Musicology in a Virtual World: A Bottom Up Approach to the Study of Musical Evolution

Martijn Bosma (1163450)

February 17, 2005

Supervisor Groningen: Rineke Verbrugge
Associate Professor of Artificial Intelligence
Team leader of the Multi-agent Systems Group of
the Institute of Artificial Intelligence and
Cognitive Engineering, ALICE
Artificial Intelligence
University of Groningen
Grote Kruisstraat2/1
9712 TS Groningen
The Netherlands
Email: rineke at ai.rug.nl

Supervisor Plymouth: Eduardo Reck Miranda
Reader in Artificial Intelligence and Music
School of Computing, Faculty of Technology
The University of Plymouth
Drake Circus
Plymouth, Devon PL4 8AA, UK
Email: eduardo.miranda at plymouth.ac.uk

Artificial Intelligence
Rijksuniversiteit Groningen

Abstract

The objective of this project is to implement and study a model inspired by the mimetic agents of Miranda [16], who proposes a society of musical autonomous agents that evolves a common repertoire of intonations from scratch by interacting with one another. These agents are furnished with a vocal synthesizer, a hearing apparatus and a simple brain. They communicate by playing imitation games [2]. The model studied in this project adds to the agent architecture a neural network brain, a simple grammar learning device, *realistic ears*, and a music culture. *Realistic ears* is a metaphor for a pitch detection algorithm that can infer a wrong pitch, depending on the timbre of the instrument.

The SARDNet [10] is the neural network brain of choice since it is able to handle sequences. The fact that music is ordered in time is a very important characteristic which is preserved by using a SARDNet. In this project we found that the SARDNet-brains of the agents learned human-made songs easier than randomly generated songs. It turned out that these human-made songs, which were successful in our world, were successful in the agent world as well. The success was measured as learnability. A second part of the agent brain is a grid of probability tables. This grid can be seen as a grammar learning device. It learns the transitions between elements of a melody.

The music knowledge of the agents does not start from scratch, but instead we assume that the culture of the agents has already a history. This cultural history is modeled as a set of songs. Some of these songs are real compositions from our world, like *Invention Number 1* of Bach, while other songs are randomly generated melodies produced by the computer. A song of this set can be linked to an agent of the society. This agent is said to "study" its culture. It means that the neural network of an agent is trained on the material of one of the songs. An agent is able to compose music based on its knowledge which is stored in the neural network brain. To evaluate these compositions, I have developed a classifying tool that compares agent-compositions to the set of songs. During learning, the compositions of an agent became less similar to the song it was assigned to, but they generally could be classified as belonging to the assigned song. Three out of four agents made compositions that were more similar to the songs they studied, than to the other human-made songs present in the set.

The timbre (sound-color) of a musical instrument has an effect on the perception of the pitch-intervals. This holds for the agents with *realistic ears* as well. I investigated the effect of different timbres of 7 fm-synthesizers on learning. The agents imitated the melodies played on these instruments. Their imitation errors were much larger when the timbre of the instrument was more percussive.

Finally, I have investigated the effect of communication by putting together two agents, who were trained on different songs, to play the imitation games. I compared two societies of two communicating agents. In one society the agents were allowed to study their own song while they communicated as well. In the other society the agents were not allowed to study their song anymore, once they started communicating. When the agents switched gradually from studying their own song, to playing imitation games, they were able to keep the knowledge of their own song. When the phase change from studying to communicating was abrupt, the agents learned a mix between their own song and the song of the other agent.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Why does music in a certain culture sound the way it does? Around the world there is a tremendous variety in music styles and cultures. Why does this variety exist? People in a certain music culture generally do not invent a complete set of musical rules. Instead they learn the music of their culture from their parents and their peers, who learned it from their parents and peers and so on. The conventions and implicit rules of a certain musical culture are learnt again and again by new individuals.

During this communication process of relearning, music changes due to different preferences of individuals. We can imagine that some melodies are sung or played more often because the inhabitants of a musical culture like them more. As a result these melodies are known by more individuals and variations of these melodies will probably occur. On the other hand, melodies that are not so popular are played less and can even disappear when no individual learns it anymore. Music is therefore shaped by social dynamics [1] [13].

Now imagine a musical culture where a very complicated type of melody exist. A melody like this has to be relearned every generation. We can imagine that some of the music learners make mistakes. They are not able to learn all the complicated details of this melody. Therefore they modify some bits of it to make the melody simpler and therefore easier to learn. They will teach their children this modified melody and in this way a simpler version will spread in the society. The point here is that not every melody can be

[1]When we look at the melodies as entities that try to survive by being played and remembered we can even speak of cultural evolution. Mutations can be seen as melodical variations on some well known main themes.

learnt or remembered easily and that music is shaped by cognitive dynamics as well.

The shaping role of social dynamics on music is studied in a number of A-life models[2]. In these models agents[3] produce musical signals that are heard and reacted to by other agents [13]. Miranda [16] proposes a society of musical autonomous agents that evolves a common repertoire of intonations[4] from scratch by interacting with one another. The agents are furnished with a vocal synthesizer, a hearing apparatus and a simple brain. They communicate by playing imitation games [2]. In these games two randomly selected agents of the society have the roles of a singer and an imitator. The singer sings a melody and the imitator imitates it. The singer listens to the imitation and gives positive feedback if it judged the imitation as similar to its own utterance. After receiving feedback the imitator agent knows whether its melody was a good imitation or not and it learns from this feedback by adapting its knowledge about melodies of the society of agents. Note that there is no central control driving these interactions. Miranda is interested in the properties and mechanisms that the agents, their environment and their interactions must possess in order to create music.

The brains of these agents are very simple. When for example a singer agent "composes" a melody, it selects one from a list of melodies that are stored in its memory. If this list is empty, it generates a random melody. The representation of a melody is symbolical, it is treated as one unit that can be selected.

There is another line of research that investigates learning of sounds and sequences (melodies). Here sequences and sound learning is modeled in neural networks. In a neural network a melody is represented on the sub-symbolical level, that is the elements of a melody are distributed over the neural network. This representation is much more biologically plausible. The representation of a melody in our brain is distributed as well. It is very unlikely to find a particular small area that stores the complete representation of a melody.

The objective of this project is to implement and study a model inspired by the society of imitating agents of Miranda [16] and to make the agent architecture more biologically plausible. In this model the agents play imi-

[2]A-life (Artificial life) is a discipline that studies natural living systems by simulating some of their biological aspects on computers [13] [12].

[3]Agents are small computer programs that operate autonomously [4]

[4]Intonations are viewed here as the basis for musical melodies.

tation games as well but now their brain is a neural network. The melody representation is therefore sub-symbolical. Furthermore the music culture of the agents does not start from scratch as happens in Miranda's society of agents, instead we assume that the culture of agents has already a (long) history. This cultural history is modeled as a set of songs. Some of these songs are real compositions from our world, like *Invention Number 1* of Bach and the *One Note Samba* of Jobim, while other songs are randomly generated melodies produced by the computer. A song of this set can be linked to an agent of the society. This agent is said to "study" its culture. This means that the neural network of an agent is trained on the material of one of the songs. In this way some knowledge of musical cultures of the real world invades the A-life model.

With a more biologically plausible brain and a cultural history that has its origins in the real world we have the opportunity to investigate some interesting topics. First of all the implementation of neural network brains allows us to study the above mentioned effect of cognitive dynamics on melodies. A neural network has its limits in how much information it can store and how fast it can learn. These limits can result in a learning bias. Some melodies will be easier to learn than others. The human made melodies that represent the cultural history in this model are very successful in the real world, since they are well-known and survived many years and generations. An interesting question is: Will they be successful in the agent world?

Secondly the cultural history allows us to study the effect of a cultural melting pot. It is possible to give two agents a different cultural history by giving them different songs. What will happen if these two agents will communicate and learn from each other? Will we be able to see a similarity to people in a multi-cultural society?

1.1 Structure of this thesis

In chapter 2, I will discuss work that has formed the inspiration of the current model. This will include a description of the imitating agents of Miranda [15], a discussion of class formation in a group of agents playing communication games, and a discussion of neural networks, that are candidates for implementation as an agent brain. This chapter ends with a discussion on the evaluation of machine compositions.

In chapter 3, I will give a detailed description of the model I propose in this thesis. The human-made songs are listed (in a special agent notation) in appendix C. Chapter 4 introduces the research questions and the hypotheses of the project. I will use chapter 5 to describe the experiments that are designed to provide answers to these questions. The parameter settings of every experiment can be found in appendix A. The results of the experiments are described in the succeeding chapter 6 and the implications for the hypothesis are discussed in chapter 7. Finally a general conclusion on the model is made in chapter 8 and some hints for improvements are given in chapter 9.

Chapter 2

Theoretical Framework

The model I propose in this project combines two areas of research. First of all there is the research on a-life and musical composition. I will discuss this in section 2.1. Then I will focus on one of these models, the *Mimetic Development of Intonation* model proposed by Miranda [16] [15], which is the inspiration for the model of this project. The second line of research is that of unsupervised neural networks. I will discuss two of these networks as candidates for the agent-brain.

Finally, I will conclude this chapter with a discussion of the evaluation of machine compositions using the a framework proposed by Pearce and Wiggins [17].

2.1 Research on artificial societies of interacting agents

There have been a number of interesting applications of A-life concerning music. In *A-life and Musical Composition: A Brief Survey* [13], Miranda and Todd discuss three approaches to the use of A-life models of interacting agents in music composition. The least musical approach from an agent point of view, is the rendering of *extra musical behavior*. In these models agents, their environment and their behavior are modeled. Some of the behavior of the agents is converted to sound, and in this way music is created. The agents are not musical themselves and their behavior does not have anything to do with music. Music has no effect on the agents either, it is just a representation of the data that result from the interactions that occur in the model.

A second more directly musical approach is inspired by genetic algorithms. Here music *does* have an effect on the agents. The agents produce musical phrases and the success of their utterances determines their chance of survival and the probability of generating offspring. In many of these models an agent poses an artificial genome which represents the ability to create music[1]. The rating of the musical phrases is done by a critic that is situated *outside* the model. This can be a human critic or another computer model [23] [13]. This critic acts as a kind of god who reigns over life and death of the agents. The agent world has no effect on the critic at all. The critic determines what characteristics the music phrases of an agent should have to make it survive.

Finally there is the cultural approach. This is the most musical of the three. The main difference with the second approach is that the critics are modeled *inside* the A-life model. This gives rise to some interesting social dynamics. The music producing agents are affected by the judgement of the critics, who determine what music is successful and what not. However the judgement of the critics is affected by some dynamics of the model.

For example Todd and Werner [23] designed a cultural a-life model that is inspired by the social dynamics involved in the effect of bird songs on mating success. They modeled male singers playing courting tunes, and female critics who judge these tunes and decide whom to mate with in order to produce the next generation of singers and critics. The offspring inherits traits of both parents and in this way tunes and tune-preferences co-evolve over time, to explore regions of melody space without human intervention.

A less explicit separation of the roles of singer and critic can be found in a-life models concerning agents playing imitation games. In these models an agent plays both roles during its lifetime.

Imitation games where developed by de Boer [3] [4] to model the formation of a vowel system in a population of agents and are a kind of language games. Language games were developed by Steels [21]. A language game is a set of interactions between agents in which they use language to communicate certain information, together with a number of rules of how the interaction should be structured and a definition of when the game is successful. In Steels' theory language is something that emerges through the interactions of language-users trying to communicate with each other and learning the

[1]See for a discussion of the design of creative evolutionary systems [22]

language in this way.

Instead of language, music can be the medium of communication. Music and language are related, as can be seen for example in *intonations*. Intonations can be considered as a bridge between language and music, since they can be viewed as *speech prosodies* or as the basis for the formation of *musical melodies*. This inspired Miranda to apply imitation games to the realm of music in his model "Mimetic Development of Intonation [16]" where a group of agents evolves a common repertoire of intonations from scratch by interacting with one another.

2.2 Mimetic Development of Intonation

Miranda [16] [15] has proposed a model wherein a small society of interactive agents furnished with the appropriate motor skills auditory skills and cognitive skills is able to evolve a shared repertoire of intonations[2]. The agents achieve this solely by imitating each other.

The goal of these agents is to be sociable. In this society an agent is sociable if its repertoire of intonations is similar to those of its peers. Therefore the agents have, apart from the ability to hear and produce sounds, an instinct to imitate.

An agents produces sounds by means of a voice synthesizer. It can control this synthesizer with two motor parameters. These parameters stand for the control of the pitch and the duration of the sound. Agents are able to hear and analyze the utterances of other agents with their hearing apparatus. The brain of an agent consists of a motor memory and a perceptual memory and associations between the two. When an agent listens to an intonation, the analyzed melody is stored in the perceptual memory of its brain. When an agent tries to sing this intonation it looks in its motor memory for the associated motor representation, and sends these motor commands to the synthesizer. Furthermore an agent possesses an innate enacting script. The enacting script tells it how to behave during its lifetime in the society.

At every time step in this agent world, two agents are selected randomly. One agent gets the role of singer, the other gets the role of imitator. The singer sings an intonation from its repertoire of intonations stored in its motor memory. If this repertoire is empty, a random intonation is generated. The imitator hears and analyzes the intonation and looks in its perceptual

[2] Intonations are viewed here as the basis for musical melodies.

memory for the most similar intonation. It retrieves the associated motor commands and sings its *imitation*. The singer agent hears and analyzes the imitation and compares it with the intonations stored in its perceptual memory. It retrieves the most similar intonation. If this retrieved intonation is the intonation the singer just had sung, it gives positive feedback by singing this intonation again. If it turns out to be another intonation then the singer remains silent, which is meant as negative feedback.

After receiving the feedback the imitator knows whether its imitation was good enough or not. If the imitation was good enough, it will reinforce the existence of the intonation that was used for the imitation, by increasing a counter that counts how many times this intonation has been successfully used. It will adapt the perceptual representation a little bit to make it even more similar to the song of the singer agent. If the imitation was not good enough, then the imitator agent adapts the motor representation of this intonation intending to make it more successful in another round. Only if this intonation was very successful in the past, it leaves the motor representation as it is, since other agents might know this intonation as well.

After several thousands of these interactions the community of agents evolves a stable amount of intonations. At this point all the agents know the same intonations. This means that their perceptual memories are similar. However, the motor representations are not always similar, which indicates that there are more perceptual-motor mappings that produce the same intonation.

2.3 Neural networks

One of the main aims of this project is to model a society of agents inspired by the mimetic agents Miranda proposes [15] and to furnish them with neural network-brains. What neural network is suitable? Firstly I will discuss a neural network that maps perceptual to motor representations and vice versa [26] in section 2.3.1. This is exactly what Miranda's mimetic agents do and therefore it seems at first site a perfect candidate for the agent brains. However I will argue that the network cannot be used.

The network of choice is the SARDNet. It is an extension to the Self Organizing Map (SOM) [11] [10]. To understand the SARDNet it is necessary to understand the SOM, therefore I will explain the SOM first in section 2.3.2, and afterwards the SARDNet in section 2.3.3.

2.3.1 Mirror Neuron Model

Westermann and Miranda [26] proposed a sub-symbolical model that integrates between a perceptual and motor representation of vowels. It is inspired by the development of vowels in an infant's babbling phase, wherein perceptual and action prototypes develop concurrently. Like the agents described above in section *Mimetic Development of Intonations* (section 2.2) [16] this model has a vocal synthesizer, a hearing apparatus, and a brain. However in this case the brain is the center of attention.

The model uses two maps, a motor map and a sensory (perceptual) map. The maps contain neurons on a multidimensional grid. These dimensions stand for motor and sensory parameters. The sensory parameters represent formants. The motor parameters are used to control a vocal synthesizer. A neuron on the motor grid is connected to all neurons on the sensory grid and vice versa, by means of hebbian connections [8].

When the model produces a sound it does this by activating a group of motor neurons, which results in an active area on the motor map. The corresponding motor commands are sent to the vocal synthesizer. At the same time the hearing apparatus hears and analyzes this sound. As a result a group of neurons on the sensory map is activated as well. Now the hebbian learning algorithm strengthens the connections between neurons that are active on both maps.

Neurons can be activated by their hebbian connections as well. If a neuron on one map is active, then this activity is transmitted through the hebbian connections to the neurons of the other map. The neurons that have a strong hebbian connection with this active neuron will therefore be activated.

Now when the model produces a sound, by activating motor neurons, this results in active perceptual neurons. This on its turn results in new activity of motor neurons, caused by the hebbian connections. This new activity of motor neurons results in a new sound, which is heard and analyzed by the hearing apparatus which causes new activity of perceptual neurons. This on its turn, results in activation of new motor neurons, and so on. This activity of the model is called babbling. In this way the model learns the associations between motor commands and the resulting formants.

This hebbian network seems to be a good candidate-brain for the agents of the model I propose in this project. It maps sensory and motor representations in a sub-symbolical way and it performs well when it learns vowels. However melodies are distributed in time. A neural network that learns melodies should be able to handle this.

One solution is to represent a melody as one point on a multi dimensional map. The maps of a Mirror Neuron Network can have as many dimensions as we wish. A melody of for example ten notes can be represented as one point on a ten-dimensional map. The problem is that we lose the sequential characteristics of the melody. The fact that for example the last note is very distant in time from the first note is gone. Due to the inability of the Mirror Neuron Network to handle time sequences it is not the agent brain we are looking for. Therefore we will investigate an alternative more suitable for handling time sequences.

2.3.2 Self Organizing Map (SOM)

A Self Organizing Map or SOM is an unsupervised neural network, that is able to learn and order statistically significant features of the input in a topologically meaningful way [7] [11]. It consists of a one dimensional input layer and a two dimensional map of output nodes. Every output node has a weight vector with the same dimensions as the input layer. The data consists of vectors of the same length as the input layer. Figure 2.1 shows an example of the network.

When we like to train the network we need to perform the following steps:

Initialization We choose random values for the initial weight vectors of the output nodes. Most of the times these chosen values are close to zero.

Sampling We draw a random vector from the set of training data. This will be the input vector.

Similarity Matching We find the best matching output node by comparing the weight vectors of all the output nodes to the input vector. The best matching output node is the one that has the weight vector with the shortest euclidian distance to the input vector.

Updating We update the weights of the best matching output node (winning neuron) and of the output nodes that are within its neighborhood in the direction of the input vector. Which nodes are in the neighborhood is determined by the neighborhood function. The size of the neighborhood shrinks during the training of the network.

Continuation We continue this process with the *Sampling* step and repeat all the steps from there until there are no weight changes anymore due to the Updating step.

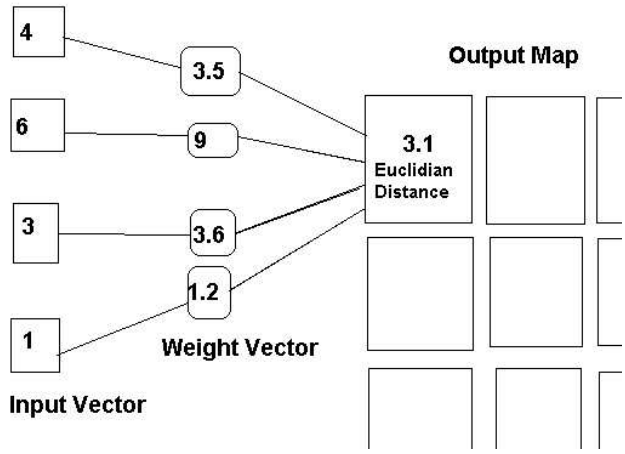

Figure 2.1: An example of an input vector, the weight vector and a node of the output map. The euclidian distance between the weight vector and the input vector is displayed in the corresponding output node.

2.3.3 SARDNet

A SARDNet is a kind of SOM adapted for sequences. SARDnet means: Sequential Activation Retention and Decay Network. After learning it yields very dense representations of the input on its output map. The prototypes of the input are packed as *sardines*, hence the name SARDNet [10].

Input of a SARDNet consists of a *sequence* of input vectors. The input layer, the weight vectors and the output map are similar to that of a SOM. It extends the SOM with an extra map that records the activity of the output nodes during the presentation of a sequence.

The input vectors of a sequence are presented to the network one by one. When an input vector is presented, the winning neuron and the neurons in its neighborhood are allowed to update their weights closer to the input, like in a SOM. Furthermore the winning neuron receives an activation of 1 on the activation map and is excluded from further competition during the rest of the sequence. In this way every input vector of a sequence is allocated to a different output node. As more input vectors come in, the activation of the previous winners decays. In other words, each sequence of length l is represented by l active nodes on the output map, with their activity indicating the order in which they were activated. The algorithm is summarized in table 2.1 and illustrated in figure 2.2.

This figure displays the presentation of a melody of three time steps to the SARNet. Every picture shows one time step. When we follow the pictures in chronological order we can follow the presentation of the sequence. Every time step one output node is activated as a result of the input vector. We see that this activation is decayed in the following time steps. In the last picture of the series we are able to follow the sequence on the output map by following the active output nodes from low to high.

The SARDNet is used in this project as the brain of the agents. In section 3.1 of chapter 3 is explained how it is implemented.

INITIALIZATION	clear all nodes to zero
MAIN LOOP WHILE	 not end of sequence
1	Find un-activated weight vector that best matches the input
2	Assign 1.0 activation to that unit
3	Adjust weight vectors of the nodes in the neighborhood
4	Exclude the winning unit from subsequent competition
5	Decrement activation values for all active nodes
RESULT	Sequence representation: activated nodes are ordered by activation values

Table 2.1: A sequence of input vectors of length one activates nodes on the output map one at a time.

2.4 Towards a Framework for the Evaluation of Machine Compositions

When we have built our society of agents furnished with SARDNet brains, we let them interact and compose music, but how do we evaluate their com-

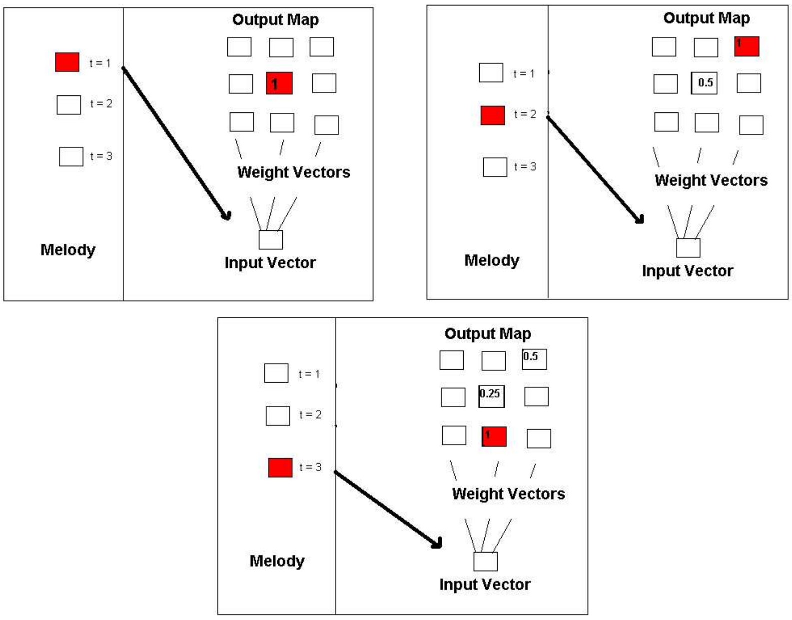

Figure 2.2: An example of what happens when a sequence, (in this case a melody) of length three is presented to the SARDNet. The input vector length is 1. Every picture displays one time step

positions? How do we treat such a subjective discipline as music objectively?

