Imperial College London

Department of Computing

Simulating Infant's Early Learning using Strong Attractors

By

Kaiyan XIAO

Submitted in partial fulfilment of the requirements for the MSc Degree in Advanced Computing of Imperial College London

September 2013

Abstract

The interactions between an infant and his/her primary caregiver in the early childhood of the infant have a strong impact on his/her action behavior in later life. In this project, I designed a system to simulate an infant's learning and decision-making mechanism regarding to his/her behavior, inspired by an arousal-based model recently introduced in a work of Hiolle et al. In this design, I apply the strong attractors as the precious memories. This system models how a caregiver responds to infant's action by probability distribution, while this response is a key factor in the infant's learning procedure. Attachment theory is a widely accepted and studied theory in developmental psychology. This system is assessed by consequences derived from this theory. A simplified version of the 'Strange Situation Procedure', which the original one is efficient at distinguishing different attachment types, is designed for the same purpose in this project. The design includes a corresponding experiment and two tasks. The system is then assessed by whether two tasks could be accomplished. Results of experiment demonstrate that the designed system has the ability to identify different baby's behavior and basic attachment types. In addition to evaluating the system, two comparisons of different choices for a component in the system and two analyses of a part of the system are attempted. The comparisons show that the current configuration is reasonable and the analyses suggest us two potential improvements.

Acknowledgements

I would like to thank my supervisor, Prof. Abbas Edalat, firstly, for offering me the opportunity to attempt this attractive project.

Thanks to my classmate and friend Zheng Lin as well, for the discussions with him in the concerned area.

And many thanks to Bingqing Qu, Chenxi Li, Siyi Jiang and Qian Zhao, for the supports and joy they bring to me.

Table of Contents

1 Introduction

1.1 Motivation

With advancement of medical science, more and more physical diseases have been found corresponding therapies. However, the mental disorders are still relatively hard to be healed, as the knowledge concerning our mental activities and our brain is far away from enough. Without enough related knowledge, human cannot develop effective psychotherapies for the mental diseases like the modern medical treatments for physical diseases.

Besides of the lack of efficient psychotherapies, the mental conditions of modern people are generally really bad. Work pressure, fast rhythm of everyday life, irregular schedule of rest and so on, all these characteristics of modern society bring us a much higher probability of having a mental disorder than the elder generations. This mental disorder is probably not a serious disease, but it could also upset your life.

In addition, the prejudice to the patient with serious mental disease makes me uncomfortable as well. Some serious mental disorder may make its patients dangerous to others, thus people shows prejudice and disgust to these patients, even after the patients are cured.

I have heard for many times that a cured patient still receives indifference and refuse from others after leaving psychiatric hospital. The faith is really unfair to these patients. They are innocent. But if we want to ameliorate this situation, we should advance our theories concerning mental disorders and improve psychotherapies.

There are many theories in psychology. Attachment theory is a famous one. After many years of development, this theory becomes the dominant theory in the study of infant behavior and in the fields of infant mental health. It is also the guiding theory of this project.

1.2 Contributions

The main achievement of this project is to build a system simulating the infant's emotional development in his/her early childhood and to assess the performance of this system with respect to Attachment Theory, a main scientific analysis in developmental psychology. The arousal-based model (Hiolle et al. 2012) heavily inspires the conception of this system.

Concisely, the emotional development is modeled by interactions between the infant and his/her caregiver, and this simulation system in fact constructs a virtual learning and decision-making mechanism for baby's behavior.

The performance of system is assessed by completion of two gradually deepened tasks within a designed experiment. The first task aims to identify different simulated virtual baby who are cared by different settings of virtual caregivers. This task has a clear success: the statistics of data recorded in experiment show obvious difference in virtual babies' action behavior. The second task requires an accurate identification of different attachment types. Recent effort makes the system able to identify the basic attachment types. With some more effort in future, accurate identifications look feasible.

For purpose of improving this simulation system, two comparisons and two analyses are performed. From these attempts, two potential improvements are found. The first improvement requires the use of multiple strong attractors in preprocessing procedure of the inputs; the other requires adding a small noise/bias to the strong attractor during storage. The core idea of improvements is to increase system's sensitivity to the external signals.

Overall, this system attempts to establish a connection between the idea of arousal level and attachment theory, but this connection is far from complete. We will describe various methods to improve the system.

1.3 Structure of the Dissertation

This dissertation consists of this introduction and four main chapters.

Chapter 2 covers the background researches during the early period of this project. Concerned knowledge is divided into different topics, and each topic takes a dedicated subsection for detailed discussion.

Chapter 3 describes the concise structure of the designed simulation system. It starts with the overall structure and then goes to the details of each piece of the whole system. In purpose of understanding the configurations easier, each piece has a corresponding illustration.

Chapter 4 constitutes the experimental studies of this project. Firstly the design of experiment is introduced, with its results. After this main experiment, two comparisons and two analyses of the system have been added, in order to understand the system better and find out potential improvements.

Chapter 5 contains the general evaluation and conclusion regarding to the designed system. The comparisons, analyses and potential improvements are mentioned as well. At last, the directions for future work are listed.

2 Background

In this chapter, we will introduce the concerned knowledge for conceiving the simulation system. This knowledge is divided into several topics, and each topic takes a subsection.

2.1 Attachment theory

The theory of attachment could be rooted to an evolutionary and ethological theory introduced in John Bowlby's famous book 'Attachment and Loss' (Bowlby, 1969). This theory postulates that a newborn baby will spontaneously seek proximity with caregivers and the interactions between the baby and his caregivers will influence how the baby will carry out the relationship with others in future. Especially, if the caregivers respond to baby's demand differently, the baby will also have different behavior on treating others.

This theory is explained by the fact that a children's brain is not fully developed in the early childhood, and consequently his/her emotional and cognitive ability is still sensitive to internal and external factors. In fact, these abilities could still be molded even after the early childhood.

According to different relationships shown between a baby and his/her caregiver, we could categorize the relationships into four major attachment patterns:

2.1.1 Secure attachment:

This is the favorite attachment type. One toddler with secure attachment type will feel free and motivated to explore the environment when the caregiver is present, will feel upset once the caregiver disappears and will feel happy to see the caregiver comes back.

Secure Attachment is helpful for emotional and cognitive development and makes the toddler familiar to exploration. The children with this attachment will be little possible to have mental disorders when they become adult.

2.1.2 Anxious-resistant insecure attachment:

This attachment type could also be called anxious-ambivalent insecure attachment. Because a toddler with this attachment will feel worry about strangers even when caregiver is present; if the caregiver goes away, the toddler will feel very depressed, and he/she will be ambivalent (want to contact caregiver but be angry as well) once the caregiver returns.

The caregivers of the toddlers with this attachment are always found to be bad at taking care of baby, do not understand baby's demand and have inconsistent behaviors.

2.1.3 Anxious-avoidant insecure attachment:

A toddler with this type attachment will neglect the presence of caregiver. He/she has no special emotional presentations whenever the caregiver is present or absent. There is neither special emotion when a stranger appears. The toddler is never motivated to explore the environment.

The caregivers of toddlers with this attachment are often impatient and insensitive to baby's demand. The toddler will normally become indifferent after growing up.

2.1.4 Disorganized attachment:

A toddler with this attachment usually has no regular behavior for reactions to environment. He/she will choose a response according to the current state of environment.

In most time, such a toddler had a scaring caregiver. The relationship between caregiver and toddler is easy to change. This is why the toddler could not have a regular behavior to respond to others.

We could observe that only the first attachment is desirable, the other three are all insecure attachment and not good for a human. In the early childhood of a baby, his/her caregiver must try to avoid these three insecure attachments.

2.2 Strange situation protocol

Mary Ainsworth proposed a method to assess the attachment type of an infant (Ainsworth et al. 1978), after her careful in-depth and longtime observations of infants with their mothers in Uganda. This method is named 'strange situation procedure'.

In this procedure, the infant and the mother will be placed in an unfamiliar room with some toys. The procedure consists of eight episodes, including the situations of mother's presence, mother's absence and mother's reunion as well as the presence of a stranger.

Concisely, these eight episodes are:

- Episode 1: mother, baby, experimenter (30 seconds)
- Episode 2: mother, baby (3 minutes)
- Episode 3: mother, baby, stranger (3 minutes or less)
- Episode 4: stranger, baby (3 minutes)
- Episode 5: mother, baby (3 minutes)
- Episode 6: baby alone (3 minutes or less)
- Episode 7: stranger, baby (3 minutes or less)
- Episode 8: mother, baby (3 minutes)

The infant's actions during the procedure will be recorded and analysed to assess which type of attachment the infant will possess.

2.3 Arousal based model

In the paper, 'Eliciting Caregiving Behavior in Dyadic Human-Robot Attachment-Like Interactions' (Hiolle et al. 2012), Hiolle et al. built up a very interesting robot system which could be implemented in a Sony AIBO robot, could act as a young baby and could explore the novel environment around it.

This innovative system is based on a measurement named 'arousal level' and the model is therefore called an arousal-based model. The arousal level, like the name describes, is a kind of evaluation of the vigilance. When the arousal level is relatively high, the robot will try to catch its human caregiver's attention and seek for care. When the arousal level is moderate, the robot will explore the environment around.

The evaluation of this arousal level makes use of the sensors of the AIBO robot, in order to gather as much information as possible about the environment the robot situates. All the gathered information will be formatted to an appropriate form and then will be passed into two neural networks. The first one is a sparsely connected associative memory and the second is a self-organized map (we could call it a Kohonen map as well). The difference of the states of two networks before and after the update will be measured and be incarnated in two values, named 'Surprise' and 'Categorization Adjustment'. The average of these two values is then the arousal level. However this arousal level is too sensitive to the changes of the networks. Thus the level will be smoothed over time and called 'instantaneous arousal level'. In the next step, the influences of touching and face recognition will be evaluated and added into this instantaneous arousal level. After all, the affected level will be smoothed another time to become the 'sustained arousal level', which is the final evaluation.

How the robot decides to act is controlled by this 'sustained arousal level'. The robot has a list of possible actions, including 'move', 'look for human' and 'bark'. If the sustained arousal level is lower than a preconfigured threshold, then the robot will move, which means to explore the novel environment; if the sustained arousal level is higher than another preconfigured threshold, then the robot will look for human, which means to seek for care; if the instantaneous arousal level, not the sustained arousal level, is high enough, then the robot will bark, which means the robot feels a little uncomfortable.

Two experiments are performed to illustrate two interesting characteristics of this arousal-based model:

i. The involvement of the human caregiver could largely influence the learning results of the model; similar to the how a primary caregiver influences his/her baby during the early childhood.

ii. Human could distinguish different configurations of the robot and will respond the robot's needs differently.

The samples of experiments are also studied for different age groups or different technology background. This ensures that the robot does not require its caregiver to have much professional knowledge and gives the robot a potential to be used widely.

2.4 Small world associative memory

The sparsely connected network, applied in arousal level model, is a two-dimensional example of small world associative memory.

The original idea of small world network could be dated back to 1998, in the excellent work of Duncan Watts and Steven Strogatz (Watts et al. 1998). In their work, they innovatively invented a network architecture whose connection topology is neither fully regular nor fully randomly, but a mix of these two configurations. In addition, the small world network is shown to possess advantageous signal-propagation speed, computational power and synchronizability.

