiator and the imitator update their set of categories based on this outcome. Note that in the game as defined here there is no telepathy (ie, the agents can not inspect eachother's 'mind') and only one bit of information is transfered.

In the example depicted in figure 3.2 the game fails because a and $arec'$ are not equal.

I have had to make some simplifications to the simulations reported in [Jansen, 2003]. In order to start simple, no complex movements with points between begin and end point are allowed. Implementing this would also increase execution time on the arm. I have removed the storage of observed trajectories because it was not trivial enough to shift and merge observations with a different number of observations (shifting and merging are explained in section 3.4). An alternative way to obtain shifted observations is to execute the shifted action and then associate it with the observed trajectory. This would cost a lot of extra time. Since the kinematics of the arm are known (see section 4.1), the action could be simulated, but then it is just as valid to map an observation of the other agent to the corresponding arm position, which is the approach I have taken. Hopefully both these simplifications can be undone in future research.

For more details about how actions are executed and observed refer to section 4.

3.3 Matching an observation

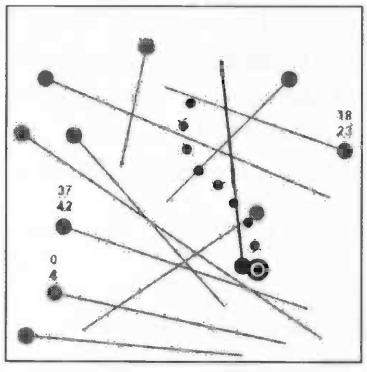


Figure 3.3: Matching an observation

When an agent has observed a movement of the arm, it takes the begin and end point of the trajectory and searches the category that is the closest match to these. This process is illustrated in figure 3.3.

The process where perception results in a one category being selected models human categorial perception. There it is also the case that one interpretation wins over all others. This is examplified by the McGurk effect [McGurk and MacDonald, 1976]. When presented with the sound 'BA' together with a movie of the lip movements corresponding to 'GA', most adults (98%) think they are hearing 'DA' - a so called 'fused respons' - where the 'D' is a result of an audio-visual illusion. This illustrates how only one interpretation (or 'category') can eventually be perceived.

Figure 3.4: Vase of faces?

Another example that illustrates this is shown in figure 3.4 in which the observer can see a vase or two faces, but not both at the same time.

In the same fashion an agent in the imitation game will perceive any observation as a category that is already in it's inventory.

3.4 Updating the inventories

Several updating schemes can be devised to update the inventories of the initiator and imitator. For all experiments the same scheme has been used in which the imitator updates it's category inventory depending on the outcome of the game. Then both agents perform updates that are independent of the outcome. Before the scheme can be described, the update mechanisms are explained first.

3.4.1 Merging

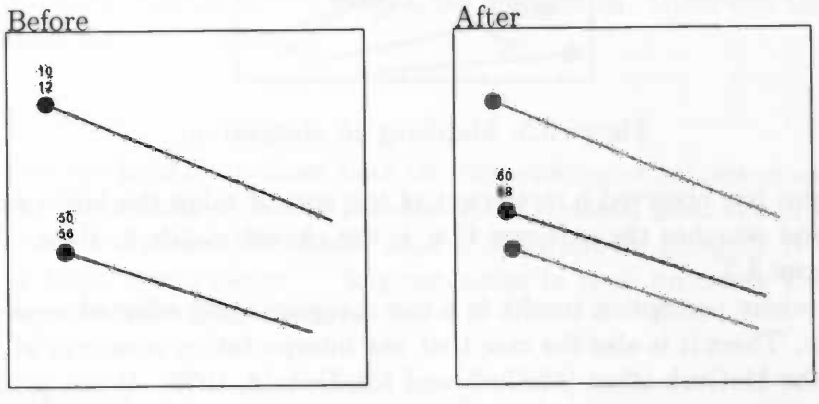


Figure 3.5: Merging

Two categories can be merged by replacing them with a new one that has the sum of the success and usage of both and a movement that is the average of the original movements. This

procedure is illustrated by figure 3.5. Merging is required to prevent categories that are very close together from interfering with each other. The distance d between two categories a and b is defined as $|\vec{a}_{start} - \vec{b}_{start}|^2 + |\vec{a}_{end} - \vec{b}_{end}|^2$ where $|\vec{x}| \stackrel{\text{def}}{=} {x_1}^2 + {x_2}^2 + ... + {x_n}^2$. When an agent performs merging on it's inventory, the distance between all combinations of two categories is computed. If a distance has a value lower than the *merging-threshold* parameter the categories are merged and the merging procedure is repeated for all combinations of categories on the new inventory. The default value of the *merging-threshold* parameter is 200 which given the distance measure corresponds to a distance in real space of approximately 10cm. The effect of varying the *merging-threshold* is treated in the merging section of the results chapter (6).

3.4.2 Shifting

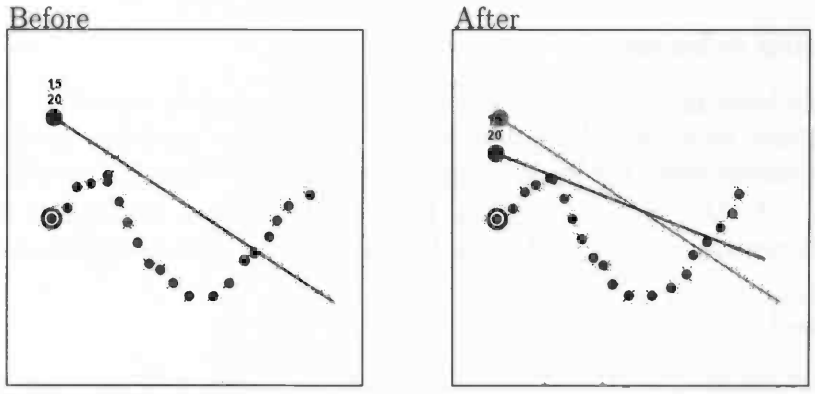


Figure 3.6: Shifting

A category can be shifted closer to an observation by bringing it's begin and end point closer to the observed points. This is done on two occasions:

1. The game was successful so the used category is assumed to correspond with the observation and is shifted closer to it.

2. The game failed and the category used in the game by the agent was less successful than the *successfulness-threshold* parameter, so this category is shifted in this case also. The default value for *successfulness-threshold* is 0.5.

The degree to which a category is shifted towards the observation is controlled by the *shift-factor* parameter which can take on any value between 0 (resulting in no shifting) and 1 (resulting in completely copying the observation). The default value for *shiftfactor* is 0.03.

3.4.3 Adoption

If the game failed and the category used by the agent was more successful than *successfulness-threshold*, it is assumed that the category codes for a different movement than the one that was just observed and should therefore be left untouched. In this case the agent will add a new category to it's inventory that exactly resembles the observation he has just done. The usage and success of the new category are both set to 0.

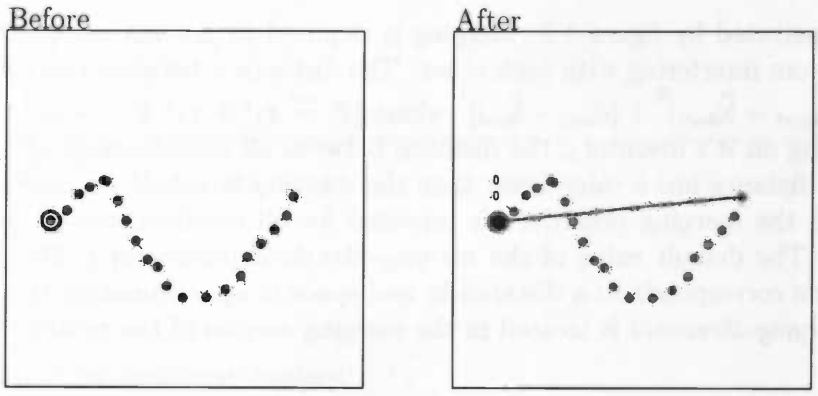


Figure 3.7: Copying from observation

3.4.4 Update scheme

After a game has been played the imitator updates it's inventory based on the outcome and both agents perform updates independent of it. This section explains the possible ways the inventories are updated using the mechanisms just described. Where aspects have already been explained along with the used mechanism, they are mentioned here again but only briefly. Mentioned mechanisms are referenced to by their section number in superscript.

Game succeeded

If the game was successful, the imitator assumes that the category it used was indeed the same as the one the initiator expressed. Therefore he shifts[3.4.2] the category toward the trajectory he has observed.

Game failed

If the game was unsuccessful the imitator has two options to update it's inventory depending on the previous success of the used category. If it has a success rate that exceeds the *successfulness-threshold* a new category is adopted[3.4.3] based on the observation. Otherwise the used category is shifted[3.4.2] towards the observation.

Other updates

First categories that are used more times than the *minimum-uses* parameter and that are less successful than the *throw-away-threshold* parameter are removed from the inventories. Then both agents have an *add-probability* chance that a new random category is added to their inventory. Finally both agents merge[3.4.1] any categories in their inventory that are within a distance of *merging-threshold* of each other.

3.4.5 Pseudocode

Table 3.1 and 3.2 provide pseudo code for the imitation game just described. They are a slightly adapted from the pseudocode in [Jansen et al., 2004b].

initiator	imitator
if $\mathcal{I}_{in} = \emptyset$ **new-category**($\mathcal{I}_{in}$) $a \leftarrow$ random from $\mathcal{I}_{in}$ $a.use \leftarrow a.use + 1$ **execute** $a.action$	

	observe Obs_1
	if $\mathcal{I}_{im} = \emptyset$
	  new-category($\mathcal{I}_{im}$)
	$a_{rec} \leftarrow$ category from $\mathcal{I}_{im}$ so $a_{rec}.action$ closest
	  to **find-action**(Obs_1)
	execute $a_{rec}.action$.

observe Obs_2
$a'_{rec} \leftarrow$ category from $\mathcal{I}_{in}$ so $a'_{rec}.action$ closest
  to **find-action**(Obs_2)
if $a = a'_{rec}$
  $a.success \leftarrow a.success + 1$
  send feedback "success"
else
  send feedback "failure"	

	update($a_{rec}, Obs_1, feedback, \mathcal{I}_{im}$)
do-other-updates()	**do-other-updates()**

Table 3.1: Pseudo code of the imitation game

update$(c, Obs, feedback, \mathcal{I})$
$c.use \leftarrow c.use + 1$
if feedback = "success"
  $c.success \leftarrow c.success + 1$
  shift-closer(c, Obs)
else
  if $c.success/c.use >$ *successfulness-threshold*
    $\mathcal{I} \leftarrow \mathcal{I} \cup$ **construct-category**(Obs)
  else
    shift-closer(c, Obs)

shift-closer(c, Obs)
$c.action \leftarrow c.action +$ *shift-factor* .(**find-action**$(Obs) - c.action$)

find-action(Obs)
find action that produces Obs using inverse kinematics

new-category$(\mathcal{I})$
$c.action \leftarrow$ random $((x_1, y_1, z_1), (x_2, y_2, z_2))$
$c.use \leftarrow 0$
$c.success \leftarrow 0$
$\mathcal{I} \leftarrow \mathcal{I} \cup c$

construct-category(Obs)
$c.action \leftarrow$ **find-action**(Obs)
$c.use \leftarrow 0$
$c.success \leftarrow 0$

do-other-updates$(\mathcal{I})$
$\forall c \in \mathcal{I}$ **do**
  if $c.success/c.use <$ *throw-away-threshold* **and**
      $c.use >$ *minimum-uses*
    $\mathcal{I} \leftarrow \mathcal{I} \setminus c$
with probability *add-probability* **do new-category**$(\mathcal{I})$
merge-categories$(\mathcal{I})$

merge-categories$(\mathcal{I})$
for all $c_1, c_2 \in \mathcal{I}$, $c_1 \neq c_2$
if $\|c_1 - c_2\| <$ *merge-threshold*
  $c_{new} \leftarrow (c_1 + c_2)/2$
  $c_{new}.success \leftarrow c_1.success + c_2.success$
  $c_{new}.use \leftarrow c_1.use + c_2.use$
  $\mathcal{I} \leftarrow \mathcal{I} \setminus c_1 \setminus c_2$
  $\mathcal{I} \leftarrow \mathcal{I} \cup c_{new}$

Table 3.2: Important procedures used in the imitation game

Chapter 4

Physical setup

The embodiment of the agents used in the experiments consists of one robotic arm and a stereo-head camera. This means that for the current experiment, agents have to share the single available body. For future experiments it is planned that two arms and two camera's opposite each other will be used to perform the imitation game.

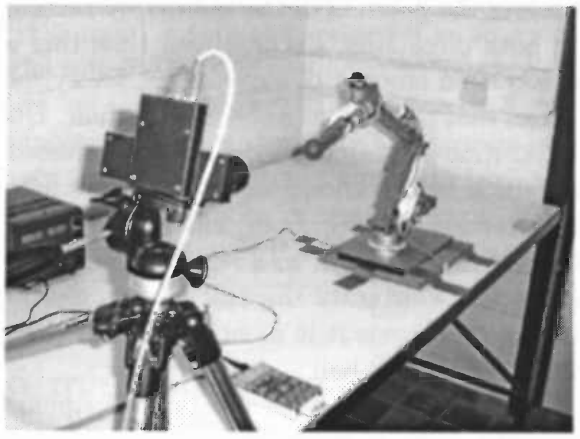


Figure 4.1: The setup consists of a 6 DOF arm and a stereo camera for respectively performing and perceiving actions.

4.1 Robotic arm

The robot arm used is the commercially available Teach-Robot[1]. It has six degrees of freedom (of which only four are relevant for the experiment). The kinematics (finding the position of the gripper, given the positions of the motors) and inverse kinematics (finding what the position of the motors should be to put the gripper at a given position in space) are known [De Vylder, 2002].

The arm is position-driven. After the desired motor positions are sent to it, one must wait for the arm to have reached the corresponding position before a new command can be issued. There is no control over the speed of movement.

[1] http://www.teach-robot.nl

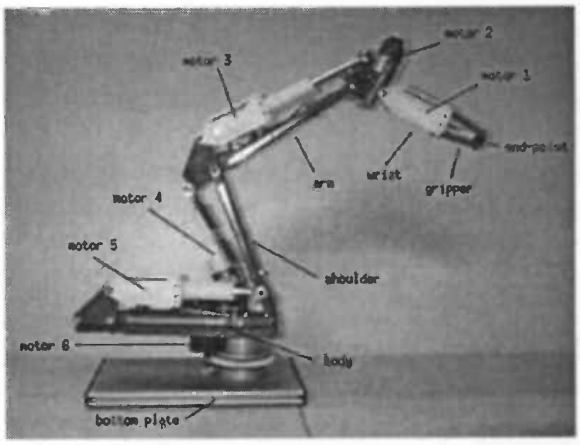


Figure 4.2: Naming of the motors and the different parts of the robot

The arm is equipped with proprioception: the current positions of the motors can be requested from the arm. This is the most reliable source of information about the position of the gripper. The arm determines the position of the motors by keeping track of the number of pulses used to send them in both directions. Because over time this will accumulate an error, the arm must be reset after a certain amount of movements. Every motor has a corresponding switch that is pressed only when that motor is at a specific position. During the reset procedure, all motors are put in such a position such that all switches are pressed and all motor positions therefore are known. This position is also known as the reset position (shown in figure 4.5).

In order to facilitate the detection of the gripper, a red ball is attached to it. The red color is usually not available elsewhere in the image, so a relatively simple color matching algorithm can be used to detect the gripper. Currently the ball is the only object that is focussed on to find the trajectory. When in this thesis it is mentionted that the gripper is observed, this always refers to the observation of the red ball.

There is a limited part of space that can be reached by the gripper. If an unreachable point is passed to the inverse kinematic code, it would be desirable if it found a motor setting that puts the gripper as close to is as possible. In many cases this is what happens, but it can not be depended upon. If actions are created at random, they are chosen to lie within a box in which all points can be reached by the gripper. This is called the visual box. The limits defined by the visual box are listed in table 4.1. It is still possible that actions are learned with one or both points outside this box, due to errors in execution or observation. Some of these actions will be possible to perform, others will not. The performable actions will be able to propagate through the population like any other action. The unexecutable (or unobservable) ones will eventually disappear again due to the characteristics of the imitation game.

Coordinate	Lower-limit (cm)	Upper-limit (cm)
x	24	40
y	-15	15
z	5	25

Table 4.1: Limits defined by the visual box

4.2 Vision

To observe the trajectory of the gripper, the commercially available MEGA-D stereo head is used, which can deliver a sequence of simultaneously taken left and right images. The Small Vision System (SVS) [Konolige, 1997] computes a depth map of these two images, containing for every pixel° in the left image that could be mapped to a corresponding pixel in the right image, the distance from the camera. To get the 3-D position of the gripper, the left image is segmented based on a prototype of the ball's color. If the ball is found and there is depth information of the corresponding pixels available, the averaged 3-D positions of all pixels is used as position for the gripper. If there is no depth information available, the right image is segmented in the same manner as the left one and the 3-D position is computed from the regions where the ball is found in both images. This procedure is far less precise than with the depth information of the SVS system.

Because the embodiment is shared, all agents have the same point of view on an action, in contrast with the situation where two agents sit opposite each other and thus have different points of view on the action. This circumvents a number of interesting issues, such as the recognition of self and other, and the use of deictic° coordinates. However embodiment sharing was a necessary simplification to test the principles of the imitation game on a robotic setup. Another simplification that is used, is that the actions that an agent can perform are limited to motion trajectories of the robot arm from one point to another.

When an agent observes an action it tracks the gripper-ball and builds a series of three-dimensional coordinates representing its trajectory. The process of image capturing and processing can be performed at a speed of about 10Hz. Since the time to execute an action depends on its trajectory, the number of observations of one can vary. It is assumed that the agents know when to start observing an action, so the observation is started just before the action execution is started.

4.3 Spaces and mappings between them

The term *space* is used for a particular frame of reference to define the position of the gripper in. In order to translate the position of the gripper between when it is observed by the camera and when it has to be computed based on the position of the robot's motors, three spaces are defined.

The spaces are:

Motor space The space defined by the positions of the arm's motors. The dimensionality of this space equals the number of motors of the arm, which is six but can be reduced to four since only four motors are relevant for the gripper position

Observation space The coordinate system of the vision system. It is a right-handed cartesian° coordinate system defined with respect to the fixed position of the left lense of the stereo-head camera. the X-axis is from left to right in the image plane, the Y-axis from top to bottom. The Z-axis is from the image plane toward the observed scene (depth) (see figure 4.4).

Action space A space defined to be used as an intermediate between the motor space and action space. It is the right-handed cartesian° coordinate system defined with respect to

the fixed position of the robot arm. It is defined with the Z-axis pointing upwards and coinciding with the vertical rotation axis of the body, the X-axis in the direction of the longest sides of the bottom plate, pointing to the front of the robot (in reset position) and the Y-axis in the direction of the shortest sides of the bottom plate (see figure 4.5). This space is not actually relevant to the agents themselves, but it is defined because it is the most convenient one to think about during implementation of the experiment.

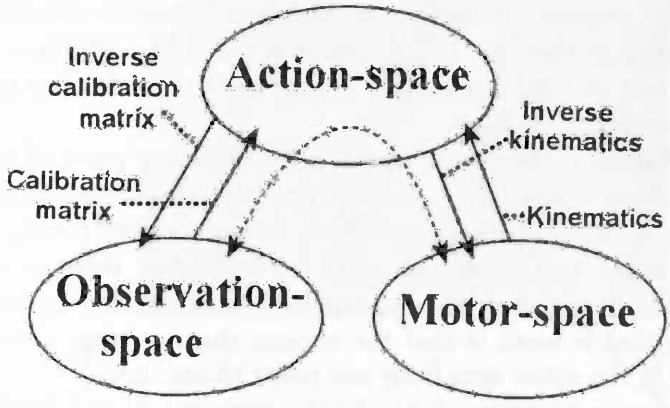


Figure 4.3: The spaces relevant for the robotic setup and the mappings between them.

Figure 4.3 schematically represents the spaces and mappings between them. The kinematics and inverse kinematics are used to map between action space and motor space, the calibration and inverse calibration matrix (see section 4.4) map between the action space and observation space. Therefore there is also a mapping between the observation space and the motor space, using the action space as intermediate.

4.4 Calibration

For calibration between the action space and the observation space, a set of eight observations is made of the gripper at the corners of the visual box. For one such a point, the arm is sent to it, the position of the arm is obtained from the proprioception of the robot and several observations are made with the camera, averaging the results.

This process results in a point in the action space (computed from the motor space using kinematics) where the gripper actually is and a matching point in the observation space where the gripper is perceived.

It is better to use the position obtained from the robot as point in the action space then the point it was sent to, because the control of the motor position is less reliable than the robot's proprioception.

From the results of the eight points a calibration matrix can be computed that gives the corresponding point in the action space for every point in the observation space. This matrix implements a rotation and translation. A point in observation space (x, y, z) can be multiplied with the matrix to obtain the corresponding point in action space (x', y', z') as follows:

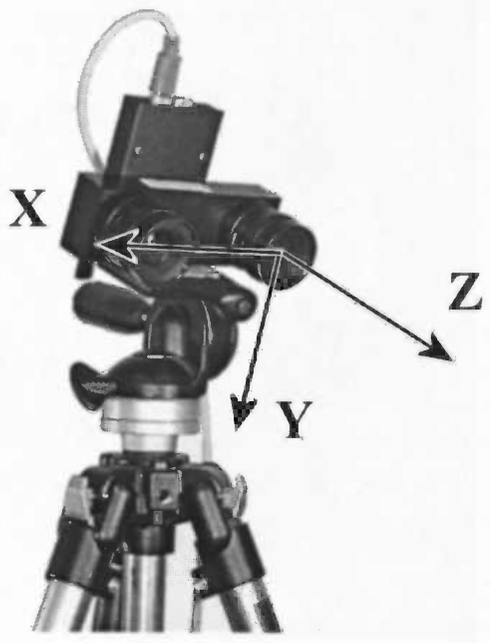


Figure 4.4: Oberservation space: Cartesian coordinate system fixed relative to the stereo head camera.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & t_x \\ m_{21} & m_{22} & m_{23} & t_y \\ m_{31} & m_{32} & m_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

The matrix specifies a translation, rotation and scaling that results in the minimal mapping error. The combination of the translation (t_x, t_y, t_z), rotation (ϕ, θ, ψ) and scaling (s) is uniquely determined by seven parameters. The optimal values for these parameters are computed using the Marquardt algorithm [Bevington, 1969; Kuester and Mize, 1973] to solve the nonlinear regression equation $ACT = F(OBS; B)$ for coefficients $B = [t_x, t_y, t_z, \phi, \theta, \psi, s]$ where OBS is the vector of observation data points, ACT the vector of gripper-position data points and F the mapping of OBS to ACT with B.

The inverse of the calibration matrix can perform the transformation from action space to observation space.

The images in figure 4.6 illustrate the workings of the vision system in general and the calibration procedure specifically. It shows the recorded left and right images and the corresponding disparity image and coordinates, both in the action space and the observation space for all eight points used for calibration. In the action space cm is used as distance unit, mm for the observation space.

The box drawn in the left image indicates where the ball was found. A box at the same position is drawn in the disparity image to indicate from what section of it depth information is used. Note that this is not always available for every pixel within the box.

Note that it is possible to see the images in figure 4.6 as 3D. For more info on how to do this, refer to appendix B.

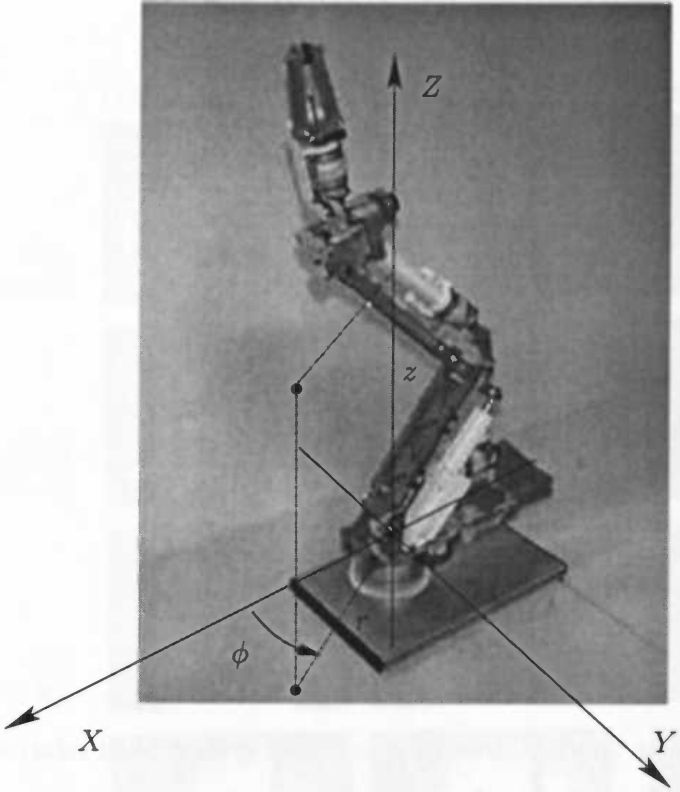


Figure 4.5: Action space: Cartesian and cylindrical coordinate frame fixed relative to the bottom plate of the robot. The XY-plane contains the link between the body and the shoulder of the robot.