Pearce and Wiggins [17] have outlined a first step towards a framework for the objective evaluation of machine compositions. The framework allows statements about those compositions to be refuted on the basis of empirical experimentation. According to Pearce and Wiggins this is fundamental if we wish to evaluate the degree to which our models achieve their compositional aims.

The framework consists of four components:

1. Specifying the compositional aims

2. Inducing a critic from a set of example musical phrases from the relevant musical genre

3. Composing music that satisfies the critic

4. Evaluating specific claims about the compositions in experiments using human subjects

We can see from component 2 that this framework is designed to evaluate those systems that generate machine compositions meant to be in a specific musical genre. Before the model is tested all the details of what the model is intending to do should be specified in great detail. For example the following questions should be answered: are we aiming to generate music from a special composer; is the model meant to generate whole compositions or just phrases; or is the focus on rhythm?

The critic is induced from a set of patterns representing the musical genre by using some machine learning technique. This machine learning technique should be clearly justified and any bias that occurs due to this technique should be mentioned. A critic induced from a machine learning technique is more flexible than one that is induced from a set of rules, since most rules have a lot of exceptions that can not be handled by the rule-critic. It would be too rigid for such a fuzzy domain as music.

After the critic has learnt the specific music style the model can be started to compose music. The results of the model are presented to the critic. Note that Pearce and Wiggins assume here that the model is able to use feedback of the critic to improve its compositions. If the model had no way of taking the judgement of the critic into account, the latter would have no effect on the music produced by the model.

Finally the generated music can be evaluated by asking human subjects to distinguish compositions taken from the data set from those generated by the system. If the machine-composed pieces are significantly misclassified as human compositions we may conclude that they are indistinguishable from human-composed pieces. It depends on the aims stated in component 1, what other experiments on the compositions have to be done with humans as the evaluator.

The aim of this framework is to be clear enough to enable machine composition researchers to evaluate their machine compositions scientifically, but it should be relaxed enough to account for a wide range of machine composition models. The question is: What does this framework imply for the model of this project?

The main aim of the model under study is to look at the effect of the implementation of SARDNet brains on a society of agents inspired by the society of mimetic agents proposed by Miranda [15]. Only a part of these effects concern the evaluation of the compositions of the agents. Remember that the model contains some human-made compositions that represent the cultural backgrounds of the agents. Only when we compare the compositions of the agents to the human-made compositions we seem to be able to use this framework. I will discuss this issue in the discussion chapter(Chapter 7).

Chapter 3

The Model: Implementation

In this chapter I will describe the model I have implemented. I will explain what it contains and how it works. I will start with discussing the agent brains in section 3.1. Then I will continue with the melody representation in section 3.2. The ears are the perception channel and the synthesizer is the motor-control channel of an agent. These communication channels of the agents are described next in 3.3. I will finish with explaining what actions an agent can perform in 3.4.

3.1 The brain

The agent brain consists of two parts, the SARDNet and the grid of transition tables. The SARDNet has already been discussed in chapter 2. I use this network in a different way than James and Miikkulainen do in [10]. I will discuss the differences below. The second part of the brain, the grid of transition tables, is a kind of grammar learning device. It learns the relations between adjacent notes of the input melodies.

3.1.1 SARDNet

The most common way to use a kohonen feature-map [11](or SOM: Self Organizing Map, a SARDnet is a kind of SOM) is to compress multidimensional input into a two dimensional output. The two dimensions are the coordinates of the output-map. Apart from compressing the input, the kohonen network also orders the input in a meaningful way on its output-map. The place of activation on this map represents statistically significant feature values of the input.

All the nodes on the output map have weight vectors. A weight vector has exactly the same dimensions as the input vector. The goal of an output node is to win the competition with its fellows on the map. It can do this by having its weight vector closer to the input vector than the weight vectors of the others. The distance measured here is the euclidian distance. The winning node is allowed to adapt this weight vector even closer to the input vector. The nodes in its *neighborhood* are allowed to do the same, but the adaptation will be less [7].

In my model I use the weight vector of the winning node as output of the network. By not using the coordinates as output, I do not use the data-compression feature of a kohonen map. What I do use is the meaningful ordering on the output map. Output nodes that are close to each other have more similar weight vectors than those that are far away from each other.

The advantage is that a weight vector always has the same dimensions as the input vector. In this model the agents have to imitate the melodies they *hear*. This means that the output the agents have to produce should be of the same type as the input they received. The weight vector is a representation of the input, but it consists of a string of floating point numbers while the input consists of a string of integers [1]. To make integers of these floating points my model rounds them to the closest integer value.

To conclude: The rounded weight vector of a winning node is the output of the network and can be seen as the interpretation of the input vector.

3.1.2 Grid of transition tables

The grid of transition tables introduces the possibility for a very simple grammar to be learnt by the agents.

Every time when a melody is presented to the SARDNet, it creates a path of active nodes on the output map. This path is a representation of the melody. Along such a path of active nodes, the transitions from one node to the next one are recorded by the grid of transition tables.

Let us say we have node X. Node X is part of a melody path. Let's call the active node in the melody path that comes after X, node Y. This transition from X to Y is recorded in the transition table corresponding to

[1]See section 3.2 for a discussion of the melody representation.

node X. The transition table is a counter. It counts where a melody goes on the output map after it visited the current node, in this case node X.

We can divide the output map in areas and assign transition tables to every area. In this way we get a grid of transition tables.

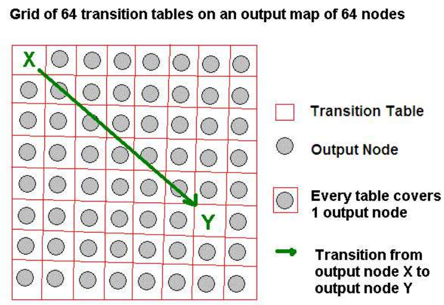

Figure 3.1: The output map of 64 output nodes is covered by a grid of 64 transition tables. Every output node has one transition table. The transition from output node X to output node Y, is recorded in the transition table that covers output node X.

A transition table does not have to correspond to one output node. It is possible to design a coarser grid of transition tables on the same output map. Here is an example of a grid of four transition tables that covers an output map of 64 output nodes:

Now we are going to look at such a transition table. It is a table in a grid of four transition tables. In the next figure we see that the it counts the number of times the corresponding area on the output map is visited. Furthermore it counts how many times the next element of the melody path went to which area. We see that from the twenty times this area was visited, the next element of the melody stayed seven times in the same area. Nine times it went to area two, three times to area three and once to area four. Every transition table of the grid works like this one[2].

The last feature is the start-transition table. This transition table doesn't correspond to an area on the output-map. It records in which area a melody

[2]An example of a complete grid of transition tables of an agent can be found in appendix B

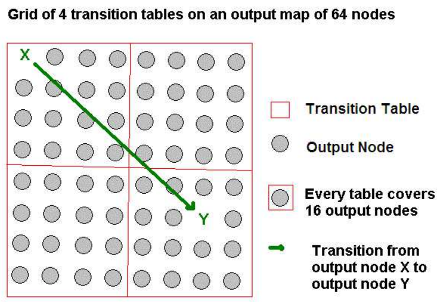

Figure 3.2: The output map of 64 output nodes is covered by a grid of four transition tables. Every transition table covers sixteen output nodes. The transition from output node X to output node Y, is recorded in the transition table that covers the area of output nodes where X belongs to.

AREA	1
times visited	20
go to AREA 1	7
go to AREA 2	9
go to AREA 3	3
go to AREA 4	1

Table 3.1: Example of a transition table

starts. This means that every time a melody is presented to the network, this start transition table is updated. For the rest it is the same as any other transition table.

3.2 The melodies

The agents communicate with a very simple music system. Music can be represented in many different ways. Examples of music notation are: the western score notation, the CARMEC notation [16] and the MIDI notation [18] to name just a view.

These representations are different in how detailed they describe the music. The CARMEC notation leaves for example much more freedom to the musician than the western score notation, since it does not describe the exact pitches. It only describes the musical contour. I have developed a very simple music representation which I will describe below.

The agents in this musical world do not know rhythm, nor do they perceive the loudness of tones. They even do not understand rests in a melody. For an agent in this world, a melody is just a sequence of a fixed length, that consists of notes of equal length.

Like most human beings the agents do not remember the pitch of the notes of the melody. What they do remember are the intervals between the notes. Therefore for example it does not matter whether a song is played in C or in G. The melody remains the same. For this reason melodies are represented as sequences of intervals, also called relative pitches. In the next table we see the names of the intervals and their agent representation.

Interval	Up	Down
prime	0	0
minor second	1	-1
second	2	-2
minor third	3	-3
third	4	-4
perfect fourth	5	-5
tritone	6	-6
perfect fifth	7	-7
minor sixth	8	-8
sixth	9	-9
minor seventh	10	-10
seventh	11	-11
octave	12	-12
etc ...	...	...

Table 3.2: The names of the intervals and their agent notation. The agent just counts on for larger intervals, for example the *ninth* interval would be 9 and played downwards it would be -9.

Figure 3.3 is an example of a chromatic scale. The same scale is written in agent notation and in western score notation.

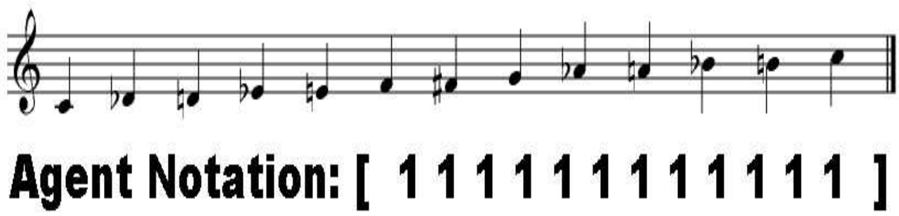

Figure 3.3: A comparison between the western score notation and the notation used in this model, of the same chromatic scale.

We see that the chromatic scale is in C, and that this fact is lost in the agent notation. The agent notation of this scale only consists of ones, because the intervals between the notes are minor seconds. Figure 3.4 is an example, of a part of a melody. Again the same melody is written in western score notation and in the agent notation. We can see that the rhythm is removed, and that the rests are ignored. Only the intervals between the adjacent notes are there. For example, the last number of the agent notation is the interval between the last two displayed notes of the melody. It is a tritone played downwards. This is the way melodies are written in the agent world.

Figure 3.4: A comparison between the western score notation and the notation used in this model, of the same melody.

3.3 Singing and listening

For an agent the synthesizer and the ear are the communication channels with the rest of the agent-world. The agent can express itself by playing the synthesizer and it can listen to expressions of other agents with its ears. Firstly I will explain how the melodies are translated to the sound representation, then I will talk about the synthesizers, and finally I will discuss the ear.

3.3.1 From relative pitch representation to sound

When an agent wants to play a melody it has to translate the representation in its head to sound. This representation has already been discussed in section 3.2.

When we speak or play an instrument, we use our muscles to translate the representation in our head to sound. In a model that links perception to action we can expect a module that is doing the motor control. For example the model of Miranda [14] focuses on the link between motor control and perception. This link is studied in more detail in [26] in the mirror neuron model. I am more interested in the perceptional side of the models, and the formation of categories by playing imitation games. Therefore the motor-control module in the model I propose is simple and straightforward. It does not pretend to be a realistic model of motor control. It just transforms a sequence of relative pitches into sound in a reliable and simple way.

The melody representation in the brain of the agent is a sequence of relative pitches. To play this sequence on a synthesizer the agent needs to perform two steps. Firstly it needs to transform the relative pitches into a sequence of note representations, where every element corresponds to a certain pitch. This is not the case with relative pitches since they only point to intervals, but once we have a reference note it is possible to deduce all the notes of the melody. When for example we have the following sequence of relative pitches: $[5, -2, 0, -1]$ and we know that the reference note is C, then we can calculate all the notes by following the intervals. The first number is 5, so we go a perfect fourth up from C to get the first note of the melody. This is an F. From here we go down a major second, and we find a $D\#$, and so on. This melody will be: $[F, D\#, D\#, D]$. The agent uses MIDI note numbers instead of letters but the idea is the same[3]. The reference note, in this case the C, is not so important, since the agents are not so interested in pitch.

Secondly the agent needs to calculate the list of frequencies from this list of MIDI note numbers. These frequencies are send to a software synthesizer, which transforms them into a wave file.

3.3.2 The synthesizers

Every agent plays a software synthesizer, and different agents play synthesizers with different timbres. I use software synthesizers from *The Synthesis Toolkit (STK)* [5]. This is a C++ library that has some good physical models of instruments, like guitar, banjo and flute. They can be controlled with C++ functions. I have implemented the most simple instruments of this library, the FM-synthesis instruments[4].

There are two reasons for choosing FM-synthesis instruments. One is that they are easy to control by the C++ functions and the second reason is that they do not cost so much CPU cycles when they are played. This will make the model faster.

3.3.3 The ears

The ear of the agents transforms the sound into the melody representation of their brains. There are many computer models that resemble the ear in many details. They perform this task in a way the ear does(e.g. [19]). A detailed model of the ear can be a project on its own, and it is outside the scope of this model. I have implemented a very simple ear that transforms a wave file into the melody representation of relative pitches. I use the C++ CLAM library to do pitch detection [1]. This library contains many algorithms to process sound. Below I will explain how I use it:

- The melodies have a fixed number of notes and all the notes have the same length, therefore we can easily break up the wave file into parts consisting of one note each. We only take a small part of the middle of the note to analyze. Every part is sliced up into audio frames, of length 1025.

[3]See for an introduction to MIDI: [18]

[4]See [24] for a description of FM-synthesis

- Every audio frame is Fourier transformed to obtain its frequency spectrum.
- The peaks of these frequency spectra are collected.
- From these spectral peaks, the fundamental frequency is taken, by the fundamental frequency detection function.
- Every audio frame of a note now has a fundamental frequency. The average of these frequencies is taken to get the estimated fundamental frequency of the note.

In this way we get a list of frequencies. Every frequency of this list is compared to a standard list of frequencies and note numbers. This is a list where 88 note numbers correspond to 88 frequencies. The note number of the closest frequency is chosen. Now we have a list of note numbers.

The last task is to translate this list to relative pitches. A reference note number is used and added in front of the list. Then the list is differentiated to get the relative pitch representation. We have for example the list of the following note numbers: $[14, 23, 18, 13]$ and a reference note 10. All agents know that its value is 10. We add 10 in front of the list: $[10, 14, 23, 18, 13]$. Now we derive the following list of relative pitches: $[4, 9, -5, -5]$. This is our desired representation.

The ears of the agents are not perfect. They make mistakes, when the notes are really low or really high, and when the timbre is murky. This is a desirable characteristic. Imagine a model where the translation from a representation to a wave file is flawless and that the other way around, from wave file to the other representation, is without error as well. Then it does not matter whether the model uses sound or not. The result is the same and sound has no effect at all. Therefore we like to see a sound dependent error. The medium the agents have to communicate within results in some addition of noise to the melodies and this noise can be different for every agent since they are playing synthesizers with a different timbre.

3.4 The actions

In my model there are two actions an agent can perform. It can learn its own culture or it can communicate with other agents (cultures) by means of melody communication.

Below I will explain what the culture learning phase is and why I have implemented it, then I will explain the communication.

3.4.1 Learning one's own culture: Music study

I have selected five very different and well-known compositions and coded them in the agent representation [5]. Every composition has its unique style of interval-use. Every agent is assigned to one of the compositions and is studying it. There is also the possibility to assign an agent to a so-called *random* song. This agent learns sequences of random relative pitch numbers. In this way the compositions and the random song form the different cultural backgrounds of the agents.

This studying phase is added to the model for two reasons. One reason is that I like to give the agents different backgrounds when they start communicating. After studying their compositions in isolation the agents have different knowledge about music. When they have to communicate they have to find a way to understand each other.

The other reason is that I like to introduce *real* music to the model. I like to know how the intervals of real, successful compositions are learnt and treated by the model.

When an agent is studying its culture, its network is trained with substrings of its composition. These substrings have a fixed length and are drawn randomly from the composition. Their length is the same as the length of the communication melodies.

3.4.2 Culture at work: Communication

Here I will describe the imitation game and the way the agents compose melodies and use feedback.

The enacting script

The enacting script I use in this model is very similar to the one of Miranda's model [16]. At each round the following happens:

- Two agents are selected randomly from the group of agents. They get the role of singer and imitator.

[5] See appendix C for the details

- The singer composes a melody and plays it on its synthesizer. How this is done will be described in section 3.4.2.
- The imitator listens to the melody and imitates it. This means that the imitator analyzes the wave file and feeds the result to its network. The output of the network is played on its synthesizer.
- The singer listens to the imitation and compares both melodies. It does this by analyzing the imitation and calculating the euclidian distance between its own composition and the analyzed imitation.
- The singer gives feedback according to the euclidian distance between both melodies.
- The imitator updates its weights while taking the feedback into account. It does this by multiplying the result of the weight update rule with the feedback. In this way, the network adapts stronger to better imitations.

Feedback

Here I will describe how the euclidian distance between the composition of the singer and the imitation is scaled to a value between 0 and 1. For this model I propose the following equation:

$$f = \frac{1}{1 + (d * s)} \tag{3.1}$$

Here f is *feedback* and d is *difference*, which is the euclidian distance between the composition and the imitation. The s stands for *scalar* and is a parameter that can be set in the initialization file[6]. This parameter determines how strong the feedback will be. If it is set to 0 feedback has no effect. If it is set to a number smaller than 1 the feedback is weakened. If it is larger than 1 the feedback is stronger. The result of this equation is always in the interval $(0, 1]$.

Now I will show how this feedback is used in the weight update rule of the SARDNet. The following equation shows the weight update rule of the SARDNet where η is the learning rate and σ is the neighborhood activation:

[6]see appendix A

$$W_j(n+1) = W_j(n) + \eta * \sigma * (X - W_j(n)) \tag{3.2}$$

Here is the same weight update rule but now sensitive to feedback:

$$W_j(n+1) = W_j(n) + feedback * \eta * \sigma * (X - W_j(n)) \tag{3.3}$$

We see that the feedback, like the neighborhood and the learning rate, acts as a kind of scalar. The difference between the weights is multiplied by these factors.

Composing

Now we turn to composing. Recall section 3.1.2 where the transition tables are explained. When a singer agent is going to produce a melody it looks in its initial transition table, called *start*. This looks for example like this:

START	.
times visited	50
go to AREA 1	17
go to AREA 2	20
go to AREA 3	3
go to AREA 4	10

Table 3.3: Example of the initial transition table: start

This agent has seen 50 melodies. Based on the knowledge it has about melodies, it infers that there is $\frac{17}{50}$ probability that a melody starts in area 1, a $\frac{20}{50}$ probability that it starts in 2, a $\frac{3}{50}$ probability that it starts in 3 and a $\frac{10}{50}$ probability of a start in 4.

The agent uses these probabilities to select the first area randomly. This is called Monte Carlo, or roulette wheel search. When it has selected an area it selects randomly a node on the output map that is situated in this area. When the grid of transition tables and the output map have the same dimensions this is not necessary, because there is only one corresponding node. It will retrieve the weight of this node and round it to the closest integer value. The resulting integer will become part of the melody.

Now the agent is ready to select the second note of its melody. It looks at the transition table of the chosen area, that is the area that is selected by the start transition table. Let us say it is area 1. Now it will look in the transition table of area 1. There it will select another area based on the inferred probabilities. The corresponding node on the output map is chosen and the rounded weight is added to the melody. Then the agent goes to the next selected area and so on. The singer agent continues this until it has collected the whole melody.

This is almost all of the story, but we have one small problem. What do we do with the first generated melodies? The first singer agent has not seen any melody. All its transition tables, including its initial transition table, are empty. This means that they only contain zeros. The agent has nothing to choose. In principle this is alright because the agent does not have any knowledge of melodies at this moment.

To come up with a solution I add the following rule:

> When an agent, while generating a melody, encounters a transition table that contains only zeros then the agent generates a random integer in the range $[0, max - note]$. The integer is added to the melody and the next area is selected randomly.

If a transition table only contains zeros this means that the corresponding area has never been visited before.

Max-note is a parameter that can be set in the initialization file. It indicates the maximum interval that can be generated in the model.

Chapter 4

Research Question and Hypotheses

In the sections below I will present four research questions to investigate the characteristics and the behavior of the society of agents. These questions result in hypotheses that are tested in the experiments described in chapter 5.

4.1 How well are the brains of the agents able to learn melodies?

The main idea behind my model is that the agents are equipped with neural networks. Therefore it is good to investigate these "brains" first. Especially because I will use the SARDNet in a less common way. Instead of outputting the coordinates of the winning neurons on the output map, the agents will give the weights of the winning neurons as output. This sequence of weights is the interpretation of the melody by the agent.

To answer this question we will look at how the networks adapt to the melody-input. This input can consist of random melodies or of real, human-made compositions. The human made compositions are very successful in the real world, since they are well-known and survived many years and generations. An interesting question is: Will they be successful in the agent world?

It is not in the scope of this model to implement the success of a song in great detail but we can at least look at the learnability of these human-made compositions. We expect that they are easier to learn by the networks than random melodies of the same size and tone range because real compositions only use a fraction of all the possible intervals that random melodies of the same *range* can use. With *range* I mean the interval between the highest and the lowest note in the melody.

> *During learning, the imitation errors of an agent will be lower when the agent learns a human-made melody, than when it learns a random melody of the same tone range and length.*

Figure 4.1: Hypothesis 1

We will look at the networks in isolation to get a clear picture of what is going on. This means that the agents do not communicate here. Their networks are just trained on melody input. We can see the agents as passive entities that absorb the information presented to them. We will give them perfect hearing capabilities as well. They will always infer the right pitches that are played to them on the synthesizer. In this way there will not be any errors due to sound-analysis. Technically this means that the agents get direct access to the relative pitch values of the played melodies. By doing this we know that the neural networks are responsible for any imitation error that will occur.

4.2 Can we see the knowledge of the learning agents in their compositions?

When we have an idea of what an agent brain is capable of, we will wake the agents up to compose some music for us. Here we will analyze their compositions while they are learning. As I explained before, the agents compose by generating relative pitch sequences with the neural-networks and the grid of transition tables.

The knowledge is coming from the coded songs, which are well known successful songs composed by humans. To obtain this knowledge an agent is said to study a song. This means that substrings of one of the coded songs are presented as input for the neural network of the agent. During the learning period an agent is composing several times. To see whether it uses its knowledge the compositions are compared with the song it is studying.