After this, Jason Bohland and Ali Minai have analyzed the performance of small world network (Bohland et al. 2001). In their work, they showed that if the one-dimensional small world architecture is applied as associative memory, the retrieval performance of this architecture could be as good as randomly connection configuration while some conditions about the total connection length are satisfied.

By following the contributions of predecessors, Lee Calcraft, Rod Adams and Neil Davey compared the performance of three concrete one-dimensional small world architectures (Calcraft et al. 2006). The first and the second configurations use Gaussian and exponential distributions in the random connections, while the last one rewires the part of random connections progressively. Their work figured that the Gaussian and exponential architecture are better than the progressively rewiring one, while the better two architectures have little difference on performance.

The studies on small world associative memory so far were all focusing on one-dimensional networks. It is still Lee Calcraft, Rod Adams and Neil Davey who extended their consequences into two-dimensional networks (Calcraft et al. 2007). Unsurprisingly, the consequences are still valid in two-dimensional space. The Gaussian and exponential architectures are still the best choices. Moreover, the authors have given out an algorithm for updating and learning stages of a two-dimensional small world associative memory. This algorithm is the one applied in arousal-based model (Hiolle et al. 2012), in the sparsely connected associative memory just after gathering the input signals. This associative memory will be used to help calculate the arousal level.

2.5 Self-organized map

The self-organized map, or called Kohonen map, due to its inventor Teuvo Kohonen, is an artificial neural network using unsupervised learning. Since the first appearance in 1982 (Kohonen 1982) this neural network has been widely used in representing the input space of training data in a low dimensional (usually two-dimensional) and discretized approach. Thus if the dimension of the input space is high, using the self-organized map is a really appropriate method to visualizing the data.

Specifically, a self-organized map consists of a certain number of neurons. These neurons construct a hexagonal or rectangular grid. Each neuron are attached a weight vector, while the dimension of this weight vector is exactly same to the dimension of input sample vector.

The self-organized map, similar to most other widely used artificial neural networks, has two modes of operation. One is 'learning', to learn what the weight vectors of the neurons are from a set of input samples; the other is 'mapping', to map a new input vector by using the learned neural network.

The aim of the learning stage is to make the topological structure of weight vectors as similar as to the structure of the input samples. A standard learning algorithm is like the statements below:

i. Initialized the weight vectors of the self-organized map. Normally this step will be done by randomly generation. But if the weight vectors are chosen from the subspace spanned by the two largest principal component eigenvectors of the input samples, the learning will be faster.

ii. For one input vector X^i:

    a. Find out the Best Match Unit (BMU) of the neurons. One possible method is to calculate the Euclidean distance from the weight vector of a neuron to the input vector for every neuron, and then to choose the one with minimal distance as the Best Match Unit.

    b. Update the weight vectors of neurons by the following formula, with the involvement of a neighborhood function:

    $$W_u(t+1) = W_u(t) + nbh(u, BMU, t)\alpha(t)\left[X^i - W_u(t)\right]$$

    The neighborhood function $nbh(u, BMU, t)$ is an assessment of how much a weight vector will be changed according to the distance from this weight vector to the BMU. The function $\alpha(t)$ is the learning rate function which is typically decreasing when time increases.

iii. Go to step ii if there is another input vector or the iteration limits is not attaint.

The most interesting characteristic of a self-organized map is that the map could preserve the topology of space of input data. This is also why a neighborhood function is introduced in the learning formula.

This characteristic is as well the reason that arousal-based model uses such a map in assessing the categorization of the data, because the topological properties of the data are not damaged in the map.

2.6 Hopfield network

Hopfield network may be the most popular artificial neural network for simulating an associative memory (Herz, J., Krogh, A., & Palmer, R. G. 1991). This network has conquered many researchers owing to the elegance and simplicity of its structure and learning/updating rules, since its invention.

A typical Hopfield network will have a fully connection configuration. In other words, for every two different neurons in the network, they are connected, bi-directionally and symmetrically. Binary neurons are the first choice for neuron in the network, while 1 stands for fire and 0 for silence.

To update a Hopfield network is not complicated. For an arbitrary neuron, S_i, we just need to sum the weighted signals from its connected neurons, noted as $h_i = \sum_{j=1,j\neq i}^{N} W_{ji}S_j$, and pass the sum to neuron's activation function, which normally is a threshold function. In the formula above, N is the total number of neurons in the Hopfield network and W_{ji} is the weight of connection from neuron j to neuron i. As the connections are symmetrically, we always have $W_{ji} = W_{ij}$.

Hopfield network applies the Hebbian rule (Hebb, D. O. 2002) as the learning rule. Hebbian rule emphasizes that 'Cells that fire together, wire together' (Doidge, N. 2007). Concisely, if a new pattern $\{\xi_i\}$ will be memorized, the increase of the weight of connection between i-th neuron and j-th neuron will be

$$\Delta Wij = \frac{\xi_i \xi_j}{N}$$

Obviously, the learning rule is incremental.

Hopfield network has a very important characteristic: if the network is updated randomly and synchronously, then we could define an 'energy function' using the formula:

$$E = -\frac{1}{2}\sum_{i,j} W_{ij}S_iS_j + \sum_{i} \theta_i S_i$$

where the θ_i is the threshold in the activation function of the i-th neuron.

2.7 Strong attractor

The stored patterns in a Hopfield network could also be called an attractor, as this pattern acts like an attractor in the space of all possible states of the network. In fact, the states in a certain range around an attractor pattern will evolve to this pattern after a certain number of updates.

However, we just store a pattern once in a Hopfield network. In this case, the capability of storage is 0.138N, where N is the number of neurons (Herz, J., Krogh, A., & Palmer, R. G. 1991). But what is the performance of the network if we store a pattern multiple times? (Edalat, A. 2013) A recent work of Abbas Edalat shows that, if there is just one attractor with being stored for d times, the capacity of storage could approximately become $0.138Nd^2$. Such a pattern is normally called a strong attractor.

Not only the capacity of storage increases, the basin of attraction of this strong attractor is also larger than the basin of a standard attractor.

3 Structure of designed system

3.1 Overall explanation

This subsection describes the structure of system briefly. The details will be introduced later.

3.1.1 Objective of the Simulation System

The built system aims to simulate the learning stage of a baby's decision-making ability of his actions, with involvement of his primary caregiver. This system could be passed to a simulation of the famous Strange Situation Protocol, to further observe whether different attachment types could be distinguished under the Attachment Theory's criterion.

An infant will learn and establish his decision-making ability during the first year after his birth. In this one year, baby's primary caregiver plays the most important role. The responses of the caregiver who in most time is baby's mother, to baby's actions, are the key factor in the learning stage. The built system therefore simulates this learning stage and its dependency on the interactions with the caregiver.

3.1.2 Overall Structure of the System

The overall structure of the Simulation System is shown in the picture below.

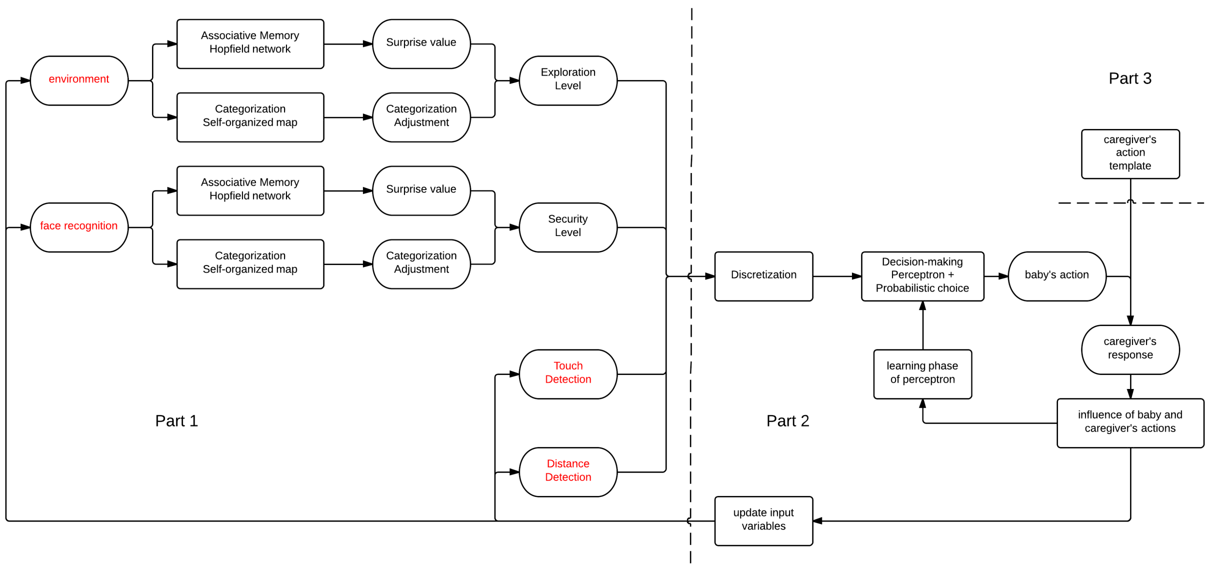


Figure 3.1 The structure of the designed system. Three parts of the system are divided by dashed lines. Rounded rectangles stands for data. The four data in red are the four input signals.

The whole system could be divided into three parts, as shown by the dashed line in the picture. The first part is the preprocessing of the inputs; the second is the main part, decision making of the baby while the third one is just caregiver's action template.

3.1.3 Preprocessing of Inputs

For this simulation system, there are four types of inputs, environment signal, face recognition signal, touch detection and distance detection. All these four types of inputs are surrounded by round rectangle in the structure picture.

For the touch detection and distance detection, the raw signals will be passed into the part of baby's decision making directly. They will be discretized there for further use.

In contrast, the environment and face recognition inputs will complete a sequence of preprocessing before entering in the next part. The treatment for these two signals are extremely similar, thus in the following paragraph only the preprocessing for environment input will be introduced.

The raw signal which is a binary vector, noted as S_{env}, will be passed into two neural network. One is a Hopfield network and the other is a self-organized map (Kohonen map). The Hopfield network plays the role of the related associative memory about environment. The input will be treated like the initial state of the network and the corresponding memorized pattern, noted as M_{env}, will be retrieved. We will then use this outputted pattern to calculate the first of the two measurements about environment input, the Surprise value, by the formula:

$$Surprise = |S_{env} - M_{env}|.$$

The self-organized map is implemented to measure the difficulty of categorization of inputs. The update algorithm will respect the standard one, however a slightly change will be applied to make the algorithm incremental, according to the idea of arousal level in his paper "Eliciting Caregiving Behavior in Dyadic Human-Robot Attachment-Like Interactions". We will cache the changes of the map's units' weights between before and after the update. All these changes of weights will be summed element by element absolutely and then be normalized. The final value is called classification adjustment and it is the second measurement about environment.

The two measurements, the surprise value and the classification adjustments, which are inspired by Hiolle et al. in their work, will be averaged equally like what they did and named as exploration level, which is somewhat self-explicated.

There is one possible deeper discussion that the weights of two measurements in the average could probably not be equal. Therefore how different weights could alter the baby's actions is a valuable topic.

The face recognition signal will have treated similarly. The only difference is that the final value will be called as Security level in place of Exploration level. After all, both these two levels will be discretized and handled in the decision-making part of system.