4.5 Modelling the physical setup

It would be desirable that the simulation resembles real observations to such an extent that there is virtually no difference between a real experiment and an experiment in simulation. In an attempt to reach this goal, results obtained from the actual physical setup have been gathered with the intention of building such a realistic model from them. This has turned out to be unfeasible due to the complexity of the setup. This failure emphasizes the importance of using an actual physical embodiment in experiments with robots, because its complexity is hard to model and easily overlooked when only working with simulations.

For the gathered data, the location of the gripper has been recorded for 91000 positions while performing regular imitation games. Every sample thus results in a point in action space and a corresponding point in the observation space. Points where the vision system could not find the gripper are not included in this set. If the observed point would be transformed to the action space using a perfect calibration and if no noise would be present in the system, both points would have the same coordinates. This is not the case in our physical setup.

At the left side of figure 4.7 are scatterplots of the x-, y- and z-coordinate respectively of the position of the gripper and the observation for every sample. The straight line represents the expected and ideal values. Deviations from this line are errors that can have accumulated from errors in proprioception, errors in the vision system and errors due to the transformation by the calibration matrix.

The plots at the right side represent the percentage of measurements whose error is below the value on the horizontal axis. Error is defined as $|position - observation|$. For all three coordinates at least 90% of the samples have an error below 4cm.

Several aspects of the physical setup are reflected in these plots. Generally perception is reliable as can be observed from the fact that most points lie nearest the ideal line. The range of the points of the scatter plots are bounded because of the limitations of the physical setup. For instance, it can be observed in the scatterplot of the x-coordinate that the arm is not capable of reaching a position with an x-coordinate greater than ~55cm. This is due to the fact that the arm is not long enough to extend beyond this value. There are some points in the plot where it is attempted to send the arm to a position with a higher x-coordinate, but this results in the gripper going to some other position near it that *can* be reached and thus in the arm being maximally stretched again. The resulting observation is thus the same as the maximal possible x-coordinate. The visual box for the x-coordinate specifies an upper limit of 45cm, so it can be noted that the system explores the space outside the visual box as well. The same principle can be observed at the lower limit of the x-coordinate and for both limits of the y- and z-coordinate.

Since the camera is placed opposite of the robot arm, the x-coordinate of the position of the gripper corresponds to the z-coordinate (or depth) of the observation space. A higher value for the x-coordinate indicates a position further from the robot and thus closer to the camera resulting in a lower value for depth. Since depth perception requires stereo vision, this is less reliably then the y- and z-coordinates of the points observed by the camera. This results in some points with a relatively large error.

One of the problems that was encountered was the fact sending the arm to a certain position gave different results depending on where the gripper was before the movement. This makes a single point to single point simulation intractable and to incorporate the previous position into such a simulation, a squared number of samples is required obtain the same number of observations for a single position.

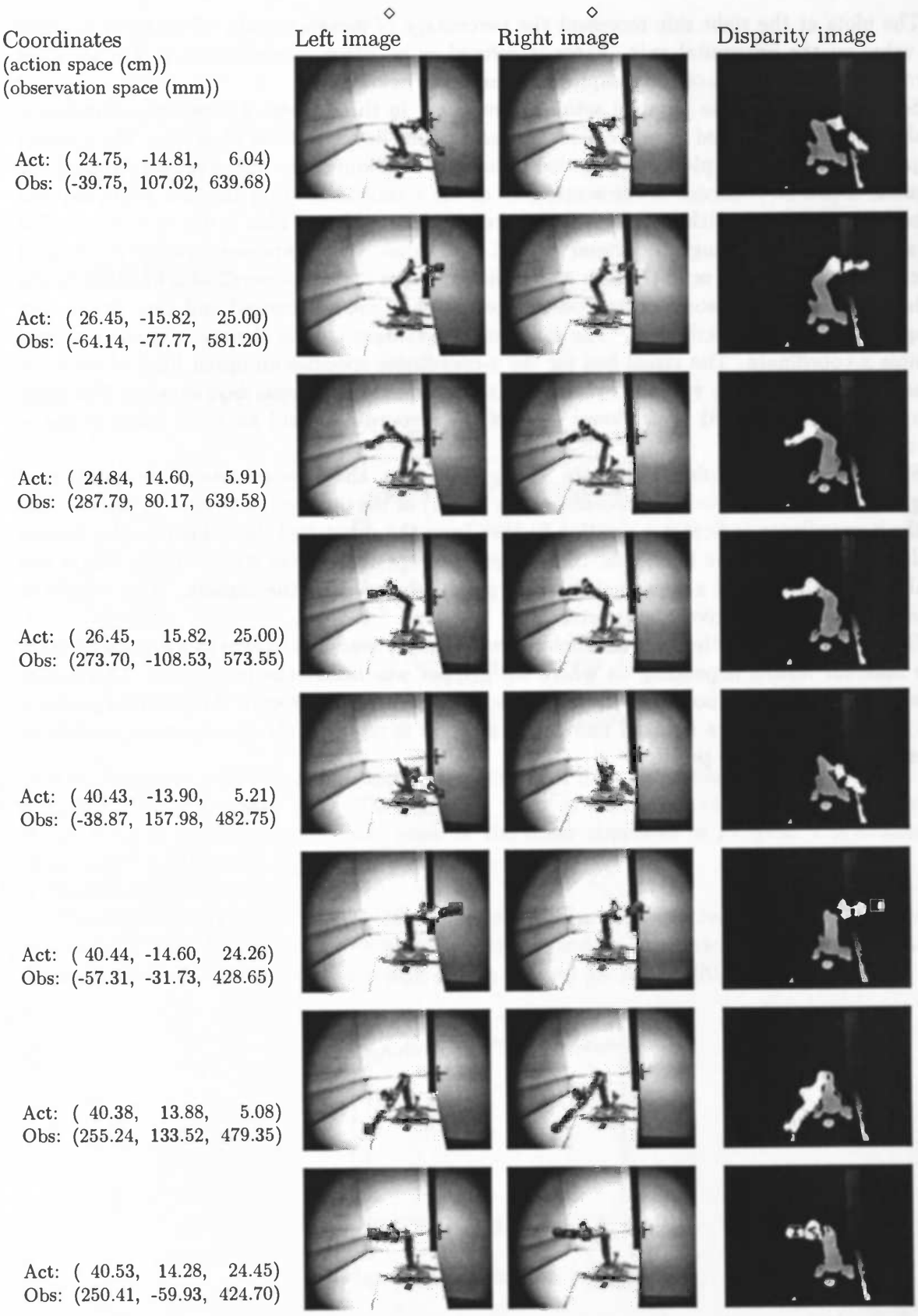


Figure 4.6: Positions of calibration

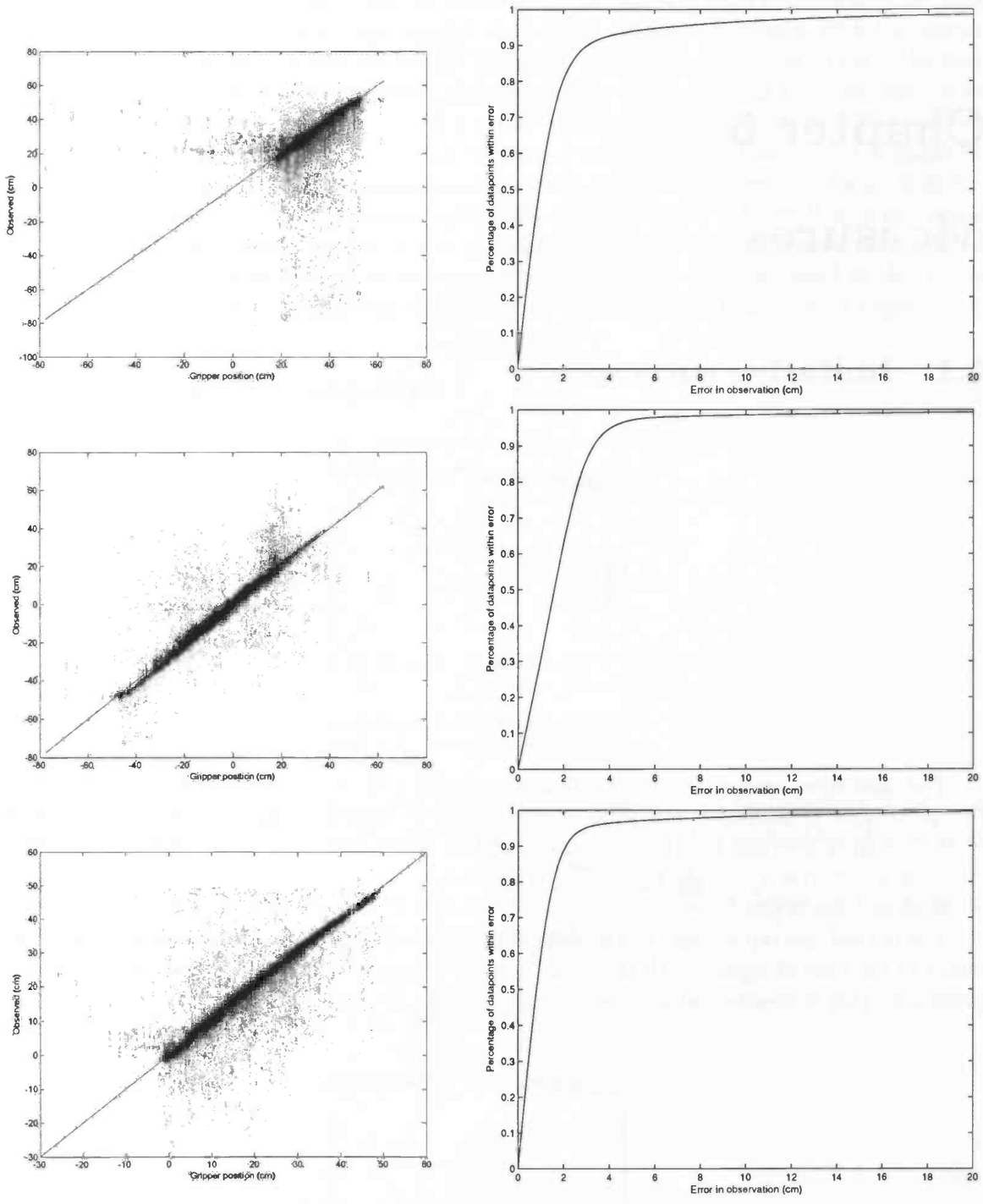


Figure 4.7: Scatterplots of coordinates of points of the gripper in action space and the corresponding observed points transformed to action space (L) and error plots (R) (x,y and z respectively)

Chapter 5

Measures

5.1 Imitative success

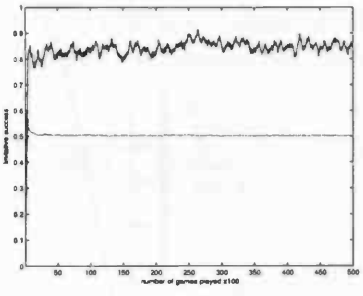


Figure 5.1: Imitative success

The imitative success of one imitation game is 1 if it was successful, 0 otherwise. In the plots this is averaged over 500 games to show trends in average success. The value of the running average for this and some of the other measures m at game t is defined as $running\_average(m(t)) = \frac{1}{500} \sum_{i=0}^{\min(t,500)} m(500 - i)$.

High success is good.

The dotted line represents the theoretical baseline value for imitative success or the expected result in the case of agents with the same number of random categories. For the mathematical proof why this is counter-intuitively high please refer to [de Boer, 1999].

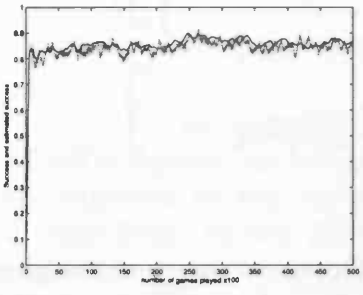


Figure 5.2: Estimated success

How the game is played can be analysed in terms of game theory by calculating a matrix

with the distances between all categories of the initiator and imitator on the columns and rows respectively. The initiator chooses a column at random, the imitator then chooses the row with the lowest distance in this column and finally the initiator picks the column with the lowest value for this row. The game is only successful if both used columns are equal. This is the case iff – in the column the initiator starts with – the distance that is smallest is also the smallest in it's row. Or in terms of game theory: if it is a nash equilibrium [Binmore, 1992]. The number of nash equilibria in the matrix can be used to estimate the chance of success of a game by dividing it with the total number of categories the initiator has. This gives the chance that the initiator chooses a category that is a nash equilibrium and therefore will result in a successful game. In figure 5.2 a running average of the estimated success is plotted over a grey colored version of figure 5.1. The estimation is in fact so good that it has not been used in this thesis because it provides almost no additional information about the performance of an experiment.

5.2 Number of categories

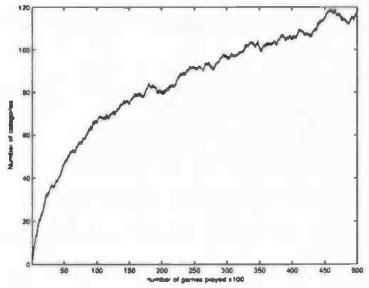


Figure 5.3: Number of categories

Every agent a in a population $\mathcal{A}$ with $|\mathcal{A}|$ number of agents has an inventory $\mathcal{I}_a$ with a number of $|\mathcal{I}_a|$ categories. The average number of categories $\overline{|\mathcal{I}|}$ of the population is defined as $\overline{|\mathcal{I}|} = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} |\mathcal{I}_a|$.

The line in figure 5.3 plots $\overline{|\mathcal{I}|}$ as a function of the number of games played. A higher value is better, but only if the success can remain high as well.

5.3 Average inventory distance

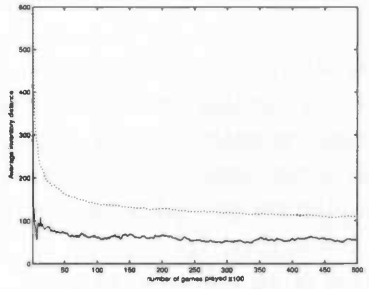


Figure 5.4: Average inventory distance

In the plot of the average inventory distance in figure 5.4 there are two lines. The solid line indicates the average value of the inventory distances $\overline{\mathcal{ID}}$ between agents, the dotted line is the expected $\overline{\mathcal{ID}}$ if the agents would have random categories. The important aspect of this measure is to what degree the inventories are better than random ones or how much lower the actual value is below the baseline value.

The average inventory distance is a measure that indicates to what degree the category inventories of the agents in the population resemble each other. This is the same measure that is called the average category distance in [Jansen, 2003]. The inventory distance measure is defined in such a way that it is lower when the inventories of two agents are more alike. The inventory distance between two inventories $\mathcal{I}_a$ and $\mathcal{I}_b$ of agents a and b is defined as

$$\mathcal{ID}(a,b) = \frac{\sum_{\alpha \in \mathcal{I}_a} \min_{\beta \in \mathcal{I}_b} d(\alpha,\beta) + \sum_{\beta \in \mathcal{I}_b} \min_{\alpha \in \mathcal{I}_a} d(\alpha,\beta)}{|\mathcal{I}_a| + |\mathcal{I}_b|}$$

Here d is the same distance measure between two categories as defined in section 3.4.1.

The average inventory distance ($\overline{\mathcal{ID}}$) of a population $\mathcal{A}$ with $N = |\mathcal{A}|$ agents is the average of the values of $\mathcal{ID}$ for every agent in the population and defined as follows:

$$\overline{\mathcal{ID}}(\mathcal{A}) = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \mathcal{ID}(\mathcal{A}_i, \mathcal{A}_j)$$

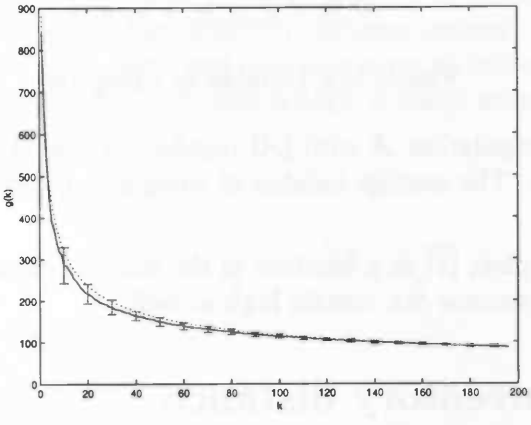


Figure 5.5: Computed value and approximation of g(k)

It has been shown in [Jansen, 2003] that the expected $\overline{\mathcal{ID}}$ for two agents a and b with respectively k and l number of random categories can be expressed as $\overline{\mathcal{ID}} = \frac{kg(l)+lg(k)}{k+l}$ where $g(k)$ is a function that can be approximated by the expression $g(k) = 8804k^{-0.4384}$. The parameters of this approximation differ from [Jansen, 2003] and have been determined by computing the value of $g(k)$ in simulation several times for different values of k. It is possible to compute this value because if $l = k$ then $\overline{\mathcal{ID}} = g(k)$. So if $\overline{\mathcal{ID}}$ is computed for two agents both with k random categories this is an estimate of $g(k)$. Doing this several times for the same k and averaging the outcomes gives a good approximation for g(k). Figure 5.5 shows the values found for g using this method, with error bars to indicate variance. The dotted

line is the exponential approximation of $g(k)$ that is used. The estimation of g is actually too low because the categories that are used to obtain figure 5.5 all lie within the visual box. In simulation and with the arm it is possible however that categories outside the box propagate through the population (see section 4.1. Therefore there is effectively more space for categories and less chance that random inventories resemble each other.

5.4 Information transfer

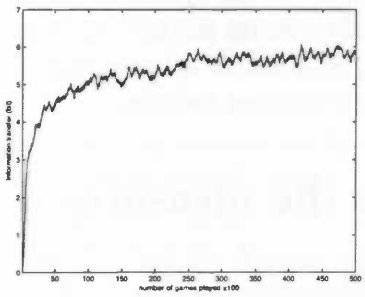


Figure 5.6: Information transfer

Neither imitative success nor the number of categories are by themselves good measures for the performance of a population because there exists a trade-off between them. Although in the experiments presented here, categories are only used in imitation games, assuming that they already have been learned this way, they can then be used for communication between agents. If more categories are shared between them this means they can transmit more information with one category. But this is only benificial if the communication is reliable. For instance: the imitation game will always be successful if both agents have one category, but with just one category no information can be transmitted. On the other hand, if two agents have identical inventories with large sets of categories that are so close together that the noise due to arm movement and observation is so large that an action will be observed as another category by the imitator, imitation games will tend to fail. If the two agents have shared categories that can not be communicated successfully on the robotic setup the transmission of information is unreliable. The definition of entropy in information theory is used to derive a measure that is higher if more categories are learned, but only to the degree that they result in successful communication.

If a category is imitated successfully, the initiator transferred enough information by performing the action for the imitator to know which action it was. For our setup we consider the initiator to be the source of the information, the robotic arm and camera as the communication channel and the imitator as the receiver. The entropy is a measure for the minimal amount of bits required to transmit the information of the action. This amount of information in a single action depends upon it's predictability: an often used one conveys less information than an action that seldom is used. In our experiments however all categories have an equal chance P of being picked by the initiator for imitation. Therefore: $\forall_{c \in \mathcal{I}_{initiator}} : P(c) = \frac{1}{|\mathcal{I}_{initiator}|}$. According to information theory the entropy of a variable is defined as $H(X) \equiv -\sum_x P(x) \log_2(P(x))$, where X is a variable than can be in several states x (categories in our case). So when $n = |\mathcal{I}_{initiator}|$ and $P(x) = \frac{1}{n}$, then $H(X) = -\sum_{1..n} \frac{1}{n} \log_2 \frac{1}{n} = -\log_2 \frac{1}{n} = \log_2 n$ (bits).

The value of *InformationTransfer* for a game t is 0 if the game failed or $\log_2 n$ if it succeeded. The plot in figure 5.6 is the running average of *InformationTransfer*. Information transfer is a good measure of the performance of the system: higher values indicate better performance.

5.5 Histograms

The two histograms are meant to give an insight into the successfulness of the categories at the end of an experiment. The left one indicates for different successfulness-bins how many categories are in it. The frequencies for all agents are stacked in the same histogram using different shades of grey. The right histogram further analyses the categories that are in the bin with the highest success (> 0.99) and shows a histogram of how often these categories have been used. This is done in the same stacked fashion.

5.6 How good were the measures used?

The defined measures give a good enough insight into the workings of the system. Imitative success and number of categories are very basic and important measures to inspect. The information transfer can be deduced from these both, but gives a good indication of the overall quality of performance, taking into account the trade-off between success and number of categories. The inventory distance seems to be the optimal measure to indicate similarity of inventories, but it can not give a perfect insight into how two sets of 6-dimensional points resemble each other. This would however be very hard to do with a one-dimensional measure. Using normal parameters it's value always tends to a value of about 50. But if there is no convergence this can be inferred from the inventory distance measure, for instance when there is high noise like in figure A.7. The histograms only give a rough indication of what the spread of success is in the resulting categories, but they are not a useful measure in their own right.

Chapter 6

Results

The experiments had already been shown to be working in simulation [Jansen, 2003; Jansen et al., 2003]. To revalidate this result I have reimplemented the experiment based on existing code. This I have done in such a way that there is no difference between simulation and experiments with the real arm, except that in the former case the observations are simulated while in the latter they are obtained from actual observations of movement of the arm.

First I will show that the experiment is still successful after reimplementation and present a sensitivity study to show the influence of the critical parameters on the system performance.

Then I will show the results obtained from a single run on the real arm that also is successful.

6.1 Simulation

6.1.1 Default results

Here it is shown that – in simulation – a shared repertoire of action categories can emerge in a population of agents. To give an impression of the influence of the used parameters, first the plots of a run with default values for parameters is given and then plots with the same values, but with one parameter that is given a different value. This section is divided in a section for every parameter that is changed. The default values for the parameters is chosen as follows:

Parameter	Value	source
Throw away threshold	0.7	[de Boer, 2000a]
Merging threshold	200	Determined from earlier experiments
Successfulness threshold	0.5	[de Boer, 2000a]
Shift-factor	0.03	[de Boer, 2000a]
Noise	10.0 (+1.0)	Error on arm and camera chosen to be $\sim 1cm$
Minimum-uses	5	[de Boer, 2000a]
Add-probability	0.02	[Jansen, 2003]

Table 6.1: Default parameter values for simulation

There are two values for noise: observation noise (in mm in the observation space) and execution noise (in cm in the action space). Both are set to an average error of $1cm$.