By learning, the knowledge of an agent increases because the network and the grid of transition tables adapt to the input. If the agent uses this knowledge we should see an effect in its compositions. This leads to the following hypothesis:

> *During learning, the compositions of an agent become more and more similar to the song it is studying.*

Figure 4.2: Hypothesis 2

The agents use the grid of transition tables to learn the relation between successive elements of the melody, represented on their output map. We can call this *grammar* learning. Now we can ask the following question: *Are the agents using the grammar in their compositions?*

To answer this one, we will compare different grids. If the grid of transition tables is very coarse, the agent is not able to learn many details of the grammar. If it is very fine, more details can be learnt. An agent can use this grammar in its compositions. If an agent indeed does so, then hypothesis 3 holds.

> *If agent A has finer grid of transition tables than agent B, then the compositions of agent A will be more similar to the song agent A is studying than that the compositions of agent B are to the song agent B is studying.*

Figure 4.3: Hypothesis 3

Finally if an agent is learning really well, and it is using a lot of its knowledge while composing, then we should be able to pick out the song it is studying by looking at the compositions of the agent. If this is true then hypothesis 4 is true.

> *After learning, the compositions of an agent are more similar to the song it studied than to another song of the set of coded songs.*

Figure 4.4: Hypothesis 4

Note that hypothesis 2 can be true while hypothesis 4 is false. When for example during learning, the compositions of an agent become more and more similar to *all* the songs of the set of coded-songs, then hypothesis 2 is true. At this point it could still be that the compositions of this agent are more similar to another song of the set of coded-songs than the one our agent is studying.

4.3 What is the effect of more realistic hearing capabilities on the learning process of the agents?

First of all I will explain what I mean by realistic hearing capabilities. I do not pretend that the ears of the agents are like human ears. What I mean here is that most people make some mistakes when they have to imitate a melody. Part of the mistakes are due to hearing the wrong intervals. The agents, like humans, will make mistakes. They will infer a wrong pitch every now and then, because the pitch analysis algorithm is not perfect. That is what is important here and that is what I mean with *more realistic* hearing capabilities. The timbre of the instrument plays an important role as well. Some instruments have very bright sounds while other instruments are more percussive. An agent should have less problems analyzing melodies played by the former than those played by the latter instruments.

Again the agents are trained with the coded songs. The hearing capabilities remain the same during learning. The ears do not get better. Also the instrument that plays the training-melodies remains the same during learning. When the agents imitate we expect that different instruments give rise to different but constant analysis errors, as is stated in hypothesis 5.

> *During learning, depending on the timbre of the instrument, the imitation error will be increased with a constant amount of noise which is the analysis error.*

Figure 4.5: Hypothesis 5

This results in different knowledge due to the noise, because the agents will learn the melodies they *hear* and not the ones that are *intended*. This means that they will adapt their weights to the analyzed wave file, instead of the substring that is drawn from the song.

> *Due to the analysis error, the agent develops a category distribution on its output map that does not resemble the category distribution of the melody.*

Figure 4.6: Hypothesis 6

4.4 What is the effect of communication by playing imitation games on the learning process of the agents?

When two agents communicate by playing imitation games, they learn from each other's compositions. In the situation of two agents studying each a different song *and* playing imitation games, we expect that hypothesis 7 holds.

> *The agents will learn a mix of their own song and the song of the other agent.*

Figure 4.7: Hypothesis 7

The agents will learn from each others culture and the result is a cultural perhaps a melting pot. Will we be able to see a similarity to people in a multi-cultural society?

Chapter 5

Experiments

In this chapter I will describe the experiments, that are designed to answer the research questions. Appendix A contains the parameter settings for every experiment.

5.1 Experiment on learning agent brains

As I described in section 4.1 we expect that human-made melodies are easier to be learned by our agents than randomly generated ones. We should see this difference in the imitation error according to hypothesis 1. Remember that learning takes place when we see a decrease in the imitation error. We set up a test to see if hypothesis 1 is true.

We will pick three different songs from our coded human-made songs collection and we will compare them with three matching random songs. Every song will be studied by four different agents. Each of those four agents will have a different brain size. This means that their output maps have a different number of output neurons which has different learning capabilities as a result.

As I explained in section **??**, the output map of a SARDNet is a two dimensional map of nodes. The length and breadth of this map can be set in the initialization file [1]. The agents of this test have the output map sizes as displayed in table 5.1.

The neighborhood is related to the size of the output map. It determines the size of the area on the output map that adapts to the input. This area

[1]See appendix A

agent	length	breadth	total number of output nodes
1	4	4	16
2	5	5	25
3	7	7	49
4	16	16	256

Table 5.1: The Agents and their different brain sizes: Length and breadth are the dimensions of the output map

is large in the beginning and shrinks during learning [2]. To make the effect of the neighborhood similar for all the networks of different size, we give the agents neighborhoods that cover precisely the whole output map during the early stages of learning.

There are six agents of every brain size, so we have one agent of every brain size for each song. We expect that the ability of those agents to learn these songs is different for every brain size, but that hypothesis 1 will hold for all agents.

First I will describe the details of the selected human-made songs, then I will explain how I generate the matching random songs. Finally I will explain the details of the experiment.

5.1.1 Human-made songs

The songs I have selected for this test are:

- Chinese Gold Snake Dance
- Invention 1: J.S. Bach
- Etude Op.10 N.1: F. Chopin

They are very different from each other as can be seen in table 5.2. We see that the compositions have a different number of notes. Especially Chopin has a very large sequence of more than 1200 notes.

[2] see section 2.3.3

song	Nr notes	input melody length	melody range
Gold Snake Dance	222	4	17
Bach	237	10	24
Chopin	1205	16	70

Table 5.2: These compositions have the same range and input-melody length as their matching random songs

The input-melody length is made different for every song. When an agent is studying a song, substrings from this song are drawn randomly. These are the input-melodies, or training melodies. I have set different lengths for the different songs, to make the learning tasks more different. We see a tendency from the short Chinese Gold Snake Dance with its small melody range, from which small input melodies are drawn, to the very long song of Chopin, with a large melody range from which longer input-melodies are drawn.

Now we are going to look at graphs of the songs. In this way it is easier to get a feeling for the training input. The relative pitches are plotted against the note position in the song[3]. The graphs are displayed in figure 5.1.

Note that the scale on the y axis of chopin is much larger than that of the other compositions. This composition is so long that the graph is split up into five blocks of length 201, to obtain a readable result. Chopin's composition has interesting changes from periods with upward intervals to periods with downward intervals. The picture of this composition looks much less chaotic than that of the other song.

5.1.2 Random melodies

The random melodies have to match the selected songs. For this reason I have implemented a random melody generator. The goal of this generator is to create a list of random relative pitches that results in a melody with a range that matches one of the three selected songs. For example, *Invention No 1. of Bach* has a melody range of 24. This means that the whole melody fits in two octaves. The lowest note and the highest note that occur in the melody have an interval of two octaves.

Now when we are going to make our random melody, we like it to have this

[3]See appendix C: C.2, C.4, and C.1 for the data of these graphs

range of two octaves. We select randomly a starting position p in the range $[1, 24]$. Let's say it is 12. Now we get a random new position np, in the same range. Let's say $np = 4$. The interval i is $np - n = 4 - 12 = -8$ To go from the old position to the new position we need to go down 8 steps. The first element of our random melody will be -8. Now the old position n becomes the new position np: $n = np$, and we select a new np from the range $[1, 24]$. This process continues until we have obtained the desired melody length.

We like to match our three random songs to the selected human-made songs. Therefore we need to generate random sequences with melody ranges 17, 24 and 70, and the melody lengths of 4, 10 and 16.

5.1.3 Experiment details

Now that we have two matching sets of songs we can run the test. The 24 agents are studying their songs for 10000 learning cycles. Every learning cycle the agents receive substrings from their songs of the specified length, if they are assigned to a human-made song. If they are assigned to a random song, a random sequence of the specified length and range is generated.

These melodies are played on the synthesizers and the agents listen to them. They can hear every interval without error since they have perfect ears. They will try to imitate the melodies they "hear". The agents receive feedback as well, which is set to a maximum. This means that the agents are sensitive to the feedback they receive on their imitations.

During the first 5000 learning cycles the neighborhoods of the networks of the agents are shrinking rapidly as is common in kohonen feature maps [7]. After 5000 training cycles all neighborhoods have the same, very small size. They will cover more or less one neuron. This is called the fine tuning phase.

After the agents studied 10000 cycles, they are "killed" so we can look at their brains to see what they have learned.

5.2 Experiment on knowledge in the agent compositions

Remember hypotheses 2, 3 and 4 in section 4.2. Hypothesis 2 says that the compositions of an agent should get closer to the song it is studying. Hypothesis 3 says that more transition tables and hence, smaller and more corresponding areas on the output map, result in compositions that are closer to the song the agent is studying. Finally hypothesis number 4 says that compared to the other songs, the compositions of the agents will get closer to the song it studied.

First I will explain how melodies are compared. Then we will look at two tests that should give answers to the hypotheses.

5.2.1 Melody comparison

To answer this research question we need a way to compare a melody composed by an agent with the song this agent is studying.

In the model *euclidian distances* are used at many places for comparison. To measure the success of an imitation we compute the euclidian distance between the imitation and the input song. The euclidian distance is also used in the competition between the output nodes. Remember that the winning neuron is defined as the neuron with the smallest euclidian distance between its weight-vector and the input-vector. Differences between melodies are calculated this way in the model, and it is therefore reasonable that we use it as well, when we compare the compositions of an agent with the song it is studying [4].

An agent does not see the whole song at once, it receives substrings as input and generates strings of the same length as output. When we would like to say something about the similarity between the song and the compositions of an agent, we need to compare them with substrings of the same length of the song.

To test the hypotheses I will create societies of two agents. Each agent in a society has a different song. They will study it for approximately 10000 learning cycles each, so the total number of training cycles in the society is $2 * 10000 = 20000$. Every learning cycle an agent is picked out randomly and presented with a input melody from its song played on a synthesizer. Again the agents have perfect hearing capabilities. The agents do not interact, they only learn their own songs. Every 1000 training cycles both agents are tested on their knowledge. This is done in the following way:

Every agent will play 1000 compositions as described in the composing section [5]. Every composition is compared to the song the agent itself is learning and to the song the other agent is learning.

---

[4]See the discussion chapter 7 for a discussion of the euclidian distance measurement and other methods of comparison

[5]See section 3.4.2

To do the comparison, all possible substrings of the same length as the composition of the agent are generated from a song. The substring with the smallest euclidian distance to the composition is selected. We could say that if the agent was composing with the knowledge of this song, this substring was most likely to be the *inspiration* for its composition. This euclidian distance is stored for each of the 1000 compositions of the agent and the mean of the resulting 1000 euclidian distances is calculated. In this way we get an euclidian distance for the average composition.

For both songs this euclidian distance is calculated, so it is done for the song the agent is *not* studying as well.

Now we have a measurement to compare the compositions to *both* songs. We can see whether the compositions of an agent are closer to its own song, or to that of the other agent's song.

5.2.2 Grid sizes and composing

We will look at agents with different grid sizes while holding the size of their output-maps constant. Also the length of the melodies will be equal for all agents. The output maps have 100 output nodes (10 times 10) and the length of the input melodies will be 10.

Firstly we will look at societies where one agent is learning *Invention no. 1* of Bach and the other is learning the *One Note Samba* of Jobim[6]. We will test four of these societies. The following table shows the different sizes of grids of transition tables:

society	Grid: length	breadth	nr of transition tables	size of an area in nr of output nodes
1	10	10	100	1
2	5	5	25	4
3	2	2	4	25
4	1	1	1	100

Table 5.3: The four societies of agents with their different grid sizes. The two agents in every society study *Invention No 1* of Bach and the *One Note Samba* of Jobim.

[6]See C.2 and C.5

See for an explanation of the transition tables section 3.1.2.

5.2.3 Two Chinese songs

Finally we look at a society with two agents that study Chinese songs. They study the *Gold Snake Dance* and the *Traditional* song[7]. These songs are much more similar to each other in their use of intervals, than *Invention No. 1* and the *One Note Samba* are. This society has almost the same parameter settings as society 1 of table 5.3. The only difference is the song the agents are studying. We compare the results of this society with the Bach and One Note Samba studying agents of this society 1. This society with Chinese song studying agents is investigated to check whether the results found before in this section are song dependent.

5.3 Experiment on hearing capabilities of the agents

Hypothesis 6 states that the knowledge of the agents will be distorted by the analysis error, caused by their less than perfect hearing capabilities [8]. Before we are going to test this, we like to know a little more about what the agents can hear, and what not. In this model melodies consist of intervals. When a melody is played on a synthesizer there is a choice. It can be played anywhere on the keyboard. For agents with perfect ears, this does not matter, since they will infer the right pitches and hence the right intervals. For agents with more *realistic* ears, this is a different story. It can be that some frequencies are outside the scope of the ear. Therefore we start by giving our agents a hearing test. This will give us an indication in what frequency-range we have to play our melodies. Once we know this we can go on to the main test.

5.3.1 Hearing test

To have an idea of the performance of the agent-ears I made a test melody of length 5. The frequencies of the notes that are played are: $125Hz$, $250Hz$, $375Hz$, $500Hz$, $625Hz$. This test-melody is played on the seven FM-instruments

[7] See appendix C: C.4 and C.3

[8] See section 4.3

of the STK-library [5]. If the pitch detection is perfect, the same frequencies should emerge as output. If the agents have problems hearing these frequencies then we should see different frequencies as output.

5.3.2 Main test: Learning and realistic ears

Based on the results of the hearing test we have an indication of what frequency range is suitable for our agents. When we are going to train the agents with a song, we will play it in this suitable frequency range.

The training song will be the *One Note Samba*, because it has a small melody range, and lots of prime intervals, so the job for our agents is not too hard[9]. The melody range is 15. This means that the highest note of the song is one octave and a third higher than the lowest.

I will look at eight agents. Seven of these agents have the same type of *realistic* ears and one agent has *perfect* ears. This latter agent has a benchmark function to compare the effect of *realistic* ears in general with *perfect* ears. Apart from the ears the agents study in the same conditions. That is, they will study the *One Note Samba* for 10000 learning cycles. They receive melodies of length 10 and they have the same brain-sizes. That is, output maps of 100 nodes (10 times 10).

The seven agents will receive their melodies, played on the seven FM-synthesizer instruments of the STK-library. For agent number 8, the one with the perfect ears, it does not matter on what instrument the melody is played, since this has no effect. Technically this means that it gets access to the sequences of relative pitches straight away.

5.4 Experiment on communication by playing imitation games

Hypothesis 7 states that the result of communication is that the knowledge of the songs the agents are studying, is mixed. In the extreme case we wouldn't recognize which agent learnt what song. If we like to study the effect of communication on the knowledge of the agents, we need a phase in the life of the agents where they study music and a phase where they communicate. I will discuss this issue first. Then I will explain the test.

[9]See C.5

5.4.1 Phase change

The problem is clear, there are two phases in the society, the *music study* phase and the *communication* phase. The question is: *How do we change from one phase to the other?*

It is possible to design a society with a sudden phase-change. This means that up to a certain point the agents always choose the *music study* action. After this phase-change point the agents always choose the *communication action*. We can say that the agents discover each other at this point and start communicating. They don't want to study their music anymore and they will spend the rest of their lives communicating.

An alternative is to change this action-preference more gradually. In the beginning both agents only study their own music, but slowly they prefer to communicate more. The older the agents get the less they study, and the more they communicate.

We can see a metaphor here. First the agents are nursed by their parents when they are infants. This is the music study phase. Then when they grow older they become more and more independent and they will act by themselves in the society. That is the communication phase. When we apply this metaphor of phase change from being a child to being an adult, to the both types of phase changes I discussed above, we can see the sudden phase change as a kind of initiation ritual to adulthood, as is common in many tribes around the world. We can see the more gradual phase change as a way many people grow up in the west.

5.4.2 Test

To study the effect of communication on the knowledge that is built up, we will test these two types of phase-changes. Both of them start with the music study phase and end with the communication phase. As a benchmark we will run a similar society where the agents do not communicate at all. These agents are isolated, and study their music in solitude. They do not know about each other's existence. Until now we have mainly studied agents in the latter situation.

In the next figure we see the action preferences plotted against the society-time. The y-axis represents the probability that an action is chosen. The red line represents the probability to choose the communication action while the blue line represents the probability to choose the music study action. When we take the sum of the probabilities of both actions at any point in time, it always adds up to 100%. The agents always choose one of the two actions.

The agents in these three societies study Bach and the One Note Samba[10]. To look at the effect of communication without too much noise the ears of the agents are made perfect again, what means that they always infer the right pitches. The grid of probability tables has dimensions of 10 times 10. The agents live for 20000 learning cycles.

In section 5.1.3 I described how the neighborhood of the SARDNet of an agent shrinks during learning in these tests. The first half of the training cycles the neighborhood is large and shrinks fast. In the second half of the life time of the society of agents the neighborhood is very small, and it shrinks slowly. In this way we generated a broad-category learning phase and a fine tuning phase. During the second half of the training the agents are not able anymore to learn completely new input, because of the small neighborhood. They only learn the details of the input.

In the societies of this experiment the communication phase is in the second half of the agent life time. When the agents reach the moment that the majority of their actions involve communication they need a neighborhood that is large enough to pick up all the new information that is coming to them from the other agent. If the neighborhood is already too small, then the agents will not be able to learn much from communication. That is, the weights of their networks do not adapt much to the input.

Therefore I have increased the amount of time where the neighborhood is large. In the societies where the agents communicate, this part takes $\frac{3}{4}$ of the total number of learning cycles. Since communication starts halfway, the agents will learn from communication.

Now we are ready to run the test. In all of the three societies, during 20000 cycles an agent is picked out randomly. It decides what it will do, *communicating* or *studying music*, according to the probabilities outlined in the graphs in this section. If the agent decides to study its music, it will receive a substring of its song as training input. It will imitate it and learn accordingly. If the agent decides to communicate, then it will compose a melody, based on its knowledge. The other agent hears this melody and will imitate it, and learn accordingly.

[10]See appendix C: C.2 and C.5

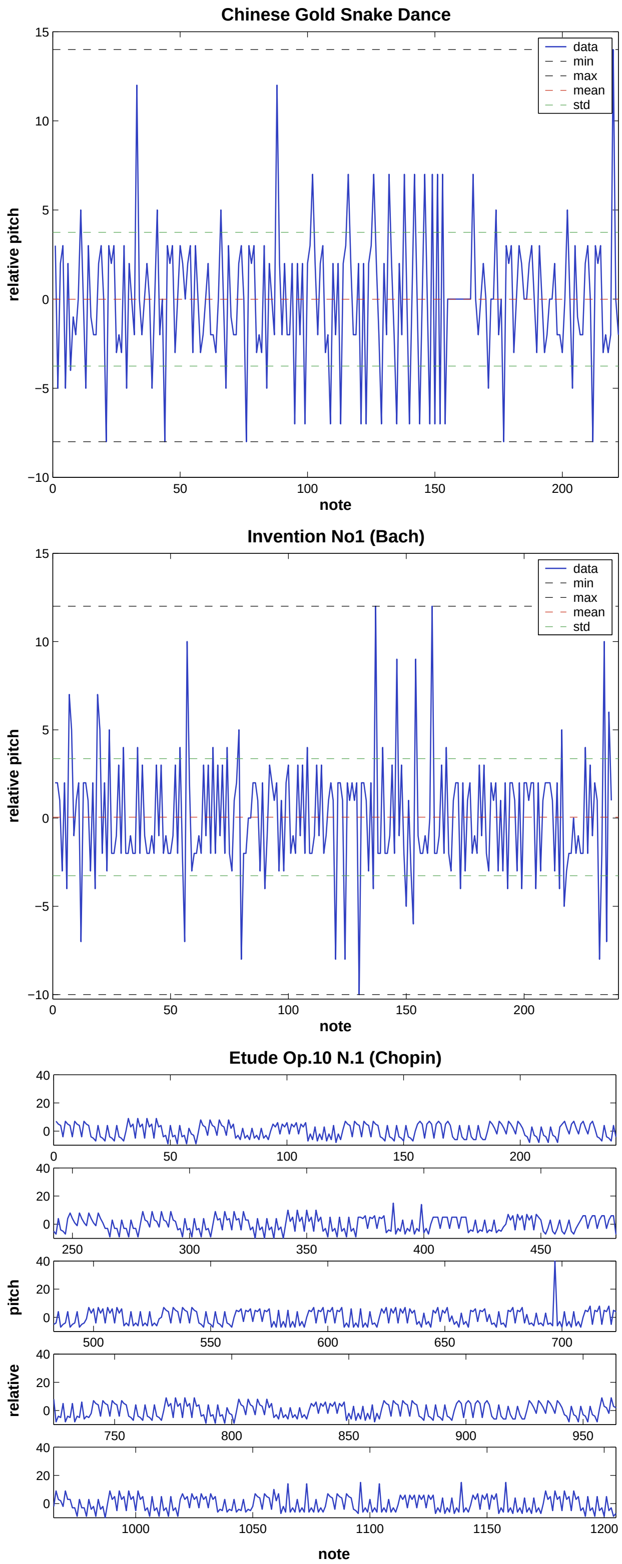

Figure 5.1: The relative pitch is plotted against the position in the song: Chinese Gold Snake Dance, Invention1 of J.C.Bach and Etude Op.10 N.1 of F Chopin

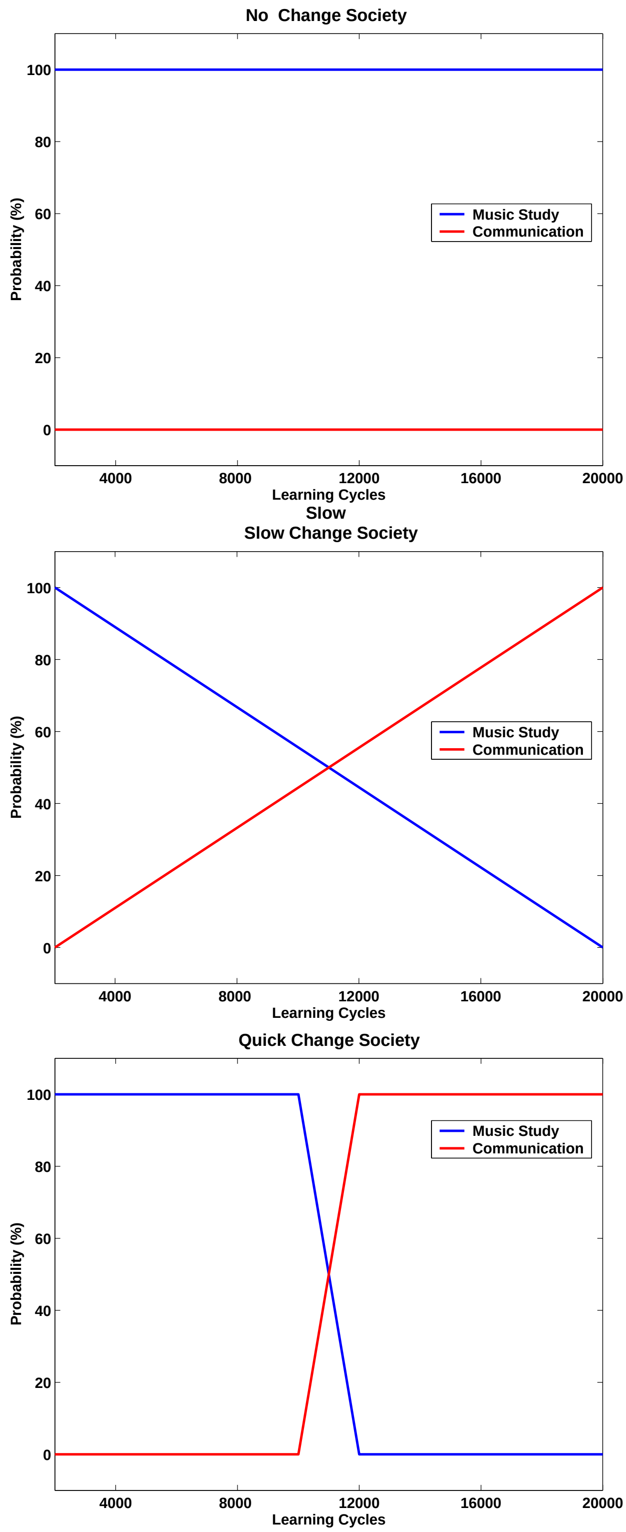

Figure 5.2: The three types of phase-changes in the societies of agents.