3.1.4 Decision-Making for baby's actions

The core piece of the decision-making part of the system is a perceptron network. This perceptron mimic baby's decision-making with supports from other pieces of the same part.

First of all, data from the previous part of system will be discretized for the ease of use in current part. In addition, the discretized data will be concatenated in order of environment, face recognition, touch detection and distance detection.

After this, the discretized and concatenated data, which is in binary format, will be passed into the perceptron. The perceptron has five output neurons, corresponding to baby's five possible actions, turn (explore), shout, move to parent, move away from parent and stay.

However, baby could just perform one action at one time. Thus we apply a probabilistic Winner-Take-All unit just after the perceptron. Concisely, the outputs of the five neurons will be normalized to a probabilistic distribution and the final action will be decided probabilistically by this distribution.

The decision of baby will interact with the caregiver's action template, the only content of the third part of system, and derive the caregiver's response to baby's action.

After baby's action and caregiver's response are determined, the influence of their actions will be assessed. This assessment has two applications, one is to update the perceptron in current part of system, and the other is to update the related part of new inputs of the whole system.

3.1.5 Caregiver's Action Template

Corresponding action templates implement how the caregiver will respond to baby's action. One template is a table of multiple probabilistic distributions. Each distribution will be used to decide caregiver's response in the corresponding baby's action and environment parameter (concretely the parameter here is the distance).

Thus the caregiver's action is also a probabilistic result, like the baby's. Different types of caregiver will have different action templates and this will differ the responses to baby's action.

3.2 Four types of input

The simulation system receives four types of signals.

3.2.1 Environment signal:

A binary vector of 100 units is used to present the perceived environment. For each unit in the vector, only two states, +1 and -1, are allowed, which reflect the word 'binary'.

The number of units could not be too small, in reason that the environment has infinity of possibility; but it could neither be too big, because of the limitation of computation capacity of my workstation. Thus one hundred is believed a good tradeoff and therefore chosen.

The signal, the binary vector in fact, will be passed to the related associative memory and self-organized map for pre-processing. The dimension of these two neural networks is carefully chosen to fit the length of this signal.

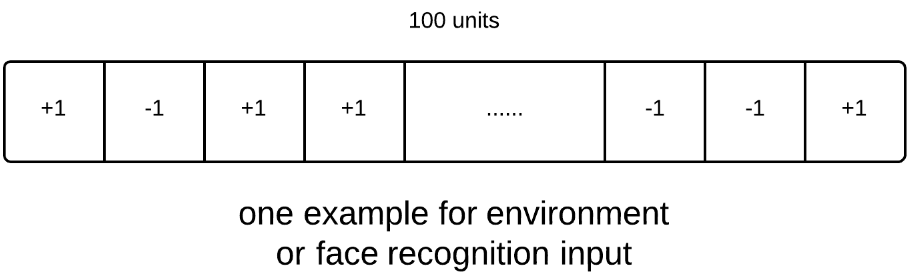


Figure 3.2 One typical example for environment input or face recognition input. These two inputs have exactly the same form: a binary vector of 100 units, while possible states are +1 and -1.

3.2.2 Face recognition signal:

The recognized face in baby's view has been modeled as well.

This face recognition signal has the identical form as the environment signal. This means that one 100 units binary vector, whose possible states are also +1 and -1, is used. In addition, there are also an associative memory and a self-organized map with adapted size after the signal.

3.2.3 Touching detection:

Touching is one of the most important types of interaction between caregiver and baby. Thus modeling the touching is not a surprise. However, as there is no enough time to handle a meticulous simulating system in a short time project, the touching is only divided into two categories, touch and strike. One touch is a positive and encouraging response given to the baby from his caregiver, and it is always related to positive influence in the learning piece of the decision-making network of the simulating system, which will be introduced concisely in section 3.4.

A two unit logical vector is used to store the result of touching detection. These two logical units are two indicators respectively for touch and strike. In other words, if the first logical indicator shows true/1, then a touch is detected; if the second shows true/1, then a strike is received; if both show false/0, then it means there is no touching detection at the moment. We must notice that these two logical indicators are mutually exclusive. Only at most one of them could be true at one time.

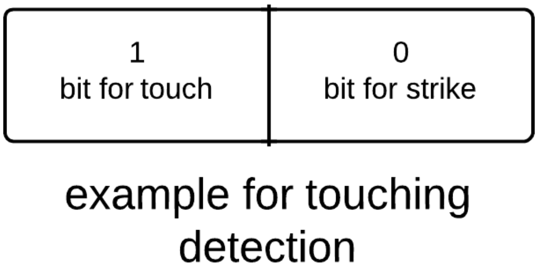


Figure 3.3 This is an example signal for touching detection. The signal consists of two logical units. The first bit corresponds to touch and the second to strike.

3.2.4 Distance detection:

The distance between baby and caregiver is another appropriate aspect to measure the interaction intensity. A caregiver could not take care of her baby very well in a far distance.

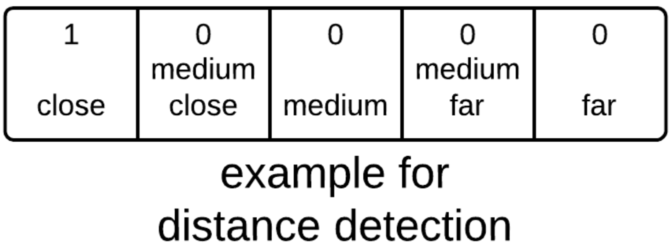


Figure 3.4 An example for the distance dection signal. This signal possesses five logical units. From left to right, these five units correspond to distance grade from close to far.

This signal is modeled into five categories, close, medium-close, medium, medium-far and far. Like the touching detection, five logical indicators are given to these five categories and these indicators are bound into a vector in the listed order. Similarly, these five units are mutually exclusive and if none of them shows true/1, then it means the caregiver is not present.

3.3 Preprocessing of environment and face recognition signals

In the four types of input, the last two, touching detection and distance detection, will be passed directly into the decision-making part of the simulating system. However, the first two, environment and face recognition, will complete a series of preprocessing in order to adapt the following subsystem better.

Specifically, any one of these signals will at first go through two parallel neural networks, of which one is a standard Hopfield network and other is a self-organized map. Secondly, the outputs after these two neural networks will be used in calculations of two measurements. At last, these two measurements will be averaged and the final value is the pre-processed signal, which will enter the next part of simulating system.

Additionally, the preprocessing operations applied to environment and to face recognition are exactly the same. Thus in the following sections, we will only use the environment signal to illustrate the related sub-system.

3.3.1 Associative memory and Hopfield network

As mentioned, the two neural networks are positioned in parallel in the structure of simulating system, for each signal of environment and face recognition. The first network, a Hopfield network, is playing the role of associative memory, which is also the normal and main application of a Hopfield network.

Hopfield network is not the only network that implements an associative memory, but these listed reasons helped me to choose it:

i. Hopfield network is a very solid network. It is invented for more than thirty years, there are numbers of relevant works on this network and I could therefore build the corresponding piece of my simulating system based on these results. In fact, the works of previous researchers really helped me a lot.

ii. A Hopfield network is very computational friendly. The structure of a Hopfield network is not complex. In addition, an easy form of activation function could totally compensate the drawback of great number of connections that need to update.

iii. The Hebbian rule applied in the Hopfield network is incremental. This means the inputs of the network could be handled one by one. This is very important in our case, because the time sequence is one of the key factors and consequently the inputs should be dealt one after another.

In Hiolle et al.'s work (Hiolle, 2012), they used a sparsely connected 2 dimensional associative memory, by applying the idea of Calcraft (Calcraft, L., Adams, R., & Davey, N. 2007). Why this type of network was not used here in my work has multiple reasons. Firstly, the related theoretic support about the sparsely connected memory is less than the classic Hopfield network. Moreover, the sparsely connected network is slightly harder to maintain.

Here, the applied Hopfield network has 100 binary neurons, where possible states are +1 and -1. The number of units, one hundred, is chosen since the environment signal has also one hundred binary units, of which the possible states are +1 and -1 as well, like introduced in the section 3.2.1. The connection's configuration is the standard one, between each pair of neurons there is a connection and no self-link exists. The weights of the connections are set to zero initially and follow the Hebbian rule for update.

Specifically, for one arbitrary input signal $\{S_i\}_{i=1..100}$, the states of the Hopfield network will be initialized to the input. After this, the network will be updated synchronously, until the states do not change any more or a limit of number of update is attaint. After the retrieve of memorized pattern, the input will be learned into the memory if the learning condition is satisfied. In our simple case, we just use a logical flag to indicate whether the learning will be performed. If learning is required, the weight of the connection between the i-th node and the j-th node will be increased by

$$\Delta W_{ij} = \frac{S_i S_j}{N}$$

where $N = 100$ (if the value is negative then it is a decrease).

3.3.2 Surprise value

The associative memory catches the environment signal as input and outputs a memorized pattern. This pattern is the environment where baby believes he situates. Therefore, the difference between this pattern and the input signal could be considered as a measurement of surprise, a surprise that how the actual environment differs to the environment in memory. Smaller the surprise value is, more familiar the perceived environment is.

The idea of this surprise value could be sourced to arousal-base model (Hiolle, A et al. 2012). The exact formula for the calculation of surprise value does not change, which is

$$Surprise = \frac{\sum_{i=1}^{N} |X_i - S_i|}{N}$$

where Xi is the retrieved pattern and N equals to 100, the number of units in the network. The only difference between my application and arousal-based model is that he uses a sparsely network and I use a standard Hopfield network. The division by N is necessary to normalize the value into the interval between 0 and 1. This normalization will facilitate the calculations in following pieces of system. This normalization is also one mistake in arousal-based model. He did not show the normalization in his formula in his paper and I noticed him about this by email.

3.3.3 Categorization and self-organized map

The surprise value measures how familiar to the related memorized pattern the situation is, but this value does not cover the aspect how difficult to categorize the situation. Therefore we need another method to evaluate this difficulty.

It is still the idea of arousal-based model that helped me (Hiolle, A et al. 2012). The authors used a self-organized map to accomplish this task.

A self-organized map, or named Kohonen map thanks to his inventor, is a neural network using unsupervised learning to represent the input space of training samples in a lower dimensional point of view by preserving the topology of the space. As the inputs are the perceived environments (or in our situation they are generated simulating data), the learned self-organized map represents the topological structure of the inputs space, then it could serve to evaluate how one input is related to other inputs in the considered space.

However, self-organized map is not easy to apply here, as the inputs must be handled in order and not within batch. Hiolle et al. solved this question by changing the standard algorithm slightly to be incremental (Hiolle et al. 2012). Concisely, he breaks the main loop of the standard algorithm, which deals one input in one loop. For each gathered input signal, the instruments in the main loop will be performed only once. The next perform of these instruments will be in the next running loop of the whole simulating system and for the next input.

What they did is what I need exactly here, so I just used his idea without change. It means a self-organized map of 100 nodes is used. For each node, a weight is attached. This weight must have the same dimension as the input vector. In other words, the weight is a vector of 100 units, like the inputs signal.