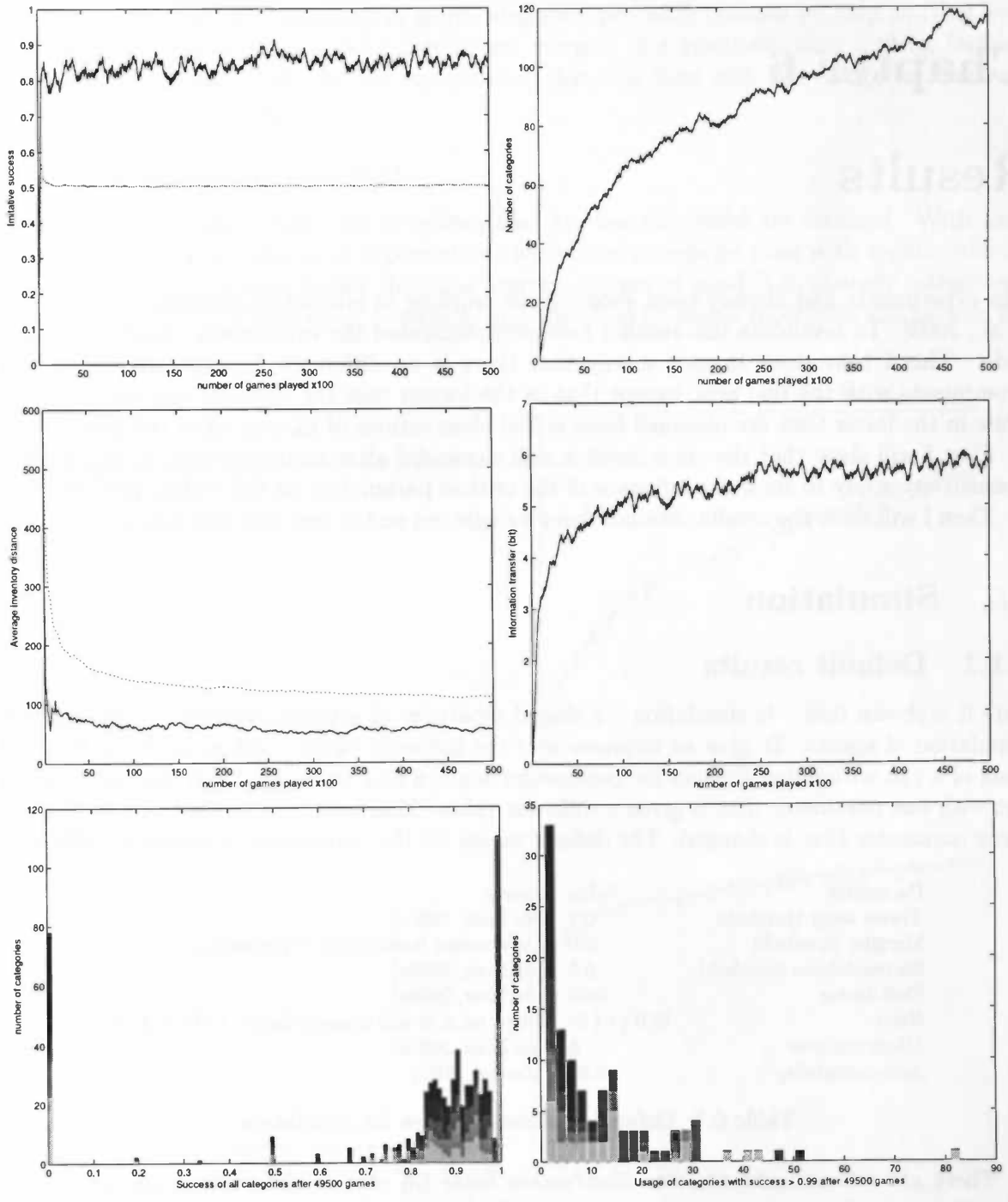


Figure 6.1: 5 agents, simulation with default settings

Figure 6.1 shows the results of the simulation for the first 50000 games with a population of five agents. The measures plotted are explained in chapter 5. It can be seen that the success remains high relative to the baseline success at 85% while the number of categories keeps growing with a decreasing speed up to a value of about 120 categories. This indicates that the agents have built up inventories of about $120 * 0.85 = 102$ shared categories by playing imitation games. The average inventory distance is well below the baseline value, also indicating that similar inventories have emerged. The information transfer reaches a level of almost 6 bits ($0.85 * log_2 120 = 5.9$), indicating that $2^6 = 64$ categories can be communicated reliably between all agents in the population. The success histogram shows that most categories have a success rate of over 80%, but this is not surprising because all categories that are less successful than 70% (the throw away threshold) disappear from the inventories. The high value for a success of 0% is due to the fact that new categories often interfere with existing ones, causing game failure. Of the most successful categories, most have been used less than 10 times. Categories that have been used more than that and have such a high score can be considered very stable. Please note that a low usage can also be caused by are very 'old' category being merged with another one that comes so close to it that they are merged and re-introduced as a new category.

Figure 6.2 shows the average of 9 simulated runs all with 5 agents and the same default settings as the experiment shown in figure 6.1, which is one of those 9 runs. This illustrates that there is some variation in the imitative success and number of categories between runs, but that the overall behavior of the system is very similar. The average inventory distance has almost no variation between runs.

6.1.2 Sensitivity study

I will attempt to describe the effects that changing a parameter can have on the performance of the system. This can sometimes be quite hard because many factors can play a role.

In appendix A many plots are provided for different values of the parameters at the risk of a graph overkill. The reader should therefore note that the graphs presented there are just meant to give an insight into how parameters influence the system and need not all be studied in detail. In the rest of this section I will explain the effects that changing a certain parameter will have on the performance of the system.

Varying the level of noise

Given two agents with certain inventories, more noise will have the effect that the agents are less likely to play a successful game.

If the noise is very large, performance will be as good as with random categories.

Varying the merging threshold

This is the most influential parameter of the system. Because it controls how close categories can be before they are merged, it effectively controls the maximum number of categories. If the threshold is high, categories further apart are merged, so there is less room in the available space for categories, so there are less of them. Even if the merging threshold is set to 0 (effectively eliminating the process of merging), the information transfer still remains relatively high at 4 bits. Although the success is low (50%), information transfer can be high because this is for a high number of categories (225). With low merging distances inventory distance is not as well below the baseline value as with higher ones.

Varying the shift-factor

The impact of different values for the shift-factor is not very large, but it can be observed that when no shifting is done, the category variance remains higher compared to shifting with a small factor. No shifting also results in more categories, probably because without shifting less categories get close enough to each other to get merged. For relatively high shifting factors the result of the system is almost the same, except that it is done with less categories which is probably due to more merging.

Varying the throwaway threshold

With a lower throwaway threshold categories that are less successful are retained. With low values the system still behaves as expected but with more categories than with higher values. If the value is set to a level higher than the average success of good (i.e. shared) categories, all categories will get thrown away eventually and therefore no stable repertoire will be able to emerge.

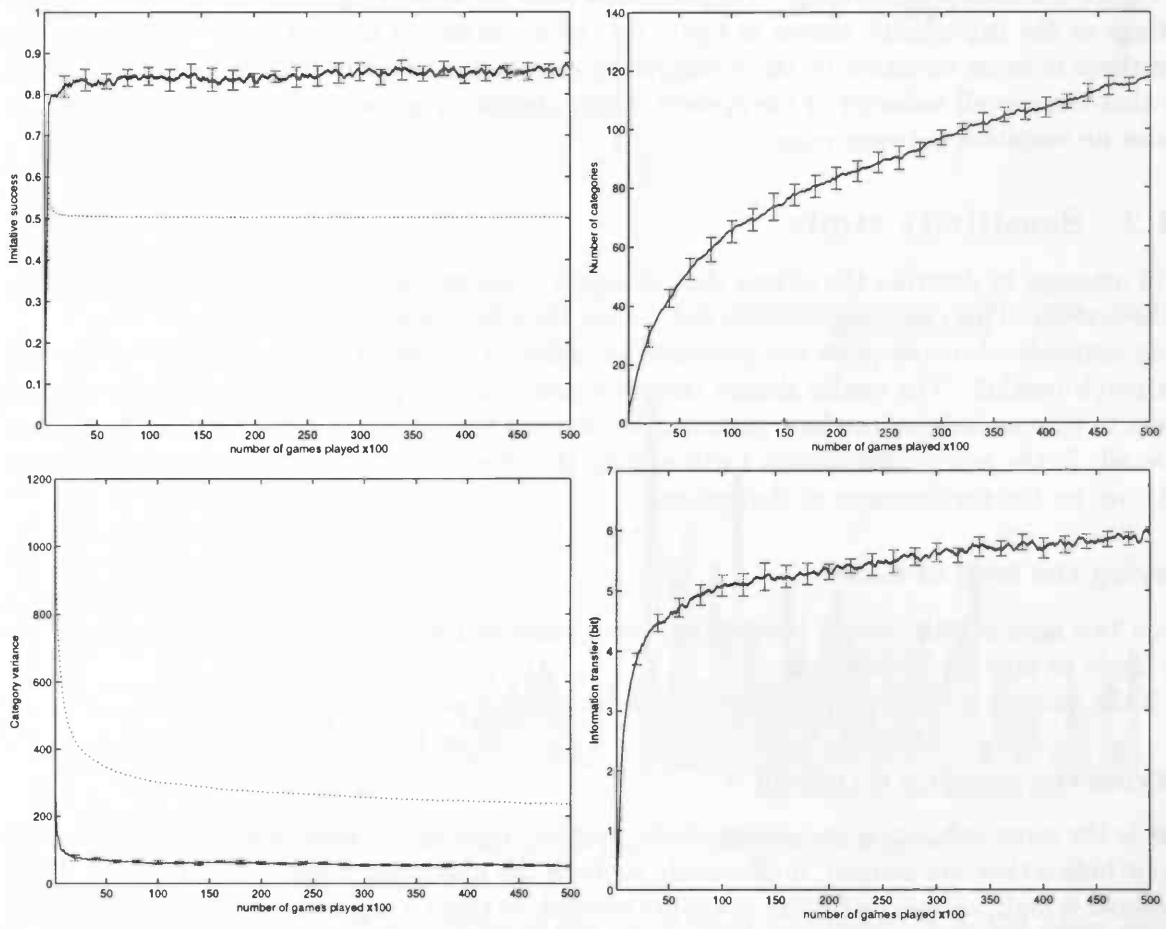


Figure 6.2: Average with errorbars of 9 simulated runs with 5 agents and default settings

Varying the successfulness threshold

The successfulness threshold has almost no effect on the system. This is the case because the only aspect that it affects is the choice between two self-organizing mechanisms in case of game failure. These mechanisms are shifting and the adding of a new category and the subsequent merging of categories. Both mechanisms will produce the 'usual' result so which is used does not have great influence.

Other parameters

The number of uses before a category can be thrown away and the chance of adding a random category are not investigated in this sensitivity study because their value does not have much influence on the behavior of the system.

6.2 Results with physical setup

The results presented here are obtained in the first successful run on the physical setup. To play 49500 games in a population of only two agents, the arm and camera have been running almost non-stop during a period of a month. Unfortunately this was also the last run because due to hardware failure there has not been an operational robotic arm since.

In figure 6.3 the measures are plotted for the run on the physical setup. Most importantly, it can be observed that success can remain well above the base-line value while the number of categories keeps growing. The fact that the level of success is lower than in simulation with five agents can be attributed to several factors.

The most important factor is that at the time of the arm experiment, there still was a bug in the algorithm that finds the closest category to an observation. Fixing this bug has resulted in improved performance in simulation. If the experiment will be run on the real arm at this moment it is expected that the results will be better for two agents then they are presented here and that the experiment will even work with a population of five agents (a short attempt at this before the final breakdown and the bugfix had shown that this did not work yet at that moment).

Another reason why the success with the arm is less than in simulation is the fact that there is an increased chance that observation of the gripper or movement of the arm fail in some way. For instance, in about 1% of the time the arm moves to what seems to be a random position in stead of where is was instructed to go. This illustates how robust the emergence of shared action inventories is in the imitation game.

The experiment was stopped after 49500 games because the number of categories did not seem to converge yet. Because the complexity of a lot of algorithms are polynomial on the inventory size, a lot of time was lost on computation at that moment. The good side of this is that a lot of categories can be fit in the available space so the arm has a great potential for expression. The down side is that we do not want to wait that long. Therefore the experiment should be repeated with a higher value for the *merging threshold* parameter (a value of 100 was used for this run).

Figure 6.4 and 6.5 show representations of the inventories of the two agents at game 3000 and 10000 respectively. Every category is plotted with its respective begin and end point of the action. The begin is marked with either a square or a circle depending on which agent's inventory the category belongs to. The more often a category is used, the larger it's marker

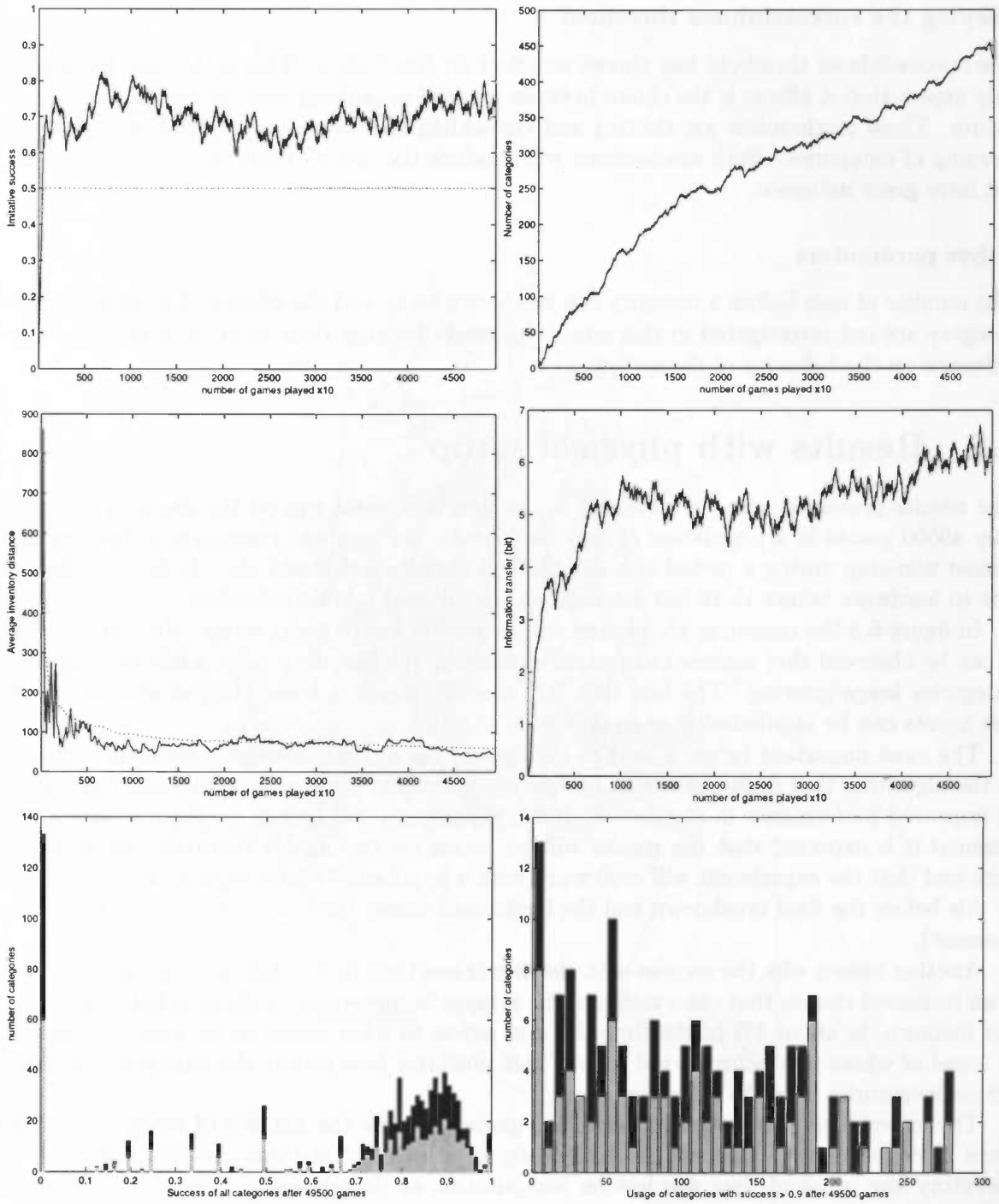


Figure 6.3: Experiment on physical setup with 2 agents

s. It can be seen that a lot of actions of one agent have a counterpart in the other agent's inventory. The 2D representation shown here is not a very good one to see how good inventories map onto each other, but no good way to map the 6D space of the categories to two dimensions has been found and it is likely that there is none.

The inventory distance measure seems to indicate that the inventories hardly resemble

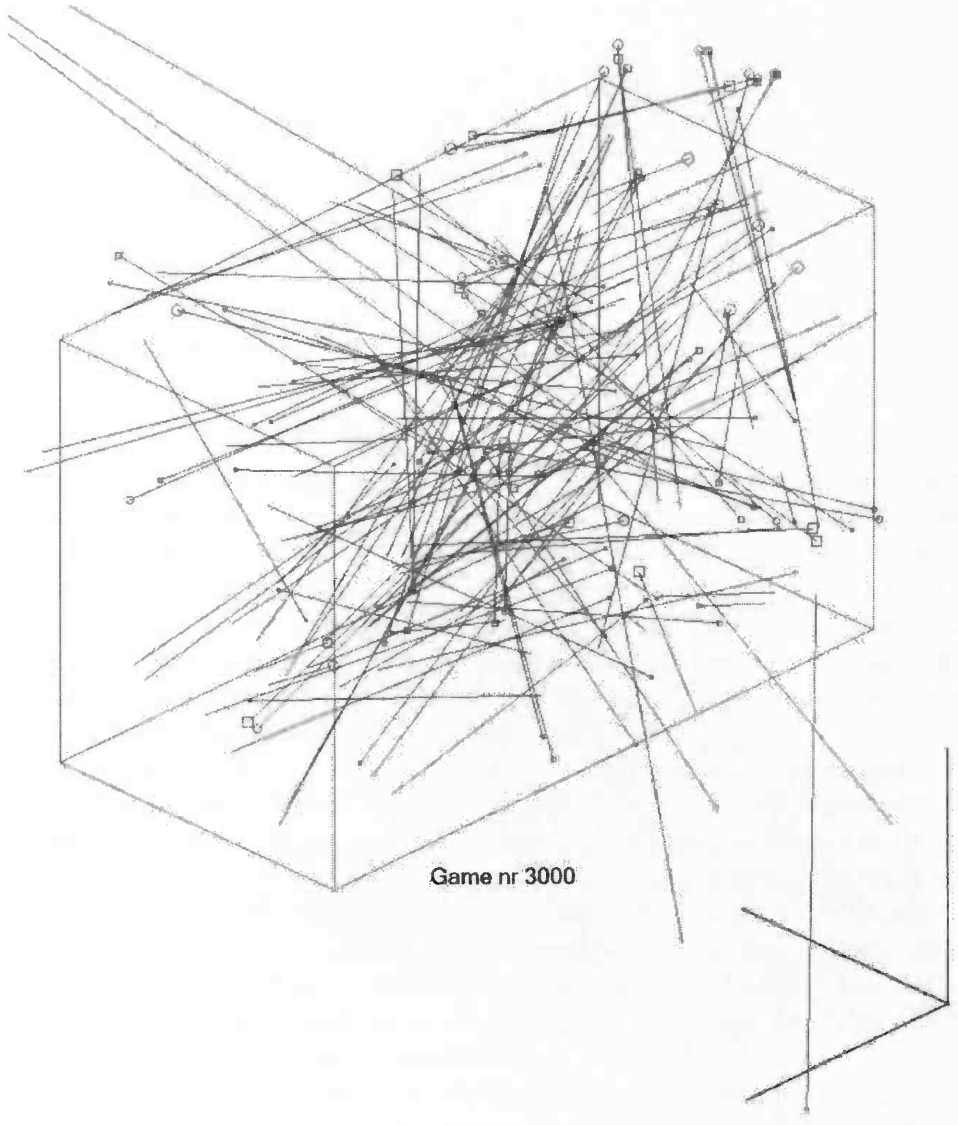


Figure 6.4: Graphical representation of the inventories of both agents after 3000 games have been played

each other more than random ones. The situation does not seem to be that bad judging from figure 6.5 and it could be true that this is due to the underestimation of the g function (described in section 5.3). It could also be that a different estimate of g is required when working with the mechanical arm in stead of in simulation.

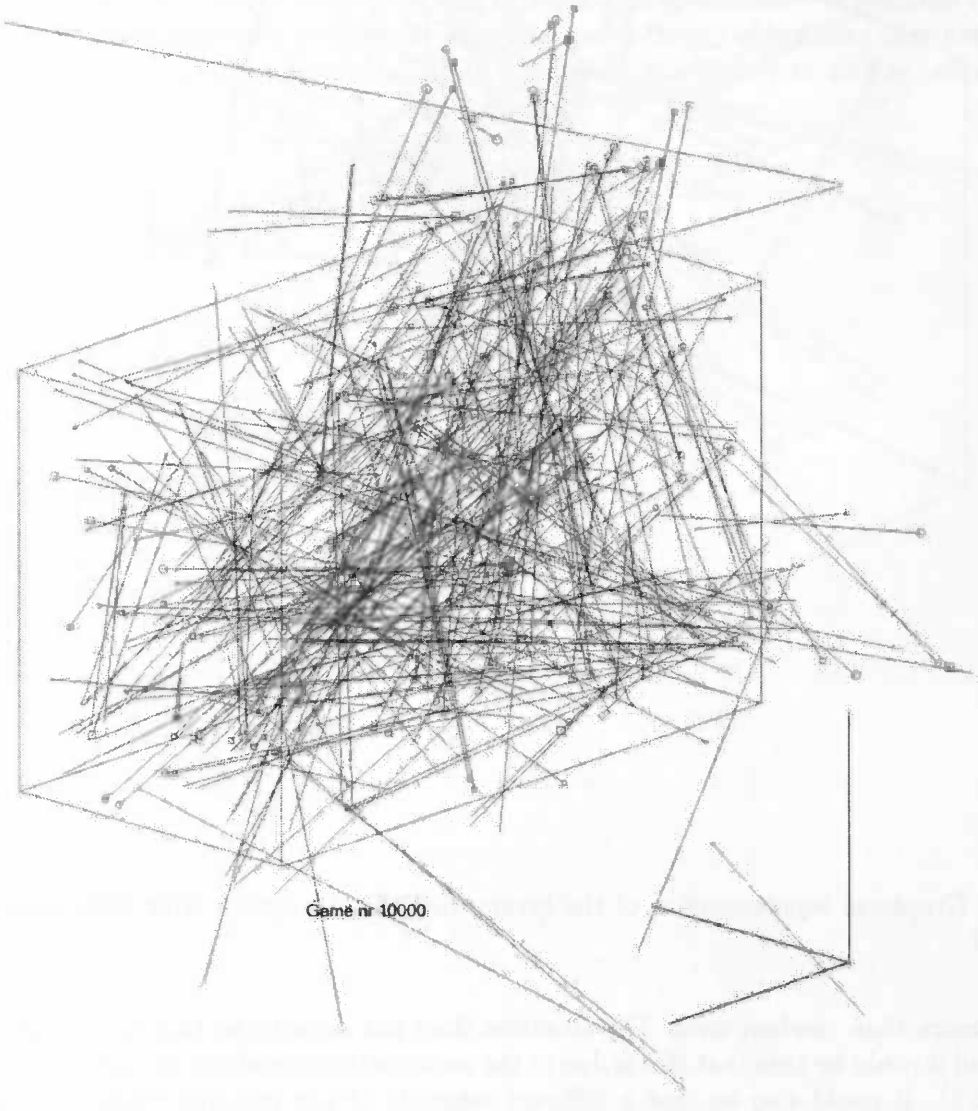


Figure 6.5: Similar representation of the inventories after 10000 games

Chapter 7

Discussion and conclusion

In this thesis it has been described how similar category inventories can emerge from imitative behavior between agents that are embodied in a physical setup. This result has been obtained without the agents being able to inspect the internal state of any other, without a form of central control, without categories being predefined and without predefined roles of iniatiator and imitator. Action categories propagate through the entire population by agents alternately taking the role of teacher and student. Because there is a pressure to add random action categories and agents adapt those categories depending on the outcome of the game, the action categories can emerge while the agents are interacting.

Because of the importance of embodiment for Artificial Intelligence, a mechanical arm was used to give an agent the potential to interact with it's world. In the experiment described in this thesis the advantage of using an arm might not seem great, but it will be impossible to reach the goal of an agent with the linguistic capabilities of a two-year old child without some kind of body. Future work should also address the issue of how categories of movement can be coupled to concepts about the world. The results obtained are less 'realistic' than when the approach was used on vowel systems because there is almost no cognitive foundation as to how humans learn and represent categories of movements. The project has none the less been an important contribution in this line of research.

The project's success has showed that the principles of self-organization and imitation can lead to shared repertoires. Using the language-game approach, models have been made that can explain universal tendencies in colour categories and vowel systems. Language-games are also used in a system currently being developed that enables agents to build up a lexicon and grammatical structure with which to communicate what is relevant to them in their environment [Steels et al., 2004].

It has been nescessary to make many simplifications in order to finish the project on the available time-scale. These include the sharing of one body in stead of imitation between seperate ones, the use of two-point movements in stead of complex ones, assuming that the kinematics are known and not using previous observations to recognize current trajectories. Hopefully these issues can be addressed in future research.