Chapter 6

Results

In chapter 4 I posed four main research questions and in chapter 5 I described the experiments designed to answer them. In the current chapter I will discuss the results of those four experiments. Firstly I will discuss the results of the experiment on the learning agent brains in section 6.1. Then I will continue with the results on the experiment that answers the question: "Can we see the knowledge of the agents in their compositions?" in section 6.2. The third experiment was on *realistic* hearing capabilities which results are discussed in section 6.3. Finally I will discuss the results of the communication experiment in section 6.4.

6.1 Results on learning agent brains

We have agents with different brain sizes, we have human-made songs of different complexity and we have matching random songs of different complexity. Now it is time to compare the two groups of agents: Those that studied human-made songs and those that studied random songs. We look at the effect of brain-size as well.

First we will have a look at the imitation errors and then we look at some snapshots of the agent brains to see what has happened there. These brains, or output maps, contain the categories the agent has learnt. When an agent has to imitate a melody, it has to build this imitation from the categories. Therefore the categories on the output map have an immediate effect on the imitation error.

6.1.1 Imitation errors

We compare the imitation errors of all four different brain sized agents that studied a human-made song, with their random-song studying brothers and sisters. In the following graphs (See figure 6.1, 6.2, and 6.3) a moving average of length 100 of the imitation errors of the agents is plotted against the *time* in the society. Training cycle 5000 is marked with a dotted line. This line indicates the start of the fine tuning phase, where the neighborhood is very small and the agents only learn the details of their input.

The agents are grouped according to the song they have studied, so we see in every graph the imitation-errors of four agents, that is one agent of every brain size. The results of studying the human-made songs are displayed on the top graph of a figure and the results of the matching random songs are displayed on the bottom graph. The scale of the Chopin graph is so large that the details of the graph disappear. I display it in this scale to match it with the random song that had errors of around 80 at the start. To see what the learning curves of Chopin's song look like I display this graph in more detail as well in figure 6.4.

We can see for every agent a steep learning curve that settles on a constant value. Here we see the effect of brain size very clearly. The constant value of the error is different for agents with a different brain size. According to the graphs an agent with a larger brain makes smaller imitation errors when it has learnt the song. In this model brain-size matters. We can say that in this model a larger brain makes the agent more intelligent, if we regard this song learning metaphorically as a kind of IQ test.

The time it takes to reach this constant imitation error value is dependent on the brain size. A larger brain needs more time. The only exception is the *Chinese Gold Snake Dance*, where the 7 times 7 agent manages to reach the same imitation errors as the 16 times 16 agent. This indicates that this song, with input melodies of length 4, is so easy to learn, that the 49 output nodes of the 7 times 7 agent are sufficient to reach an imitation error of around zero.

When we compare the graphs of the human-made songs with their random counterparts, we can see in all the three matching songs the difference in error. In all conditions the error is much smaller in the human-made songs. It is smaller at the start and it is smaller at the point where the agents settle their imitation errors. The error difference is larger when the composition and the random song are more complex.

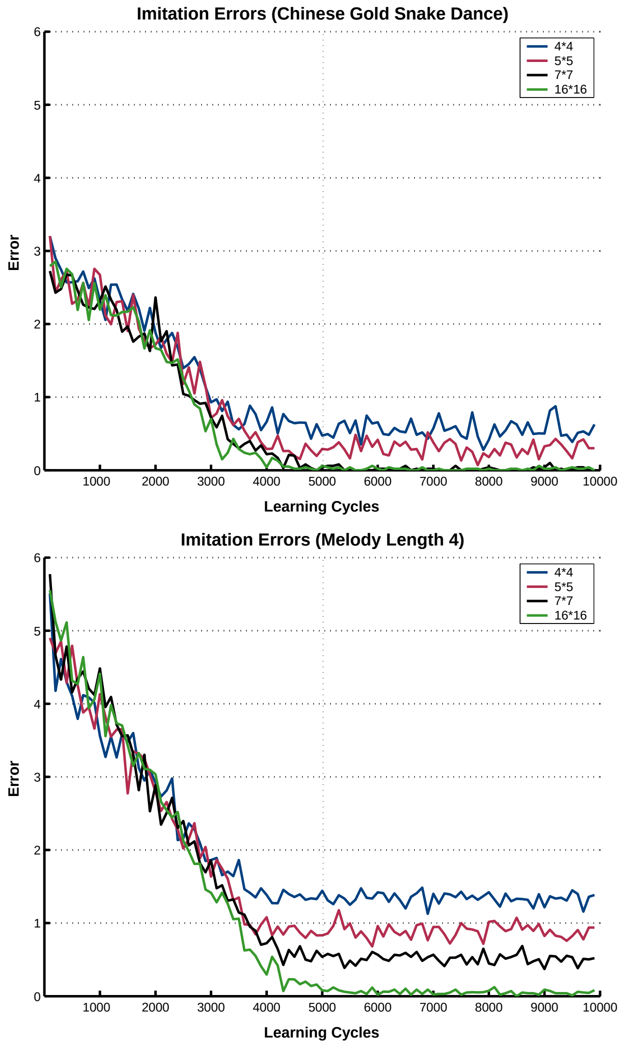

Figure 6.1: Comparison of the imitation errors of the agents, trained on the Chinese Gold Snake Dance and a similar random song

It is interesting to see that in all the graphs of the random songs, the error decreases globally in a straight line. In the human-made compositions this is not the case. In the Chinese Gold Snake Dance, and in the composition of Bach the error decreases faster and faster. The inclination in the graph becomes steeper and steeper.

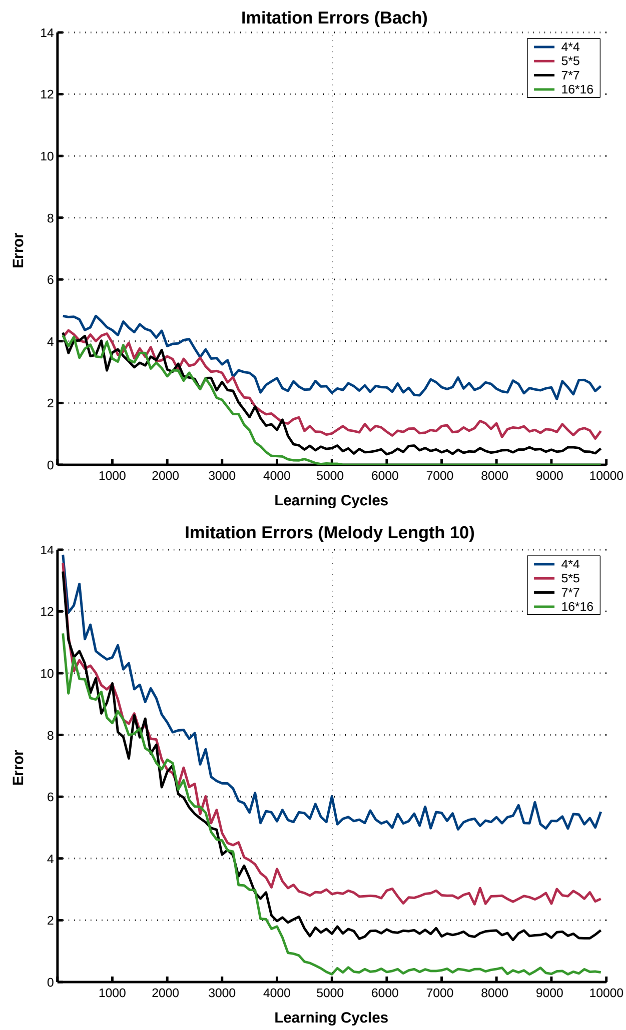

Figure 6.2: Comparison of the imitation errors of the agents, trained on Invention 1 of J.S.Bach and a similar random song

The detailed picture of the error of the Chopin composition does not show a clear way of learning. What is interesting here is that the 4 times 4 agent does not learn much and that this is the only condition where the error differences between the agents are large from the start. In all the other graphs, the errors of the agents are more or less the same in the beginning, and diverge during learning. This does not happen with the Chopin studying

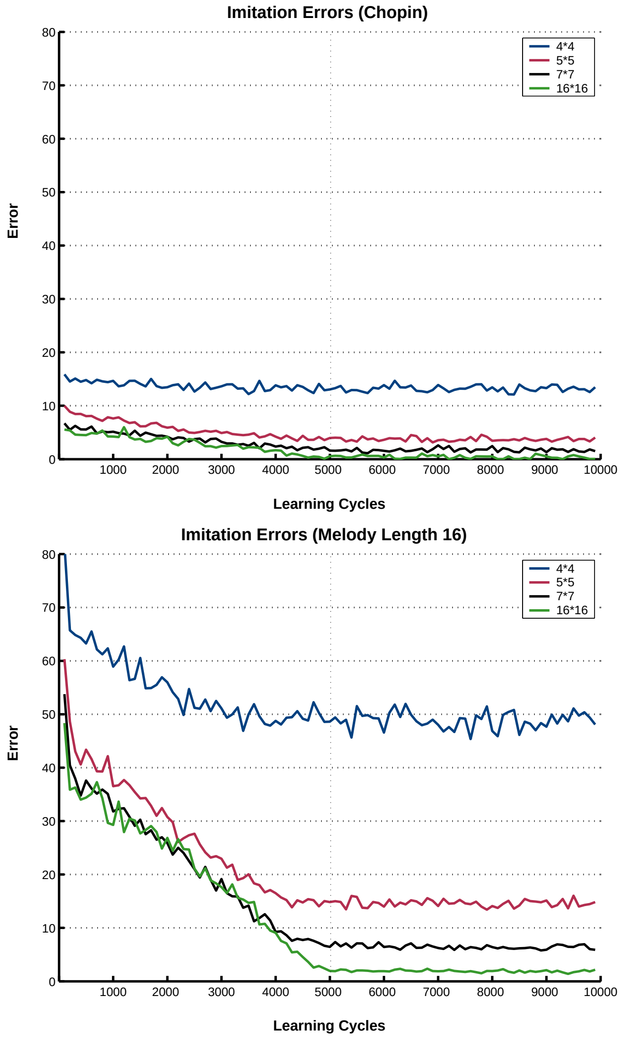

Figure 6.3: Comparison of the imitation errors of the agents, trained on Etude Op.10 N.1 of F Chopin and a similar random song

agents, nor with the *random counterpart* studying agents.

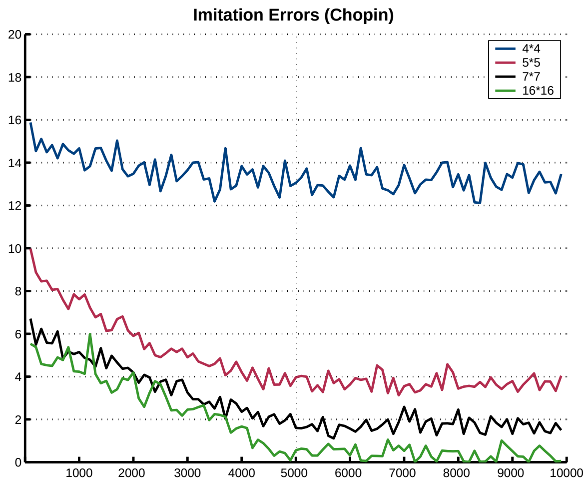

Figure 6.4: Smaller scale graph of the same data: the imitation errors of the agents trained on Chopin

6.1.2 Brains

What categories are learnt by the agents? Here we take a look at the brains after learning. We look at the 16 times 16 agents that studied the Gold Snake Dance and Chopin and the same brain sized agents that studied the matching random songs [1]. Snapshots of their brains after learning are displayed in figure 6.5.

Note that every agent chose another corner to store its highest and lowest categories and that there is a large difference between the agent in the lower right corner and the others.

The colors are coded from -60 to 60. This is more or less the range of the categories on the map of the agent in the lower right corner. This agent studied the random song matching Chopin. This song and Chopin's composition have an extremely large melody range of 70. This means that for the random melody, intervals of 70 and -70 are possible. Chopin's composition uses the maximum and minimum intervals of 40 and -10. This is much less than in the random melody. The Gold Snake Dance has maximum and minimum intervals of 14 and -8. It has a range of 17, so the random song has 17 and

[1]The agents with another brain size showed the same overall result

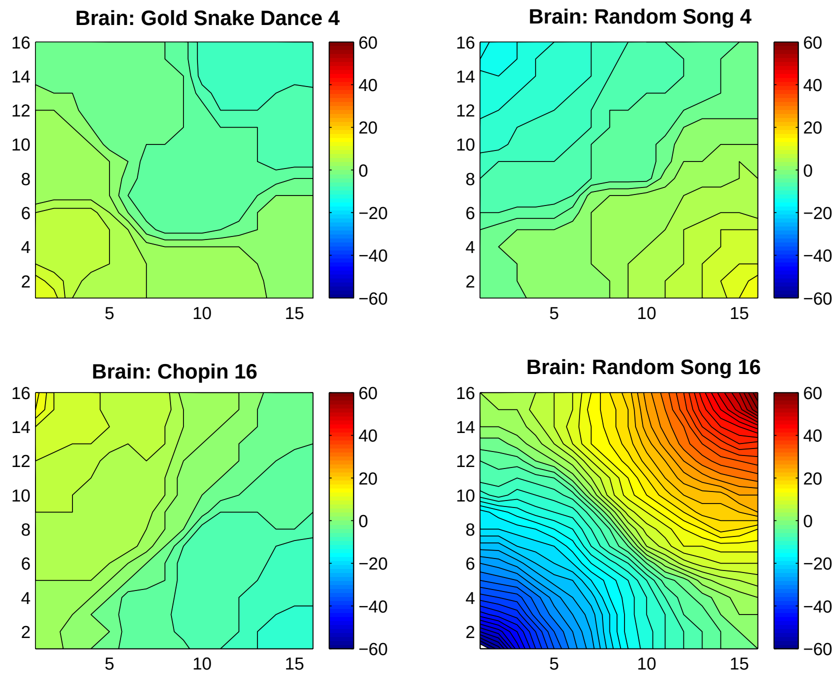

Figure 6.5: Comparison between the brains of the agents after training on the Chinese Gold Snake Dance, Etude Op.10 N.1 of F Chopin and the two similar random songs.

-17 as its maximum and minimum. This is why the agent in the right corner has a so much wider range of categories than the other agents.

Now we look at the highest and lowest categories in the brains and the largest intervals in the input songs. The highest and lowest categories we see on the agent brains are the limits of what the agents have learnt. If the agents have learnt the songs perfectly, the highest category would be equal to the largest upward interval of the song. The same holds for the lowest category, which should be equal to the largest downward interval of the song. Table 6.1 shows these intervals, together with the highest and lowest categories on the output maps of the agents that learnt these songs.

We see a large difference between the highest category in the Chopin composition and on the brain scan of the corresponding agent. This large interval of 40 occurs only ones in the whole composition. Furthermore the composition is 1205 notes long and an input melody, which is a substring of

	song high	map high	difference	song low	map low	difference
gold snake dance	14	12	2	-8	-8	0
random song1	17	14	3	-17	-15	2
chopin	40	15	25	-10	-10	0
random song 3	70	62	8	-70	-63	7

Table 6.1: This should clarify the last brain scan. The highest and lowest categories on the brains are displayed, together with the highest and lowest categories in the songs.

the whole song, is only of length 16. This means that there is a probability of $\pm\frac{1}{75}$ that this interval is present in an input melody, when substrings of length 16 are drawn from the 1205 note long composition randomly.

6.2 Results on knowledge in the agent compositions

First of all we are going to look at the effect of the size of the grid of transition tables. For this test we have two agents of every grid size[2], studying the *One note Samba* and *Bach*. Then we will look whether the results we found can be generalized, by investigating a society of agents where the agents study two Chinese songs [3].

6.2.1 Grid sizes and composing

In the experiment section 5.2.1 I proposed a similarity measure to compare the compositions of the agents with the songs. This is done by calculating the average euclidian distance of 1000 compositions of the agents and the closest substring of the song.

First we look at the distances of the compositions of the agents to the songs in the societies where the agents have grids of 10 times 10 and 1 times 1 transition tables. If the transition tables have any effect, it should be most

[2]See the experiment description in section 5.2.2

[3]Appendix B contains the complete grid of transition tables of the 5 times 5 agent that studied the One Note Samba. This is meant as an example to get an idea of what it looks like after the agent has learnt.

clear in these two extreme conditions. We look whether the distances are different or not.

Then we will display a classification of the agents of all four societies of the different grid sizes test. Remember that there are two agents in every society, each studying a song independent from each other. Both agents in a society have the same grid size. The compositions of an agent have more characteristics of the song the agent studies if the distance to this song is smaller than the distance to the song of the other agent. We subtract the distance to its own song from its distance to the song of the other agent. If we are left with a positive number, then the agent is classified as closer to its own song. If the number is negative then the agent is closer to the song of the other agent and classified accordingly.

We divide the agents in two groups according to their songs, to look how the agents of every grid size are classified. Then we will calculate the average of the classification of all the agents that study Bach and that of all the agents that study the One Note Samba. In this way we can look at the effect of the song on the classification measure.

A comparison of the effect of a grid size of 10 times 10 and of 1 times 1

What we see in figure 6.6 is that the distances to both of the songs are growing! No matter what song an agent is learning, its melodies will become more different from both songs. This happens in both grid size conditions. We can see a difference between the agents. The One Note Samba agent produces melodies that are closer to its own song. On top of that, its melodies are also closer to Bach than the compositions of the Bach studying agent are to Bach. The Bach agent produces compositions that are closer to the One Note Samba.

When the agents reach their stable phase after 5000 learning cycles, we see in the 10 times 10 condition that the distance of the compositions to the song become a little bit smaller. This does not happen in the 1 times 1 condition.

The Bach agent doesn't seem to benefit from more transition tables since its compositions are more distant to its own song in the 10 times 10 condition than in the 1 times 1 condition. Based on this graph, there is no difference between the conditions for the One Note Samba studying agent.

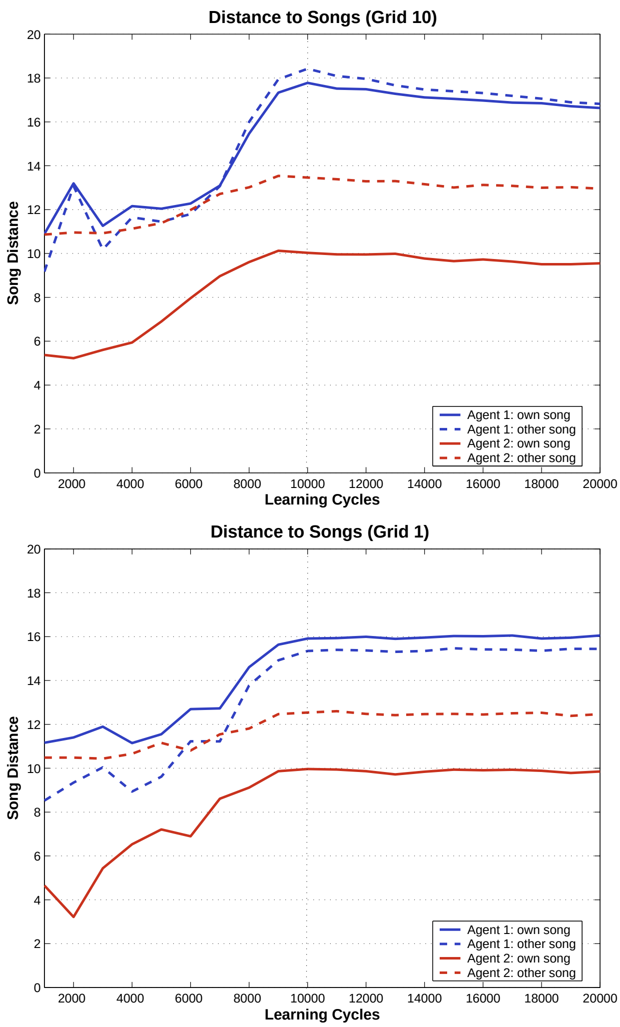

Figure 6.6: The distance to songs of the melodies generated by an agent trained on Bach and on the One Note Samba. The solid lines are the distances to the agents own song, and the dashed lines are the distances to the song of the other agent. The grid sizes are 10 times 10 and 1 times 1.

Classification and Grid Size

Now we subtract the distance of the compositions of an agent to its own song, from the distance to the other agent's song, to obtain our classification measure. Figure 6.7 displays the classifications. The agents grouped in the graph “Agent 1” studied Bach. The agents in “Agent 2” studied the One Note Samba. The graphs here show the classifications of both agents in all four grid-size conditions. Remember that positive numbers indicate that the agent composes songs closer to the song it studies.

The main difference between the agents is that the One Note Samba agent always composes songs that are closer to the One Note Samba, while the Bach agent produces songs that are more similar to the One Note Samba than to Bach. Only in the 10 times 10 condition the classification of the Bach-agent is in favor of Bach. We did some more runs and it turns out that this is due to noise. In general we don’t see a close relationship between the grid size and the classification.

Classification and song

When we take the average of the four conditions for each agent and plot it, we see that both agents are different. The song an agent is studying is much more important than the size of its grid. We also see here that both the lines in the graph have an almost opposite shape. When the classification line of one agent goes in the direction of its own song, the line of the other agent becomes more like the song it is *not* studying. Remember that the agents are *not* communicating. They don’t know about each other’s existence.

Part of this shape can be explained by the nature of the One Note Samba and the way agents learn. The agents start with zeros on their output map. This means that they assume that melodies consist of prime intervals, zeros. The One Note Samba lives up to this innate expectation, it has for most part zeros. This means that in the beginning the agent is producing melodies that are very close to the One Note Samba. However, during learning the agent is forming other categories than zeros in its brain. The result is that its melodies come closer to Bach, because Bach has larger intervals. The probability that the agent is producing melodies that do not exactly match its One Note Samba becomes greater as well. This results in a classification that is less in favor of the One Note Samba.