Once the map gathers an input, it will select the Best Match Unit (BMU) from the one hundred nodes the map possesses. The method used by Hiolle et al. for this selection and the method by standard algorithm is not the same. The original one will select the node whose weight is closest to the input vector, in the sense of Euclidean distance. This calculation is performable since one node's weight represents one vector in the input space as well. The Hiolle's one select the node whose weight has the highest activity. For a given input, one node's weight's activity is defined as the dot product of the weight vector and the input vector, following the formula: $Y_i = \sum_{j=1}^{100} W_{ij}X_j$, where W_{ij} means the j-th element of the weight of the i-th node. In fact, this activity is a kind of similarity. More identical bits weight and input have, higher the activity is. If the weight and the input are exact the same, then the activity is the maximal, equal to 100; if the weight is the inverse of the input, then the activity is the minimal, equal to -100. Here, we follow Hiolle's idea in his paper.

After selecting out the Best Match Unit, we will update the weight of the one hundred nodes. The update formula is as below:

$$W_{ij}(t+1) = W_{ij}(t) + nbh(BMU, i)\alpha(t)(X_j(t) - W_{ij}(t))$$

The function $nbh(BMU, i, t)$ is the neighborhood function, which will multiply the increase of weight by a factor based on the distance between the BMU and the current weight vector. Normally, further the distance is, smaller the neighborhood function returns. In other words, the weight vectors closer to the BMU's weight vector will receive higher increase, and the weight vectors far away from the BMU's weight vector in the input space will receive just few portion of increase or even zero. Here the concrete neighborhood function is a piecewise one:

$$nbh(BMU, i) = \begin{cases} 1 & if\ |d(BMU, i)| \leq a \\ -\dfrac{1}{3} & if\ a < |d(BMU, i)| \leq 3a \\ 0 & if\ |d(BMU, i)| > 3a \end{cases}$$

The function $\alpha(t)$ is the learning rate which decreases as time passes. Concretely, the value of learning rate follows:

$$\alpha(t) = \frac{\alpha}{1 + t\kappa}$$

where α is the learning rate constant and κ is a decrease rate.

After the update, the state of self-organized map will be used to calculate related measurement. The next chance to use this map is when the system gathers the next input.

3.3.4 Categorization Adjustment

Like the Surprise value after the Hopfield network, we will calculate another measurement after the self-organized map. This measurement is called categorization adjustment. It is a complement of the surprise value and will measures 'how difficult to categorize the new input', as stated in the work of arousal-based model (Hiolle, A et al. 2012). His reason is that 'the variations of the weights are, though by a changing factor, proportional to the distance between the perceptual inputs and this internal variables correlates to the wanted difficulty'.

Thus, as the explication says, we could define the Categorization Adjustment by the formula below:

$$Cat_{adj}(t) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\left|W_{ij}(t) - W_{ij}(t-1)\right|}{NM}$$

where N is the number of nodes, equal to 100; M is the dimension of input vector and weight vector, equal to 100 as well. The normalization is like the one in surprise value and therefore works for facilitating the following calculations.

3.3.5 Exploration level

The Surprise value and Categorization Adjustment are two measurements about the environment input signal. However, we just need one evaluation about the input. So we need to combine the two calculated measurements. The most natural idea is to use the average and it is the applied method in my system.

$$Exploration = \frac{Surprise + Cat_{adj}}{2}$$

Thus, we average the surprise value and the categorization adjustment to have the final evaluation, which is named exploration level, as new environment is always tied to exploration. The value of this level is also in the interval from 0 to 1, since the two measurements are normalized. After calculation, exploration level will be passed into the next part of the system and used for decision-making.

We just simply average the two measurements here. However, the weight of two measurements in this average may not be equal. Then the exploration level becomes:

$$Exploration = \frac{\alpha * Surprise + \beta * Cat_{adj}}{\alpha + \beta}$$

In this case, choosing appropriate values for α and β could be a very interesting topic and worth to study in the future work.

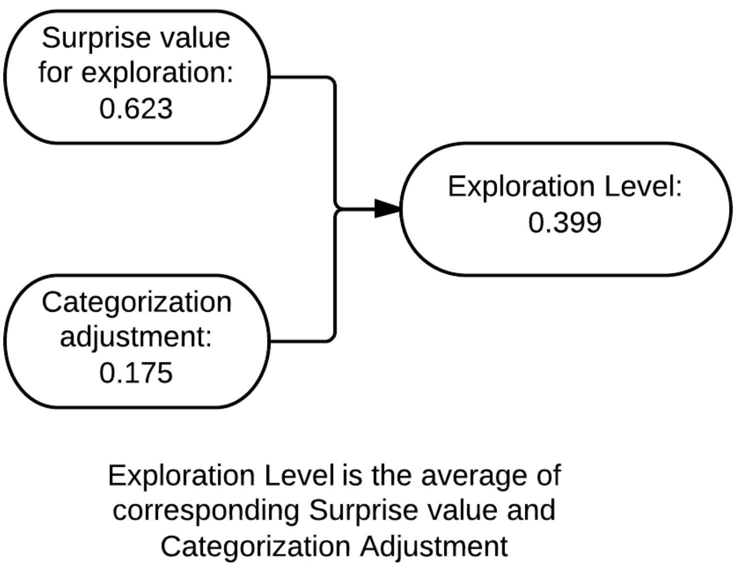


Figure 3.5 This figure illustrates how the exploration level is calculated with a numeric example.

3.3.6 Security level

The Security level is extremely similar to the Exploration level introduced above, however the Security level is for evaluating the face recognition input signal. We call it security because recognized face could let people feel secure.

Like the Exploration, Security is the average of the Surprise value for face recognition and the Categorization Adjustment for face recognition:

$$Security = \frac{Surprise^{face} + Cat_{adj}^{face}}{2}$$

Evidently, this security level is calculated for decision-making as well.

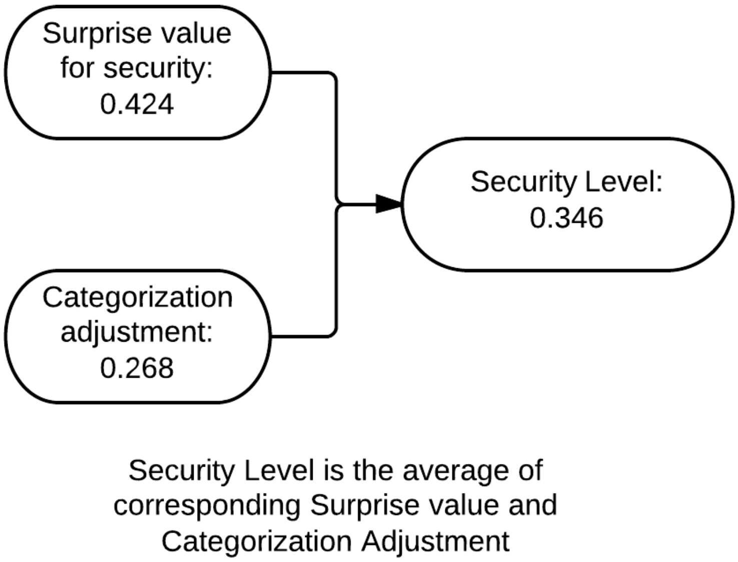


Figure 3.6 This figure shows how the security level is calculated from coresopnding surprise value and categorization adjustment. A concrete example is also provided.

3.4 Decision-making in the system

The decision-making part is the core of this built system.

This part will receive the outputs of the previous part, which are two pre-processed signals, exploration level and security level, and two raw inputs, touching detection and distance detection. The system will perform the decision-making with the help of these received signals.

The decision, which is the action of the simulated baby concretely, will affect the action of the simulated caregiver. Soon afterwards, the influence of both actions will be evaluated. The influence interacting with the environment will be incarnated in the next inputs; the influence about how appropriate the baby's action is will be treated as parameter to update the neural network in decision-making part.

3.4.1 Discretization

Specifically, the four received signals of the decision-making part, including exploration level, security level, touching detection and distance detection, will be discretized at first, in order to facilitate computations in neural network of this part. Moreover, the four discretized signals will be concatenated as one signal. And the neural network will make use of this one final signal as input.

The exploration level and the security level are both a number between 0 and 1. The interval from zero to one will be equally divided to 32 sub-intervals and the discretization is based on this division. These thirty-two sub-intervals are assigned a number, from 0 to 31. The final result after the discretization is the binary representation of the number of corresponding sub-interval. Thus the format of this result is a logical vector of 5 units. The first unit stands for the highest bit and the last unit stands for the lowest bit.

An example may let the understanding be easier. Supposed the exploration level is 0.763. This value falls into the sub-interval no. 24 (0.763*32=24.416, we choose the floored value).

The binary representation of 24 is 11000, then the final result of exploration level after discretization is a logical vector of values [true, true, false, false, false] (or [1,1,0,0,0] in Matlab).

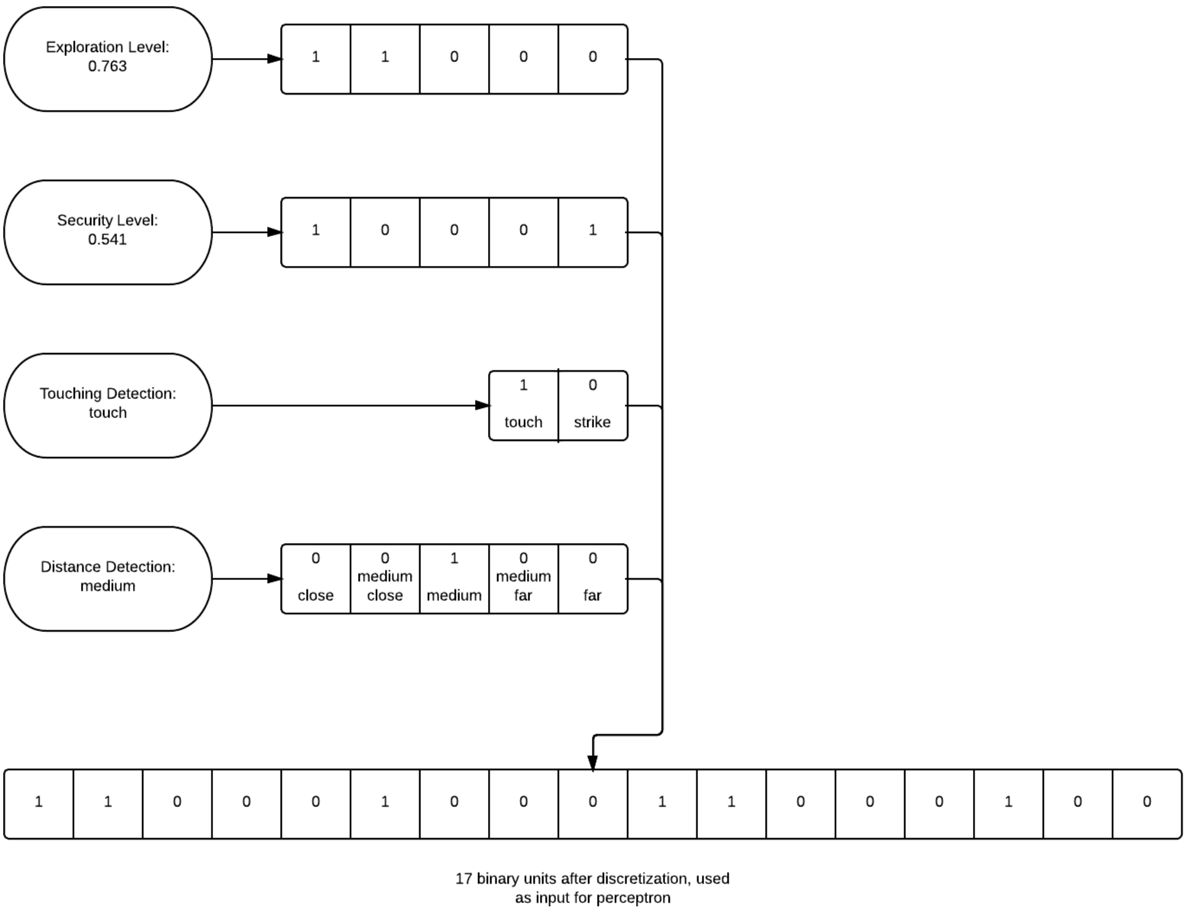


Figure 3.7 The example illustrated at the end of this subsection, showing how the discretization is done. All four signals are transformed to in total 17 logical/binary untis.