Ongoing research at the AI-Lab is focussing on imitation of intentions [Jansen et al., 2004a; Jansen, 2004] in stead of 'direct imitation'. It is proposed that the process of human development consists of several stages that are differentiated by their type of intention (gestural or utilitarian) and degree of social interaction. For instance: individual gestural development corresponds to motor babbling where social utilitarian development has to do with imitation of the use of objects. Motor babbling can be an alternative for having to know the kinematics of

the robotic arm. I find it a good approach to first attempt to build a robot capable of learning how to control it's body by observing it's own actions. Ultimately imitation is learning by observing somebody else's body and it seems likely that the ability to learn from observing one's own body is a prerequisite for this.

An interesting future project would be to model motor babbling and imitation using a cognitive architecture like ACT-R [ACT-R Research Group, 2004]. I think it is possible to implement the research presented in this thesis using ACT-R and that it will provide at least some cognitive foundation. One of the advantages of using this architecture is that the combination of ACT-R's partial matching mechanism and it's general theory of learning eliminate the usage of the merging threshold parameter which in my model is arbitrary as well as influential.

Appendix A

Plots of simulation sensitivity study

A.1 Default

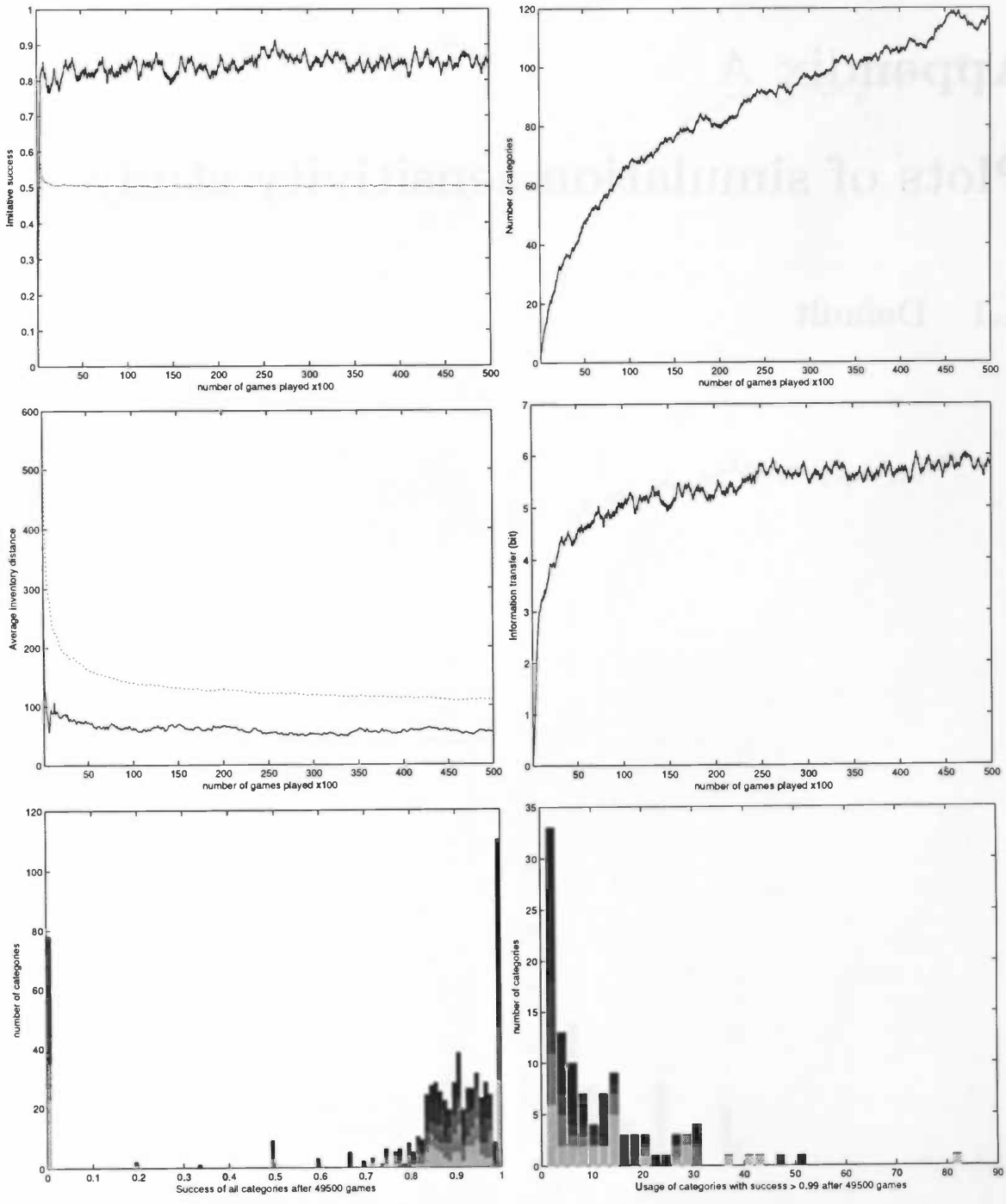


Figure A.1: 5 agents, 1 run with default settings

A.2 Noise

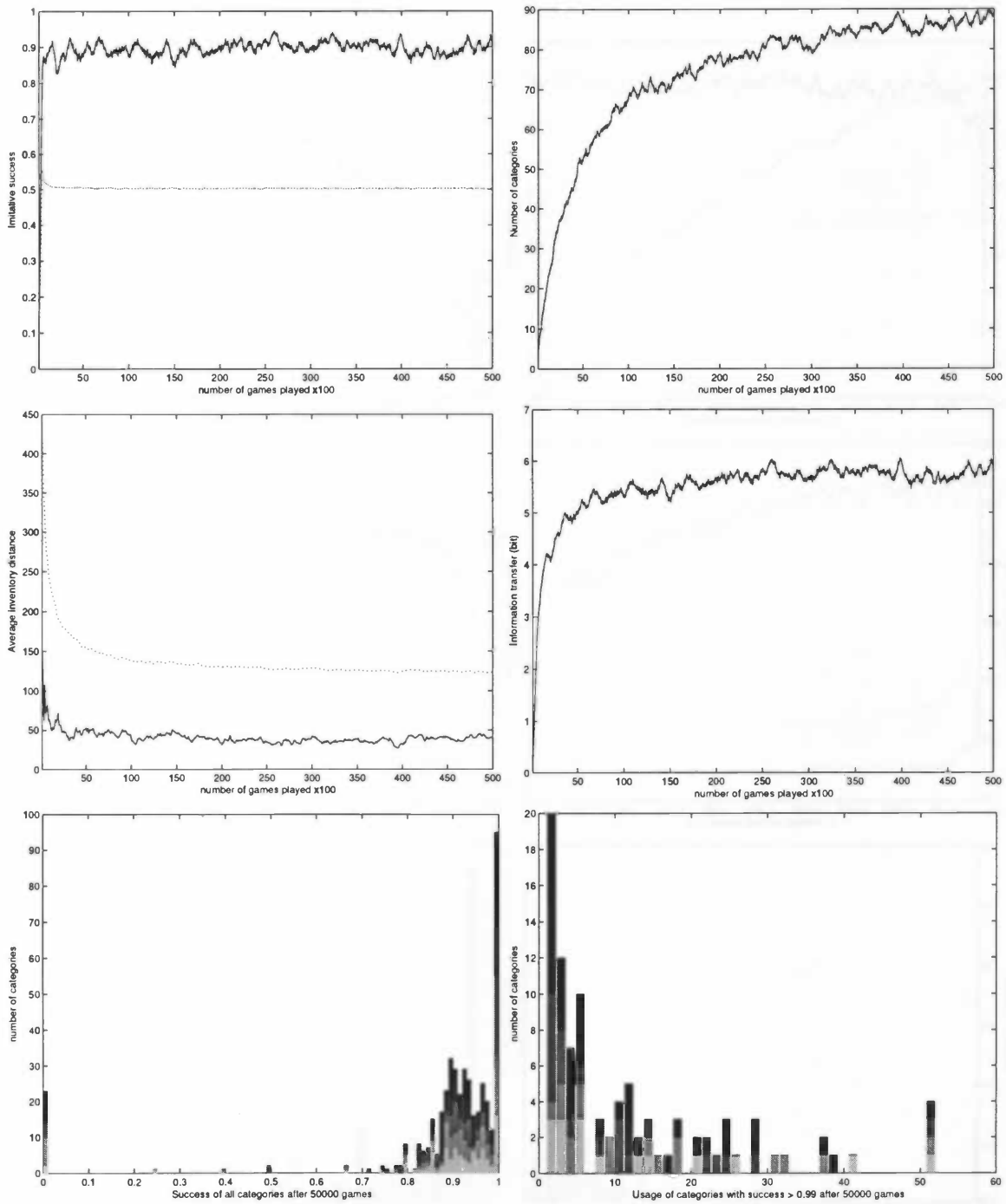


Figure A.2: 5 agents, noise = 0.0

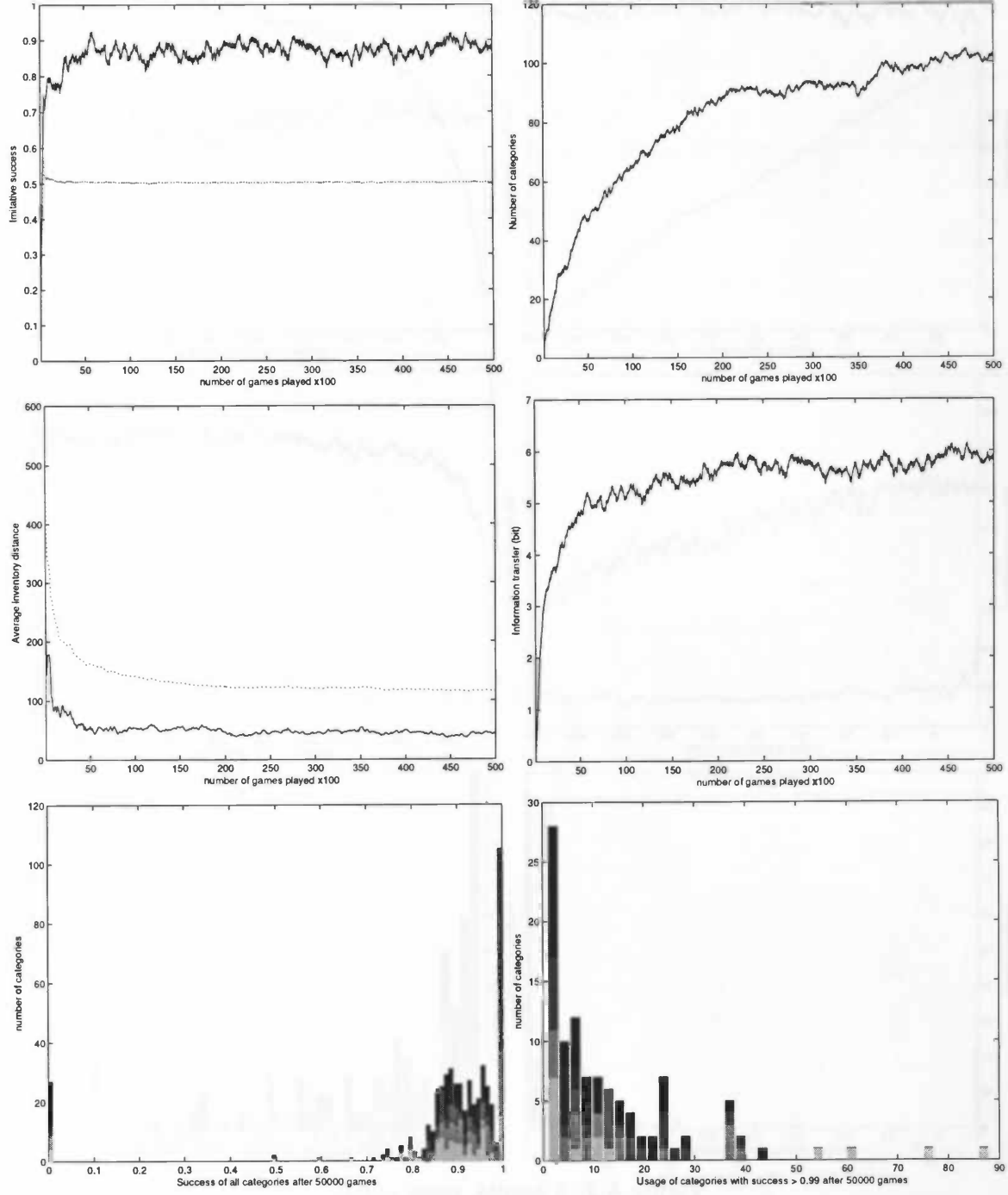


Figure A.3: 5 agents, noise = 10.0

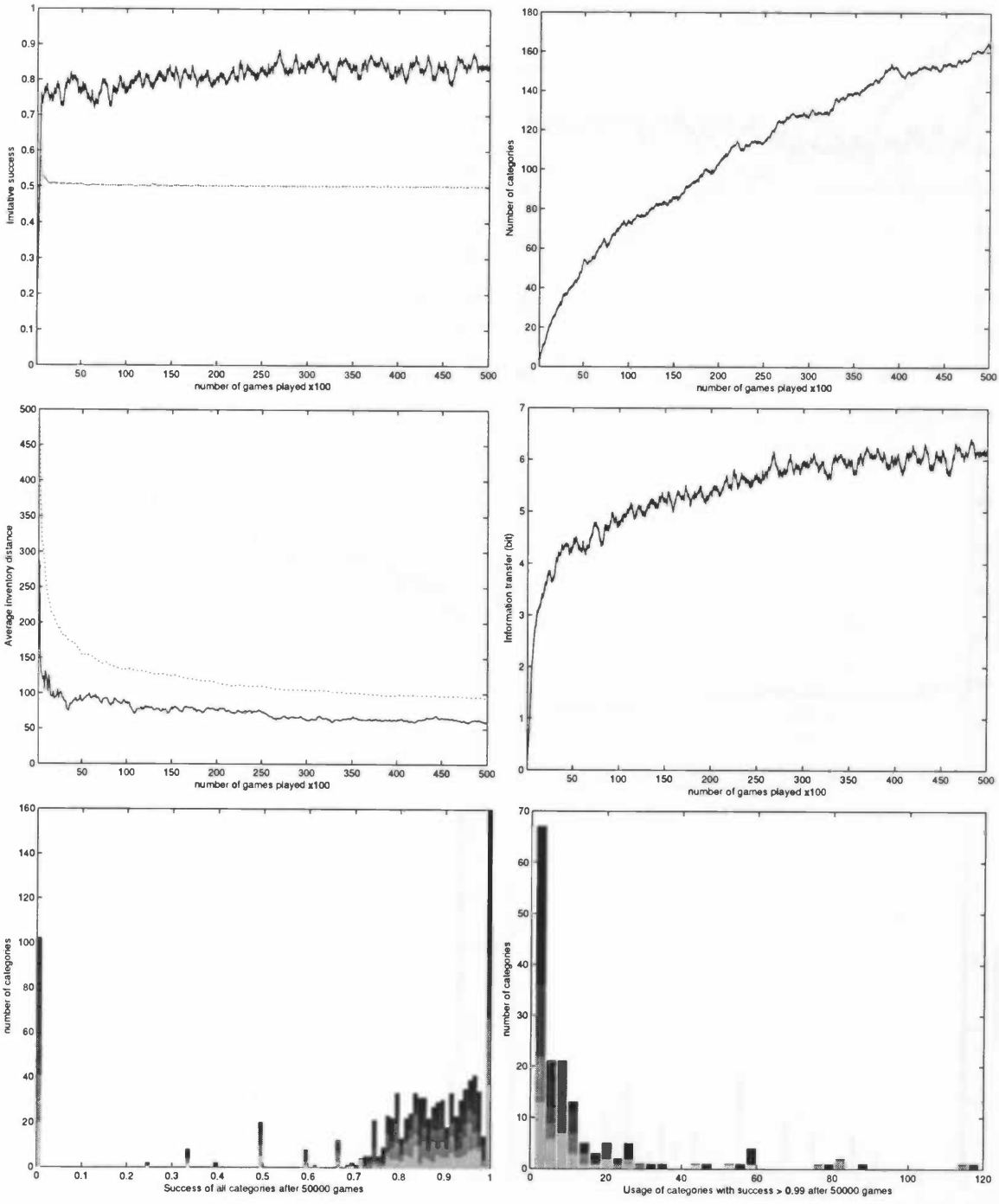


Figure A.4: 5 agents, noise = 20.0

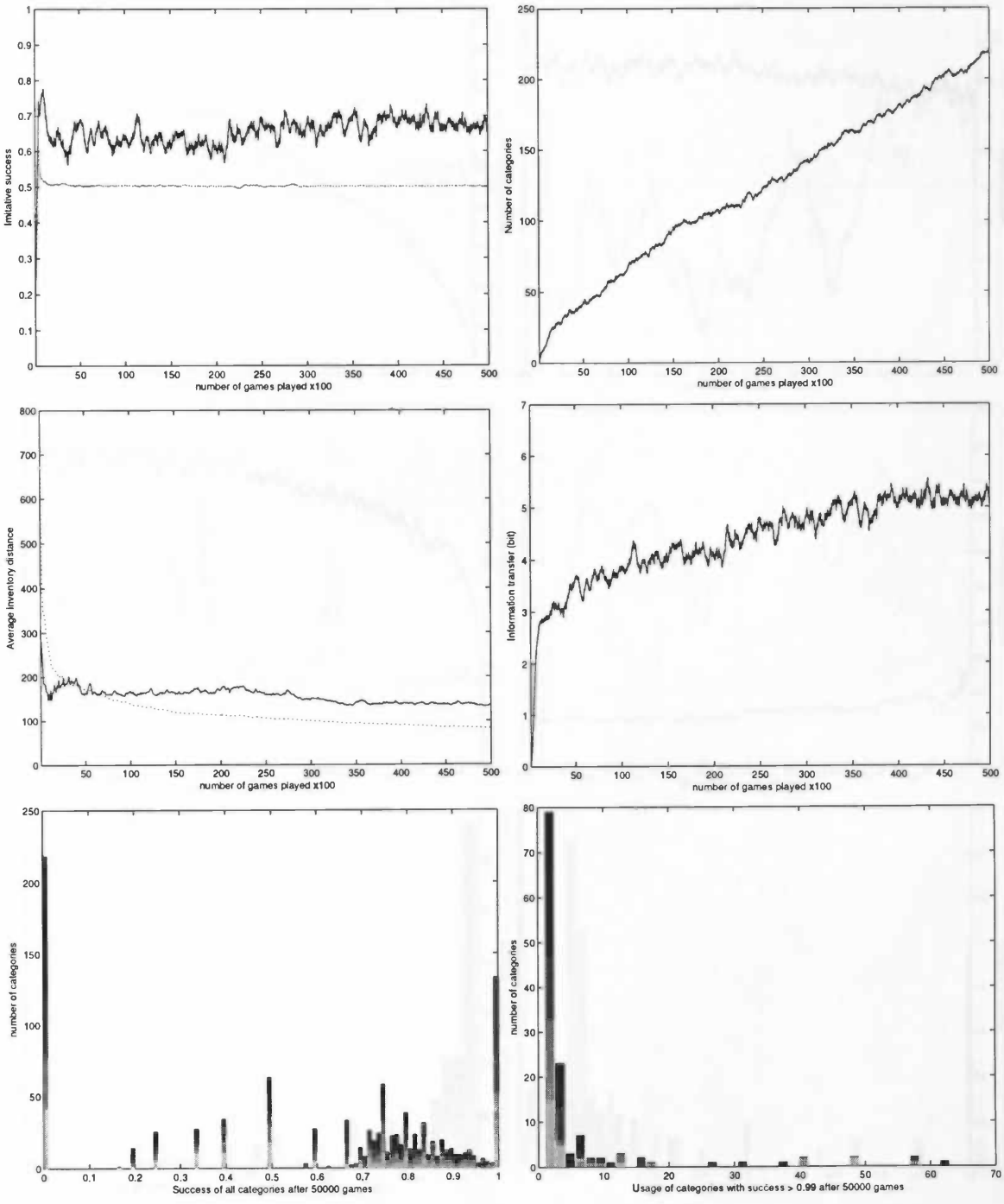


Figure A.5: 5 agents, noise = 30.0

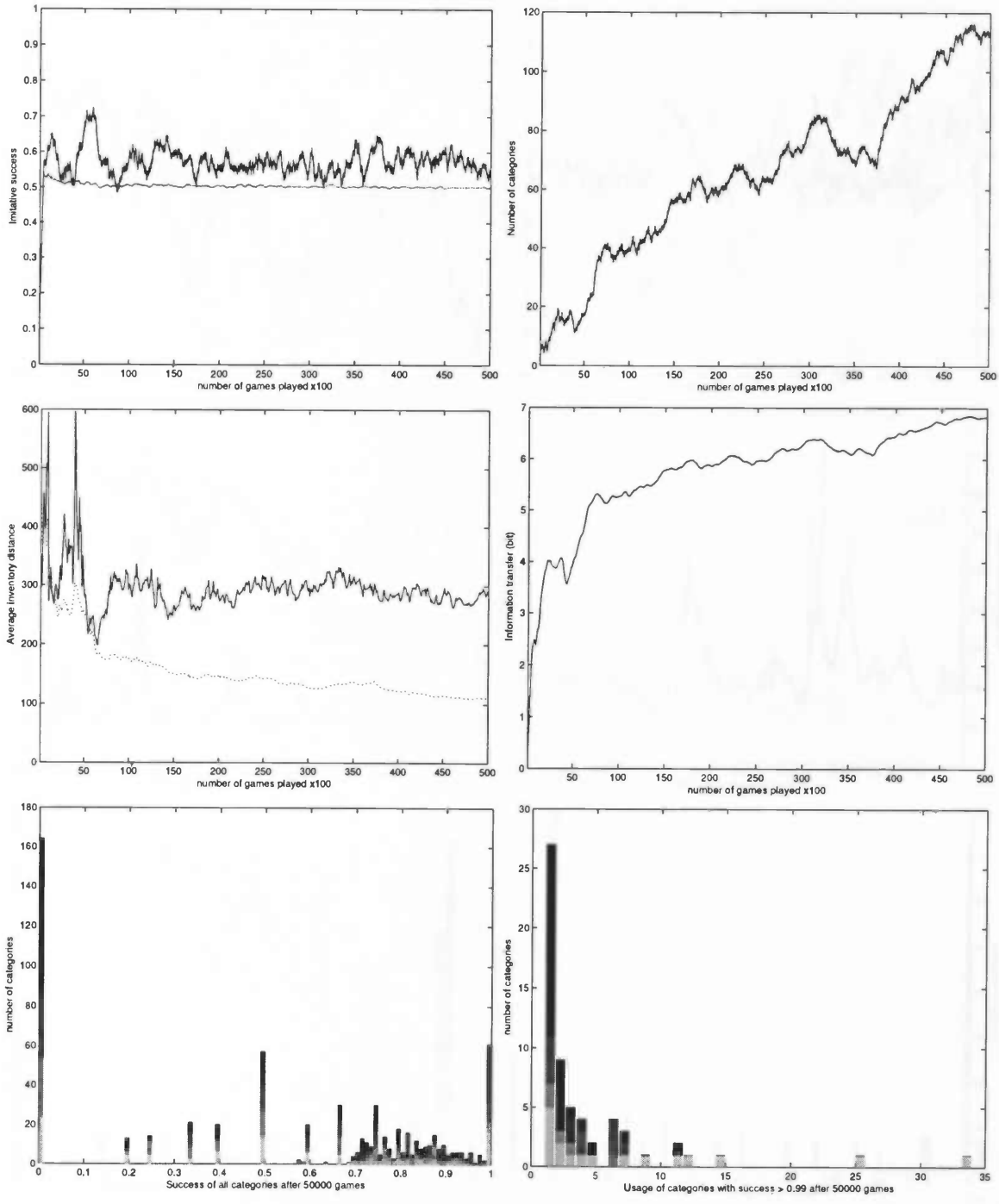


Figure A.6: 5 agents, noise = 40.0

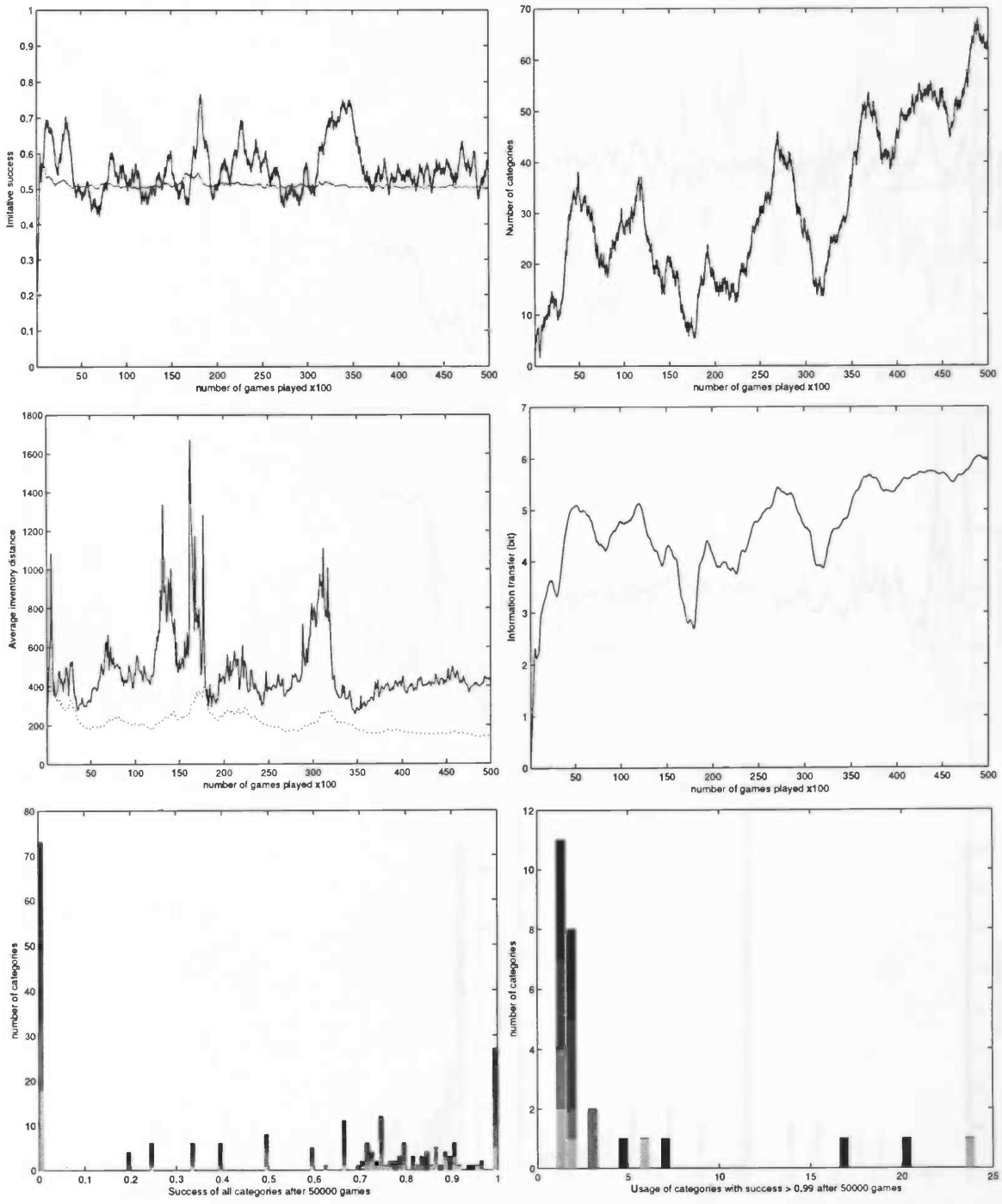


Figure A.7: 5 agents, noise = 50.0

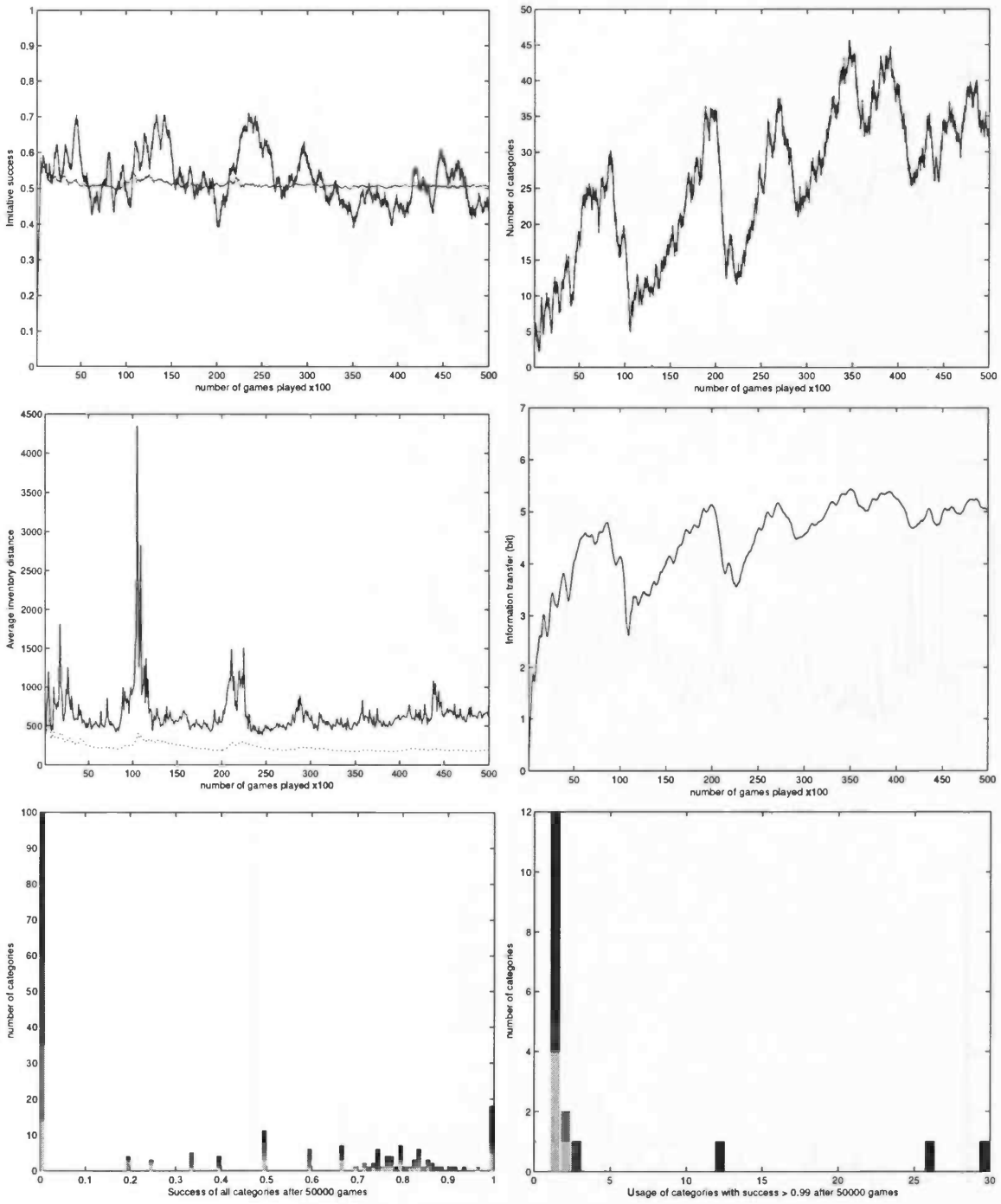


Figure A.8: 5 agents, noise = 60.0

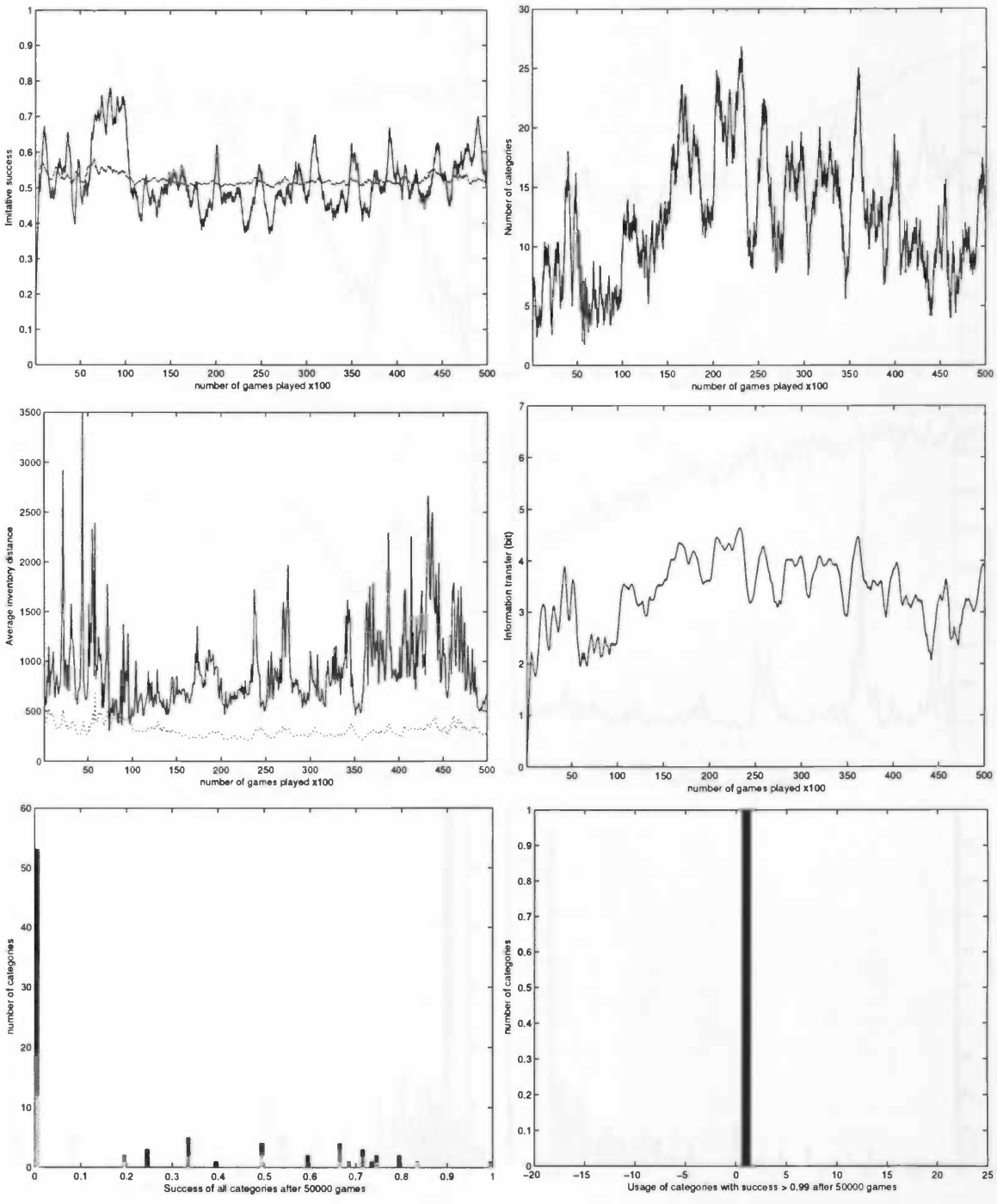


Figure A.9: 5 agents, noise = 70.0

A.3 Merging threshold

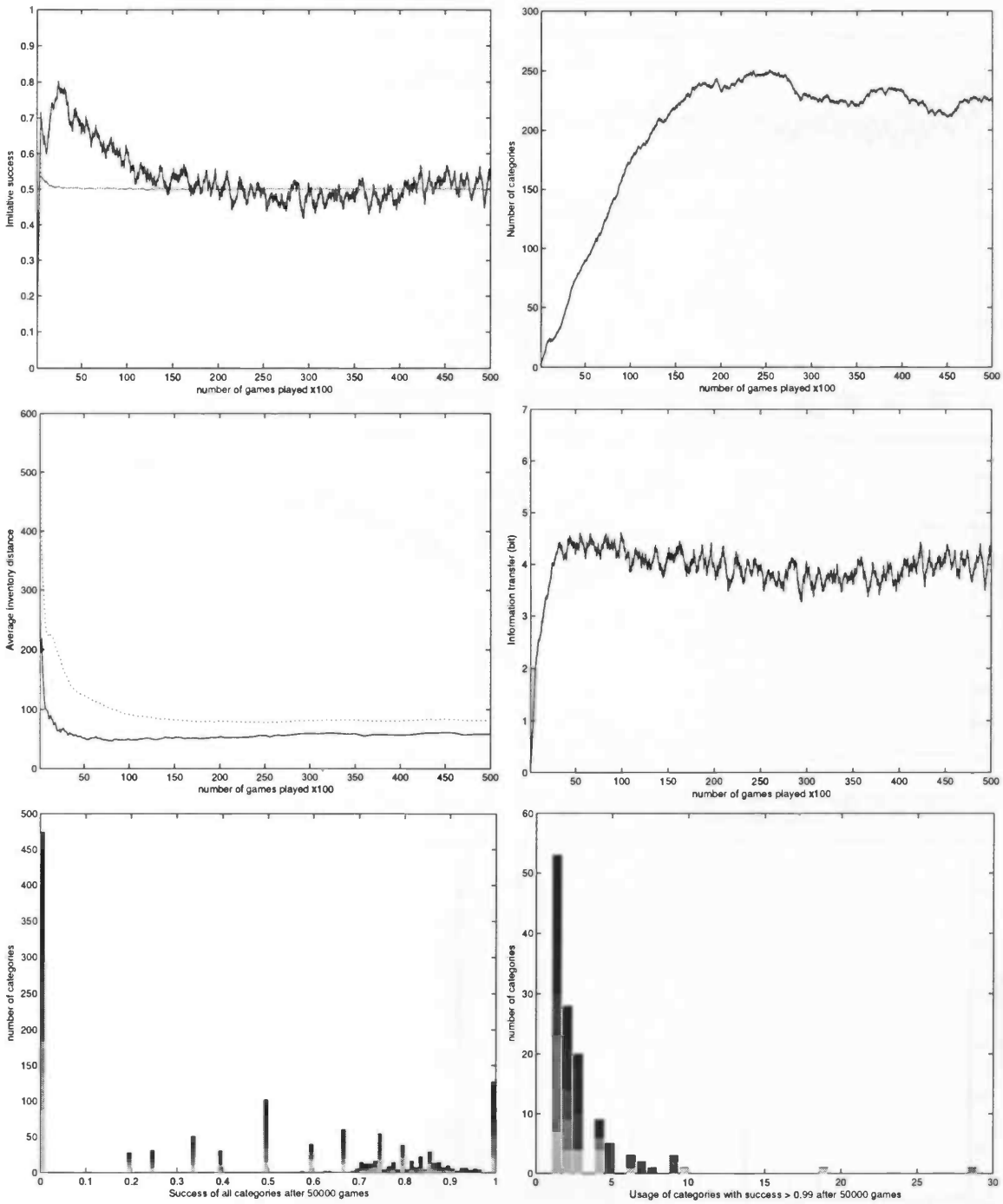


Figure A.10: 5 agents, merging threshold = 0

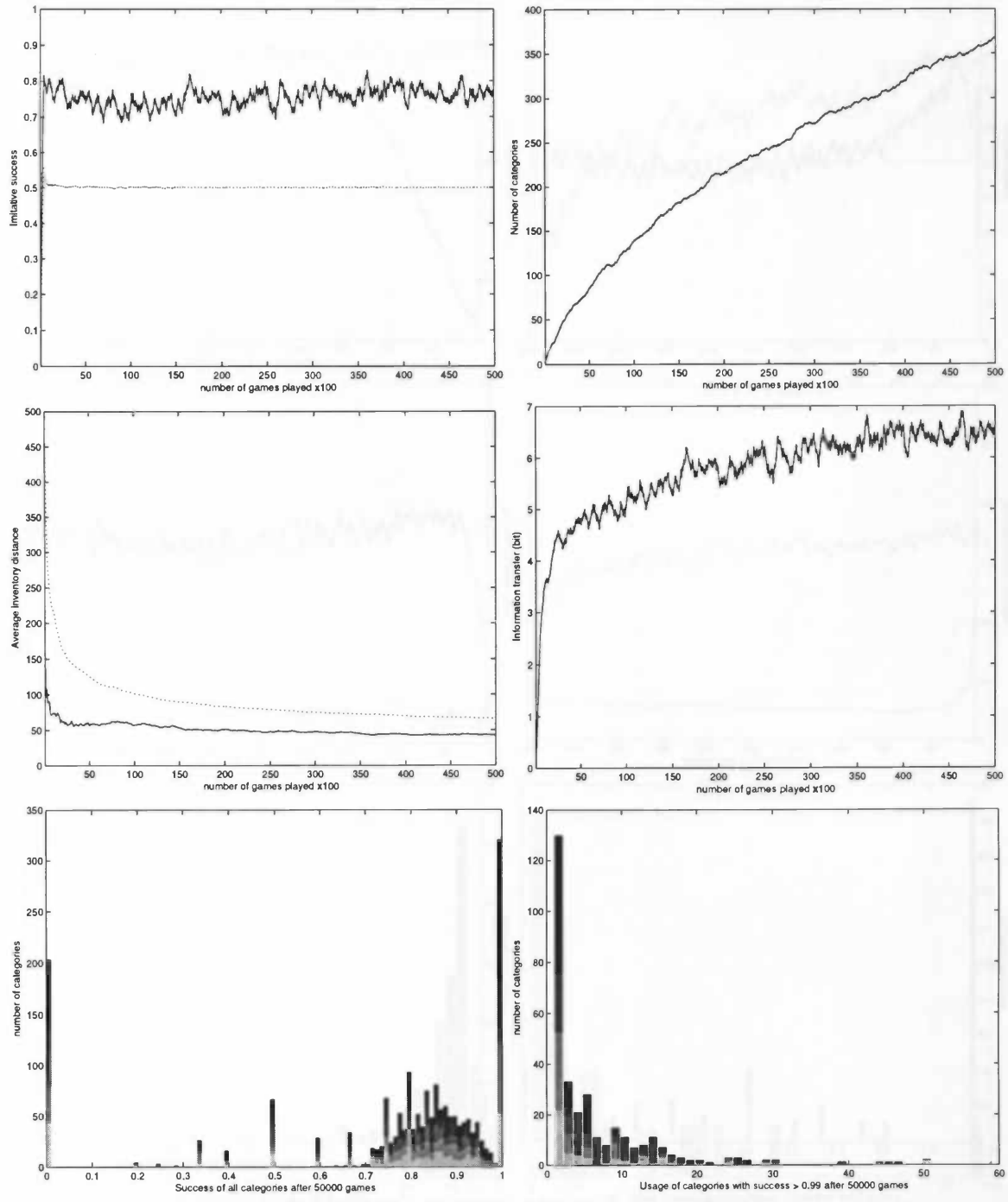


Figure A.11: 5 agents, merging threshold = 100

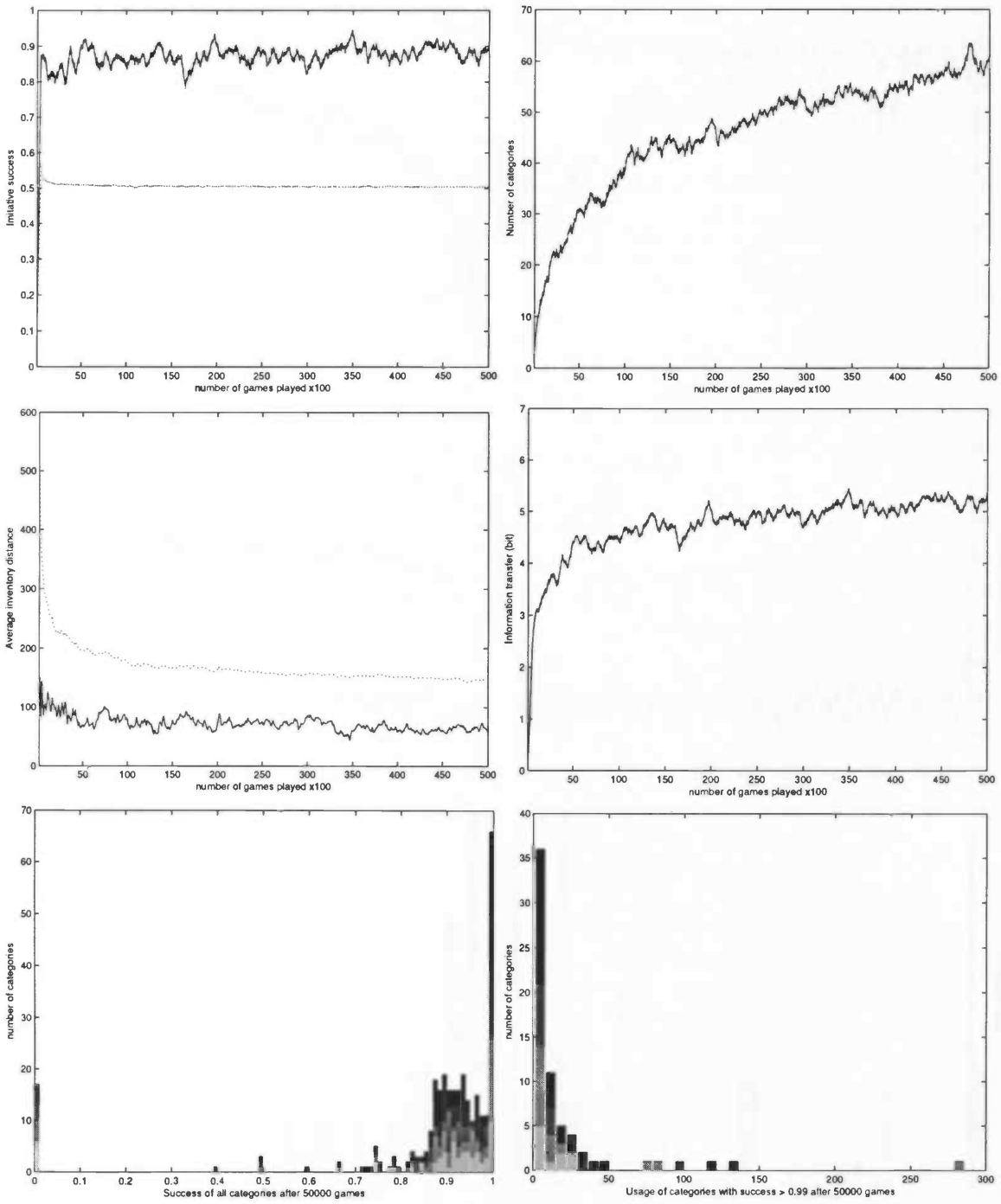


Figure A.12: 5 agents, merging threshold = 300

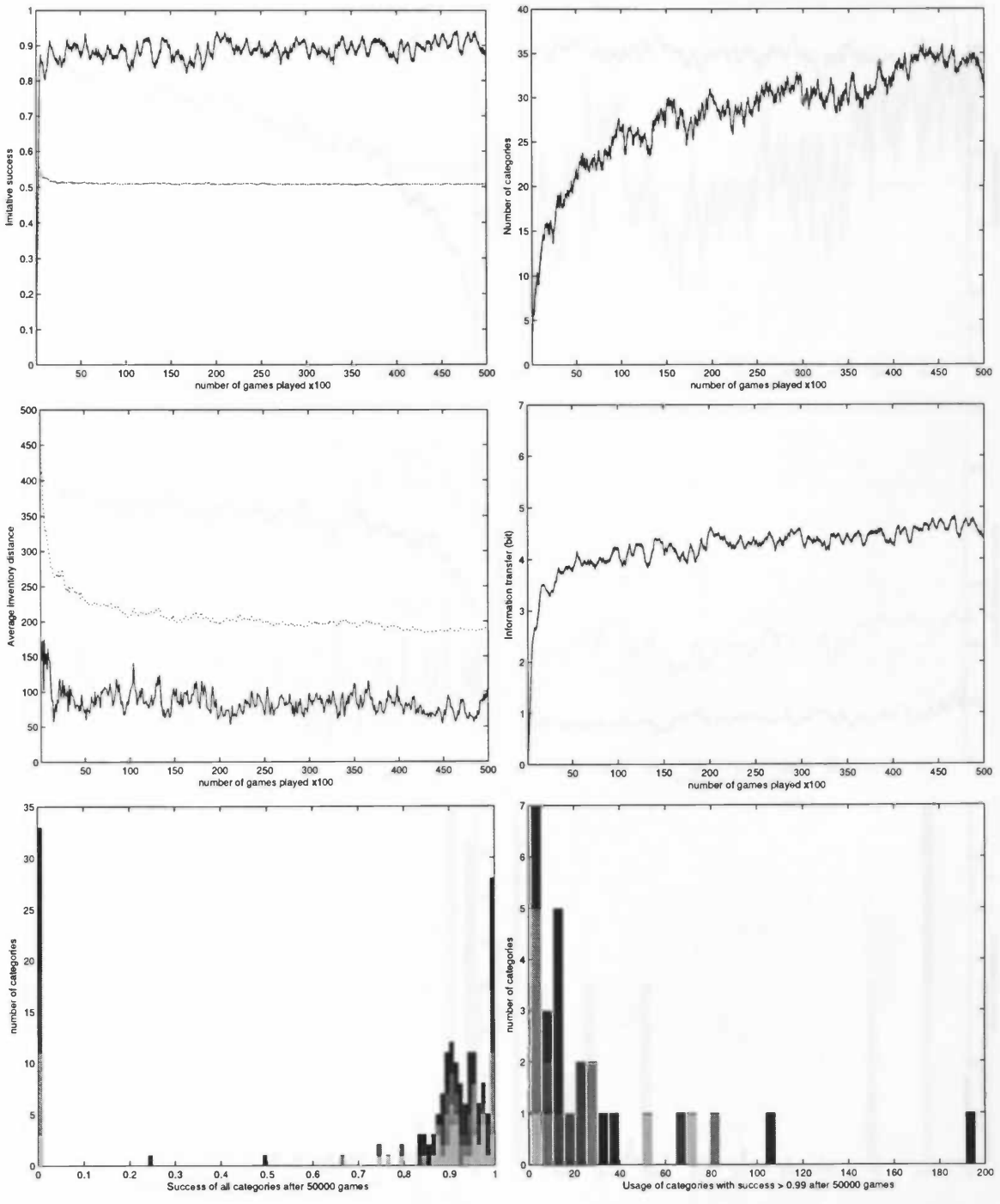


Figure A.13: 5 agents, merging threshold = 400

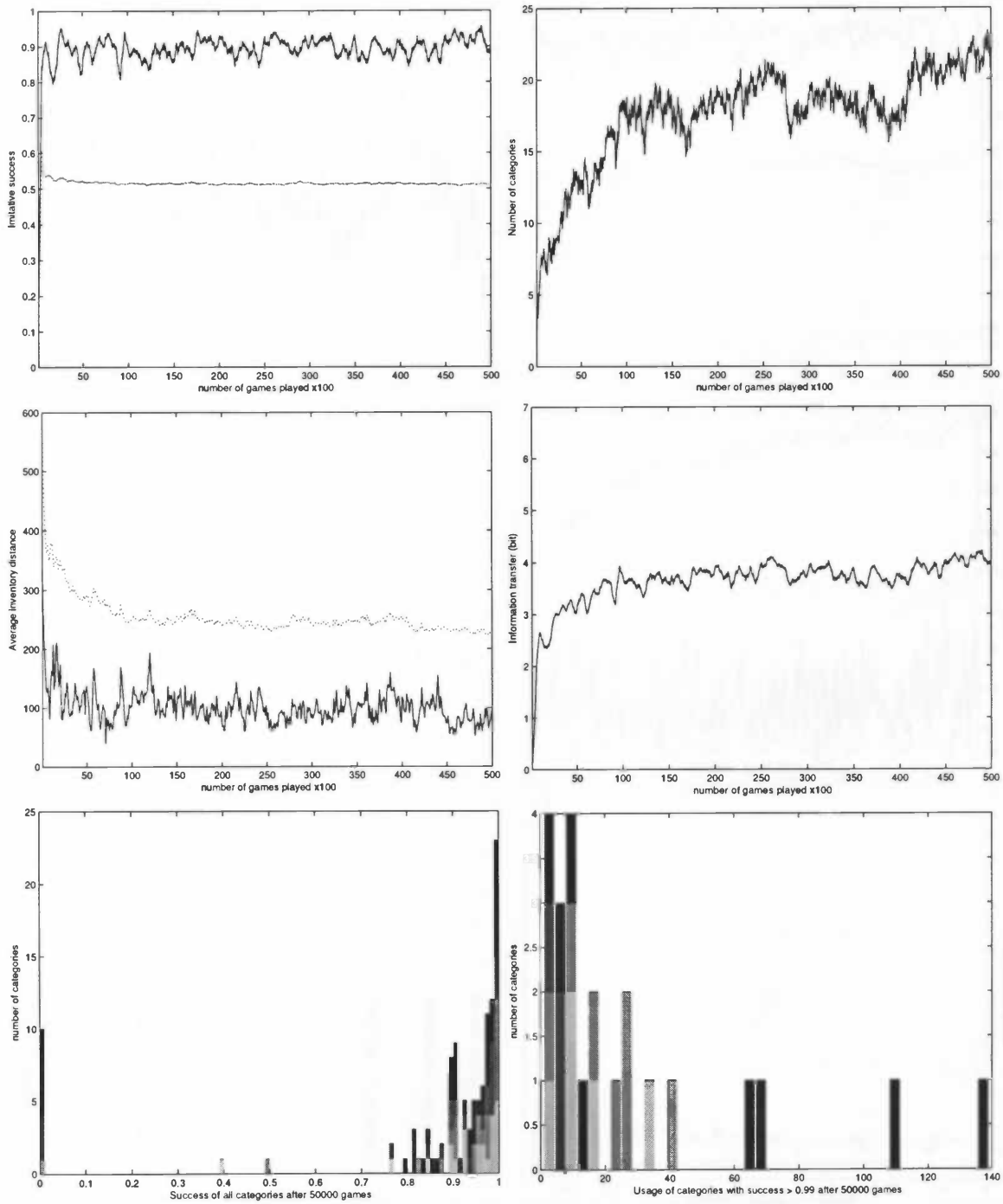


Figure A.14: 5 agents, merging threshold = 500

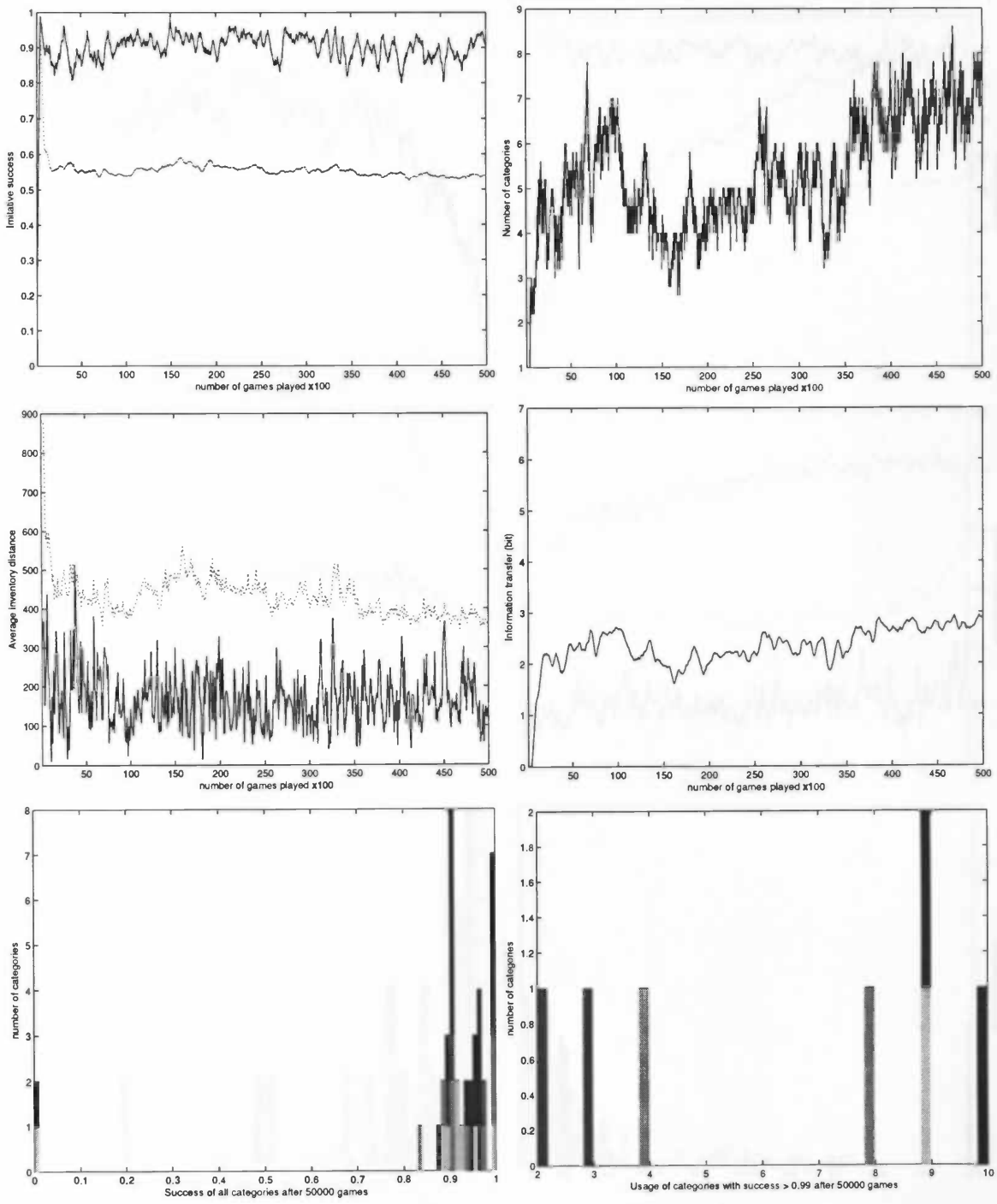


Figure A.15: 5 agents, merging threshold = 800

A.4 Shifting factor

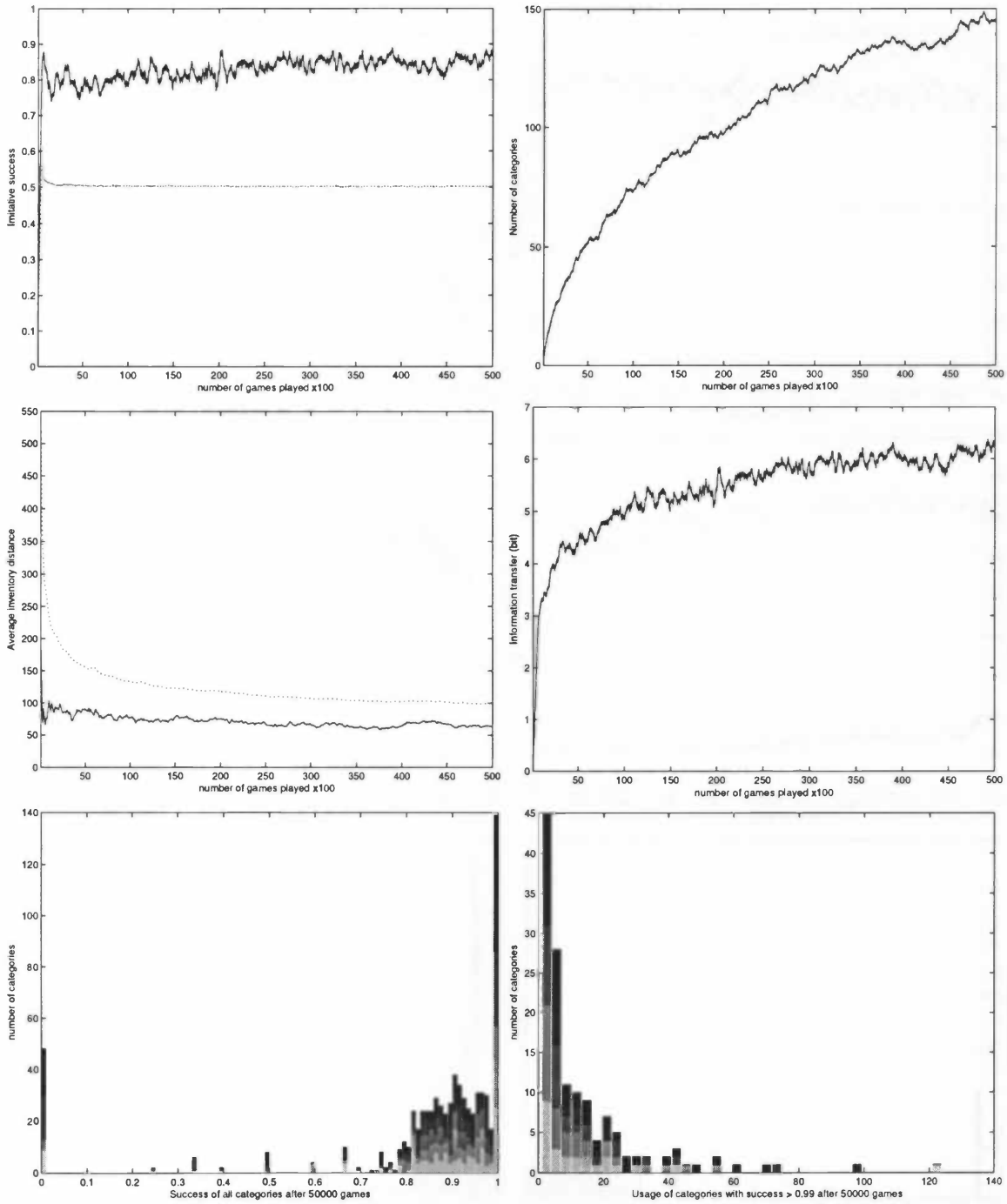


Figure A.16: 5 agents, shifting factor = 0.0

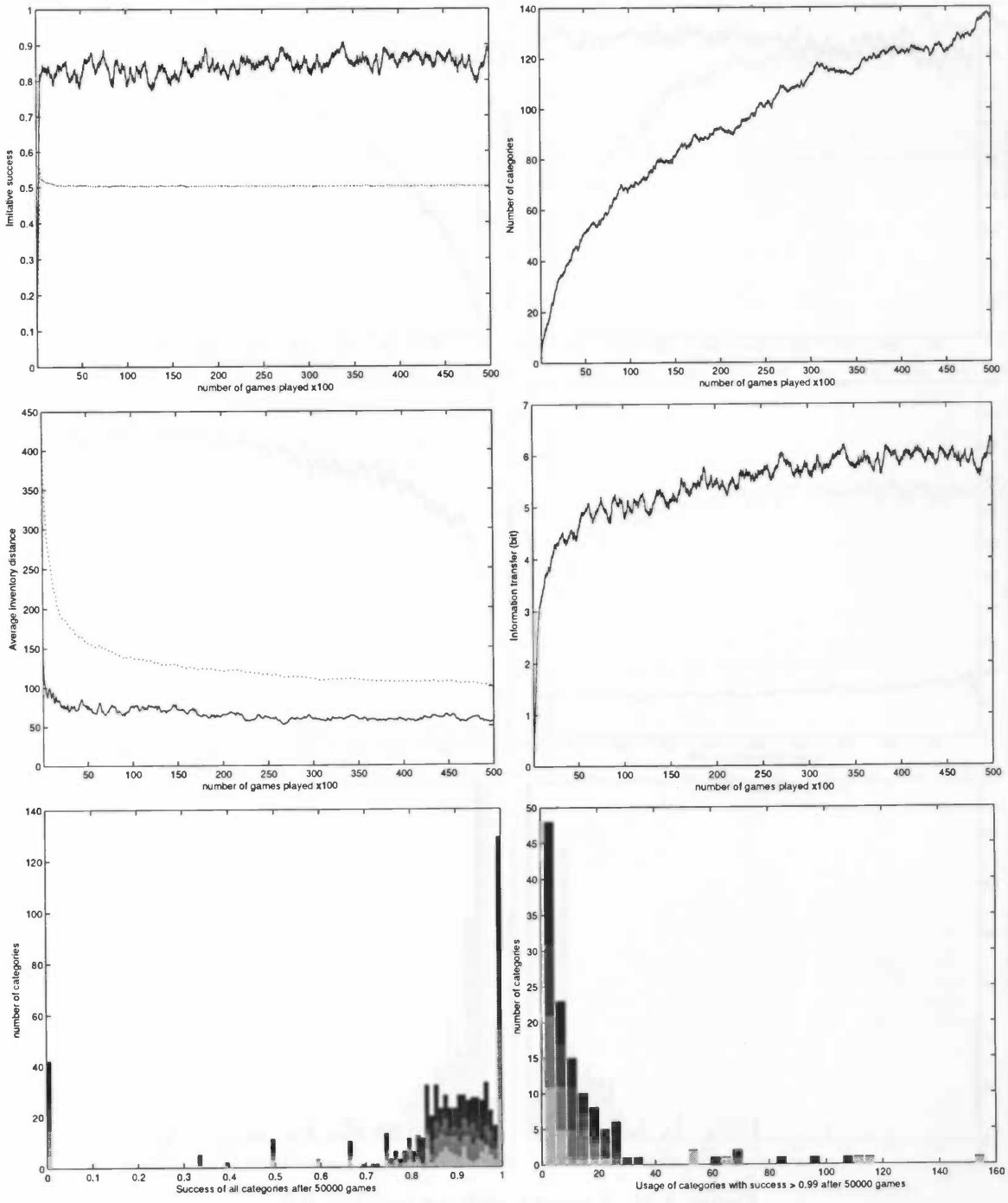


Figure A.17: 5 agents, shifting factor = 0.01

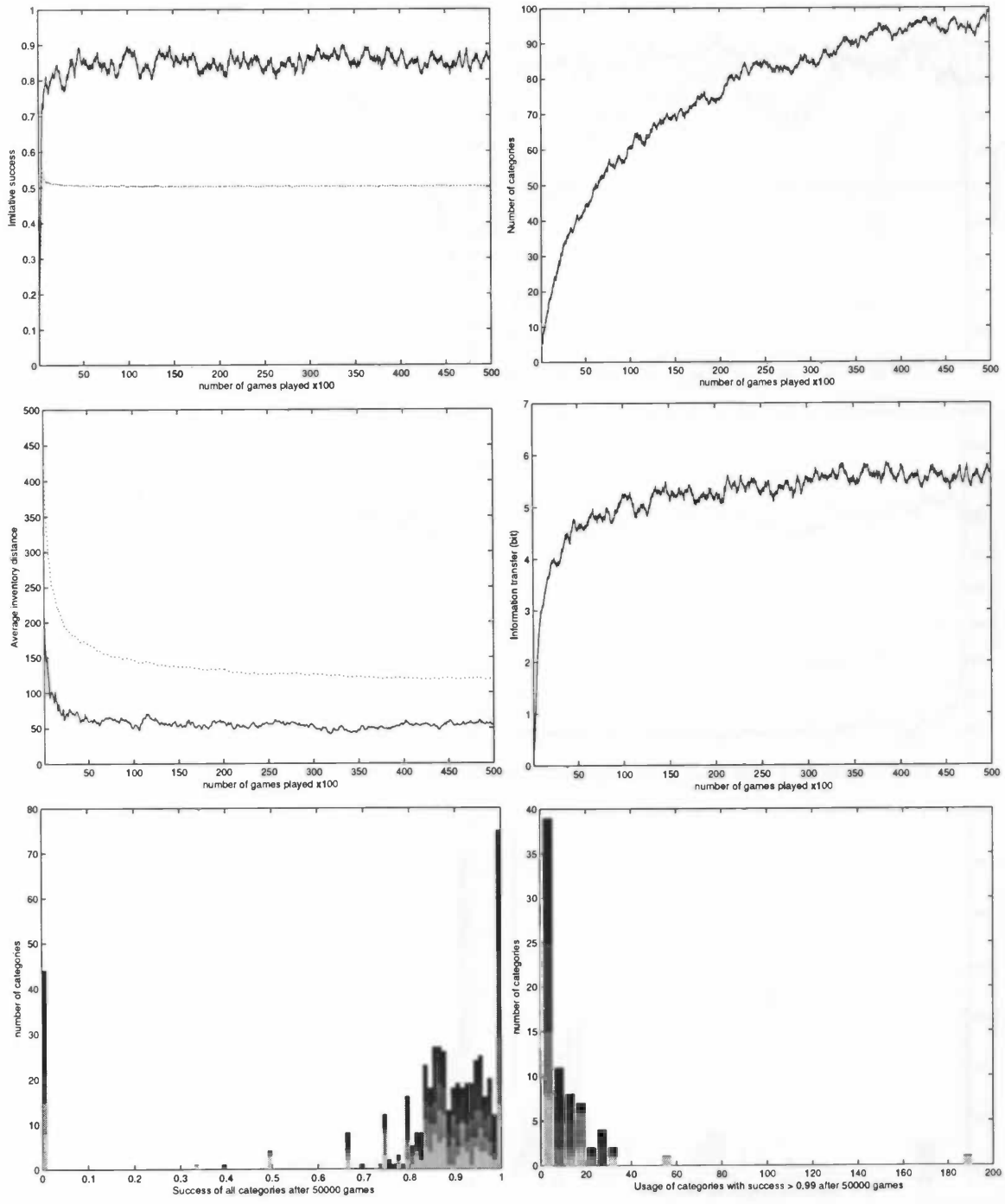


Figure A.18: 5 agents, shifting factor = 0.05

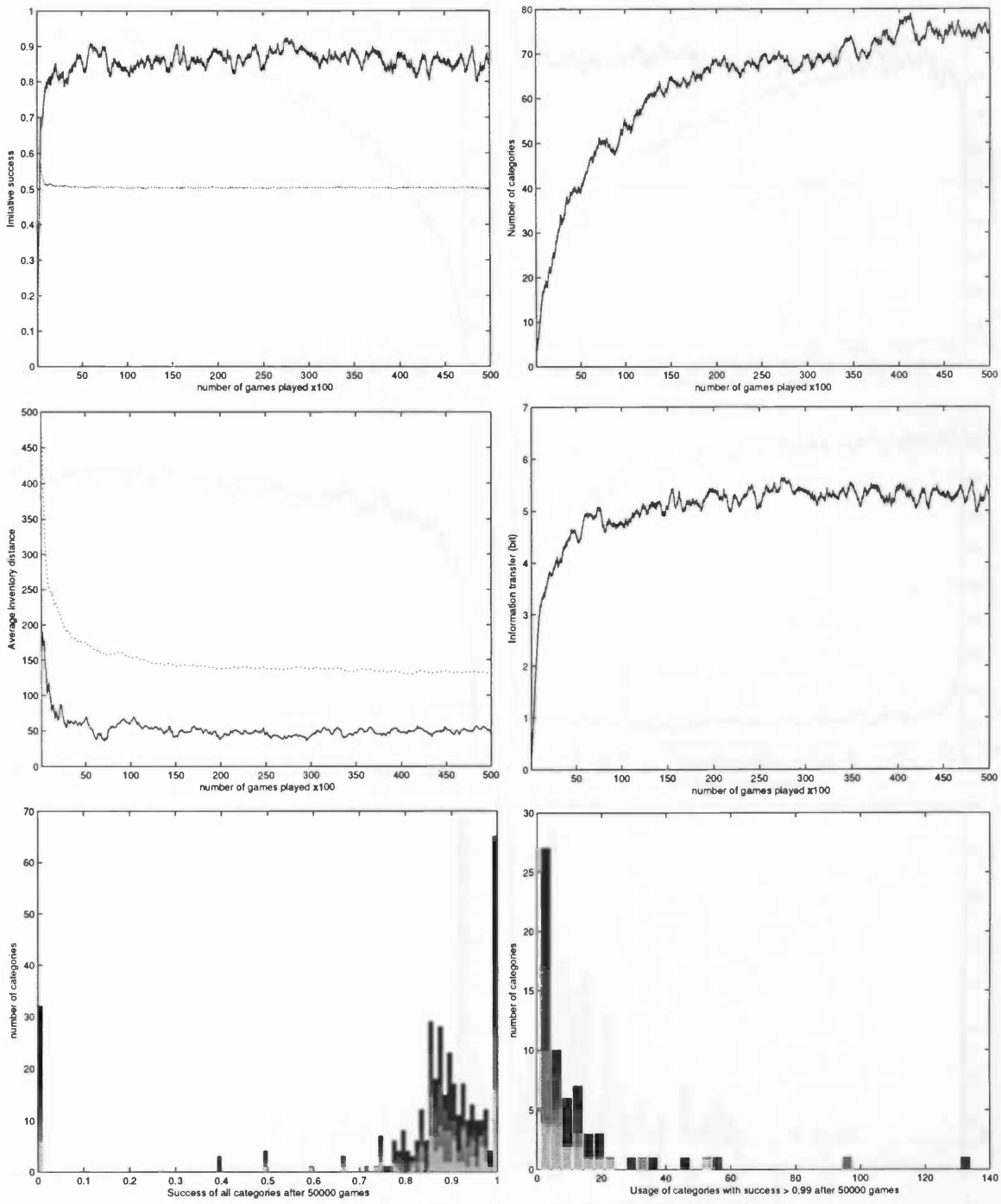


Figure A.19: 5 agents, shifting factor = 0.1

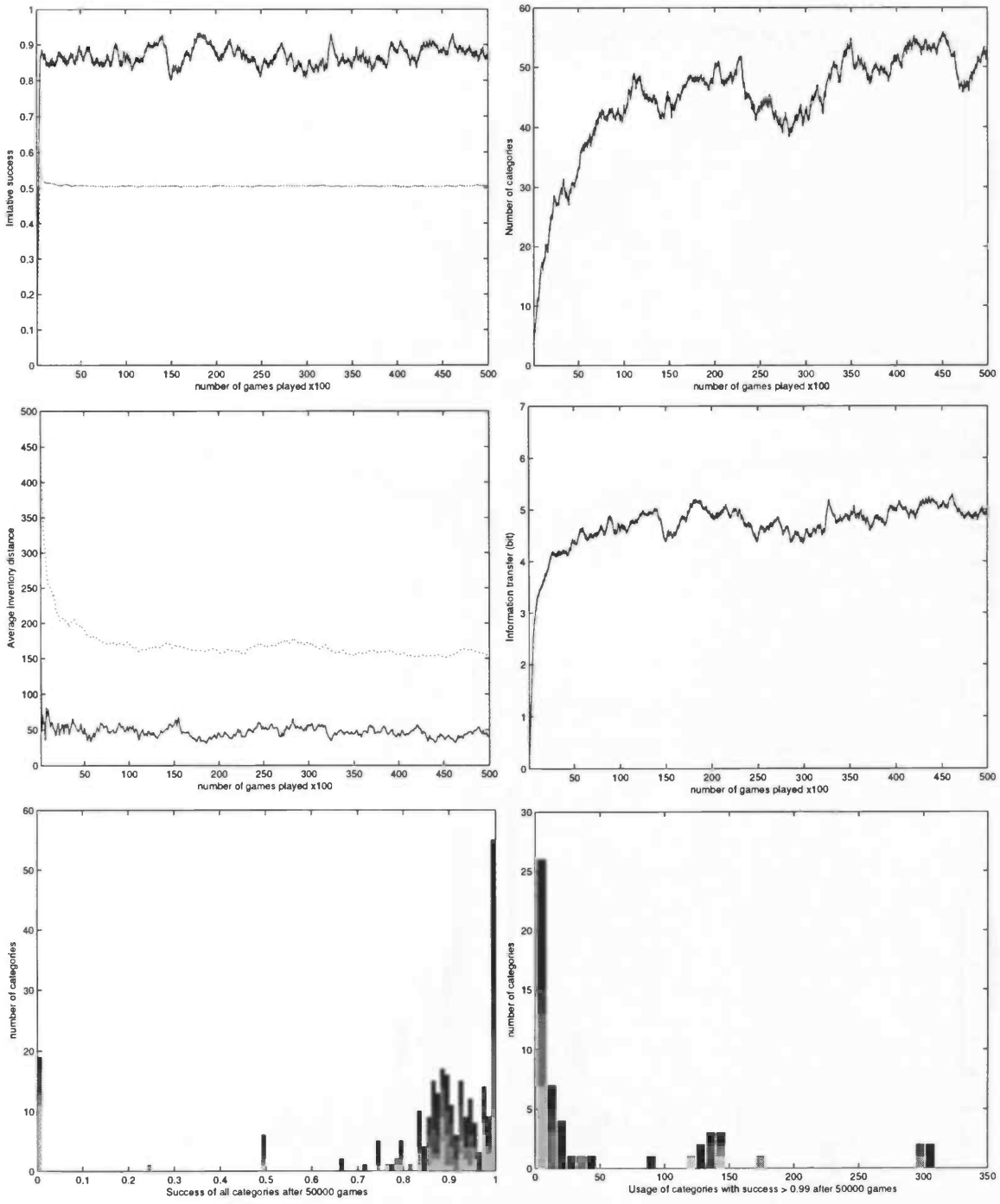


Figure A.20: 5 agents, shifting factor = 0.3

A.5 Throwaway threshold

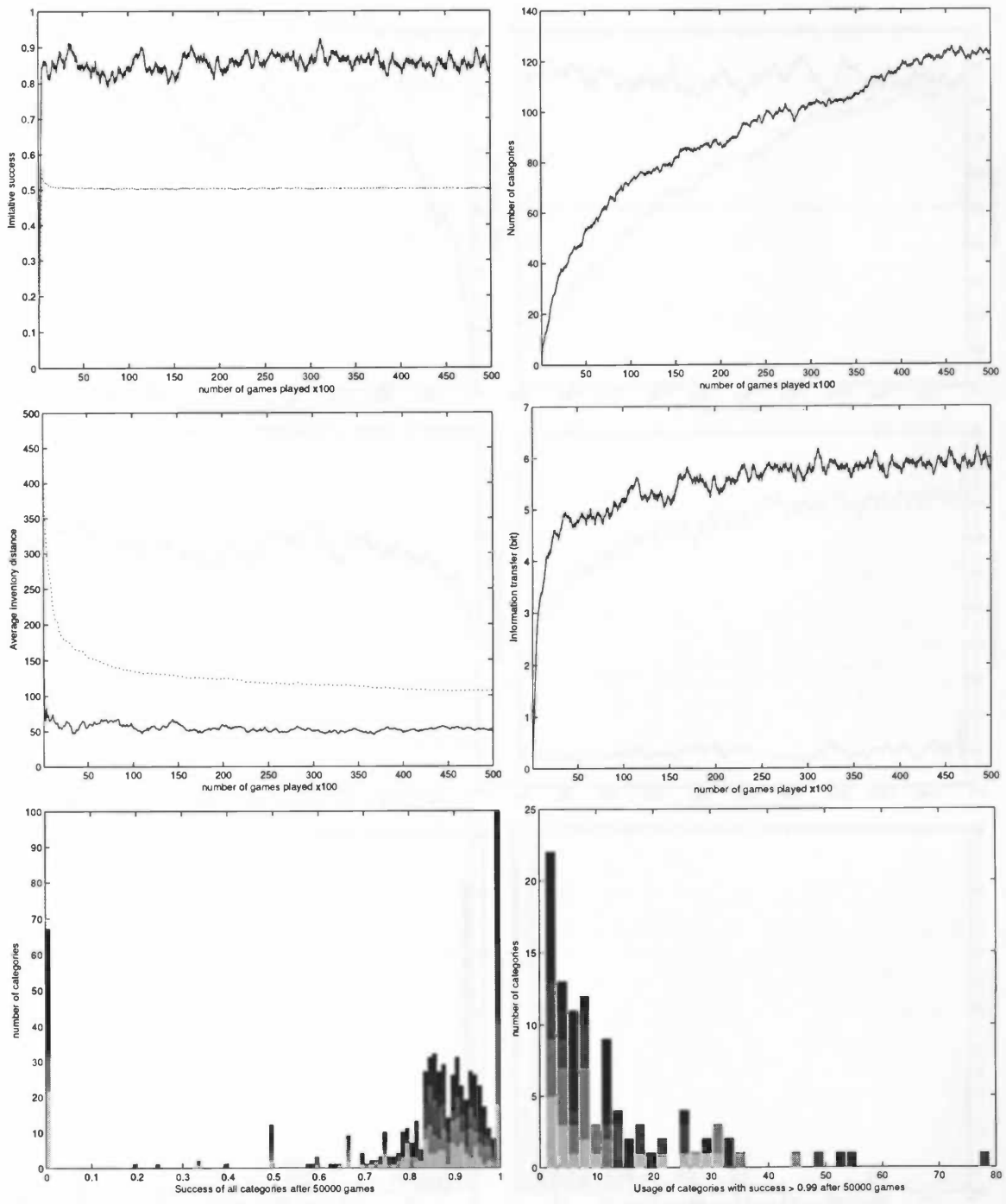


Figure A.21: 5 agents, throwaway threshold = 0.5

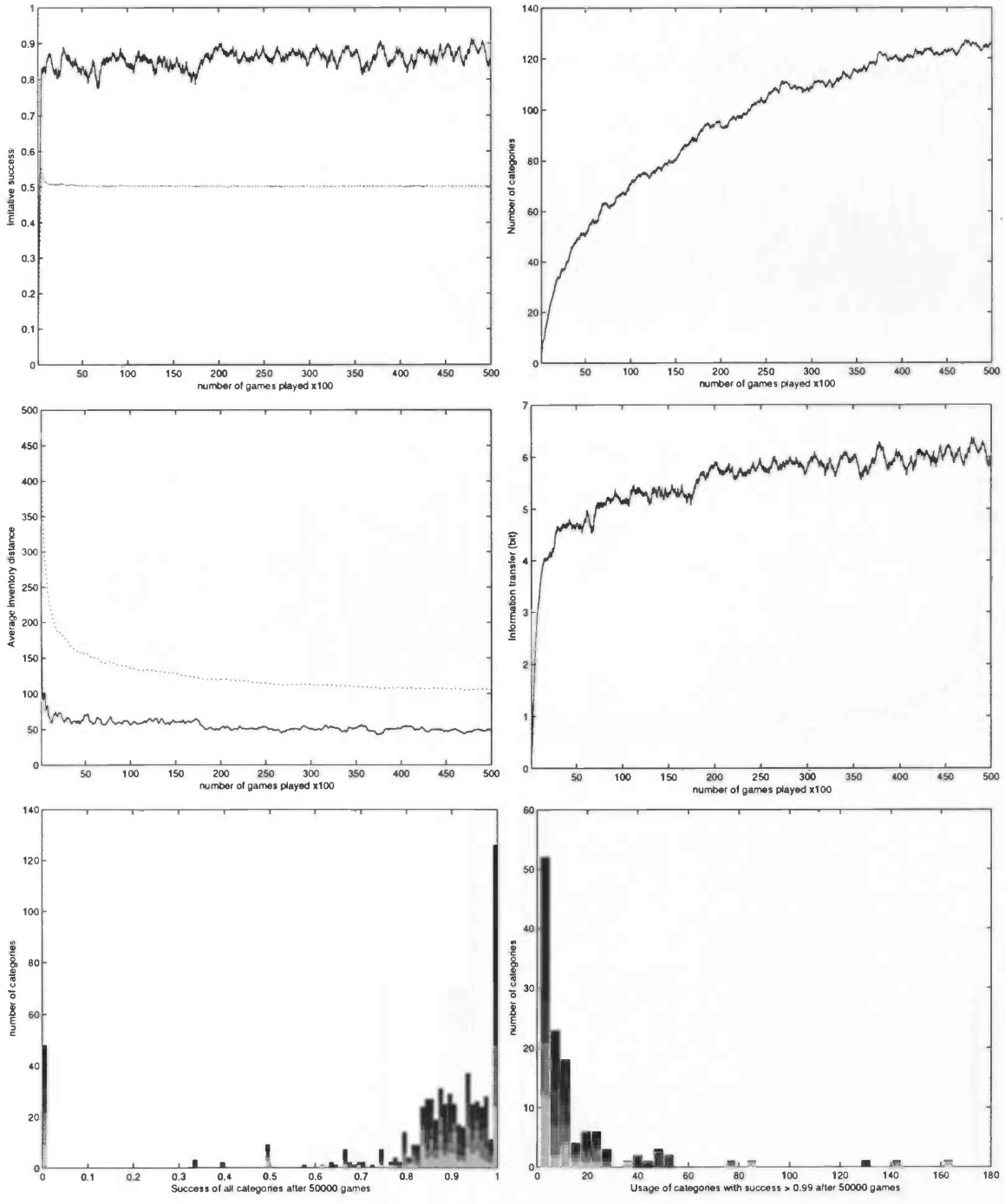


Figure A.22: 5 agents, throwaway threshold = 0.6

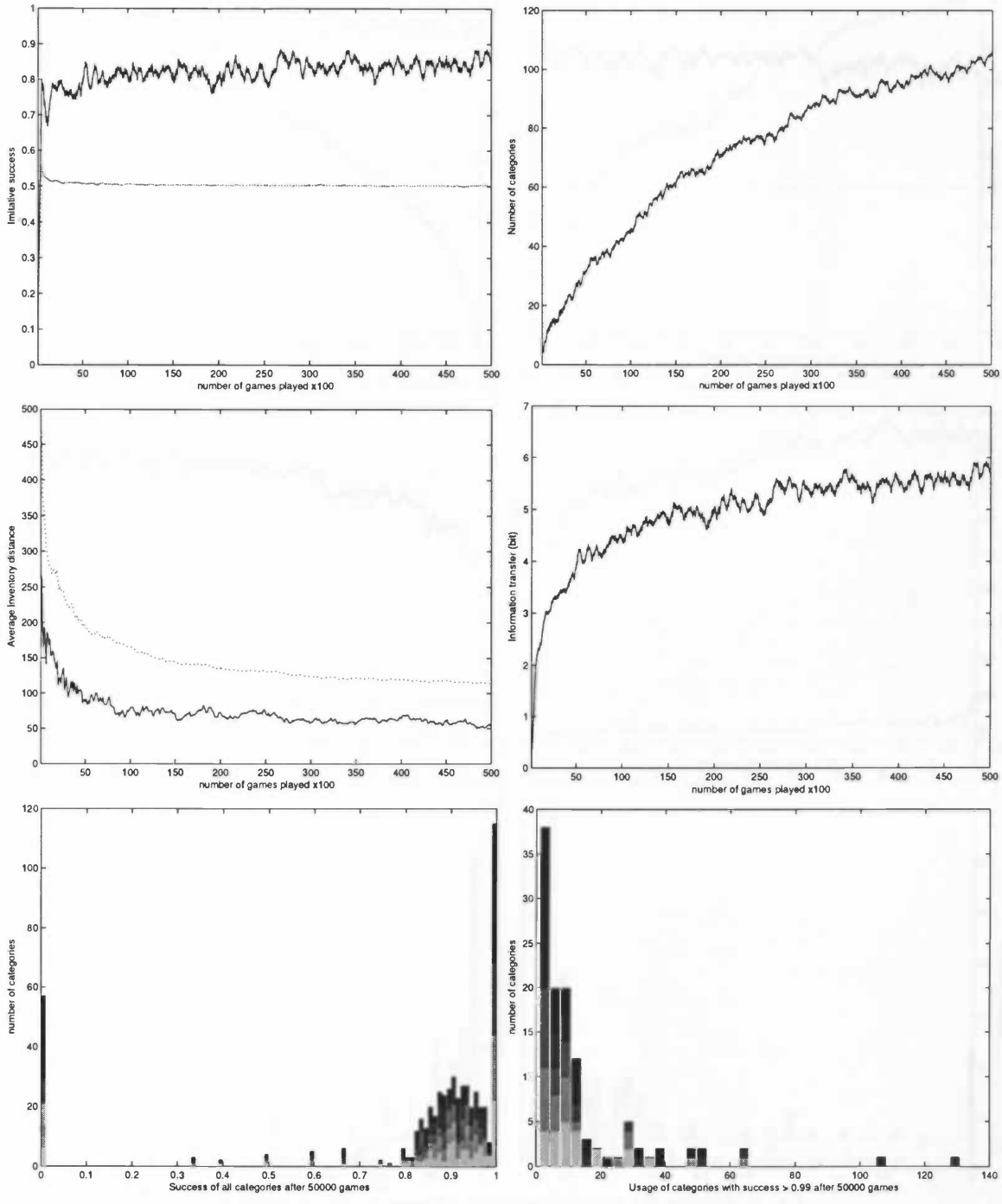


Figure A.23: 5 agents, throwaway threshold = 0.8

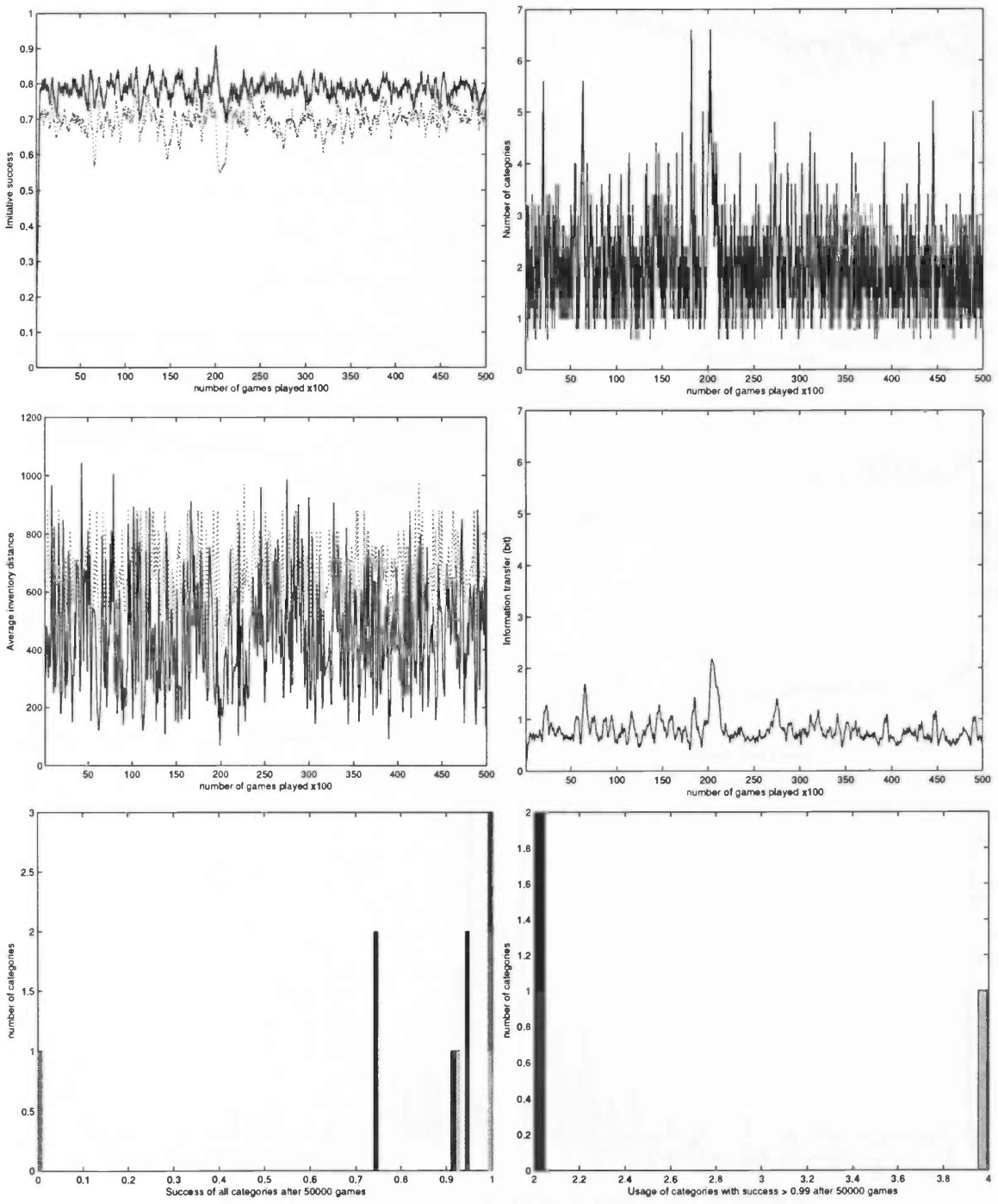


Figure A.24: 5 agents, throwaway threshold = 0.9

A.6 Successfulness threshold

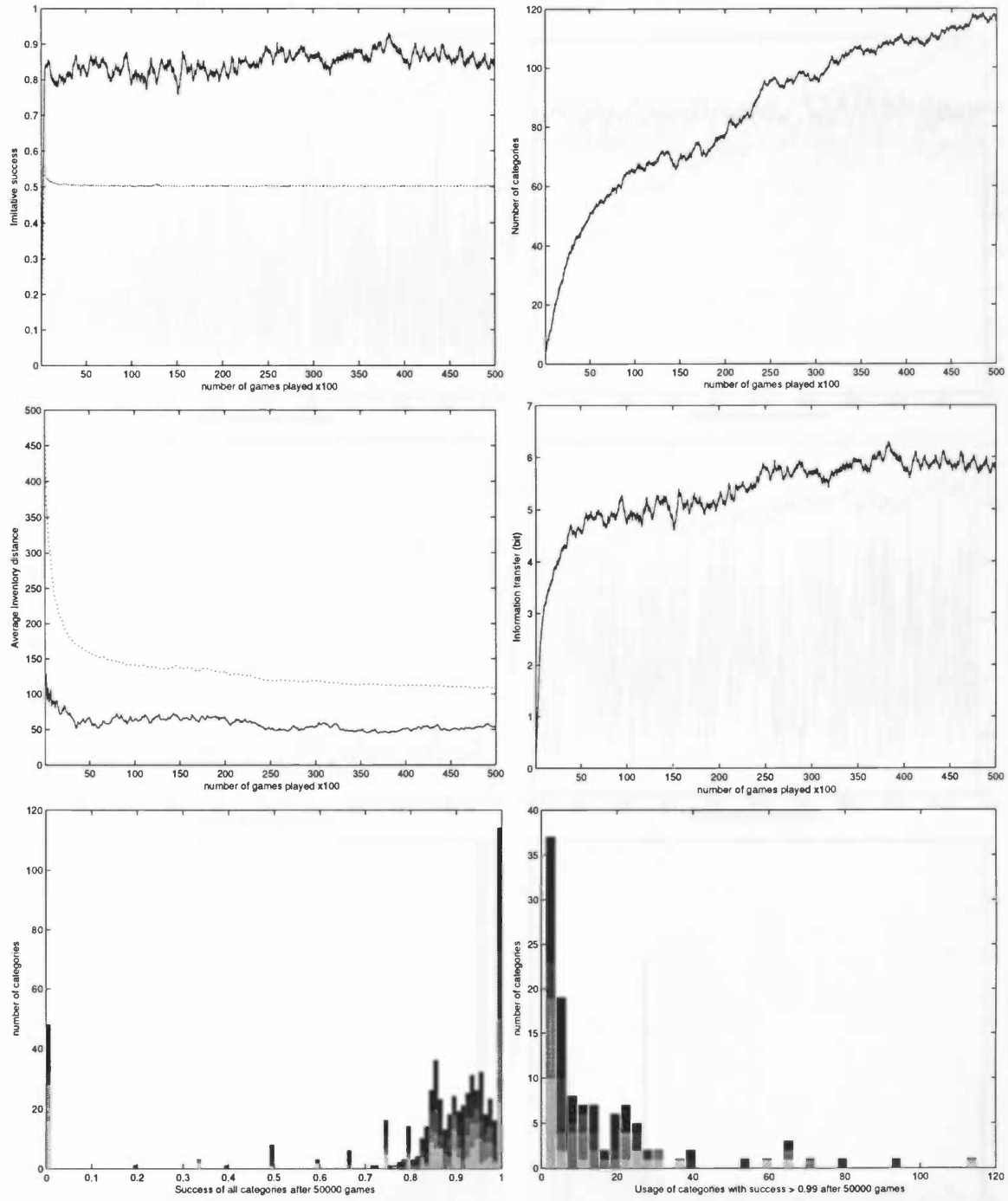


Figure A.25: 5 agents, successfulness threshold = 0.3

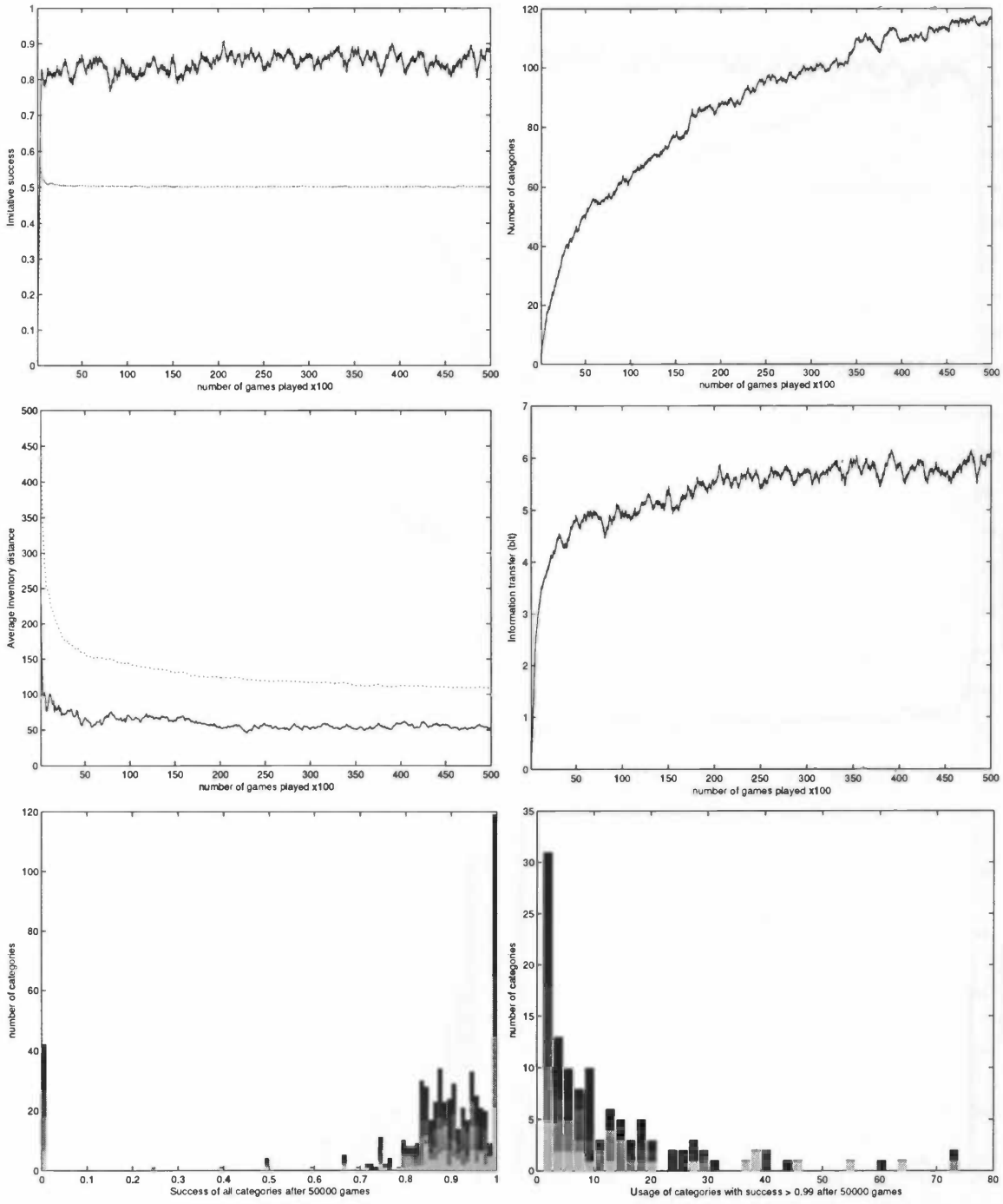


Figure A.26: 5 agents, successfulness threshold = 0.4

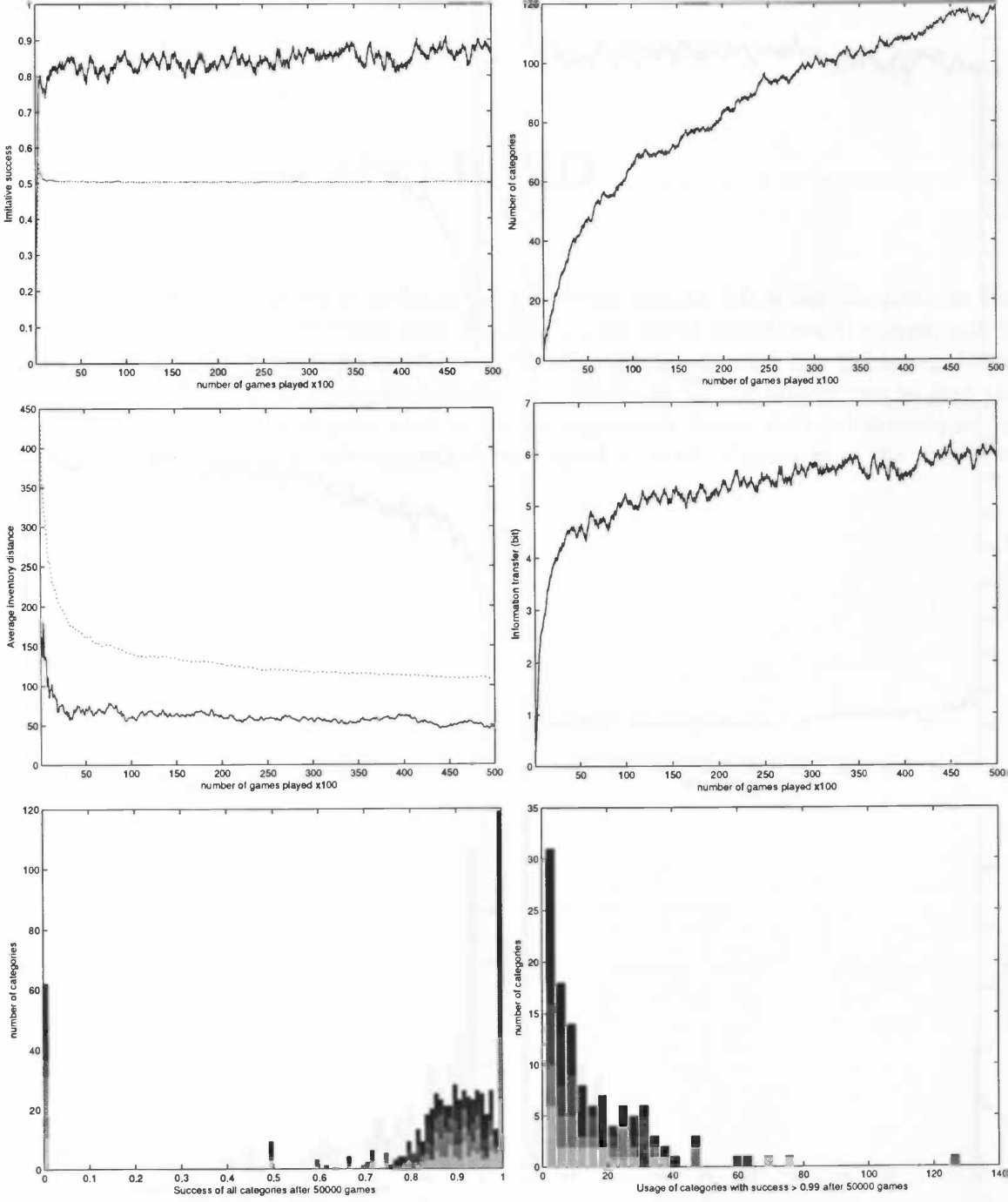


Figure A.27: 5 agents, successfulness threshold = 0.6

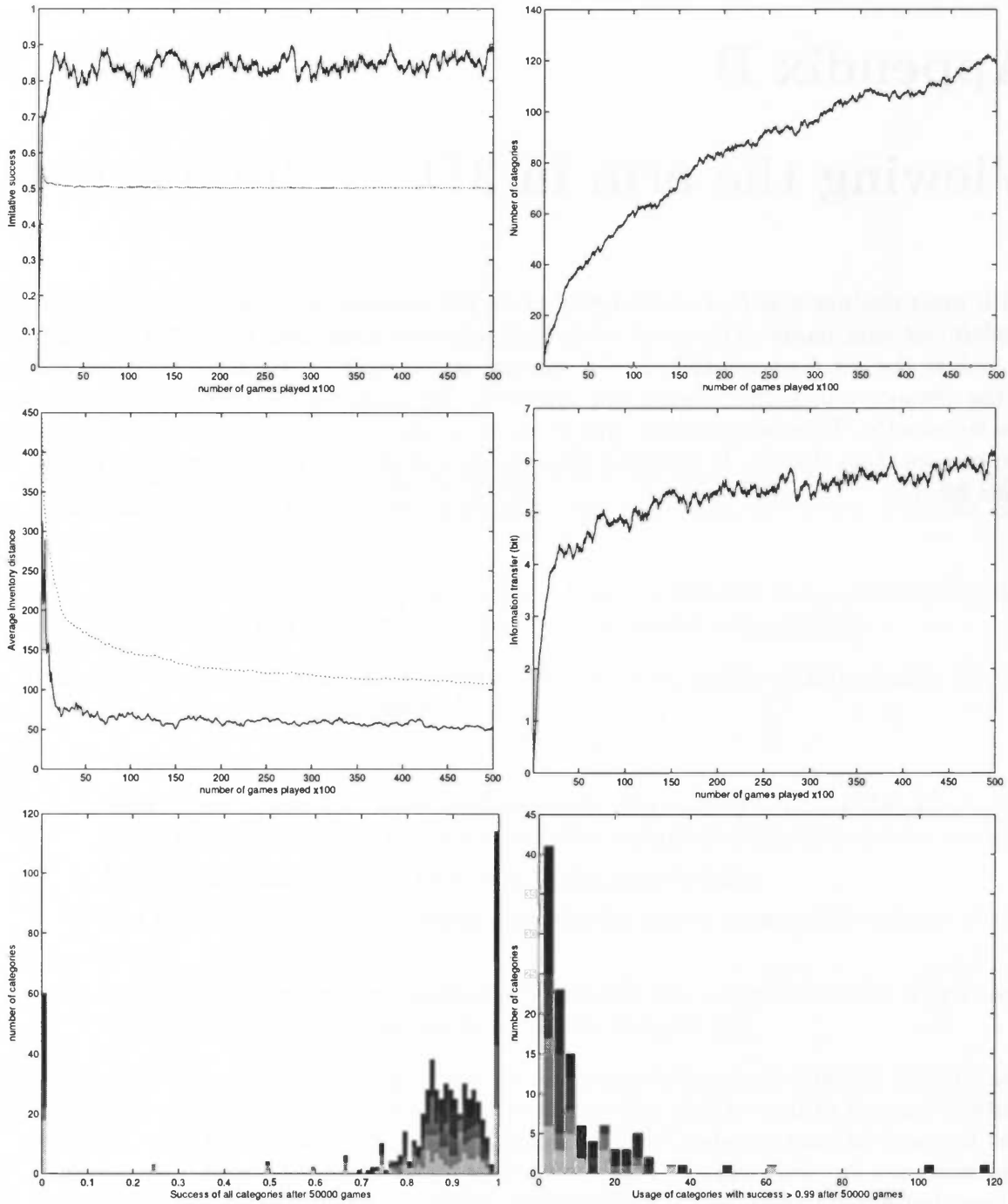


Figure A.28: 5 agents, successfulness threshold = 0.7