At the same time the Bach agent starts with the prime interval assumption as well. Therefore it is classified more as a One Note Samba agent than as a Bach agent. During learning it develops larger and larger categories on its output map. This results in compositions that are closer to its own song, hence the opposite effect. Nevertheless we see that its melodies still remain

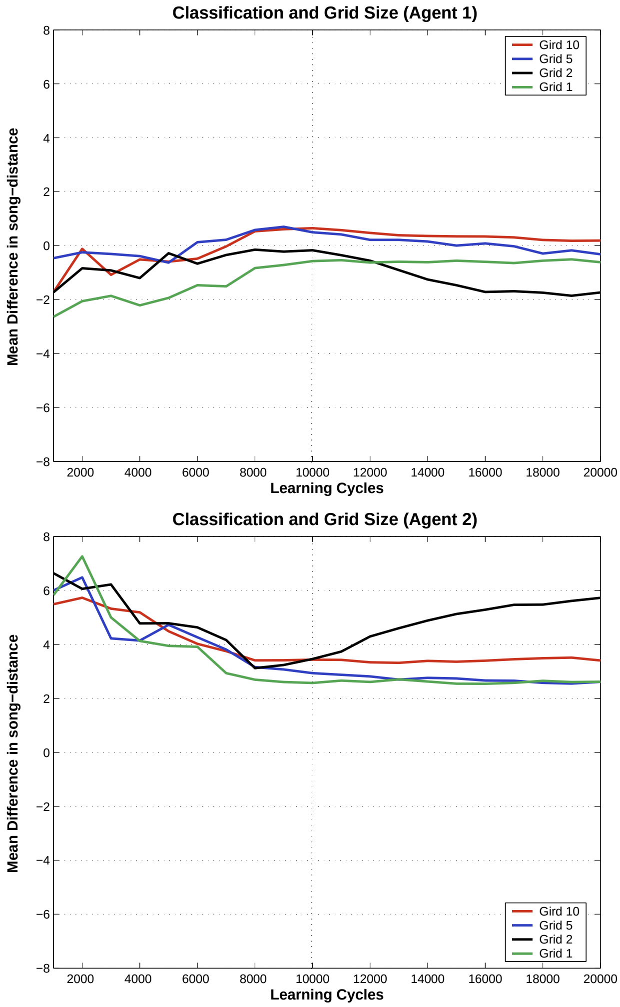

Figure 6.7: The classification is the difference between the distance to the agents own song and to that of the other agent. Four grid sizes are compared: 10, 5, 2 and 1. Agent 1 contains the Bach agents and Agent 2 the One Note Samba agents.

closer to the One Note Samba

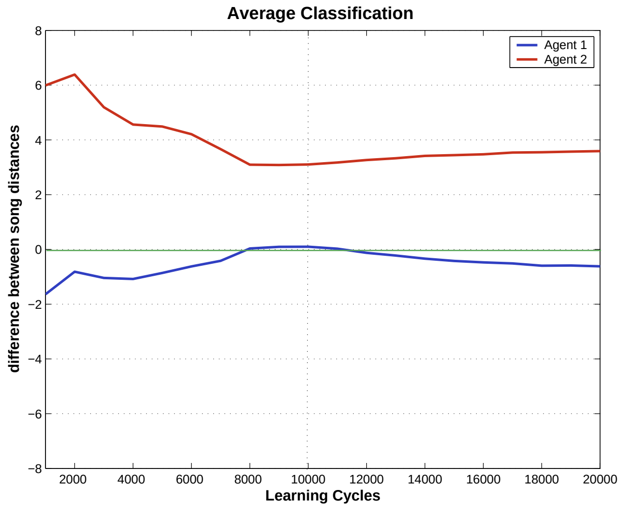

Figure 6.8: The average of the classifications of all the grid sizes is taken. In this graph we can compare the effect of the song that the agent studies. Agent 1 studies Bach and agent 2 studies the One Note Samba.

6.2.2 Two Chinese songs

The left graph of figure 6.9 shows the distances of the compositions of the agents to the songs. The right graph shows the classification of both agents. This society of agents can be compared with the society of the last test, having grid size 10 times 10.

When we look at the left graph of figure 6.9 and compare it to the 10 times 10 society of the last test in figure 6.6, we see the same pattern. The distance to both songs increases and when the agents reach a stable state the distance decreases slightly. Again, one agent is making compositions that are closer to both songs when compared to the compositions of the other agent.

The two Chinese songs are much more alike than the songs of the previous test[4]. The effect is that the dashed lines and their solid counterparts are much more similar. This results in less variation in the classification graph on the right. The agents are classified the same, both classifications are positive and close to 0, no matter how much they have learnt.

[4]see section 6.2.1

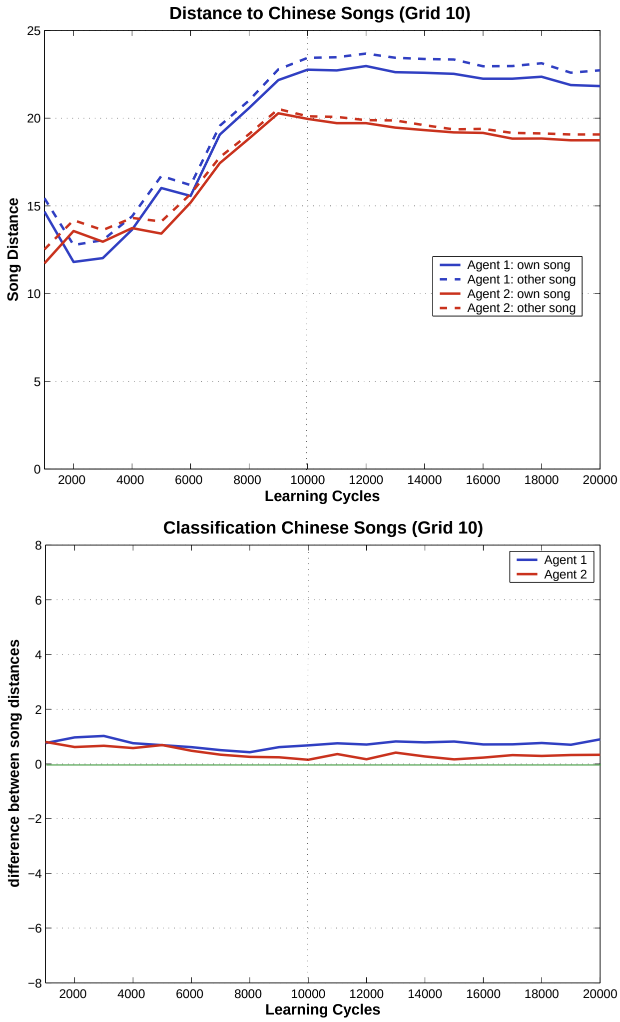

Figure 6.9: Two agents are trained on Chinese melodies. The grid size is 10.

6.3 Results on hearing capabilities of the agents

We start with our findings on the hearing test and then we will use the result in the main test.

6.3.1 Result on the hearing test

In the next table we see the result of the hearing test of our agents. The agents with realistic ears listened to five frequencies played on seven synthesizers of the STK-library. Each synthesizer having a different timbre.

input note	1	2	3	4	5
freq(Hz)	125	250	375	500	625
instrument					
BeeThree	285.544	278.775	374.589	483.096	363.577
FMVoices	244.127	249.47	374.31	482.525	345.166
HevyMetl	124.54	249.867	249.673	266.607	332.863
PercFlut	...	332.053	280.63	324.471	297.854
Rhodey	266.931	249.604	374.151	483.274	311.989
Wurley	...	252.405	374.233	265.873	298.314
TubeBell	222.155	245.313	348.53	402.109	336.987

Table 6.2: The agent-ear had to detect the frequencies in a melody of length 5, played on different synthesizers. The frequencies in the melody went up from 125 Hz to 625 Hz.

The dots in this table mean that the agent couldn't hear anything. Therefore its pitch detection algorithm didn't come with a frequency at all. Based on this table we have to determine the frequency range of the melodies. We know that the frequency doubles every octave. When we set the base frequency at 150 Hz and we allow the melody's highest note to be two octaves higher than the lowest note, then the highest possible frequency that can be played will be 600 Hz and of course the lowest will be 150 Hz. We see that the frequencies 125 and 625 are analyzed very poorly, but that the middle frequencies of this test melody are analyzed better. At least this is true for some of the instruments. When we set our range to [150, 600] Hz then we already know that the borders of our range are analyzed better than the last and the first note of this test melody. We know as well, that very high notes and very low notes cause more errors than the notes in the middle of our range.

Based on the hearing test our base frequency setting is $150 Hz$. Now we go on to the main test.

6.3.2 Result on the main test: Learning and realistic ears

First of all we will have a look at the mean of the analysis errors of the agents. The analysis error is the difference between what the agent hears, and what melody is actually played on the instrument [5]. Again we use the euclidian distance to measure this difference. These errors give us an idea of the brightness of the instruments.

After this we will look at a comparison between the analysis errors and the imitation errors. Remember that the imitation error is the difference between the melody that is presented to the agent and the melody that is played *by* the agent as a response. When an agent hears the wrong melody, then it tries to imitate the melody it hears, and not the melody that was actually played. If the agent is a perfect imitator but it has bad ears, then the imitation error is equal to the analysis error. If the agent is making mistakes in the imitation as well, then the imitation error is the sum of the error caused by the SARDNet and the analysis error.

When we look at the effect of sound on the learning process of the agents, it is also interesting to look at the input melodies after the agents have analyzed them. In this way we can see what they hear, and look whether the different instruments result in different distortions of the played melody. Recall that the agents have studied 10000 input-melodies of length ten drawn from the One Note Samba. If we put all the input-melodies together we can calculate the mean input-melody. In the case of the One Note Samba, this mean melody is a string of ten zeros. When we play the input melodies on the synthesizers and the agents analyze the sound, the analysis error will have an effect on the input-melody. If this effect is biased in a certain direction and it is large enough, it can be present in the average input-melody as well. If it is just random noise, then the averaging process should filter this out and the average input melody the agents hear should still contain only zeros. We will look at the average melodies of all the agents to see if this is true.

Finally we compare the brains of the agents. We expect that the distortion of the melodies by the analysis mistakes has an effect on the categories that are formed in the agent brains.

[5] During an imitation game it is the difference between the imitation melody that is played and what the singer agent hears, as well.

Analysis Errors

In table 6.3 we see the mean analysis errors for all the agents. Of course the "perfect ears" agent does not make analysis errors. We see that the differences between the agents are very large. The FMVoices, Rhodey and HevyMetl instruments give rise to relatively small analysis errors. Then we have the BeeThree and the TubeBell instruments that result in medium errors. Finally we have the Wurley and the PercFlut that produce very large errors.

Agent listening to instrument:	mean analysis error
Perfect ears	0
FMVoices	0.9446
Rhodey	1.9973
HevyMetl	3.0419
BeeThree	8.3059
TubeBell	10.2638
Wurley	23.8467
PercFlut	35.0994

Table 6.3: Mean analysis errors of the five agents.

Imitation Errors and Analysis Errors

Here I compare the analysis errors with the imitation errors. I have divided the instruments into two groups. There is one group with large errors and one group with small errors. I have done this because the differences between the errors of the agents are so large that the graphs need different scales to display them clearly.

Figure 6.10 displays the large errors. The left graph shows the a moving average of 100 of the analysis errors, and the right graph shows the moving average of the imitation errors. What we expect is a noisy but constant analysis error and a very noisy learning curve in the imitation error. Instead we see changes in the analysis errors as well. A very steep rise in the analysis error of the PercFlut and a more moderate but steady rise in the analysis error of the Wurley. On the other hand we see a kind of learning curve in the analysis error graph of the BeeThree instrument.

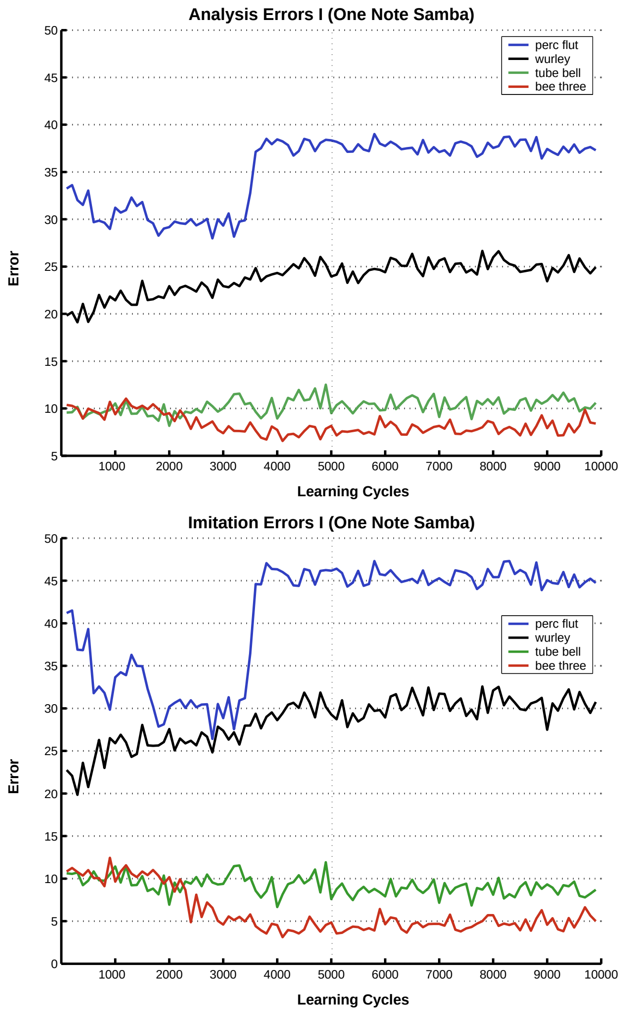

Figure 6.10: Imitation and analysis errors for the worst performing agents. They studied the instruments: PercFlut, Wurley, TubeBell, and BeeThree.

When we look at the imitation errors, we cannot see the characteristic learning curves we have seen in the other experiments, although the BeeThree might be an exception. This is not surprising. If we compare the errors with the results of the agent with the perfect ears, we see that the imitation error starts at around 2, and goes to 0. The imitation error of the agents with the realistic ears consists of such a large analysis error component that the contribution of the “clean” imitation error disappears in the total error. That’s why we cannot see any learning curves anymore.

We will turn to the instruments that give rise to smaller errors. The small-error instruments are displayed in figure 6.11. Note that the scale of the graphs is very different. We see no clear trends in the analysis errors. These analysis errors are so small that we are able to see the learning curves in the imitation errors graph.

The red line in the imitation error graph is the learning curve of the agent with perfect ears. It does not experience any effect from sound on its learning, therefore it is not displayed in the analysis error graph.

Analyzed Input-Melodies

Now we look at the average melodies to see if the instruments give rise to some systematic distortions of the analysis by the agent ears. Table 6.4 shows the average melodies. The instruments are sorted according to the mean analysis error found in section 6.3.2. Again we can divide the instruments into two groups: Those that have a biased average melody and those that have not. It looks like that a larger mean analysis error results in a more biased average melody. Note that these instruments are sorted according to their mean analysis errors, where the instrument that gives rise to the lowest error is put on top and the instrument that gives rise to the highest error is put on the bottom of the table.

We can think of several ways of measuring this bias in an average melody. For example we can count the number of biased notes or we look at the amount of bias in each note. When we look at the average melodies that result of the BeeThree, the TubeBell, the Wurley and the PercFlut instruments, we see different numbers of biased notes, and different amounts of bias. When we compare the BeeThree with the more noisy PercFlut the case is clear. The average melody of the PercFlut has more biased notes, and the bias is generally larger. The TubeBell is a hard case. The bias is smaller than the bias in the average melody of the BeeThree, but the number of biased nodes is larger. Note as well, that a bias in a certain direction in a note is most of the times compensated with a bias in the opposite direction of the following note. Only the average melody of the PercFlut does not show this pattern.

The average melody of the One Note Samba consists of zeros so the bias is not coming from the song. It is clear that the bias is caused by the ears of the agents, although it is not clear why notes at some positions in the melody

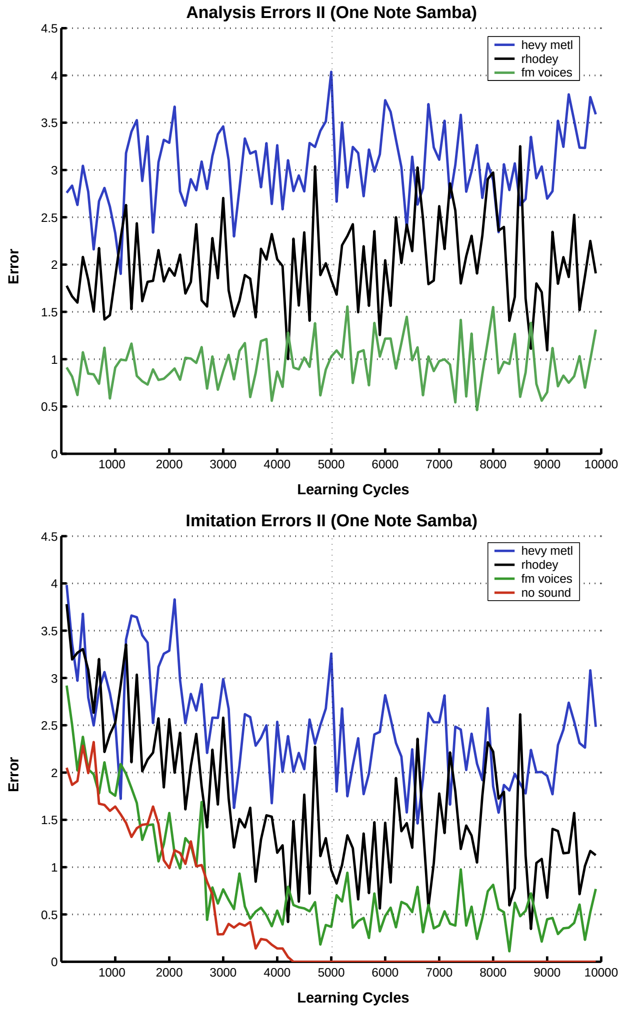

Figure 6.11: Imitation and analysis errors for the best performing agents. The agents have the instruments: hevyMetl, rhodey and FMVoices. There is one agent that doesn't play an instrument. This last agent therefore has no analysis errors.

are biased and others are not. Why are there biased notes in the first place?

However it is not the aim of this test to investigate the pitch detection algorithm itself. The aim of this test is to investigate the *effect* of it.

Perfect ears	0	0	0	0	0	0	0	0	0	0
FMVoices	0	0	0	0	0	0	0	0	0	0
Rhodey	0	0	0	0	0	0	0	0	0	0
HevyMetl	0	0	0	0	0	0	0	0	0	0
BeeThree	0	0	4	-4	0	0	0	0	0	0
TubeBell	-1	1	-2	2	0	0	1	-2	0	0
Wurley	-1	1	0	0	0	0	-5	4	0	0
PercFlut	-10	0	12	-7	6	0	0	-6	-2	0

Table 6.4: The mean analyzed input-melody for all different agents.

Brains

The color code on the displayed brains of the agents in figure 6.12, ranges from -19 to 19. We can see that the PercFlut agent has the largest number of categories. This agent has a range from 19 to -17. The noSound agent, which is the agent with the perfect ears is on the other end of the brain spectrum. It only learned the categories that were present in the One Note Samba. This agent has as highest and lowest categories 10 and -5. The other agents are between these extremes. We see here that the agents with realistic ears indeed learn pitches that are not there.

6.4 Results on communication by playing imitation games

We are going to compare the three societies by looking at the imitation errors, the brains, the differences in the category distribution between the agents and at the effect of communication on classification.

6.4.1 Imitation Errors

The mean of the imitation errors of both agents is taken and a moving average of length 200 of this error is plotted in figure 6.13. The dotted vertical line is the moment that the agents of the Quick Change society start to communicate. Before this line all the societies show the same pattern. In fact in this period the No Change and the Quick Change societies are similar.

The agents only study their music. The Slow Change society is almost equal to the others. Here the agents have a very low probability to communicate. After the dotted line, the Quick Change society starts to deviate from the familiar imitation error curve. The error becomes suddenly much smaller. In this phase the other two societies still show the familiar pattern. Finally all the societies reach the point where the error becomes zero, but then the imitation error of the Slow Change society starts to grow again. At this point the agents of this society start to communicate so much that this action starts to dominate the error.

When the agents have learnt their song well, they have a lot of knowledge of their song and the imitation error of their music study actions should be close to zero. However, the imitation error of the communication actions does not have to be zero at all. We can expect that the brains of the agents are completely adapted to their own song. We can say that they expect melodies like those of their own song and therefore the growth of the imitation error in the end of the agent lifetime can be explained by the growing percentage of communication actions, since the other agent plays different music than expected.

6.4.2 Brains

Figure 6.14 shows the brains of the agents of the Slow Change and the Quick Change societies, after learning. We see that the agents of the Quick Change society have a smaller range of categories on their output map than the agents of the Slow Change society. Remember that learning starts with the middle categories that are present in the input. Gradually during the learning process the range of categories grows on both sides and the more extreme categories emerge on the output map. Therefore we can say that the agents of the Quick Change society did not grow up completely! Only the middle categories that were present in the songs are on the output maps. The agents did not learn the large intervals. On the contrary, the Slow Change agents have learnt almost all the intervals of the songs they studied.

Because of the abrupt phase change after 6300 cycles of the Quick Change society, the agents there spend the rest of the time communicating. At that moment they did not learn the large intervals yet and therefore they did not use them in their compositions when they communicated. After the phase change the agents had no way of learning those extreme categories.

On the other hand the Slow Change agents started communicating while they still studied their songs. This dual learning, that is learning from each other *and* from their songs continues nearly until the end. Therefore they were able to learn all the categories.

6.4.3 Difference in Categories

Here we compare the categories on the output maps of the agents. There are 100 output nodes on a map. We sort them according to the category values they have by putting every output node in a "bin" according to its category. This gives us the category distribution, which is a list where every element represents a category and the value represents the number of occurrences of that category on the output map. The average category distribution of a society is calculated based on the category distributions of both agents. Now the euclidian distance between the category distribution of an agent and the average category distribution is calculated. When a society has two agents, this distance is the same for both agents. The distance to the average category distribution is a measure for the similarity of the categories on the output maps of both agents. This distance is what is plotted in the graph of figure 6.15. We expect that communication makes the category distributions of the agents more similar, especially in the quick change society. Therefore we should see a decrease in the distance to the average category distribution. This means that the lines in both communication societies should move closer to zero.