The original format of touching detection and distance detection is already discretized. The touching signal is a logical vector of 2 units, where the first one shows for 'touch' and the second shows for 'strike'. The distance signal is similar. It is a logical vector of 5 units, where these units shows for 'close', 'medium-close', 'medium', 'medium-far', 'far' respectively. The details are introduced in the section 3.2.

Overall, the four discretized signals will be concatenated into one signal. The order is exploration level, security level, touching detection and distance detection. The exploration level and security level have 5 logical units for each, the touching detection has 2 units and the distance detection has 5 units as well. Thus the final signal must have 17 logical units to contain all four signals.

One illustration may be helpful. We supposed the exploration level is 0.763, the security level is 0.541, there is a strike and the distance is medium. Then the four discretized signals are 11000, 10001, 01, and 00100. Consequently the final signal is 11000100010100100.

3.4.2 Neural Network: a Perceptron

The most important piece in the decision-making part is the neural network. A good choice of neural network must simulate as similar as possible the decision-making mechanism of a baby. This usually indicates a complex structure of network. However, the limitation of computation resources requires the structure could not be too complex to update. Thus a dilemma appears and we need a tradeoff.

The solution is to use a perceptron. The structure of a perceptron is simply enough to be updated very quickly. The drawback of a perceptron is that it may not simulate very similarly to a baby's natural decision-making, because a perceptron has just one layer. However, as this is a first try, I think one layer could show us the main characteristics of the mechanism. As regards the details, to cover them in future works is not bad. In addition, the simple structure and the linear computation of a perceptron could greatly decrease the difficulty in the mathematical and theoretical analysis of the simulating system.

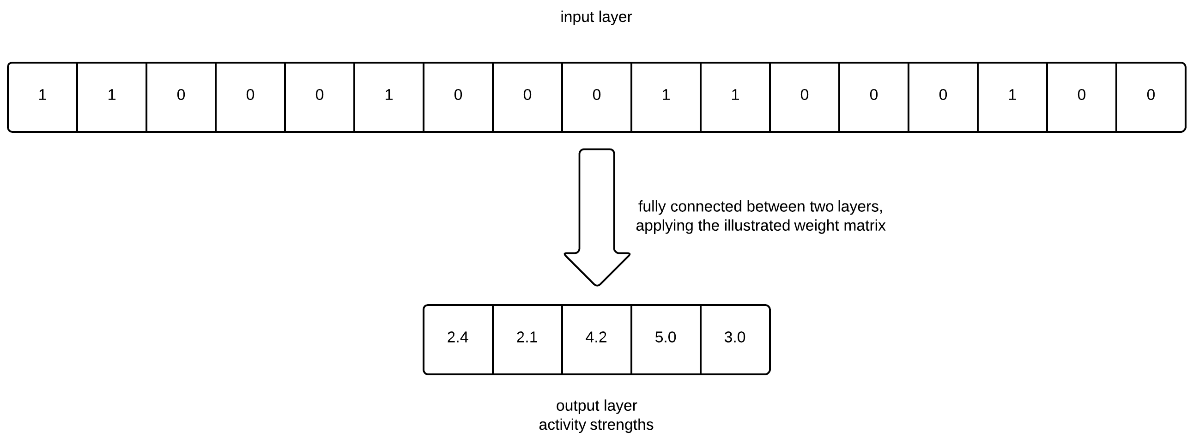


Figure 3.8 A calculation illustration of the perceptron. The applied weight matrix is the one supposed below. The input of perceptron is the output of discretization, having 17 units.

The applied perceptron has 17 input neurons and 5 output neurons. The seventeen input neurons correspond to the 17-bits signal after discretization. One neuron works for one bit. If the corresponding bit is true/1, then the neuron is initialized to 1; if the bit is false/0, then the neuron is initialized to 0. The five output neurons correspond to the five possible actions of our virtual baby, which are to turn, to shout, to move to caregiver, to move away from caregiver and to stay still. The output value of a neuron is the activity strength of the corresponding action. How these activity strengths could help decide the action will be introduced later.

Let us to understand the network by an illustration. We use the example value of the discretization section, 11000100010100100. This is the 17 bits of the seventeen input neurons. The weights of connection constitute a matrix of 5 lines and 17 columns. For one arbitrary element in this matrix, noted as W_{ij} where 'i' is between 1 and 5 and 'j' is between 1 and 17, this element represents the weight of connection from j-th input neuron to i-th output neuron. If we suppose the whole connection weight matrix like below:

The 17 neurons in the input layer

	0.6	0.3	0.2	0.6	0	0.1	0.5	0.6	0.9	0.3	0.7	0.5	0.4	0	0.6	0.7	0.6
	0.4	0.7	0.6	0.4	0.8	0.1	0.6	0.1	0.9	0.1	0.5	0.8	0.3	0.9	0	0.5	0.8
5 neurons in the output layer	0.7	0.3	0.7	0.7	0.9	0.9	0.5	0	0.7	0.7	0.1	0.7	0.2	0.7	0.9	0.4	0.2
	0.9	0.9	0.9	0.9	0.1	0.6	0.8	0.6	0.6	0.8	0.4	0.9	0.2	0.1	0.9	0.2	0.6
	0.4	0.9	0.8	0.6	0.2	0.3	0.7	0.5	0.5	0.4	0.1	0.9	0.6	0	0.1	0.3	0.5

Table 3-1 This table contains the weight matrix used in the example of calculation in perceptron. The connection weight for connection from i-th input neuron to j-output neuron could be found at the cell at j-th row and i-th column.

Then we could calculate the output by the formula

$$V_{output} = W_{connection} V_{input}.$$

Thus the output vector is [2.4, 2.1, 4.2, 5.0, 3.0]. It means the activity strengths for the five possible actions are these five calculated values.

3.4.3 Probabilistic decision using activity strengths

After the calculations in the perceptron, we now have the activity strength for each possible baby actions. These activity strengths could be considered as the potential that the corresponding action would be chosen. We should therefore make the final decision by using these strengths.

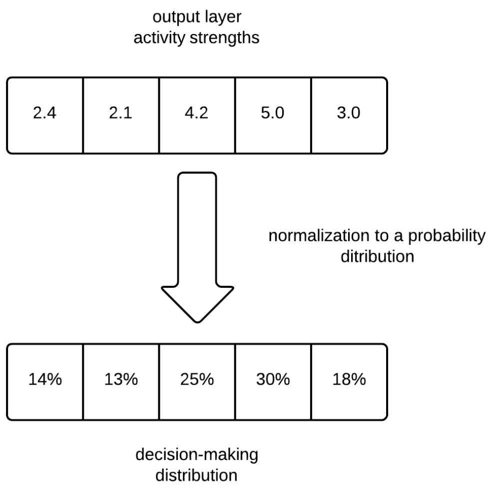


Figure 3.9 The probability distribution used in decision-making is generating by normalization of the output vector of perceptron.

There are two intuitive ideas. The first is deterministic that we just choose the action having the highest activity strength. The second is probabilistic that we normalize the five strengths to a probabilistic distribution and then make the final decision randomly.

The first idea is straightforward and easy to understand and to analyze. However, the perceptron is a little slightly simple in structure, and the probability in the second idea may bring the system a little more vitality. Hence the second idea is applied in the system. In fact, in my point of view, the probabilistic decision may be more biological plausible, since we could consider it as the human's hesitation between multiple choices.

So if the activity strengths are [2.4, 2.1, 4.2, 5.0, 3.0], the calculated values in the previous example. Then the normalized probabilistic distribution will be [0.14, 0.13, 0.25, 0.30, 0.18]. It means the simulated virtual baby has 14% chance to turn around, 13% to shout, 25% to move to caregiver, 30% to move away from caregiver and 18% to stay still. The final decision will come out from this normalized distribution.

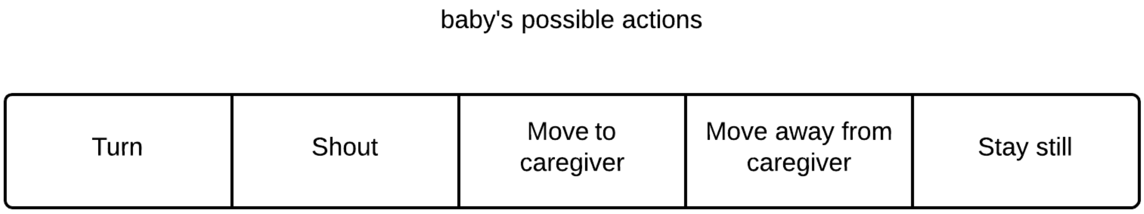


Figure 3.10 Baby's five possible actions.

In fact, the above distribution is just an example. The weights of connections in the perceptron are chosen randomly. That is why the five percentage values have not too big difference. However, if the simulating system works well, there will be just one, sometimes two, values are relatively high and the others are relatively low. Because in most times, the decision needed to make is obvious for a given environment. Even when the hesitation exists, the possible choices are not too much.

3.4.4 After having made decision

After the decision has been made, the third part of the system, which is also the smallest part of the system, will involve. This part simulates the caregiver's decision-making of response to virtual baby's action, but is just based on a very simple idea. The structure of this part is really uncomplicated, in contrast to the decision-making part of the virtual baby that we are currently introducing.

Thus in order to not confuse with what we are talking about now, how the third part of the system involves in computations will be shown in a separate section 3.5 later. What we need to know at this moment is that the caregiver's response will be given after the involvement, where the response is one of the 7 possible actions: to move to baby, to move away from baby, to touch baby, to strike baby, to stay still, to appear and to disappear.

caregiver's possible actions

Move to baby	Move away from baby	Touch	Strike	Stay still	Appear	Disappear

Figure 3.11 Caregiver's all seven possible responses.

3.4.5 Influence of baby's and caregiver's action

Once baby's action and caregiver's response are decided, we need to evaluate these actions' influence. This evaluation is extremely important, because it will change the perceived environment (even though the environments in our experiments are virtual, the change still exists) and the learning of the perceptron will make use of it as well.

As the evaluation will influence two different elements, we could divide this evaluation into two relatively separate aspects. The first is the change in environment and the second is the feedback to the perceptron.

3.4.5.1 Change in environment

The baby and caregiver's actions will mostly influence the distance, the touching and the face recognition.