Appendix B

Viewing the arm in 3D

With some practice it is possible in figure 4.6 to see the arm in 3-D when one looks at both the left and right image at the same time. Hold your eyes about 5cm above the paper and try to look at the left diamond with the left eye and at the right one with the right eye. Gazing in the distance will result in seeing four diamonds. Try lining up the middle two so that they overlap exactly. Then keeping your eyes in the same position, focus on the diamonds so that you can see them sharply. If this succeeded, try to look at a set of pictures of the arm in the same manner.

Appendix C

Things done

When working with robots or implementing code, a lot of work is done for which there is no good place to report on them in a thesis, but that did take up a significant amount of time. Here is a list with some of the things that have eaten bits of my time:

- Learn lisp.
- Upgrade the robot setup, first by taping the base to the table, later on by making a base that can be attached to the table on which a robot can be put so that it is fixed but can easily be replaced.
- Fit the gripper with a bright red ball (several times because the arm smashing onto the table often caused loss of the ball and resulting failure of the experiment).
- Face the problems of operating a mechanical robot arm, luckily with invaluable help of 'the *Barts*'. These hardships included:
  - Jammed motors
  - General smellyness that could fortunately be remedied by utilizing the fan present in the lab and giving the arm some rest after a number of games had been played
  - Broken controllers that had to be sent to the manufacturer
  - The manufacturer sending back a controller for a new incompatible version of the arm
  - A working controller that mysteriously inverted one motor's direction which took quite some time and a keen eye from BartdV to figure out
- Find out why captures from the camera do not agree with expected result. This turned out to be caused by a mysterious undocumented buffer that seemed to be present when working with the camera. An obtained image effectively always was the image of two captures ago. In order to test if this realy was the case, a large counter on a monitor was recorded at several time intervals. To prevent this effect from causing unexpected results, a variable is kept in the read-out code that registers how long ago the previous capture was made. If this is longer than some small interval ago, two captures are made of which the results are ignored, then a normal capture is performed and used for gripper-detection.

- Find the error we had made that caused a segmentation violation for which the Konolige-black-box really should have given a warning. Without the help of Peter Stuer there is the possibility that I still would be looking for it.
- Implement capability to write captured images to a file.
- Implement capability to view any desired captured image and turn each one on and off while the experiment is running.
- Build in extensive logging capabilities for the state of agent's inventories and outcome of the games. This has been done in such a way that data is written to disc after every game and that an experiment can continue after it has crashed or otherwise failed to complete. Different types of logging can be turned on and off while the experiment is running. All parameters that influence the systems are also written into the log-file so that it is easy to check later on how a logfile was produced.
- Introduce a CVS system for the experiment's codetree in order to be able to work on the project together with other members.
- Implement platform detection in lisp and rewrite code that runs the experiment to automatically use the correct paths for the current platform.
- Build a simple lisp parser that can interpret the log of the agent's inventories and that produces matlab code that creates a data structure that can be used to plot the categories in space (e.g. figure 6.5). From this data structure, the matlab code can also produce a movie of the development of categories. This has proved very useful to gain insight into the inventories of agents, especially when things were still going wrong.