We see that the category distributions of both agents become more similar in *all* societies, also in the society where the agents do not communicate with each other. This can be explained. In the beginning when the neighborhood of the SARDNet is large the whole output map of an agent adapts to the input [6]. This results after a couple of learning cycles in a map filled with one or two categories. These categories have the value that is most common for the input. During learning, slowly more categories emerge on the output map. When the most common category values of the input are different for both songs of the agents, this results in output maps filled with different categories. For example the One Note Samba has the 0 as most occurring category, and let us say that Bach has category 3 as most occurring category. In that case the Bach studying agent will have an output map full of threes and the One Note Samba Agent will have an output map full off zeros. Then

[6] See section 2.3.3 for a description of the network

slowly they will learn more categories that are the same for both agents and hence their category distributions become more similar.

The agents of the Quick Change society end up with the most similar category distributions. When they start communicating they adapt themselves very rapidly to each other. We see that the other two societies are more or less similar to each other until they reach the stable phase, but that in the end the agents of the Slow Change society are still learning from each other. Their category distributions become still closer to the average category distribution. This does not happen in the other two societies. It is in line with what we see in the imitation error graph of figure 6.12. There the activities of the other societies stopped while the Slow Change society went on even while the learning rate and the neighborhood where very small.

6.4.4 Classification

Now we look at the classification of the agents. We do not care whether an agent was studying Bach or the One Note Samba. We are only interested in the difference between the societies and the effect communication has on classification. Therefore we take the average classification of both agents. This means we have one classification measure for every society. In this way we can compare the effects of the different communication trajectories. The results are displayed in figure 6.16.

Based on the results so far, we expect that the Quick Change society is less well classified than other two societies. The agents have more similar output maps and they did not learn all the categories of the songs they studied. Therefore their compositions might have less characteristics of the songs they studied. This prediction is true. The Quick Change agents are composing melodies that could be more inspired by both songs.

Again we see that classification of the Slow Change society looks similar to that of the No Change society, but in the end the former shows a decrease of the classification value, probably due to the communication actions.

The dashed line is a society run in another test. This society is exactly the same as the No Change society, apart from the input melody length. This is set to 25. I have added this result to show that apart from the different communication phases, the input melody length is very important for the classification of the agents. A longer input melody forces the agents to learn more of the structure of the song. Compare the two extreme cases where the agents are trained with input melodies of length one, and with input melodies of a length that is equal to the song they study. In the former case, the agents do not learn any structure of the song, in the latter they will learn the complete song and hence its structure.

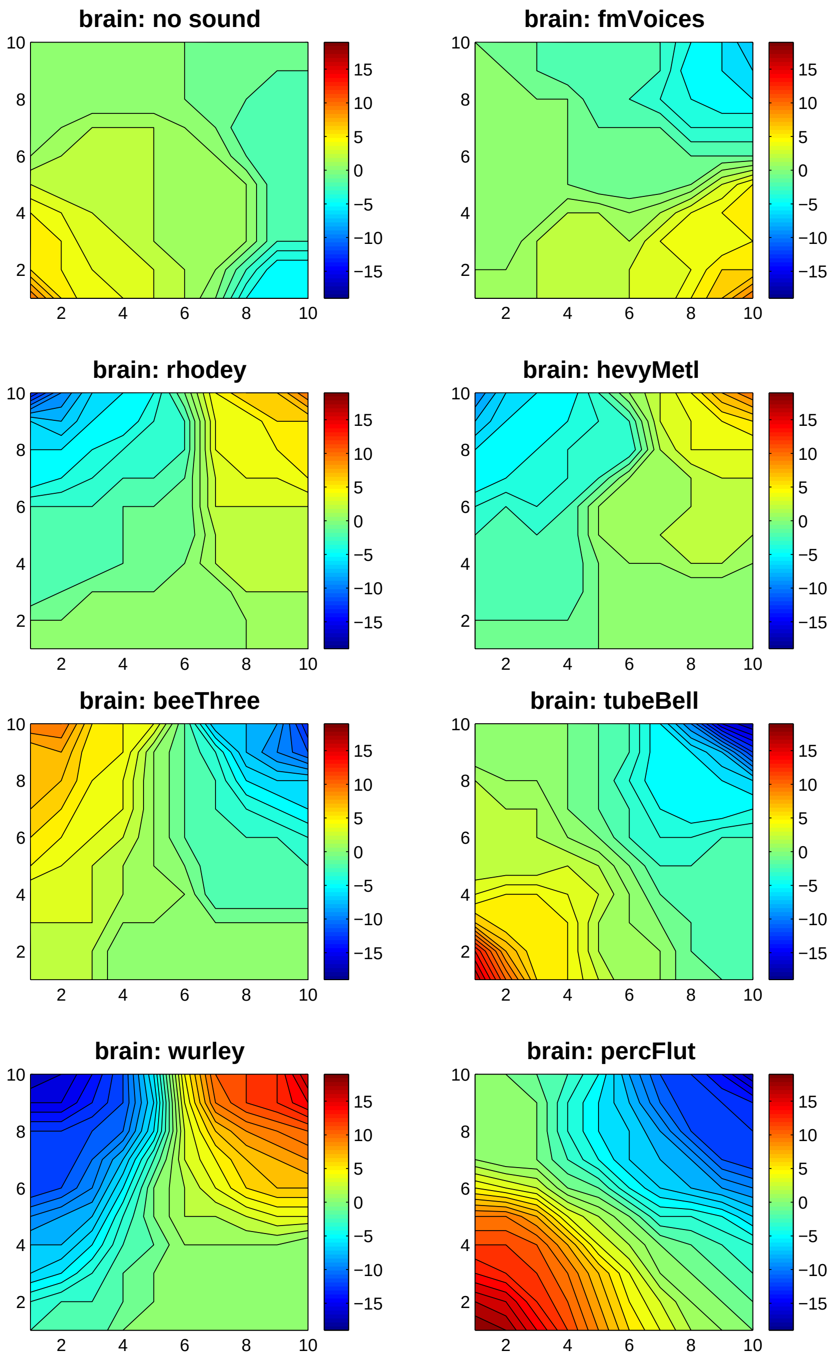

Figure 6.12: The brains for all different agents after learning.

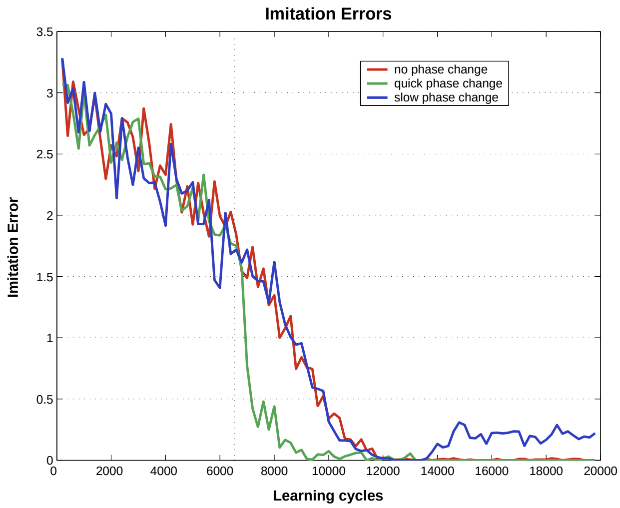

Figure 6.13: Imitation errors of three societies: *no change, quick change* and *slow change*. The vertical dotted line is the moment that the quick change society starts to change to the communication phase.

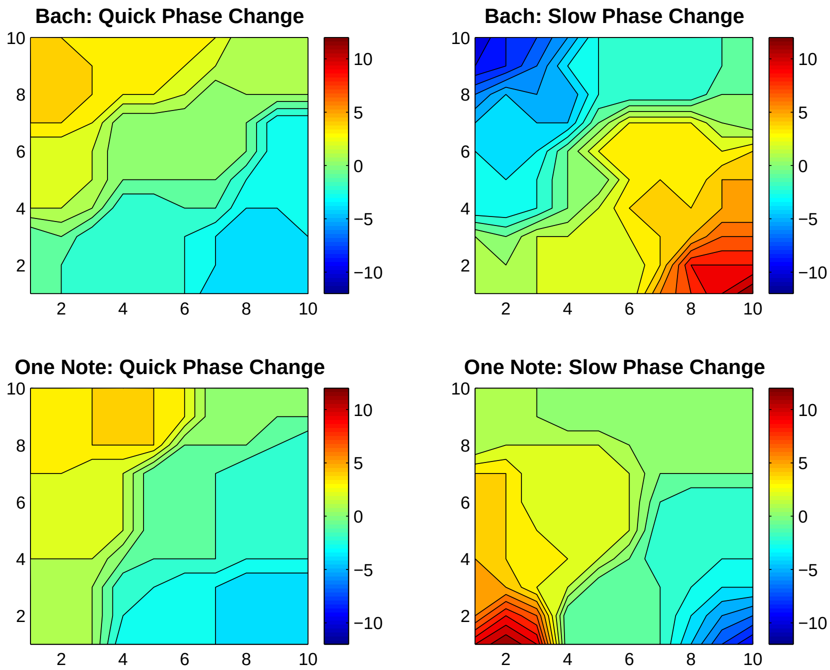

Figure 6.14: The brains of both agents (One Note Samba and Bach) of the *slow change* and the *quick change* societies.

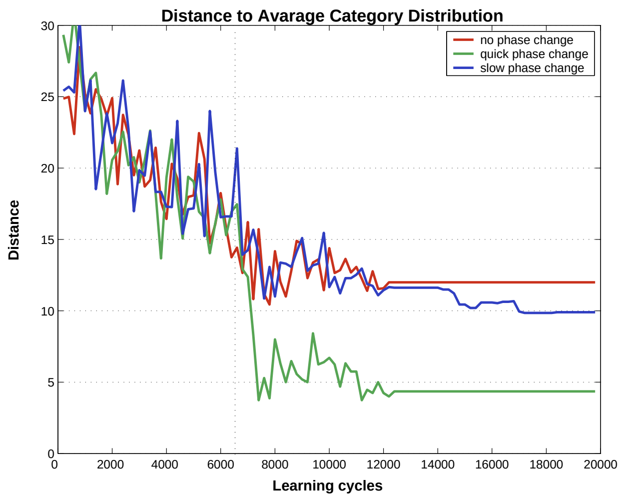

Figure 6.15: These are the distances to the average category distribution. The category distribution is calculated, for both agents. Then the average is calculated. The euclidian distance to this average distribution is the same for both agents. This distance is plotted in this graph.

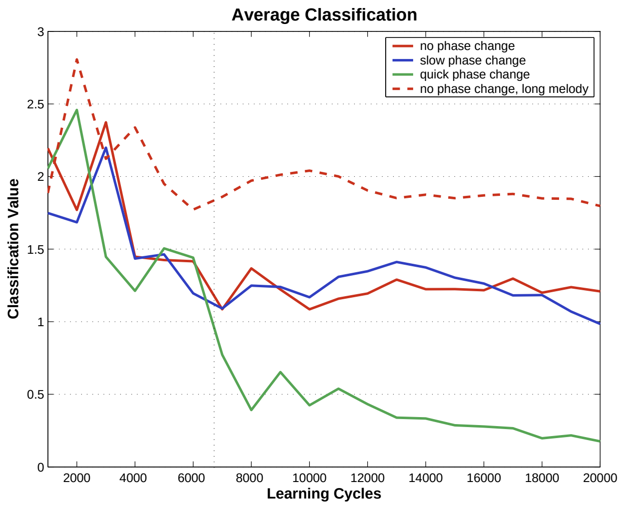

Figure 6.16: The average classification of the agents of all three societies. The dotted red line is the average classification of a society from another test. The length of the compositions of the agents of this society is 25 instead of 10 for the other societies. The other test is the grid size test, this society has a grid of 10 times 10 like all the societies here)

Chapter 7

Discussion

In this chapter the results presented in chapter 6 are compared with the hypotheses as stated in chapter 4. In section 7.1 all the hypotheses will be displayed again together with short discussions on the results. Section 7.2 ties the *framework for the evaluation of machine compositions* of Pearce and Wiggins [17], as discussed in section 2.4, to the evaluation of the agent compositions. Finally section 7.3 compares this society of communicating agents with the mimetic agents proposed by Miranda [15].

7.1 Hypotheses

Section 7.1.1 answers the question whether human made songs are easier to learn than randomly generated songs. Section 7.1.2 discusses the results concerning the agent compositions. Section 7.1.3 discusses the effect of learning with *realistic ears* and section 7.1.4 discusses the effect of communication on the knowledge of the agents.

7.1.1 Discussion on learning agent brains

> *During learning, the imitation errors of an agent will be lower when the agent learns a human-made melody, than when it learns a random melody of the same tone range and length.*

Figure 7.1: Hypothesis 1

According to the results in section 6.1 hypothesis 1 is correct. No matter what the size of the agent brain is, the error on the human-made song is always smaller. Successful melodies in our human society are successful in the agent world as well [1]. The results indicate that this is caused by the fact that a human-made song uses only a subset of all the possible intervals of the melody range. The brain scans of the agents show more categories on the agent brains that studied the randomly generated songs compared to their human-made counterparts. The melody space of a human-made song is not so dense. From all the allowed intervals of the melody range only a subset is used, and only a small subset is used frequently.

7.1.2 Discussion on knowledge in the agent compositions

During learning, the compositions of an agent become more and more similar to the song it is studying.

Figure 7.2: Hypothesis 2

To investigate hypothesis 2 I have developed a tool that measures the distance of the compositions of an agent to the song it is studying. A smaller distance indicates more similarity. Therefore we should see a decreasing distance to the own song of the agent during learning for hypothesis 2 to be true. According to the results in section 6.2, the agent compositions do not become more and more similar to the song it is studying. The distance is growing, so during learning the compositions become more and more different instead. We can argue that when an agent has seen more substrings of its song it can play around with them. It has a choice of what composition to generate based on more and more probabilities on its grid of transition tables. We can see this as a development of a kind of creativity. The agent creates new material based on knowledge of existing material and it communicates its *interpretation* of the song it is studying by means of its compositions.

For hypothesis 3 we test whether the idea of a grid of transition tables really works. From figure 6.7 of the Results chapter we can see that there are

[1] Note that success in the agent world is the learnability of the songs. We have no idea whether the agents "like" the songs.

> *If agent A has finer grid of transition tables than agent B, then the compositions of agent A will be more similar to the song agent A is studying than that the compositions of agent B are to the song agent B is studying.*

Figure 7.3: Hypothesis 3

no significant differences between agents with different grid sizes. A larger grid does not result in agent compositions that are closer to the song it studies. There is an explanation for this.

During the early stages of learning, the weight changes are large, due to a large learning rate. This together with a large neighborhood is necessary for the overall topology of the output map to develop. The standard ordering pattern that emerges is that of an adaptation of the nodes in one corner to the highest categories of the input, and an adaptation of the nodes in the opposite corner to the lowest categories of the input. What corner is chosen is completely random. We can see this for example in the brain scans of figure 6.12 and 6.14. The agents have chosen different corners for their highest and lowest category adaptations.

This choice of corner changes during the early stages of learning. For example the highest categories can be stored in the top left corner for the first 200 training cycles and then the ordering changes and the highest categories are stored in the bottom right corner.

Now we turn to the grid of transition tables. Every transition table refers to an area on the output map, and this connection is *hard wired*. It means that during the whole lifetime of an agent the transition table refers to the same output nodes.

In figure 7.4 we see two snapshots of an output map of sixteen output nodes during two moments in the life time of an agent. The picture on top is the first snap shot, and the picture at the bottom is the second snap shot, taken a couple of hundreds of learning cycles later. The numbers are the rounded weight vectors of the output nodes, which stand for categories. The red line is the most likely route on the output map at the moment the snapshot is taken, according to the probability tables. We see that the global ordering on the output map has changed in the time between these snapshots. We know that this is a very common situation. The transition tables cannot follow this change of ordering, since they are *hard wired*. However the

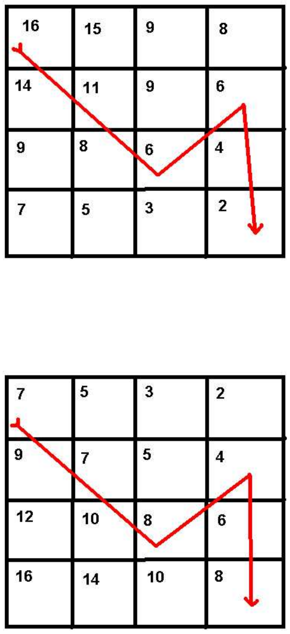

Figure 7.4: The same route on the output map during different learning cycles. The most likely route on the output map is the same but the categories that are stored on this route have changed. Route 1: [16, 6, 6, 2] Route 2: [7, 8, 4, 8]

probabilities, stored in the transition tables *can* adapt to this new situation, but this is a slow process, and due to past learning-inference this adaptation is not perfect.

The result is that the meaning of the most likely route on the output map has changed as well. In the top figure it refers to melody [16, 6, 6, 2] and in the figure on the bottom the same route refers to melody [7, 8, 4, 8]. This latter melody does not have to correspond to any input that is characteristic for the network. It is the result of the probabilities of the transition tables that were not able to adapt to the new ordering on the output map.

This example illustrates that the connection between a melody phrase that appears in the input many times and the probabilities that are generated on the output map is far from clear.

After learning, the compositions of an agent are more similar to the song it studied than to another song of the set of coded songs.

Figure 7.5: Hypothesis 4

The results on hypothesis 4 do not show a clear picture. We see that this depends completely on the song the agent studies and to the songs to which the compositions of the agent are compared. According to the results of the experiment the compositions of the agents that studied the Chinese songs and the One Note Samba were classified correctly. This means that the compositions of these agents were more similar to the song they studied than to the song they were compared with. Only the compositions of the Bach studying agent were classified as the One Note Samba. The group of songs and the number of songs the compositions are compared with is very small, and the results are not pointing in one direction. We do not have enough evidence to conclude that this hypothesis is true, nor to conclude that it is false.

7.1.3 Discussion on hearing capabilities of the agents

During learning, depending on the timbre of the instrument, the imitation error will be increased with a constant amount of noise which is the analysis error.

Figure 7.6: Hypothesis 5

While investigating hypothesis 5 we discovered a strange phenomenon. The analysis errors of the two noisiest instruments were not constant. They rose during learning. Why do they change during learning?

There are two melodies that are analyzed every training cycle. We have the substring, drawn from the song, and the imitation melody of the agent. Both melodies are played on the same instrument, and the errors are saved. Every 100 training cycles the mean is calculated from all the errors saved during this interval.

The randomly drawn substrings do not change systematically during the learning process, nor do their analysis errors. The imitation melodies however, *do* change, and so do their analysis errors. In the beginning the weights of the network are set to zero, so the imitation will be zero, whatever the input melody was. During the learning process categories will form. This happens almost always in the same way. First the middle categories emerge on the output map, and then more and more higher and lower categories will emerge one by one. This means that the imitations will be able to contain higher and lower categories during the learning process, while in the beginning they only could contain the middle categories. We have seen that the analysis errors are different for different frequencies. This is even stronger in the instruments that give rise to larger errors.

The answer to the rise in the analysis errors is that the range of categories that is present in the imitation melodies becomes wider during learning. In the hearing test of section 6.3.1 we have seen that the remote categories of a range result in different and most of the time larger errors in the analysis.

> *Due to the analysis error, the agent develops a category distribution on its output map that does not resemble the category distribution of the melody.*

Figure 7.7: Hypothesis 6

Hypothesis 6 is true. The agent with *perfect* ears only learned the categories that were present in the song. All the agents with *realistic* ears learned a wider range of categories. This means that they learned categories that were not present in the song as well. Instruments with more noise gave rise to more "ghost" categories.

7.1.4 Discussion on communication by playing imitation games

> *The agents will learn a mix of their own song and the song of the other agent.*

Figure 7.8: Hypothesis 7

Whether hypothesis 7 is true or not depends very much on the way the communication phase is introduced. If the phase change from song study to communication is very quick, that is, the agents change their action preferences in just a couple of learning cycles, then the knowledge of the agents is completely mixed. We can see as well that at the moment of the phase change, this knowledge is not much yet. The agents have not developed all the categories that were present in their songs. When we allow a slow phase change, the knowledge of an agents does not mix much.

Now we turn back to the idea of the metaphor of the multi-cultural society. Can we see a similarity? The sudden phase change resulted in a mix of songs partly because the knowledge of the songs was not completely learned yet. This would be the same as when we put a group of children from different cultures together for the rest of their lives. This would certainly lead to a kind of creole culture [9].

In the slow phase change condition the agents had the possibility to stay in contact with their own culture. They were able to choose the song study action until the end. The result was that the cultures of the agents did not really mix. The metaphor we see here is that of some big American cities. In those cities people from all kinds of cultures are living in separate neighborhoods. They interact with each other, but they do not mix their cultures.

7.2 A framework for the evaluation of agent compositions

The framework of Pearce and Wiggins [17] discussed in the theoretical background chapter in section 2.4 states that we should adhere to four components, if we wish to evaluate the compositions of a computer model scientifically. I mentioned that this framework is meant for machine compositions that intend to play in a specific style. This is not the main aim of my model, however my model consists of many small machine composers. The agents are composing the music and they do try to compose in a certain style, especially when they study their culture which is represented by a human-made song. Therefore I will discuss the four components on the agent level.

7.2.1 Specifying the compositional aims

When an agent is learning its culture it is trying to compose in the style of the song it is studying. The human-made songs in this model are simplified. Everything is removed from the score. Only a list of relative pitches remains. From this list an agent receives substrings of a certain length. Therefore the aim of an agent is to produce a list of relative pitches that is in the style of the song it studies. Note that the song has the role of a music genre and a substring of the song has the role of a song.

When an agent is imitating another agent, it tries to compose in the style of the singer agent. In this case the style is defined by the knowledge of the singer agent which is stored in the weights of its network and on its grid of transition tables.

We can look at the compositions of the singer agent as well. We can ask ourselves, how do we evaluate them? However, the singer agent is free to compose whatever it likes and it is not trying to compose music in a specific style.

7.2.2 Inducing a critic from a set of example musical phrases

Ideally this critic should be induced by means of some kind of machine learning technique. The SARDNet is a very good candidate and we know that the learning agent has a SARDNet itself. Therefore an agent of our model could be such a critic. In the imitation games this is what happens in a modified way.

We have a critic which is the singer agent. The knowledge of this agent, which is stored in its SARDNet, defines the musical style the imitator agent tries to compose in. In a society of agents the critics are learners as well and therefore there is no fixed direction the compositions are going.

This is different in the music study phase. There the compositions the agent has to imitate are coming directly from the song.

To evaluate the compositions of a singer agent I have designed a comparison measure that compares its compositions to the song it studies. We can see this measurement as a kind of rule based critic. As I mentioned in the *compositional aims* section: On this level the society of agents is not designed to compose music in a specific style. Therefore the composing singer agents have no means of adapting their compositions to satisfy this rule based critic.

We like to leave the composers to compose what they "feel" because that is part of what we study here. Therefore we regard this comparison measure as a descriptive research tool and not as a critic.

7.2.3 Composing music that satisfies the critic

This has already been discussed in the previous component. The imitators are adapting their weights and probability tables to melodies that are sung by the critics: the singer agents. These singer melodies represent the demanded style and imitations with small errors will definitely satisfy them. To compare two melodies, the euclidian distance between them is calculated.