Concisely, the 'moving to' and the 'moving away from' actions, both for the virtual baby and the virtual caregiver, are the two actions influencing the distance. If a 'moving to' action is made, then the distance will decrease one grade, e.g. from medium-far to medium; if a 'moving away from' action is made, then the distance will increase one grade, e.g. from close to medium-close. The influences of baby's action and of caregiver's action are separately calculated. This means if both the baby and the caregiver choose to move to other, the distance will decrease two grades. In contrast, if both choose to move away from other, then there is a two-grades increase. Moreover, if one chooses to move to and the other choose to move away from, the distance will not be changed. In addition, the distance could not be closer than 'close' and further than 'far'. So if we need to decrease one grade in case of 'close', the distance will still be 'close', and similarly to 'far'.

For the touching signal, mainly two caregiver's actions could influence it, 'to touch' and 'to strike'. How the signal will be influenced is very clear, as the names of the action show. If the caregiver chooses to touch the baby, then the next touching signal will be [true, false]/[1, 0]; if the caregiver chooses to strike the baby, then the next signal will be [false, true]/[0, 1]; if none of these two actions is made, then the signal will be [false, false]/[0, 0]. In other words, the touching signal is totally under control of caregiver's decision.

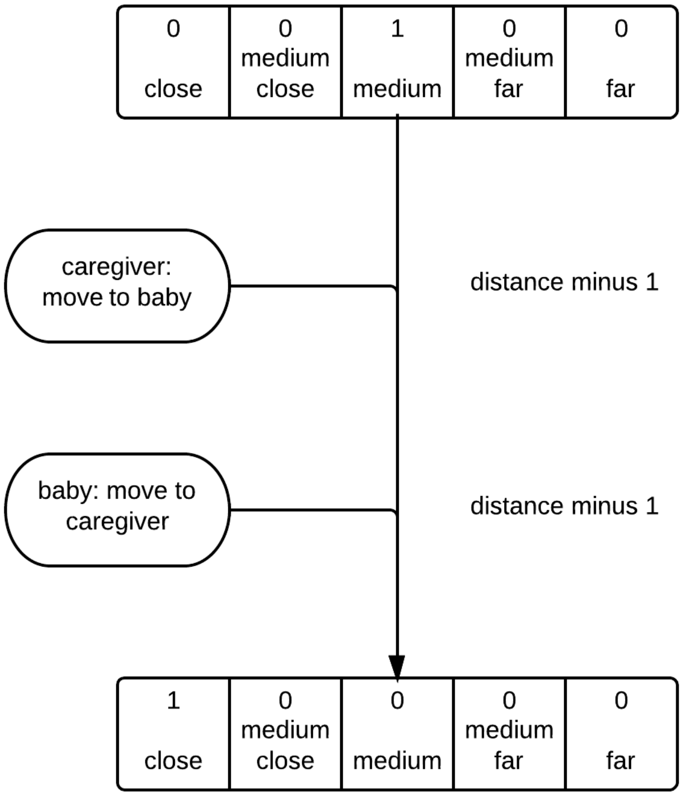


Figure 3.12 Illustration about how the actions influence the distance sdetection.

The face recognition signal has a similar situation as the touching signal. Only two caregiver's actions could influence it, 'to appear' and 'to disappear'. The caregiver in fact has two presence states, being present and being absent. The two actions are the bridges between these two states. If the caregiver is present, then the face recognition signal has a high probability to be the face of caregiver. This is based on the observation of everyday life that, in most time, a mother will stand in her baby's view if the mother is present around the baby. Because mothers tend to let their baby know they are aside. If the caregiver is absent, then the face recognition signal will be randomly generated, as these random signals stand for other people or noise signals. Besides the face recognition, the 'to appear' and 'to disappear' actions have also an indirect influence on the distance signal. The five distance grades are only applicable when the caregiver is present; if the caregiver is absent, then the distance must be [false, false, false, false, false], which means none of the five grades corresponds to the situation.

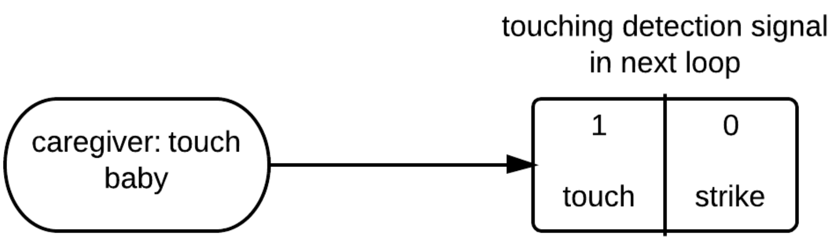


Figure 3.13 How the actions influence the touching detection.

3.4.5.2 Feedback to perceptron

The feedback to perceptron is a vector of 5 values. Each value corresponds to the influence of a baby's possible action.

Although the feedback is a vector, the calculation of this vector is based on the calculation of a number. This number defines how appropriate the virtual baby's decision was. Once this number is out, the feedback vector will be almost determined. Specifically, in the feedback vector, the value corresponding to the chosen action is three times of the number, while other values are exactly the number. It means the influence of the chosen action is three times to other rejected actions.

This higher weight of the chosen action is easy to understand. The confusion requiring explication is why other rejected actions have a non-null feedback. The reason is that the neuron of a rejected action could be considered fired although it has not been chosen, because the rejection is after operations of the perceptron. Thus according to Hebb's rule, 'Cells that fire together, wire together' (Doidge, N. 2007), it is reasonable to give a non-null feedback to the rejected neurons.

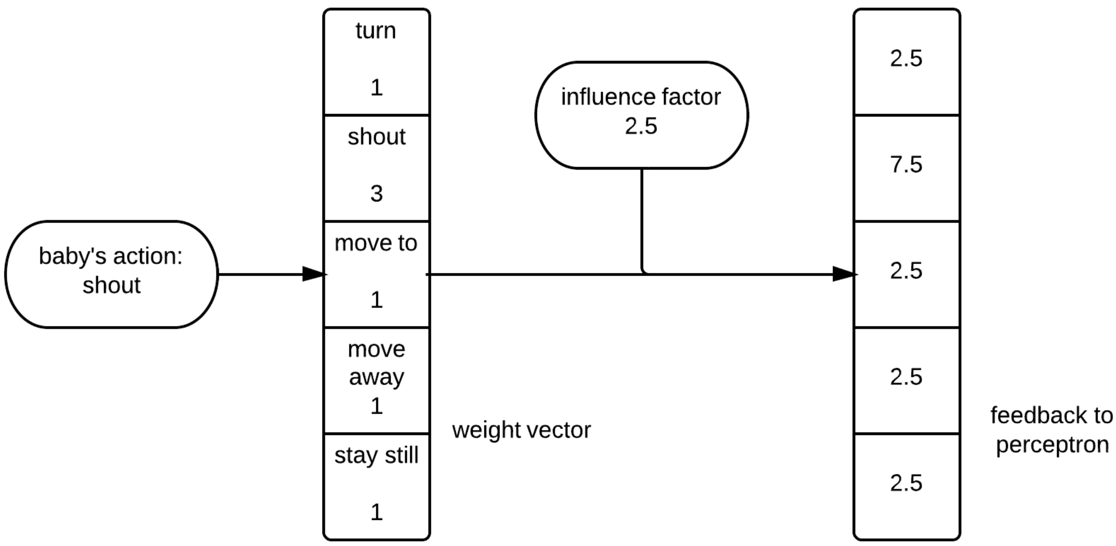


Figure 3.14 This figure shows how the feedback vector is calculated given baby's action and influence factor. The influence factor could be found in the following table.

So, if the number is 2.5 and the chosen action is the second action, 'to shout', then the feedback vector must be [2.5; 7.5; 2.5; 2.5; 2.5].

The calculation of the special number is the key step. This key step only depends on baby's action and caregiver's response. The value of this number could be retrieved from the following table:

	Move to	Move away	Touch	Strike	Stay	Appear	Disappear
Turn	1	-1	2	-2	0	2	-2
Shout	-2	2	-3	3	0	-3	3
Move to	3	-3	3	-3	0	4	-4
Move away	-3	3	-2	2	0	-4	4
Stay	1	-1	2	-2	0	2	-2

Table 3-2 This table is used to calculate the influence factor depending on baby's action and caregiver's response.

One important notice, the sign of values in the table signifies that caregiver's response will promote or restrain the paired baby's action. It never means the response is positive or negative.

3.4.6 Learning stage of perceptron

Once the feedback is evaluated, we need to use this feedback in the learning stage of the perceptron. Concisely, as said in the previous subsection, the feedback will involve in the learning formula.

The formula is a slightly modified version of the delta rule:

$$\Delta W_{ij} = \alpha * feedback_i * distribution_i * input_j$$

The subscript 'i' is the index of the output neuron in the output layer of the perceptron and the subscript 'j' stands for the index of the input neuron in the input layer. The valid range of first subscript is from 1 to 5, while the range of second subscript is from 1 to 17.

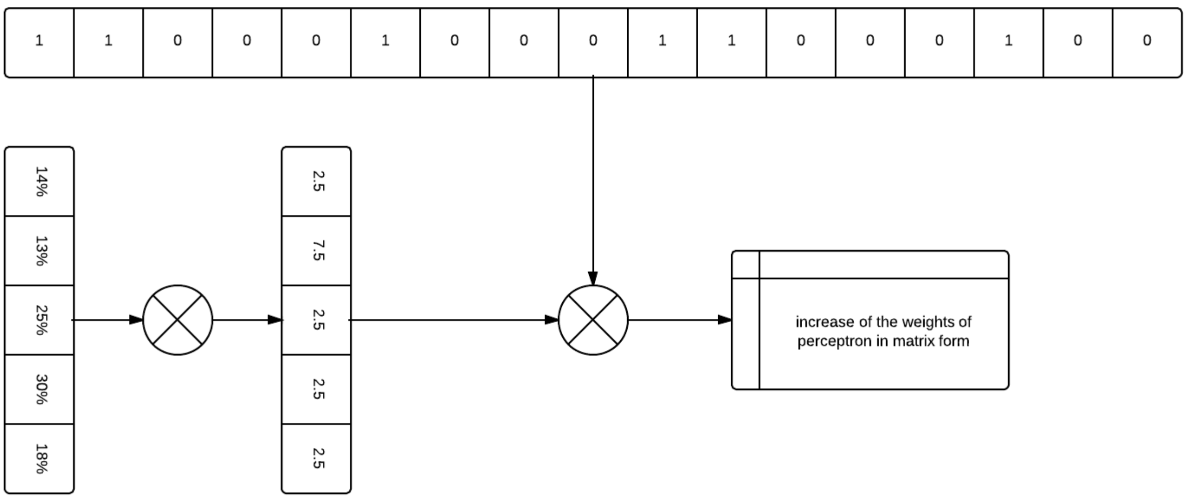


Figure 3.15 How the feedback vector, distribution vector and input vector contribute in learning phase of the perceptron. The result matrix is the matrix in the following table.

Variable '$feedback$' in the formula is the feedback vector calculated in previous section. Variable '$distribution$' is the probability distribution obtained by normalizing the output values of the perceptron. Variable '$input$' is the input vector of the perceptron. Variable 'α' is the learning rate, fixed at 0.5.