- Plot trajectory of gripper in space compared to the expected positions to check if movement and detection of arm works as expected.
- During the experiment with the physical setup, store all observations with a corresponding position of the arm and use this data for the error analysis of the setup.
- Define a class for a body that can use it's calibration matrix and robot arm properties to translate any point in any space to another space automatically.
- Improve the merging algorithm in such a way that only categories that have changed are considered for merging instead of considering every pair of categories. This is equivalent to the old method and has improved the running time of the simulation greatly especially when inventories got large.
- Write matlab code to give insight into the quality of a calibration.
- Write matlab code to plot many results of imitation games and save them to files.
- Implement a web-based BibTex database in which the BibTex files of the members of the AI-Lab have been joined to easily re-use literature references.

- Attempt to build a client-server architecture that can make agents 'hop' between setups on different computers very much like with the Talking Heads. The main reason for this was that I wanted to implement an ACT-R version of an agent, but with the current version of ACT-R it was not possible to implement multiple agents in one lisp-process. This would be solved by a hopping architecture because every agent could then be a separate process that connects to the server. Another benefit of such an architecture would be that multiple setups can be used to run an experiment in parallel. Unfortunately it was not possible to finish this architecture in the available time-frame. I have been informed that the next version of ACT-R will allow multiple agents.
- Fabricate a red/green image of the stereo images from the camera and the glasses to view it. This did not contribute to the outcome of the project, but was great fun and makes for a nicer presentation of the experiment when showing it to others.

Appendix D

Learned lessons

In this appendix I would like to draw some attention to aspects that have greatly helped me complete my project successfully.

D.1 Use version control

Version control is a system that can automatically perform the administration of different versions of your code and texts. In effect it keeps a history of every change made and allows you to go back to any state that the code has been in. The code 'repository' can be stored on a central place from where everybody working on the 'code tree' can retrieve a copy, edit it and add his changes along with a log message indicating what has been changed. There is also a mechanism that warns you if conflicting changes have been made and that aids in how to resolve these. Linux has a default version control system called 'cvs' to which I will refer here. Some disadvantages of a cvs are:

- It takes some effort to learn how it works and how best to set it up.
- It takes extra time to add changes to the repository, especially because in the ideal case all changes to the code should be accurately described by an appropriate log message.
- Conflicts can only be resolved if all people working on the project submit their changes regularly (i.e. at least once every day).
- Changes made by others can break working code for instance due to dependencies that they are not aware of.

Advantages of a cvs are:

- Changes that have broken the system can be undone.
- Accidentally deleted files can be recovered.
- Without a cvs it is not humanly possible to work on code with multiple developers.
- The log messages provide an additional documentation of the code on a meta level.
- While the repository resides on one computer, local versions can be used and edited on other machines as well.

All in all I have experienced that it was well worth the trouble to introduce a cvs for the codetree we were working on. A good quick guide to how cvs works is the website 'Open Source Development With CSV' (http://cvsbook.red-bean.com/cvsbook.html).