7.2.4 Evaluating claims about the compositions in experiments using human subjects

This component is at the moment outside the scope of this model. First of all the representation of melodies in this agent world is so different from music we are used to hear that a human being has problems evaluating it. For example the one note samba contains mainly one note. The characteristics of this melody are entirely in the rhythm, which is removed in the agent representation.

Secondly it is not an aim of the agents to compose in the style of the song they study. Even if this would have been so, human beings had problems determining if the agents succeeded since they have problems recognizing the songs in agent representation.

7.3 Comparison with Mimetic Development of Intonation

What are the differences between the model in this project and the *Mimetic model* of Miranda [15]? There are many differences on every level and I will list the most important ones below:

agent brains The agents of the mimetic model poses brains with two symbolical memories and associations between these two. The agents of the model of this project possess SARDNet brains. There is one memory, which can be regarded as a sub-symbolical perceptual memory.

melody representation The mimetic agents store lists of complete melodies, and they do this in two representations. There is the CARMEC representation, which is a notation for melody contour, and functions here as the perceptual representation. The motor representation is a list of synthesizer commands. In the model of this project, there is one type of representation, which is a number representing an interval. The melodies are distributed over the agent brains and there is no place where a complete melody can be found. The knowledge of intervals is stored in the weight vectors and the knowledge of sequences of notes can be found in the transition tables.

agent ears The mimetic agents do not make errors when analyzing melodies. They have perfect ears. The agents in the model in this project can have ears that are biased and result in analysis errors.

agent voices All mimetic agents have the same voice, and they learn to use it by giving motor commands. In my model every agent can have a different voice, but the motor commands are modeled in less detail. In fact the synthesizer is controlled by outputting MIDI note-numbers

initial knowledge The mimetic agents start from scratch. They have no initial knowledge. My agents are able to learn the knowledge from their culture before they start to communicate.

feedback Feedback in the mimetic agent world is given by playing back the singer-melody for the second time or by remaining silent. The feedback is boolean, it is a *yes* or a *no*. In my model feedback has more values. It can have every floating point number between (0, 1]. Where 1 means similarity and 0 means very different (orthogonality).

The main difference between the results of this model and the mimetic model is that in this model there is no fixed number of melodies (intonations) that evolves in the society. The neural network brains are noisy systems. There is however a certain melody *type* that emerges in the system as can be seen in the classification of the songs in section 6.2. The results in section 6.4 indicate that communication has an effect on the knowledge of melodies of the agents. The agents learn from each other and we can see this in their compositions.

Chapter 8

Conclusion

Music is shaped by cultural and cognitive dynamics. It has to be relearned over and over again by new generations of people. Cultural dynamics are studied in several a-life models, where agents play the role of learning members of a society [13] [16] [23] [20].

The main aim of the model proposed in this project was to investigate the effects of neural network brains on a society of agents. This society was inspired by a society of *mimetic agents* that evolves a repertoire of intonations from scratch [16]. The SARDNet was a good choice for an agent brain because it is able to process time sequences in an efficient way. The fact that music is ordered in time is a very important characteristic which is preserved by using this neural network. A second line in this project was the introduction of human-made compositions, which represent culture. By letting the agents learn these compositions as well a connection with our musical culture was simulated.

Like their *mimetic* fellows, the agents in this model played imitation games. In this way they learned melodies sung in their society. Melodies were represented as pitch intervals, and the interval distribution of the melodies were learned and stored on the output map of the SARDNet of an agent. The grid of probability tables was a part of the agent brain too, and can be seen as a grammar learning device. It learned the transitions between elements of a melody. The agent was able to hear melodies with a very simple ear, which could sometimes infer wrong intervals from the melodies that were played to it. When an agent composed a melody it played it on its synthesizer. The control of this synthesizer was kept simple and consisted of MIDI note-numbers. Different agents could play synthesizers with a different timbre.

To investigate the effects of the SARDNet brains on the communication games we had to investigate the learning abilities of this network first. We took the chance to compare the human-made songs with randomly generated songs. It turned out that these human-made songs, which were successful in our world, were successful in the agent world as well. The success was measured as learnability.

We investigated the compositions of the agents as well. Every agent was assigned to a human-made song. A simple tool was developed to compare the agent-compositions with the human-made songs. Three of the four agents made compositions that were more similar to the songs they studied, than to the other human-made songs present in the set.

In the first two experiments described above the ears were flawless. The agents were able to classify every heard interval correctly. The third experiment the ears are made more *realistic*, which meant that it infers sometimes the wrong interval. The effect of different timbres of the synthesizers on learning was investigated. It turned out that the imitation errors were much larger when the timbre of the instrument was more percussive.

Finally the agents were allowed to play the imitation games. In the beginning of the simulation the agents had a culture learning phase in which their networks were trained on a human-made song that was assigned to them. By putting two agents with different songs together to play imitation games, a culture clash was simulated. The *culture clash* is a metaphor for the effect of imitating an agent that studied a different song. When the agents switched gradually from studying their own song, to playing imitation games, they were able to keep the knowledge of their own song. When the phase change from studying to communicating was abrupt, the agents learned a mix between their own song and the song of the other agent.

Compared to the *mimetic* agents there was no fixed number of agent-compositions that was shared by the whole society.

Chapter 9

Future Work

The model as it is now can be seen as a first step in the investigation of the integration of neural networks in a society of musical agents playing imitation games. It provides a glimpse of the effects of adding human musical culture to this society as well. However, there are a lot of elements of this model that need to be improved in order to make its claims more powerful. Furthermore hypothesis 4 is not completely answered yet. In this chapter I will discuss hypothesis 4 first in section 9.1. Then I will discuss improvements on the representation of music in section 9.2. Improvements on the agent-brains are discussed in section 9.3 and finally improvements on the comparison of melodies are discussed in section 9.4.

9.1 Hypothesis 4

> *After learning, the compositions of an agent are more similar to the song it studied than to another song of the set of coded songs.*

Figure 9.1: Hypothesis 4

To answer this hypothesis we need to compare the compositions of an agent with many more songs. In the experiments done in this project the compositions of an agent were compared to its own song and to one other song. We have seen for example that the Bach studying agent produced compositions that were more similar to the *One note samba* than to *Invention*

No. 1 of Bach. What do we know after this comparison? Is it the composition of Bach that is so hard to learn or is it the One note samba that is so easy to learn? We can answer this question by coding many more compositions. Imagine that we create a library of 1000 melodies. We let our agent study *invention no. 1 of Bach*, and compare its compositions to all the melodies of this library with the tool developed in this project. Once we have done this we can sort the melodies of the library on their euclidian distances to the agent compositions. If the agent was able to learn its song really well, then *Invention No.1* of Bach should be somewhere in the front of this sorted list.

We probably will be able to group the melodies of this library according to their distances to the agent compositions. This melody grouping will tell us something about the learnability of the song of the agent, but it will tell us something about the similarity between the melodies of the library as well. I guess for example that many other works of Bach will be grouped together.

Remember that in an experiment of this project an agent generated 1000 compositions every 1000 training cycles. We have seen that the similarity of its compositions to the songs, changed over time. When we compare these compositions to our library and we order this library every time according to the agent-composition versus song distances, we probably will be able to see interesting changes in the ordering of the songs, during the lifetime of the agent.

9.2 Music representation

In this model the representation of music is very simple. Music in the human world has many features as rhythm, loudness, polyphony and differences in tempo. All these features are removed in the current model. Even rests are removed, and therefore we can hardly speak of music, but this representation can easily be extended to make it more similar to human-music. A SARD-Net is a very flexible network, and we could start with adding the features *loudness* and *duration* to the notes that are presented to the network. In this way we get input vectors of length three: [*loudness, duration, relative pitch*]. Care should be taken to scale these three elements in such a way that the weight of every element represents the importance in the perception of real music.

9.3 The brain

Another approach is to use another network as agent-brain. Dominey proposes a more biologically plausible but much more complicated neural network that models parts of the brain in more detail [6]. We could as well turn back to the hebbian network proposed by Westermann and Miranda [26]. This model performs exactly the perceptual to motor mapping that the mimetic agents [16] have to learn. The problem with this network is that it is not suitable for time sequences. Remember that the "SARNet trick" makes a SOM suitable for learning sequences.

The challenge is to apply a (modified) SARDNet trick to the hebbian network. During this project I have started a first attempt to accomplish this. However it was too much work to finish this in this master project. The challenge is still open, and I think this is the best option to improve the brains of the agents for four reasons: Firstly the brain is more biologically plausible than the SARDNet. Secondly the network is still simple and easy to understand. Thirdly, it provides the exact perceptual to motor mappings the mimetic agents use. (In the SARDNet the motor mapping is not explicitly modeled.) Finally, the number of dimensions on the perceptual side as well as on the motor side of the network can be set freely. Therefore we can in principle model any musical feature and muscle contraction we wish.

9.4 Melody comparison

At first I will discuss how melodies can be compared in section 9.4.1. Then I will discuss the melody classification in section 9.4.2.

9.4.1 Melodic similarity

Throughout the model the euclidian distance is used for melody comparison. This is a quick and dirty way to compare melodies and should again be regarded as a first attempt. It does not always result in a desirable outcome as is illustrated in the next example.

Let us say we compare the following two melodies of length 4: melody $a = [\,0, 1, 0, 0\,]$ and melody $b = [\,0, 12, 0, 0\,]$, with the basic melody: $[\,0, 0, 0, 0\,]$. The basic melody consists of four prime intervals. According to the euclidian distance melody a is more similar to the basic melody. However harmonically this melody is a very different. By going up one minor second, different notes are played that can be outside the scale of the melody. Melody b goes up one octave. The distance of an octave results in a note that is harmonically very close to the note where it is derived from, while the distance of a minor second results many times in a note that is harmonically dissonant.

A whole line of research is dedicated to similarity measures of melodies and the euclidian distance measure is certainly not the most sophisticated one. Measures as for example the *Earth Mover's Distance (EMD)* (discussed by Typke, Veltkamp and Wiering [25]) take note duration and note position into account while comparing two melodies. The ordering of the notes on the pitch axis is another point. The euclidian distance measure as I used it here and the EMD as used by Typke, Veltkamp and Wiering orders the notes according to their (relative) pitch and not to their harmonical function in the melody. A harmonical ordering could improve the comparison as is illustrated in the example of this section.

9.4.2 Melody classification

In this model I have proposed a simple tool to classify the agents according to the compositions they produced. I tried to characterize the style the agents composed in.

Westhead and Smaill [27] described a system that automatically characterizes musical styles by using motifs. Here motifs are patterns of rhythms or pitches that are common to more pieces in the music style. Their model uses a machine learning technique that builds up a set of motifs belonging to the style by learning from examples. It builds up a set of motifs that does not belong to the style as well. After learning this model distinguishes between different styles with a success rate of over 95%.

This model could be a good improvement for analyzing the agent compositions. It can classify the agents and hence show the effect of communication by playing imitation games.

Chapter 10

Thanks

At this place I would like to thank a couple of people. Without them I could not do this project and all of them really helped me.

First of all I like to thank my supervisor in Plymouth Eduardo Miranda for all the help he gave me. You were an inspiring supervisor. The answers to questions in my emails were most of the times exactly what I needed to know, and in our discussions on the project I admired your speed of thinking, and your ability to focuss immediately on the most important topics.

Then I like to thank my supervisor in Groningen Rineke Verbrugge. Onze scriptie besprekingen waren echt een steun voor mij, en de sfeer was altijd relaxed. Dat hielp me door deze vier weken van schrijven heen. Ik wil je ook bedanken voor het telkens weer zo grondig nakijken en becommentariëren van delen van mijn scriptie. Verder ben ik blij dat je me steunde om terug te gaan naar Plymouth. Hierdoor kan ik met veel meer voldoening terugkijken op mijn afstudeerproject.

I like to thank the people of the Computer Music Research lab for being there. You were nice company and I am happy you dragged me away from my computer every now and then for lunch or coffee. Good luck with your projects.

Finally I like to thank Prof. Lambert Schomaker for giving me the idea of the grid of transition tables. Je hebt me hier erg mee geholpen, al roept het gebruik van deze transitietabellen uiteindelijk veel meer vragen op. Je zou een heel afstudeerproject kunnen wijden aan het ontwikkelen van een SARDNet dat echt profiteert van deze vondst.

Appendix A

Parameters

The society of agents can be designed by setting the values in the parameter file *INIT.ini* and running the model. The program will read this file and build the society accordingly. The next section will display the list of parameters. I will give a short description of their functions. The rest of the appendix displays the parameter settings of the experiments.

A.1 Parameter file: INIT.ini

Table D.1 displays the list of parameters that are used for the experiments.

ears has two possible values: *perfect* and *realistic*. When it is switched to perfect, it means technically that the sound is switched off in the model. If the agent ears are set to realistic, it means that the sound is switched on. If the sound is switched *on* the agents will make wave files with their instruments and produce and analyze real sound samples. When the sound is switched *off*, the agents communicate with the raw melodies directly. That means that they have immediate access to the lists of relative pitches. The perfect ears situation is the same as when the agents would analyze the wave files without error, however not analyzing wave files saves a lot of unnecessary calculations.

output map length is the length of the two dimensional output map of the SARDNet. Values have to be between 0 and 1000.

output map breadth is the breadth of the two dimensional output map of the SARDNet. Values have to be between 0 and 1000. Together, the

ears base frequency instruments
output map length output map breadth transition table grid length transition table grid breadth sequence length
number of agents number of train iterations feedback scaling
highest note sigma trajectory learning rate trajectory
music study trajectory communication trajectory songs

Table A.1: The parameters of the initialization file, INIT.ini

length and the breadth determine the shape of the output map. The total number of output neurons is: length times breadth.

transition table grid length is the length of the two dimensional grid of transition tables. The length has to be proportional to the *output map length* in such a way that: *output map length* modulo *transition table grid length* = 0.

transition table grid breadth is the breadth of the two dimensional grid of transition tables. The breadth has to be proportional to the *output map breadth* in such a way that: *output map length* modulo *transition table grid breadth* = 0.

sequence length is the length of the input melody. It is the number of notes the melodies will have. Values have to be between 0 and 1000.

base frequency is the lowest frequency of the frequency range. This frequency range is attached to a list of note numbers. For example, the base frequency is set to 400 then number 12 which is one octave higher represents frequency 800. From this list, the list of relative pitches is derived. See for details the section of *the ears* in the *Implementation* chapter. Works only if *sound* is switched on. Values have to be between 0 and 10000.

number of agents sets the number of agents in the society. Values have to be between 0 and 1000.

instruments Assigns the agents to their FM-synthesizers, the choice is: HevyMetl, BeeThree, TubeBell, FmVoices, PercFlut, Wurley, and Rhodey.

number of train iterations is the number of training cycles that occur in the society. It can be seen as the time in the society, or the life time of the agents. Values have to be between 0 and 1000000.

feedback scaling is the parameter that sets the importance of the feedback for the imitator agent. The result of the weight update function of the imitator agent is multiplied with the result of the feedback function. This feedback function always outputs a number between $(0, 1]$. It has a decreasing effect on the weight update. When *feedback scaling* is equal to zero, the output of the feedback function is always one. In that case the feedback does not influence the weight updates. The feedback function is:

$$\frac{1}{1+(d*s)}$$

Where $f =$ *feedback*, $d =$ *difference*.
Which is the euclidian distance between the melody and the imitation. and $s =$ the scaling factor that can be set here. Values of this scaling factor have to be between 0 and 1000.

highest note This value sets the melody range. For example when it is set to 24, this means that there is a melody range of two octaves. This is used by the random melody generator. The values have to be between 0 and 88.

sigma trajectory The SARDnet and the kohonen map have a neighborhood function that is used in the weight update rule. One of its parameters is σ. σ affects the size of the neighborhood. The size of the neighborhood usually decreases during training. This can be realized by increasing the σ. The *sigma trajectory* specifies how the σ value changes. The σvalue will follow this trajectory during the learning cycles. At cycle one it starts at the first value of the trajectory and at the last learning cycle it has reached the last value. The longer the trajectory, the more specified the the changes in the σ movements are. Values have to be between 0 and 100.

learning rate trajectory Most neural networks have a learning rate parameter (η). The size of η usually decreases during training. This slows down learning and makes fine tuning of the weights possible: To learn the details of the input data. The *learning rate trajectory* specifies how the η value changes. The η value will follow this trajectory during the learning cycles. At cycle one it starts at the first value of the trajectory, and at the last learning cycle it has reached the last value. The longer the trajectory, the more specified the the changes in the η movements are. Values have to be between 0 and 100.

music study trajectory The numbers in this trajectory represent the values of probability weights. The weights of the music study and the communication trajectory together produce probabilities that an action is chosen by the agent. When for example one weight has a large value and the other a small one, the probability that the action of the first weight is chosen is very high. The actions of the agents are communicate and study. The values of these weights will follow the trajectories during the learning cycles like those of the learning rate and the σ trajectories.

communication trajectory See the explanation of the *music study* parameter.

songs Assigns the agents to their songs. This list of songs has to have the same size as there are agents in the society. See appendix C for details.

A.2 Parameters settings of the experiments

For every experiment there are *general* and *specific* parameter settings. The general settings are the parameter values in the experiment that are kept constant. The specific settings are the parameters that are under study and hence varied for every agent or society of agents. Not all parameters matter in an experiment. When for example the ears of the agents are perfect, then the use of real sound has no effect on the agents, since they always can analyze the right pitches. Therefore in this case the choice of the instrument and the base frequency setting of the melody range, become irrelevant. If a parameter is irrelevant in an experiment, it will be labeled *not important*.

ears	perfect
base frequency	not important
instruments	not important
output map length	varied
output map breadth	varied
transition table grid length	not important
transition table grid breadth	not important
sequence length	varied
number of agents	24
number of train iterations	10000
feedback scaling	1
highest note	varied
sigma trajectory	varied
learning rate trajectory	1 - 0.1 - 0.001
music study trajectory	100 - 100
communication trajectory	0 - 0
songs	varied

Table A.2: General parameters: Experiment on the brain.

agent	output map: length	breadth	sequence length	highest note	sigma trajectory	song
1	4	4	4	17	0.99 - 0.1 - 0.001	random
2	5	5	4	17	1.32 - 0.1 - 0.001	random
3	7	7	4	17	1.98 - 0.1 - 0.001	random
4	16	16	4	17	4.95 - 0.1 - 0.001	random
5	4	4	4	17	0.99 - 0.1 - 0.001	Gold Snake
6	5	5	4	17	1.32 - 0.1 - 0.001	Gold Snake
7	7	7	4	17	1.98 - 0.1 - 0.001	Gold Snake
8	16	16	4	17	4.95 - 0.1 - 0.001	Gold Snake
9	4	4	10	24	0.99 - 0.1 - 0.001	random
10	5	5	10	24	1.32 - 0.1 - 0.001	random
11	7	7	10	24	1.98 - 0.1 - 0.001	random
12	16	16	10	24	4.95 - 0.1 - 0.001	random
13	4	4	10	24	0.99 - 0.1 - 0.001	Bach
14	5	5	10	24	1.32 - 0.1 - 0.001	Bach
15	7	7	10	24	1.98 - 0.1 - 0.001	Bach
16	16	16	10	24	4.95 - 0.1 - 0.001	Bach
17	4	4	16	70	0.99 - 0.1 - 0.001	random
18	5	5	16	70	1.32 - 0.1 - 0.001	random
19	7	7	16	70	1.98 - 0.1 - 0.001	random
20	16	16	16	70	4.95 - 0.1 - 0.001	random
21	4	4	16	70	0.99 - 0.1 - 0.001	Chopin
22	5	5	16	70	1.32 - 0.1 - 0.001	Chopin
23	7	7	16	70	1.98 - 0.1 - 0.001	Chopin
24	16	16	16	70	4.95 - 0.1 - 0.001	Chopin

Table A.3: Specific parameters: Experiment on the brain. *Gold Snake* refers to *Chinese Gold Snake Dance: Traditional* , *Bach* to *Invention 1: J.S. Bach*, and *Chopin* to *Etude Op.10 N.1 in C maj: Chopin F.* See appendix C for details

ears	perfect
base frequency	not important
instruments	not important
output map length	10
output map breadth	10
transition table grid length	varied
transition table grid breadth	varied
sequence length	10
number of agents	10
number of train iterations	10000
feedback scaling	1
highest note	not important
sigma trajectory	2.97 - 0.1 - 0.001
learning rate trajectory	1 - 0.1 - 0.001
music study trajectory	100 - 100
communication trajectory	0 - 0
songs	varied

Table A.4: General parameters: Experiment on agent compositions

agent	transition table grid: length	breadth	song
1	10	10	Bach
2	10	10	One Note Samba
3	5	5	Bach
4	5	5	One Note Samba
5	2	2	Bach
6	2	2	One Note Samba
7	1	1	Bach
8	1	1	One Note Samba
9	10	10	Gold Snake
10	10	10	Chinese Traditional

Table A.5: Specific parameters: Experiment on agent compositions. *One Note* refers to *One Note Samba: Antonio Carlos Jobim*, *Bach* to *Invention 1: J.S. Bach*, *Gold Snake* to *Chinese Gold Snake Dance: Traditional* and to *Chinese Traditional* is *Chinese Traditional Song: Traditional*. See appendix C for details

ears	realistic
base frequency	150Hz
instruments	varied
output map length	10
output map breadth	10
transition table grid length	not important
transition table grid breadth	not important
sequence length	10
number of agents	8
number of train iterations	10000
feedback scaling	1
highest note	not important
sigma trajectory	2.97 - 0.1 - 0.001
learning rate trajectory	1 - 0.1 - 0.001
music study trajectory	100 - 100
communication trajectory	0 - 0
songs	One Note Samba

Table A.6: General parameters: Experiment on realistic ears

agent	instrument
1	none
2	HevyMetl
3	BeeThree
4	TubeBell
5	FmVoices
6	PercFlut
7	Wurley
8	Rhodey

Table A.7: Specific parameters: Experiment on realistic ears. All instruments are FM-instruments from the STK-library. See [5] for details

ears	perfect
base frequency	not important
instruments	not important
output map length	10
output map breadth	10
transition table grid length	10
transition table grid breadth	10
sequence length	10
number of agents	6
number of train iterations	20000
feedback scaling	1
highest note	not important
sigma trajectory	2.97 - 1.5 - 0.1 - 0.001
learning rate trajectory	1 - 1.5 - 0.1 - 0.001
music study trajectory	varied
communication trajectory	varied
songs	varied

Table A.8: General parameters: Experiment on communication

society	trajectory: music study	communication	agent	song
no change	100 - 100 - 100 - 100 - 100	0 - 0 - 0 - 0 - 0	1 2	One Note Bach
slow change	100 - 75 - 50 - 25 - 0	0 - 25 - 50 - 75 - 100	3 4	One Note Bach
quick change	100 - 100 - 0 - 0 - 0	0 - 0 - 100 - 100 - 100	5 6	One Note Bach

Table A.9: Specific parameters: Experiment on communication. *One Note* refers to *One Note Samba: Antonio Carlos Jobim*, and *Bach* to *Invention 1: J.S. Bach*. See appendix C for details

Appendix B

Grid of Transition Tables, an Example

In this appendix all the transition tables of an agent with grid size 5 times 5 are displayed as an example. The agent is trained on the One Note Samba for 10000 learning cycles. The output map of this agent is 10 times 10, so every transition table refers to an area of four output nodes.