If we continue to uses the examples calculated in previous subsections, where '$feedback$' is [2.5; 7.5; 2.5; 2.5; 2.5], '$distribution$' is [0.14, 0.13, 0.25, 0.30, 0.18] and '$input$' is 11000100010100100, then the increase of the weights will be the two-dimensional vector below:

The 17 neurons in the input layer

5 neurons in the output layer	0.175	0.175	0	0	0	0.175	0	0	0	0.175	0	0.175	0	0	0.175	0	0
	0.4875	0.4875	0	0	0	0.4875	0	0	0	0.4875	0	0.4875	0	0	0.4875	0	0
	0.3125	0.3125	0	0	0	0.3125	0	0	0	0.3125	0	0.3125	0	0	0.3125	0	0
	0.375	0.375	0	0	0	0.375	0	0	0	0.375	0	0.375	0	0	0.375	0	0
	0.225	0.225	0	0	0	0.225	0	0	0	0.225	0	0.225	0	0	0.225	0	0

Table 3-3 This table contains the result matrix of the example in this subsection. The value in this table corresponds to the increase of related connection weight.

3.5 Simulation of caregiver

The simulation of how the caregiver makes his response to baby's action is the last part of the whole system. The main requirement of this simulation is that this part must be possible to simulate some different caregivers, like a kind caregiver, an impatient caregiver, an indifferent caregiver or even a stranger. Because we need at least two different caregivers to show that the simulated virtual baby will have different behavior when the baby was under cover of different caregivers. In addition, in the 'strange situation protocol', a stranger is required.

We did not choose to use a neural network in this part. Multiple reasons exist:

i. An adult's behavior is already developed and hardly to be altered, so there is no need to perform a new learning by using the neural network.

ii. A neural network will usually consume much more computational resources than a simple structure. We have already used five neural networks in the system; thence one more network may be too burdened.

iii. The structure of the whole system is such simple. Therefore a simple structure in a local part could make the system more reliable.

In the end, I decided to seek help from probabilistic distribution once more.

For one chosen type of caregiver, and for one decided action of the virtual baby, we have four fixed probabilistic distributions. Each distribution has 7 probabilities, corresponding to the seven possible actions of the caregiver, 'to move to baby', 'to move away from baby', 'to touch baby', 'to strike baby', 'to stay still', 'to appear' and 'to disappear'. And each distribution is used in a different distance of 4 categories, 'close', 'medium', 'far', 'disappeared'. The 'medium' category includes the three distance grades beginning with medium: 'medium-close', 'medium' and 'medium-far'.

Here we use different distribution for different distance. The reason is that the distance between the baby and the caregiver has a extremely important role in the evaluation of the feedback to perceptron, and we must therefore treat it very carefully. In addition, as we want the system to be more real, we suppose that the touching and strike actions could only be performed when the distance is in the 'close' category, and this also requires the categorization of distance.

There are five possible actions for baby. So for one type of caregiver, there are in total 20 probabilistic distributions (5×4 = 20). These five distributions constitute a probabilistic table, of five rows and seven columns, where each row is a distribution. The table is an illustration:

		Move to	Move away	Touch	Strike	Stay still	Appear	Disappear
Baby's action: Turn	close	0.29	0.00	0.31	0.02	0.27	0.04	0.07
	medium	0.18	0.21	0.08	0.10	0.14	0.12	0.17
	far	0.10	0.06	0.34	0.12	0.08	0.13	0.17
	disappear	0.50	0.01	0.05	0.11	0.26	0.03	0.04
Baby's action: Shout	close	0.03	0.26	0.13	0.22	0.13	0.15	0.08
	medium	0.12	0.03	0.18	0.20	0.06	0.24	0.17
	far	0.11	0.19	0.18	0.12	0.19	0.06	0.15
	disappear	0.19	0.26	0.19	0.00	0.07	0.17	0.12
Baby's action: Move to caregiver	close	0.07	0.20	0.09	0.09	0.16	0.20	0.19
	medium	0.20	0.18	0.16	0.18	0.15	0.08	0.05
	far	0.22	0.15	0.15	0.14	0.11	0.11	0.12
	disappear	0.24	0.13	0.06	0.12	0.14	0.10	0.21
Baby's action: Move away from caregiver	close	0.16	0.20	0.20	0.08	0.18	0.09	0.09
	medium	0.15	0.16	0.02	0.25	0.18	0.23	0.01
	far	0.24	0.18	0.06	0.16	0.07	0.20	0.09
	disappear	0.14	0.10	0.05	0.25	0.16	0.22	0.08
Baby's action: Stay still	close	0.10	0.17	0.13	0.18	0.03	0.29	0.10
	medium	0.15	0.17	0.18	0.06	0.07	0.31	0.06
	far	0.20	0.08	0.10	0.05	0.16	0.22	0.19
	disappear	0.05	0.23	0.09	0.17	0.16	0.11	0.19

Table 3-4 This table is an example for caregiver's action template. For each baby's action and each distance category, there is a probability distribution (a row in this table) for deciding caregiver's response.

The bold border divides the whole table into five sub-tables, where each corresponds to one action of the virtual baby. In each sub-table, there are four distributions, corresponding to the four distance categories. Anyone of the twenty row sums to 1, confirming that it is a valid probabilistic distribution.

If the baby chose to stay still, which is the last possible action, and the distance is medium-close, which is the second distance category ('medium'), then the retrieved distribution from table is [0.15, 0.17, 0.18, 0.06, 0.07, 0.31, 0.06]. The caregiver's response will be chosen randomly by this distribution.

One such probabilistic table represents one caregiver's response mechanism. Different tables correspond to different caregivers. So we could very easily create many different response mechanisms. The only thing in the creation is to generate the table by following the characteristics of a given mechanism. This is the one big advantage of using the probabilistic distributions instead of other structure.

4 Experimental study

The designed system is built up to simulate the baby's action mechanism with regard to the caregiver's responses and perceived environment. With using this system, we could test and verify that different kind of caregivers could make their baby different in emotional presentation and behavior to others. In addition, an essay to categorize these differences by applying attachment theory can be considered. This categorization could evaluate the system in guidance of theoretical supports.

4.1 Various caregivers

The caregiver's responses to his/her baby's actions are very important to baby's emotional development in early childhood. How the responses influence baby's action mechanism is implemented by the learning phase of the Perceptron (decision-making neural network of the built system). In order to generate different baby's action mechanism, we need different types of caregivers. This difference could be achieved by using different caregiver's action template in the system.

In fact, three different action templates are generated, respectively for simulating a good enough caregiver, a not good enough caregiver and a stranger.

The action template for the good enough caregiver will be generated by respecting the following principle: the caregiver will step closer to the baby if the baby does not resist and if they are not close enough; the caregiver will never strike the baby as this action is really undesirable; the caregiver will comfort the baby once the baby feel unfamiliar; the caregiver will not disturb baby's exploration. The concise caregiver's action template is shown below. As introduced in the section 3.5, each row is a probability distribution and there are twenty distributions (five possible baby's actions and four possible distance categories).

		Move to	Move away	Touch	Strike	Stay still	Appear	Disappear
Baby's action: Turn	close	0.00	0.10	0.10	0.00	0.80	0.00	0.00
	medium	0.20	0.10	0.00	0.00	0.70	0.00	0.00
	far	0.25	0.00	0.00	0.00	0.70	0.00	0.05
	disappear	0.00	0.00	0.00	0.00	0.25	0.75	0.00
Baby's action: Shout	close	0.00	0.05	0.90	0.00	0.05	0.00	0.00
	medium	0.90	0.05	0.00	0.00	0.05	0.00	0.00
	far	0.90	0.00	0.00	0.00	0.05	0.00	0.05
	disappear	0.00	0.00	0.00	0.00	0.10	0.90	0.00
Baby's action: Move to	close	0.00	0.10	0.30	0.00	0.60	0.00	0.00
	medium	0.60	0.10	0.00	0.00	0.30	0.00	0.00

caregiver	far	0.60	0.00	0.00	0.00	0.30	0.00	0.10
	disappear	0.00	0.00	0.00	0.00	0.25	0.75	0.00
Baby's action: Move away from caregiver	close	0.30	0.10	0.00	0.00	0.60	0.00	0.00
	medium	0.60	0.10	0.00	0.00	0.30	0.00	0.00
	far	0.80	0.00	0.00	0.00	0.10	0.00	0.10
	disappear	0.00	0.00	0.00	0.00	0.25	0.75	0.00
Baby's action: Stay still	close	0.00	0.10	0.10	0.00	0.80	0.00	0.00
	medium	0.20	0.10	0.00	0.00	0.70	0.00	0.00
	far	0.25	0.00	0.00	0.00	0.70	0.00	0.05
	disappear	0.00	0.00	0.00	0.00	0.25	0.75	0.00

Table 4-1 This is the action template for the good enough caregiver.

The action template for the not good enough caregiver will follow the principle that the caregiver does not pay enough attention to his/her baby. In most of time, the caregiver will just do his/her own work (i.e. to stay still for our virtual caregiver); the caregiver has no tendency to step closer or responds to baby's demand frequently. The concise built action template is the table below. The fifth column, which corresponds to action 'staying still' for caregiver, has the highest probability of all columns.

		Move to	Move away	Touch	Strike	Stay still	Appear	Disappear
Baby's action: Turn	close	0.0	0.2	0.0	0.0	0.8	0.0	0.0
	medium	0.2	0.2	0.0	0.0	0.6	0.0	0.0
	far	0.2	0.0	0.0	0.0	0.6	0.0	0.2
	disappear	0.0	0.0	0.0	0.0	0.8	0.2	0.0
Baby's action: Shout	close	0.0	0.0	0.0	1.0	0.0	0.0	0.0
	medium	1.0	0.0	0.0	0.0	0.0	0.0	0.0
	far	1.0	0.0	0.0	0.0	0.0	0.0	0.0
	disappear	0.0	0.0	0.0	0.0	0.0	1.0	0.0
Baby's action: Move to caregiver	close	0.0	0.2	0.0	0.0	0.8	0.0	0.0
	medium	0.2	0.2	0.0	0.0	0.6	0.0	0.0
	far	0.2	0.0	0.0	0.0	0.6	0.0	0.2

	disappear	0.0	0.0	0.0	0.0	0.8	0.2	0.0
Baby's action: Move away from caregiver	close	0.0	0.2	0.0	0.0	0.8	0.0	0.0
	medium	0.2	0.2	0.0	0.0	0.6	0.0	0.0
	far	0.2	0.0	0.0	0.0	0.6	0.0	0.2
	disappear	0.0	0.0	0.0	0.0	0.8	0.2	0.0
Baby's action: Stay still	close	0.0	0.2	0.0	0.0	0.8	0.0	0.0
	medium	0.2	0.2	0.0	0.0	0.6	0.0	0.0
	far	0.2	0.0	0.0	0.0	0.6	0.0	0.2
	disappear	0.0	0.0	0.0	0.0	0.8	0.2	0.0

Table 4-2 The action template for the NOT good enough mother.

The action template for a stranger is much more simple. We suppose the stranger will just stay still to see what the baby does and there will be no response to baby's demand. This is a very extreme assumption. But it is enough to give an instructive result. Consequently, the table of the action template is very simple as well. Only the values in the fifth column (for action 'staying still') are 1, the others are all 0.