D.2 Log

It pays off to spend a lot of time on good logging facilities. Implementing a way to log almost everything that is relevant to the system but also a way to turn it off if it is not relevant can easily make up in a later stage for the time lost in developing such a logging system.

If at all possible, log-files that show at least the most important aspects of the systems should be kept for later reference. For this purpose it is also important to provide an automated way to put *all* parameters that influence the system into the log-files. This way one only needs to inspect a log-file to see how it is produced. I have started out by trying to keep track of this on paper but this quickly became too cumbersome and confusing.

Appendix E

Glossary

Agent Computer program that acts and that distinguishes itself from mere "programs" with attributes such as operating under autonomous control, perceiving its environment, persisting over a prolonged time period, adapting to change and being capable of taking on another's goals Russell and Norvig [2003].

Circle Ellipse in which the two axes are of equal length.

Deictic Specifying identity or spacial or temporal location from the perspective of a speaker or hearer in the context in which the communication occurs.

Imitator The agent that is chosen to imitate the action of the initiator°

Initiator The agent that has to perform the action that is to be imitated by the imitator°

Inventory The set of action categories in an agent's repertoire

Language game The interaction (possibly robotic) agents engage in in a specific environment that models some aspect of the learning of language Steels [2000].

Pixel An abbreviation of the term 'picture element.' A pixel is the smallest picture element of a digital image.

System A group of independent but interrelated elements comprising a unified whole.

Bibliography

ACT-R Research Group. Act-r: Theory and architecture of cognition, 2004. URL `http://act-r.psy.cmu.edu/`.

Tony Belpaeme, Bart de Boer, Bart De Vylder, and Bart Jansen. The role of population dynamics in imitation. In K. Dautenhahn and Chrystopher L. Nehaniv, editors, *Second International Symposium on Imitation in Animals and Artifacts 2003*, pages 171–175. The Society for the Study of Artificial Intelligence and the Simulation of Behavior, 2003. ISBN 1-902956-30-7.

Philip R. Bevington. *Data Reduction and Error Analysis for the Physical Sciences.* McGraw-Hill, New York, 1969.

K. Binmore. *Fun and games.* D.C. Heath and Company, Lexington, MA, 1992.

Klaus A. Brunner. What's emergent in emergent computing? In R. Trappl, editor, *Cybernetics and Systems 2002 : Proceedings of the EMCSR 2002*, volume 1, 2002. ISBN 3852061601. URL `http://winf.at/ klaus/emcsr2002.pdf`.

Morten H. Christiansen and Simon Kirby. Language evolution: The hardest problem in science? In M.H. Christiansen and S. Kirby, editors, *Language Evolution: The States of the Art.* Oxford University Press, 2003. URL `http://www.ling.ed.ac.uk/ simon/0-19-924484-7.pdf`.

Richard Dawkins. *The Selfish Gene.* Oxford University Press, Oxford, 1976.

Bart de Boer. *Self Organisation in Vowel Systems.* PhD thesis, Artificial Intelligence Lab, Vrije Universiteit Brussel, 1999.

Bart de Boer. Emergence of sound systems through self-organisation. In Michael Studdert-Kennedy, James R. Hurford, and Chris Knight, editors, *The Evolutionary Emergence of Language : Social Function and the Origins of Linguistic Form.* Cambridge University Press, Cambridge, 2000a.

Bart de Boer. Self-organisation in vowel systems. *Journal of Phonetics*, 28(4):441–465, 2000b.

Bart de Boer. *The origins of vowel systems.* Oxford University Press, Oxford, UK, 2001.

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

Terrence W. Deacon. *The symbolic species: the co-evolution of language and the brain.* W.W. Norton, New York, 1997.

Dario Floreano, Francesco Mondada, Andres Perez-Uribe, and Daniel Roggen. Evolution of embodied intelligence. In Fumiya Iida, Rolf Pfeifer, Luc Steels, and Yasuo Kuniyoshi, editors, *Embodied Artificial Intelligence*. Springer, 2004.