Every table has the name of its coordinates. These coordinates are the numbers outside the borders of dashed lines. The element in every table that refers to itself is marked with a star: "*". For example the element in the left top corner of table [0,0] is marked with a star, because this element refers to table [0,0]. The probabilities in the tables are displayed in percent. Apart from rounding errors, all the values of one table should therefore add up to 100.

```
----------- start table -----------

      0   1   2   3   4
   --------------------
  0|   3   4  52   3   2
  1|   0   0   1   0   0
  2|   5   3   1   3   7
  3|   1   1   1   1   2
  4|   3   2   1   1   4
```

```
----------- table [0, 0] -----------

      0   1   2   3   4
    --------------------
  0|  *9   6   5  36  18
  1|   1  15   3   1   0
  2|   1   0   0   0   0
  3|   0   0   0   0   0
  4|   3   0   0   0   1

----------- table [0, 1] -----------

      0   1   2   3   4
    --------------------
  0|   7 *37  10  32   2
  1|   0   2   4   1   0
  2|   1   0   0   0   0
  3|   0   0   0   0   0
  4|   3   0   0   0   1

----------- table [0, 2] -----------

      0   1   2   3   4
    --------------------
  0|   4  38 *25   6  15
  1|   0   1   4   1   0
  2|   1   0   0   0   0
  3|   0   0   0   0   0
  4|   3   0   0   0   1
```

----------- table [0, 3] -----------

	0	1	2	3	4
0	31	4	21	*8	6
1	1	2	17	3	1
2	1	0	0	0	0
3	0	0	0	0	0
4	3	0	0	0	1

----------- table [0, 4] -----------

	0	1	2	3	4
0	35	2	21	4	*8
1	1	2	16	3	2
2	1	0	0	0	0
3	0	0	0	0	0
4	3	0	0	0	1

----------- table [1, 0] -----------

	0	1	2	3	4
0	6	3	4	14	7
1	*18	16	14	2	1
2	2	2	1	0	1
3	1	1	0	0	0
4	5	1	0	0	1

----------- table [1, 1] -----------

	0	1	2	3	4
0	8	3	3	6	38
1	5	*7	11	10	0
2	1	1	1	0	0
3	0	0	0	0	0
4	5	0	0	0	1

----------- table [1, 2] -----------

	0	1	2	3	4
0	4	7	6	10	27
1	2	26	*6	4	1
2	0	0	1	0	0
3	0	0	0	0	0
4	4	0	0	0	1

----------- table [1, 3] -----------

	0	1	2	3	4
0	4	7	8	13	16
1	1	4	13	*10	9
2	0	0	3	3	2
3	0	0	0	0	0
4	3	0	0	0	1

```
----------- table [1, 4] -----------

       0   1   2   3   4
    --------------------
  0|   3   4  17   8   9
  1|   1   0   4  11 *11
  2|   1   2   1   2   6
  3|   0   0   3   2   9
  4|   1   0   1   2   1

----------- table [2, 0] -----------

       0   1   2   3   4
    --------------------
  0|   0   0   0   0   0
  1|   0   0   0   0   0
  2| *17  17  17   0  12
  3|   2   2   1   1   3
  4|   2   2   4   1  18

----------- table [2, 1] -----------

       0   1   2   3   4
    --------------------
  0|   1   0   0   0   0
  1|   1   0   0   0   0
  2|  60  *1   2   0   0
  3|   4   5   1   0   0
  4|   6  16   1   0   0
```

```
----------- table [2, 2] -----------

       0   1   2   3   4
    --------------------
  0|   1   2   1   1   1
  1|   0   2   3   2   0
  2|  30   1  *2   1   0
  3|   3   3   1   0   0
  4|   1  33  10   0   0

----------- table [2, 3] -----------

       0   1   2   3   4
    --------------------
  0|   1   1  10   0   0
  1|   0   0   1   1   1
  2|   1  10   1  *1  25
  3|   0   1   1   5  29
  4|   0   0   1   8   1

----------- table [2, 4] -----------

       0   1   2   3   4
    --------------------
  0|   0   0   1   0   0
  1|   0   0   0   1   2
  2|   9   0   0  35  *9
  3|   1   1   0   6  22
  4|   5   0   1   3   1
```

----------- table [3, 0] -----------

	0	1	2	3	4
0	0	0	0	0	0
1	3	0	0	0	1
2	8	6	0	3	2
3	*5	10	6	3	11
4	4	10	3	5	19

----------- table [3, 1] -----------

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	10
2	5	5	1	14	1
3	6	*7	5	7	5
4	6	9	5	2	11

----------- table [3, 2] -----------

	0	1	2	3	4
0	2	1	1	1	1
1	0	0	1	0	0
2	4	3	5	1	1
3	11	13	*6	2	1
4	7	16	13	4	2

----------- table [3, 3] -----------

	0	1	2	3	4
0	0	0	4	0	0
1	0	0	0	0	1
2	5	4	18	9	21
3	1	2	2	*4	9
4	3	2	2	8	3

----------- table [3, 4] -----------

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	4
2	4	0	0	14	12
3	2	1	0	28	*9
4	7	2	1	12	4

----------- table [4, 0] -----------

	0	1	2	3	4
0	8	8	10	11	9
1	1	4	6	2	1
2	10	0	1	1	1
3	5	3	1	1	1
4	*4	4	3	1	5

```
----------- table [4, 1] -----------

      0   1   2   3   4
    --------------------
  0|   0   0   0   0   0
  1|   0   0   0   0   1
  2|   4   2   0   1   1
  3|   5  29   3   2   4
  4|   3  *5   5   3  28

----------- table [4, 2] -----------

      0   1   2   3   4
    --------------------
  0|   1   0   1   0   0
  1|   0   0   0   0   0
  2|   5   2  19   0   4
  3|   4   6   9   5   4
  4|   4   9  *8   6  11

----------- table [4, 3] -----------

      0   1   2   3   4
    --------------------
  0|   0   0   1   0   0
  1|   0   0   0   0   4
  2|   5   0   0   4   8
  3|   2   2   2  19  12
  4|   8   2  11 *12   5
```

```
----------- table [4, 4] -----------

      0   1   2   3   4
    --------------------
  0|   4   4   4   3   2
  1|   0   1   2   0   0
  2|   0   0   0   1  15
  3|   0   0   0  10  19
  4|   1   0   0  22 *10
```

Appendix C

The Songs

C.1 Etude Op.10 N.1 in C maj: Chopin F

7 5 4 -4 7 5 4 -4 7 5 4 -4 7 5 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 0 9 3 5 -5 9 3 5 -5 9 3 5 -5 9 3 4 -4 -3 -9 4 -4 -3 -9 4 -4 -3 -9 2 -2 -3 -9 -1 8 4 3 -3 8 4 3 -3 8 4 3 -3 8 2 5 -5 -3 -6 2 -5 -3 -6 2 -5 -3 -6 2 -5 -3 -6 0 5 3 6 -2 5 3 6 -2 5 3 6 -2 5 3 6 -7 -2 -6 3 -7 -2 -6 3 -7 -2 -6 4 -8 -2 -6 1 7 5 4 -4 7 5 4 -4 7 5 4 -4 7 5 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 0 5 7 5 -5 5 7 5 -5 5 7 5 -5 5 7 4 -4 -6 -6 4 -4 -6 -6 4 -4 -6 -6 4 -4 -6 -6 0 7 5 2 -2 7 5 2 -2 7 5 2 -2 7 5 2 -3 -4 -8 3 -3 -4 -8 3 -3 -4 -8 3 -3 -4 -8 3 5 7 2 -2 5 7 2 -2 5 7 2 -2 5 7 2 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 4 8 4 1 -1 8 4 1 -1 8 4 1 -1 8 4 1 -3 -3 -9 3 -3 -3 -9 3 -3 -3 -9 3 -3 -3 -9 0 9 3 2 -2 9 3 2 -2 9 3 2 -2 9 3 2 -4 -3 -9 4 -4 -3 -9 4 -4 -3 -9 4 -4 -3 -9 0 9 3 4 -4 9 3 4 -4 9 3 4 -4 9 3 3 -4 -2 -10 4 -4 -2 -10 4 -4 -2 -10 4 -4 -2 -10 0 10 2 5 -5 10 2 5 -5 10 2 5 -5 10 2 5 -5 -3 -9 5 -5 -3 -9 5 -5 -3 -9 5 -5 -3 -9 5 5 4 6 -3 5 4 6 -3 5 4 6 -6 -4 -5 15 -7 -3 -5 3 -7 -3 -5 3 -7 -3 -5 14 -6 -3 -7 0 5 5 5 -3 5 5 5 -3 5 5 5 -3 5 5 5 -6 -4 -5 3 -6 -4 -5 3 -6 -4 -5 3 -6 -4 -5 -2 0 7 3 6 -4 7 3 6 -4 7 3 6 -4 7 5 3 -5 -7 -3 3 -5 -7 -3 3 -5 -7 -3 3 -5 -7 -3 0 3 6 6 -3 3 6 6 -3 3 6 6 -3 3 6 6 -7 -5 -4 4 -7 -5 -4 4 -7 -5 -4 4 -7 -5 -4 -1 7 3 6 -4 7 3 6 -4 7 3 6 -4 7 3 6 -6 -4 -6 4 -6 -4 -6 4 -6 -4 -6 4 -6 -4 -6 -1 0 7 5 4 -4 7 5 4 -4 7 5 4 -4 7 5 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 0 5 4 6 -3 5 4 6 -3 5 4 6 -3 5 4 6 -7 -3 -7 5 -7 -3 -7 5 -7 -3 -7 4 -6 -3 -7 0 5 4 7 -4 5 4 7 -4 5 4 7 -4 5 4 7 -7 -4 -7 6 -7 -4 -7 6 -7 -4 -7 4 -5 -4 -7 0 6 3 7 -4 6 3 7 -4 6 3 7 -4 6 3 5 -5 -3 -7 3 -5 -3 -7 5 3 7 -3 5 3 7 -3 1 -5 -3 -7 3 -5 -3 -7 5 4 6 -3 5 4 6 -3 2 -5 -4 -7 4 -5 -4 -7 5 4 7 -4 5 4 7 -4 2 -5 -4 -6 3 -5 -4 -6 3 -5 -4 -6 40 -6 -3 -7 4 -6 -3 -7 4 -6 -3 -7 0 5 4 8 -5 5 4 8 -5 5 4 8 -5 5 4 8 -8 -4 -5 5 -8 -4 -5 5 -8 -4 -5 6 -6 -4 -5

-2 7 5 4 -4 7 5 4 -4 7 5 4 -4 7 5 4 -4 -5 -7 4 -4 -5 -7 4 -4 - 5 -7 4 -4 -5 -7 0 9 3 5 -5 9 3 5 -5 9 3 5 -5 9 3 4 -4 -3 -9 4 -4 -3 -9 4 -4 -3 -9 2 -2 -3 -9 -1 8 4 3 -3 8 4 3 -3 8 4 3 -3 8 2 5 -5 -3 -6 2 -5 -3 -6 2 -5 -3 -6 2 -5 -3 -6 0 5 3 6 -2 5 3 6 -2 5 3 6 -2 5 3 6 -7 -2 -6 3 -7 -2 -6 3 -7 -2 -6 4 -8 -2 -6 1 7 5 4 -4 7 5 4 -4 7 5 4 -4 7 5 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 4 -4 -5 -7 0 5 7 5 -5 5 7 5 -5 5 7 5 -5 5 7 4 -4 -6 -6 4 -4 -6 -6 3 -3 -6 -6 3 -3 -6 -6 0 7 5 2 -2 7 5 2 -2 7 5 2 -2 7 5 2 -3 -4 -8 3 -3 -4 -8 3 -3 -4 -8 3 -3 -4 -8 1 9 3 2 -2 9 3 2 -2 9 3 2 -2 9 3 3 -3 -3 -9 3 -3 -3 -9 3 -4 -2 -9 3 -4 -2 -10 0 9 3 5 -5 9 3 5 -5 9 3 5 -5 9 3 5 -5 -3 -9 5 -5 -3 -9 5 -5 -3 -9 5 -5 -3 -9 3 7 3 5 -3 7 3 5 -3 7 3 5 -3 7 3 5 -6 -4 -5 3 -6 -4 -5 3 -6 -4 -5 3 -6 -4 -5 -2 7 5 4 -4 7 5 4 -4 10 2 7 -7 -2 -6 14 -6 -3 -6 3 -6 -3 -6 14 -6 -3 -6 3 -6 -3 -6 -2 7 5 4 -4 7 5 4 -4 7 5 4 -4 -5 -7 15 -6 -3 -6 3 -6 -3 -6 14 -6 -3 -6 3 -6 -3 -6 -1 6 3 6 -3 6 3 6 -3 6 3 6 -3 6 3 6 -7 -3 -6 4 -7 -3 -6 4 -7 -3 -6 15 -6 -3 -6 -1 6 3 7 -4 6 3 7 -4 6 3 7 -7 -3 -6 15 -6 -3 -7 4 -6 -3 -7 4 -6 -3 -7 4 -6 -3 -7 0 9 3 5 -5 9 3 5 -5 9 3 5 -5 9 3 5 -5 -3 -9 5 -5 -3 -9 5 -5 -3 -9 5 -5 -3 -9 -7

C.2 Invention 1: J.S. Bach

2 2 1 -3 2 -4 7 5 -1 1 2 -7 2 2 1 -3 2 -4 7 5 -2 2 -3 5 -2 -2 -1 3 -2 4 -2 -2 -1 -2 -2 4 -2 3 -1 -2 -2 -1 -2 3 -1 3 -2 -1 -2 -2 -1 3 -2 4 -2 -7 10 2 -3 -2 -2 -1 -2 3 -1 3 -2 4 -2 3 -1 3 -2 4 -2 -3 1 2 5 -8 -2 -2 0 0 2 2 1 -3 2 -4 -1 3 2 1 2 -3 1 -3 2 3 -2 -1 -2 3 -1 3 -2 4 -2 -2 -1 3 -1 3 -2 -1 1 2 1 -8 2 2 1 -8 2 1 2 1 2 -10 2 2 1 -3 2 -4 12 -2 -2 4 -2 -2 -1 3 -2 9 -1 3 -2 -5 1 -3 -6 9 -1 -2 -2 -1 -2 0 12 -2 -2 -1 3 -2 4 -2 -3 1 2 2 -4 2 -3 1 2 -2 -1 -2 3 -1 3 -2 -3 2 1 2 -3 1 -3 2 -4 2 2 1 -3 2 -4 2 2 1 2 2 -4 2 -3 1 2 2 2 1 -3 2 -4 5 -5 -3 -2 -2 0 -2 -1 -2 -2 4 -2 3 -1 2 1 -8 -2 10 -7 6 1

C.3 Chinese Traditional Song: Traditional

0 3 2 -5 -2 2 -2 0 2 3 2 -10 3 -3 5 -2 2 -2 -3 3 2 0 3 2 2 0 -2 -2 2 2 -2 -2 -3 -2 5 -3 -2 0 2 3 2 -10 12 -12 12 5 2 -2 -2 2 -2 -3 -2 2 -2 -2 2 2 -4 2 5 2 3 4 -2 0 2 3 -3 -2 -2 2 -2 4 -2 -2 0 2 2 3 -5 2 -2 -12 12 2 3 -3 -2 -2 0 0 4 -2 2 0 -2 -5 0 0 3 0 2 -2 -3 -2 0 5 2 5 -19 -3 -2 0 2 3 2 -10 12 12 0 -12 5 2 -2 -2 2 -2 -3 -2 2 -2 -2 2 2 -4 2 7 2 -2 2 -2 2 -2 2 -2 2 -2 2 -2 2 -2 2 -2 -2 2 2 -2 2 -2 2 -2 2 -2 2 -2 2 -2 2 -2 -5 5 2 -4 0 2 -2 0 0 2 2 -7 0 0 3 2 -7 0 2 3 -3 -2 -2 2 -2 -3 3 2 2 3 -5 0 17 -3 -2 0 2 -2 2 3 -7 2 -2 4 -2 -2 0 2 -2 2 2 -7 2 -2 2 -2 0 3 2 -7 -3 0 12 -12 3 -3 12 -12 3 -3 12 -12 3 -3 5 -5 5 -2 5 -3 5 2 -4 0 4 -2 -5 3 -5 7 -5 7 -4 2 -5 -2

5 -8 5 -7 5 7 5

C.4 Chinese Gold Snake Dance: Traditional

3 -5 2 3 -5 2 -4 -1 -2 0 5 0 -5 3 -1 -2 -2 2 3 0 -8 3 2 3 -3 -2 -3 3 -5 2 0 -2 12 0 -2 0 2 0 -5 0 5 -2 0 -8 3 2 3 -3 0 3 2 0 2 3 -3 3 0 -3 -2 0 2 -2 -2 -3 0 5 0 -5 3 -1 -2 -2 2 3 0 -8 3 2 3 -3 -2 -3 3 -5 2 0 -2 12 2 -2 2 -2 -2 2 -7 2 -2 2 -7 2 3 7 2 -2 2 3 -3 -2 -7 2 -2 2 -7 2 3 7 2 -2 -2 2 -7 2 -7 2 3 7 2 -2 -7 2 -2 7 2 -2 -7 2 -2 7 0 -7 0 7 0 -7 0 7 0 -7 7 -7 7 -7 7 -7 0 0 0 0 0 0 0 0 0 0 0 7 0 -2 0 2 0 -5 0 0 5 -2 0 -8 3 2 3 -3 0 3 2 0 0 2 3 0 -3 3 0 -3 -2 0 0 2 -2 -2 -3 0 5 0 -5 3 -1 -2 -2 2 3 0 -8 3 2 3 -3 -2 -3 -2 14 0 -2

C.5 One Note Samba: Antonio Carlos Jobim

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0 2 1 2 -2 -1 -2 -2 -2 -1 -2 -2 -1 1 2 2 -5 -2 2 1 4 -5 -2 10 2 1 2 -2 -1 -2 -2 -2 -1 -2 -2 -1 1 2 2 -5 -2 2 1 4 3 -1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Bibliography

[1] X. Amatriain. An oo framework fordeveloping audio and music applications. In *Proceedings of 17th Annual ACM Conference on Object-OrientedProgramming*, 2002.

[2] Bart de Boer. A first report on emergent phonology. Technical report, Vrije Universiteit Brussel, AI-lab, 1996.

[3] Bart de Boer. Self organisation in vowel systems through imitation. In J. Coleman, editor, *Computational Phonology, Third Meeting of the ACL SIGPHON.*, pages 19–25, July 1997.

[4] Bart de Boer. Self organization in vowel systems. *Journal of Phonetics*, 28(4):441–465, 2000.

[5] P. R. Cook and G. P. Scavone. The synthesis toolkit(STK) version 2.1. In *Proceedings of the International ComputerMusic Conference*, Beijing, 1999.

[6] P. F. Dominey. A shared system for learning and temporal structure of sensori-motor sequences? evidence from simulation and human experiments. *Cognitive Brain Research*, (6):163–172, 1998.

[7] S. Haykin. *Neural Networks: Comprehensive Foundation.* Prentice Hall, second edition, 1999.

[8] D. Hebb. *The organization of behavior: a neuropsychological theory.* Wiley, New York, 1949.

[9] J. Hurford. The evolution of language and languages. In R. Dunbar, C. Knight, and C. Power, editors, *The Evolution of Culture*, pages 173–193, Edinburgh, 1999. University press.

[10] Daniel L. James and Risto Miikkulainen. Sardnet: A self-organizing feature map for sequences. *Advances in Neural Processing Systems*, (7), 1995.

[11] Tuevo Kohonen. *Self-organizing maps*. Springer, Heidelberg, New York, 1995.

[12] C. G. Langton, editor. *Artificial Life: An Overview.* MIT press, Cambridge (MA), 1997.

[13] Eduardo R. Miranda and Peter M. Todd. A-life and musical composition: A brief survey. In *IX Brazilian Symposium on Computer Music: Music as Emergent Behaviour*, pages 59–65, Campinas, Brazil, 2003. UNICAMP.

[14] Eduardo Reck Miranda. *Composing Music with Computers.* Focal Press, Oxford, 2001.

[15] Eduardo Reck Miranda. Emergent sound repertoires in virtual societies. *Computer Music Journal*, 26(2):81–94, 2002.

[16] Eduardo Reck Miranda. Mimetic development of intonation. In *Proceedings of the 2nd International Conference on Music and Artificial Intelligence.* ICMAI, Springer Verslag, September 2002. Lecture Notes on Artificial Intelligence.

[17] M. Pearce and Wiggins. Towards a framework for the evaluation of machine compositions. In *Proceedings of the AISB01 Symposium on Artificial Intelligence and Creativity in Arts and Science*, pages 22–32, 2001.

[18] J. Rothstein. *MIDI: A comprehensive introduction.* A-R editions, Madison, Wisconsin, USA, 1992.

[19] Malcolm Slaney and Richard F. Lyon. A perceptual pitch detector. In *Proceedings of the 1990 International Conference on Acoustics, Speech and Signal Processing*, pages 357–360, Albuquerque, 1990. NM, IEEE.

[20] K. Smith. The cultural evolution of communication in a population of neural networks. *Connection Science*, 14(1):65–84, 2002.

[21] Luc Steels. Synthesising the origins of language and meaning using co-evolution, self-organisation and level formation. In J. Hurford, editor, *Evolution of Human Language*. Edinburgh University Press, Edinburgh, 1997.

[22] Tim Taylor. Creativity in evolution: Individuals interactions and environments. In Peter J. Bentley and David W. Corne, editors, *Creative Evolutionary Systems.*, pages 79–108. Academic Press, London, 2002.

[23] Peter M. Todd and Gregory M. Werner. Frankensteinian methods for evolutionary music composition. In N. Griffith and P.M. Todd, editors, *Musical Networks: Parallel Distributed Perception and Performance*. MIT Press, Cambridge MA, 1999.

[24] T. Tolonen, V. Vlimki, and M. Karjalainen. Evaluation of modern soundsynthesis methods. Technical Report 48, Helsinki University of Technology, Laboratory of Acoustics and Audio Signal Processing, Espoo, Finland, 1998. Available on-line: http://www.acoustics.hut.fi/publications/.12.

[25] Rainer Typke, Remco C. Veltkamp, and Frans Wiering. Searching notated polyphonic music using transportation distances. In *proceedings blablabla*, 2004.

[26] Gert Westermann and Eduardo Reck Miranda. Modelling the development of mirror neurons for auditory-motor integration. *Journal of New Music Research*, 31(4):367–375, 2002.

[27] M. Westhead and A. Smaill. Automatic characterisation of musical style. In W. Woods, editor, *In Music Education: an AI approach*, pages 157–170. Springer-Verlag, 1993.