Steve Grand. *Growing up with Lucy*. Weidenfeld and Nicolson, 2004. ISBN 0297607332.

Bart Jansen. The emergence of shared action categories in robotic agents. Master's thesis, Vrije Universiteit Brussel, Brussels, 2003.

Bart Jansen. Towards goal directed imitation. Paper in progress, 2004.

Bart Jansen, Bart de Boer, and Tony Belpaeme. You did it on purpose! towards intentional embodied agents. In Fumiya Iida, Rolf Pfeifer, Luc Steels, and Yasuo Kuniyoshi, editors, *Embodied Artificial Intelligence*. Springer, 2004a.

Bart Jansen, Bart De Vylder, Tony Belpaeme, and Bart de Boer. Emerging shared action categories in robotic agents through imitation. In K. Dautenhahn and Chrystopher L. Nehaniv, editors, *Second International Symposium on Imitation in Animals and Artifacts 2003*, pages 145–152. The Society for the Study of Artificial Intelligence and the Simulation of Behavior, 2003. ISBN 1-902956-30-7.

Bart Jansen, Tom Ten Thij, Tony Belpaeme, Bart De Vylder, and Bart de Boer. Imitation in embodied agents results in self-organization of behavior. In *Proceedings of The Eighth International Conference on the Simulation of Adaptive Behavior*, Los Angeles, USA, 2004b. The MIT Press, Cambridge.

Kurt Konolige. Small vision system: Hardware and implementation. In *Proceedings of the Eighth International Symposium on Robotics Research*, 1997. http://www.ai.sri.com/konolige/svs/Papers.

James Kuester and Joe Mize. *Optimization Techniques with Fortran*. McGraw-Hill, New York, 1973.

H. McGurk and J. MacDonald. Hearing lips and seeing voices. *Nature*, 264:746–748, 1976.

Giorgio Metta. *Babybot: a study into sensorimotor development*. PhD thesis, LIRA-Lab, DIST, September 2000. URL `http://www.liralab.it/papers/theses/giorgio.metta.2000.pdf`.

Stefano Nolfi and Dario Floreano. *Evolutionary Robotics. The Biology, Intelligence, and Technology of Self-organizing Machines*. MA: MIT Press, Cambridge, 2001.

Martin A. Nowak, Natalia L. Komarova, and Partha Niyogi. Evolution of universal grammar. *Science*, 291:114–118, 2001.

Michael Oliphant and John Batali. Learning and the emergence of coordinated communication. *The newsletter of the Center of Research in Language*, 11(1), 1997.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Inc., Upper Saddle River, New Jersey 07458, second edition, 2003. ISBN 0-13-080302-2.

L. Steels, J. De Beule, N. Neubauer, and J. Van Looveren. Fluid construction grammars. In *Proceedings International Conference on Construction Grammars*, Marseille, 2004.

Luc Steels. The talking heads experiment. Available from the VUB Artificial Intelligence Laboratory, Brussels, Belgium, 1999.

Luc Steels. Language as a complex adaptive system. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Proceedings of Parallel Problem Solving from Nature VI*, Lecture Notes in Computer Science. Springer, Berlin, Germany, 2000.

Luc Steels and Tony Belpaeme. Coordinating perceptually grounded categories through language. A case study for colour. Accepted as target article for Behavioral and Brain Sciences., 2004.

Luc Steels and Paul Vogt. Grounding adaptive language games in robotic agents. In Phil Husbands and Inman Harvey, editors, *Proceedings of the Fourth European Conference on Artificial Life (ECAL'97), Complex Adaptive Systems*, Cambridge, MA, 1997. The MIT Press.

P. Vogt. 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, 2000. URL `citeseer.nj.nec.com/vogt00grounding.html`.

B. Webb and R.C. Consi. *Biorobotics -Methods and application-*. MIT Press, Cambridge, Mass., 2000.

Barbara Webb. Neural mechanisms for prediction: do insects have forward models? *Trends in Neurosciences*, April 2004.

Ludwig Wittgenstein. *Philosophical Investigations*. Macmillan, New York, 1953.

Wordnet. A machine-readable lexical database for the english language, 2004. URL `http://wordnet.princeton.edu/`.

Willem Zuidema. *(to appear) The major transitions in the evolution of language*. PhD thesis, University of Edinburgh, 2004.