		Move to	Move away	Touch	Strike	Stay still	Appear	Disappear
Baby's action: Turn	close	0	0	0	0	1	0	0
	medium	0	0	0	0	1	0	0
	far	0	0	0	0	1	0	0
	disappear	0	0	0	0	1	0	0
Baby's action: Shout	close	0	0	0	0	1	0	0
	medium	0	0	0	0	1	0	0
	far	0	0	0	0	1	0	0
	disappear	0	0	0	0	1	0	0
Baby's action: Move to caregiver	close	0	0	0	0	1	0	0
	medium	0	0	0	0	1	0	0
	far	0	0	0	0	1	0	0
	disappear	0	0	0	0	1	0	0
Baby's	close	0	0	0	0	1	0	0

action: Move away from caregiver	medium	0	0	0	0	1	0	0
	far	0	0	0	0	1	0	0
	disappear	0	0	0	0	1	0	0
Baby's action: Stay still	close	0	0	0	0	1	0	0
	medium	0	0	0	0	1	0	0
	far	0	0	0	0	1	0	0
	disappear	0	0	0	0	1	0	0

Table 4-3 The action template for a stranger.

4.2 Simplified Strange Situation Procedure

For the purpose of distinguishing different action mechanisms for multiple babies, and furthermore to categorize the attachment types of the babies, the 'Strange Situation Procedure', introduced in the section 2.2, is the best tool to accomplish this task.

However, there is one constraint making us unable to use the complete version of the procedure: the face recognition network used here could only handle one signal. It means this simulating system could only recognize one present human, no matter a caregiver or a stranger. In other words, the episodes that needs the presence of both primary caregiver and stranger are hard to be implemented.

The substitutive solution is to use a simplified version of the 'Strange Situation Procedure'. In this simplified version, we have five episodes instead of eight:

- Episode 1: caregiver and baby
- Episode 2: baby alone (caregiver leaves)
- Episode 3: caregiver and baby (caregiver returns)
- Episode 4: stranger and baby (caregiver leaves and stranger comes in)
- Episode 5: caregiver and baby (caregiver returns and stranger leaves)

The episodes 1,2 and 3 are arranged together to observe baby's actions in case of leaving of caregiver and return of caregiver. The episodes 3,4 and 5 are planned together in order to observe baby's action when the presence of a stranger replaces the presence of the caregiver and when the caregiver returns.

According to attachment theory, an emotionally well-developed infant, who usually possesses a secure attachment, will shows motivated to explore the environment, happy to see the caregiver and not too nervous when a stranger is present. In situation of our system and experiments, such a baby will be more likely to choose the action 'to turn' or 'to move to caregiver'.

In contrast, a not well developed infant, who normally labeled one of the insecure attachment types, will show less interest in exploration, be ambivalent or avoidant to the caregiver and feel unfamiliar to the presence of a stranger. In situation of our system and experiment, such a baby will have higher probability to perform actions like 'to bark' or 'to move away from caregiver', than other possible actions.

4.3 Procedure of the experiment

Since three different caregiver's action templates are generated and a simplified version of the 'Strange Situation Procedure' is constructed to adapt the built system, we could finally define the workflow of the experiment.

Generally, the whole experiment has two main stages. The first one is the learning stage, simulating the early childhood of an infant. The virtual baby in our experiment will be initialized in the beginning of this stage, and then to learn how to react with caregiver and environment using the built system. Concisely, the system will be executed for 1000 times. The five neural networks, which consist of two Hopfield network, two self-organized map and one perceptron, will be trained in this stage. We will use two copies of the simulating system, one with the action template of a good enough caregiver, while the other with the action template of a not good enough caregiver in contrast.

The second stage is the testing stage. The trained virtual babies will be tested by the simplified version of the 'Strange Situation Procedure'. Concretely, in each episode of the five, the simulating system will be executed thirty times. Consequently, the whole simplified version needs 150 executions of the system. These executions are not like the ones in the previous learning stage. They use the action template for acting a stranger instead of the template of a good enough caregiver or a not good enough caregiver. Moreover, the five neural networks in the simulating system will not perform their learning phase. In other words, these neural networks will be operated in 'testing' mode instead of the 'learning' mode in the previous stage.

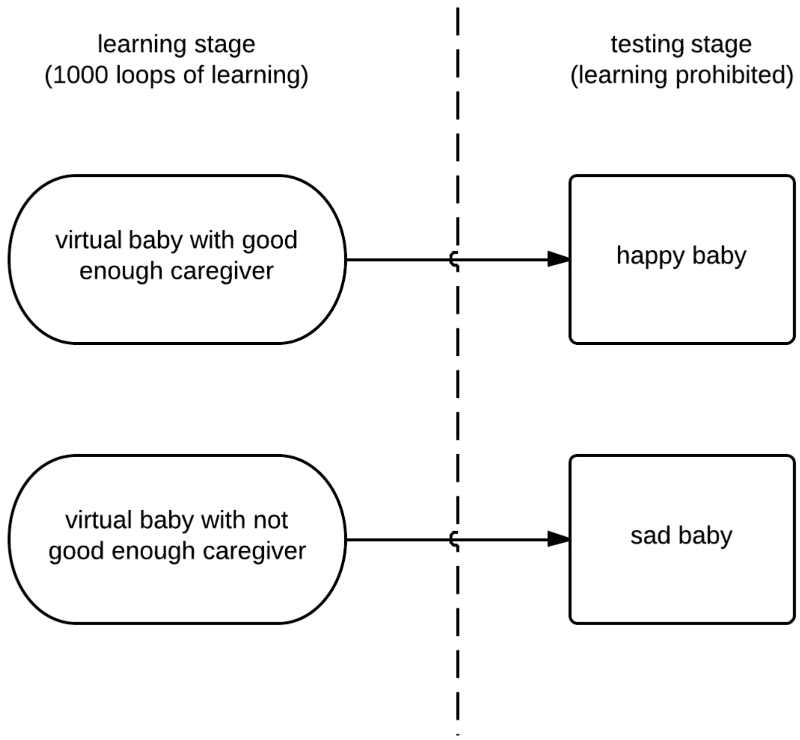


Figure 4.1 A simplified structure of the designed experiment.

4.4 Results of the experiment

In the second stage of the experiment, the actions of the two virtual babies, who are actually two copies of the simulating system with different caregiver action templates, will be recorded. These actions will be compared to verify whether these two virtual babies have different action mechanism after interacting with different virtual caregiver. After the comparison, we will evaluate whether these actions could distinguish attachment types.

The procedure of the experiment will be executed 100 times. The following analyses are based on the recorded data.

4.4.1 Babies acts differently

We focus on the dominant action and occurrence probability of each action of a tested virtual baby during the second stage. The dominant action in one execution is the action with highest occurrence. The occurrence probability of one action in one execution is the value of number of occurrence over number of total actions (100). These two measurements are calculated for both kinds of virtual baby. The results are figured below:

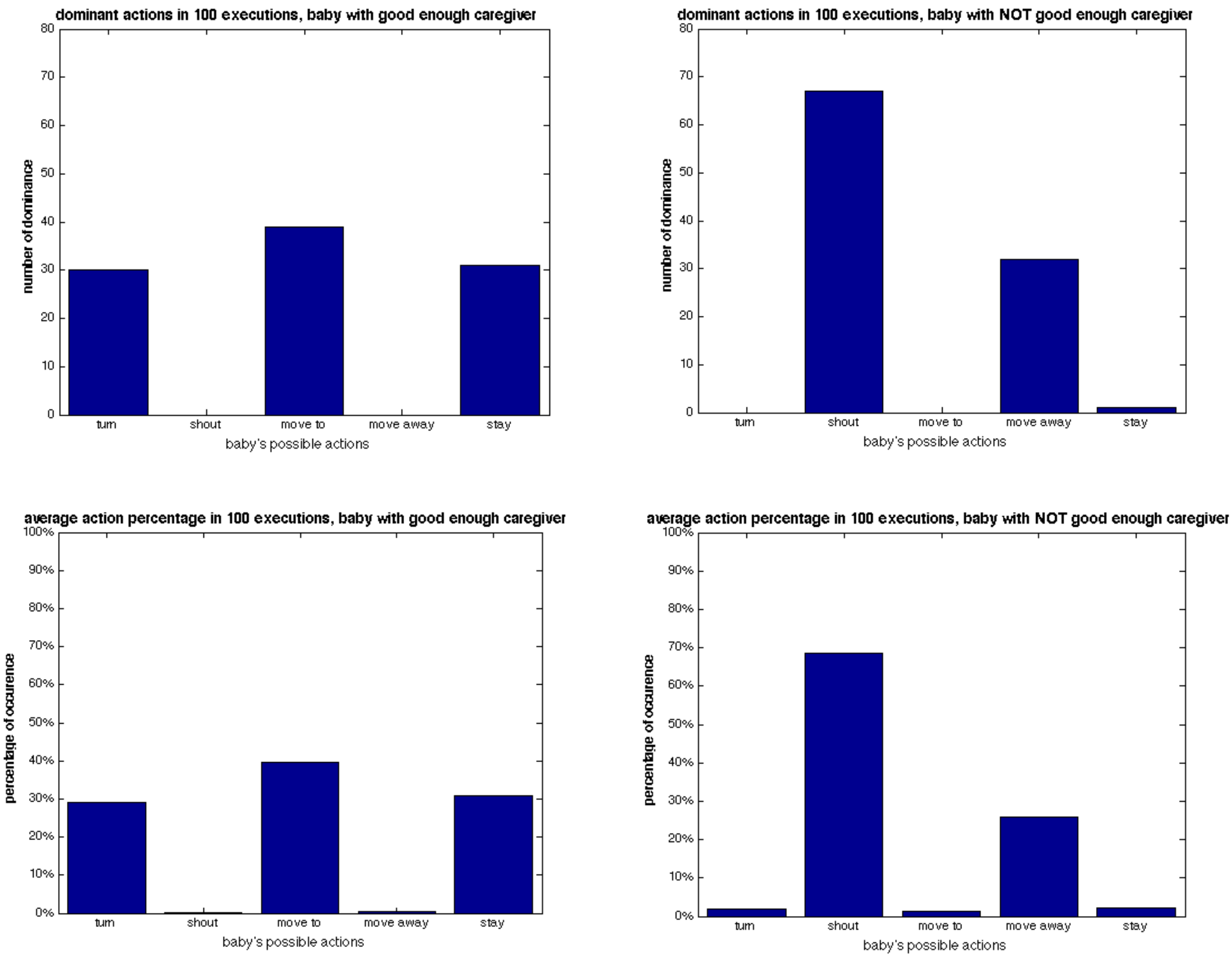


Figure 4.2 Statistics of dominant actions and probability of occurrence. We could observe the happy baby (left column) and the sad baby (right column) have totally different behavior. The clearer figures are provided in Appendix.

In the four figures above, the two in the first row are the statistics of the dominant actions; the two in the lower row are the calculated percentage of occurrence for each action. The left column is related to the virtual baby with action template for good enough caregiver; in contrast, the right column is the results about the virtual baby with action template for NOT good enough caregiver.

For ease of introduction, we will call the baby with good enough caregiver’s action template as ‘happy baby’; relatively, the baby with not good enough caregiver’s action template will be called ‘sad baby’.

By means of comparing the listed two measurements, we confirm that our two virtual babies have really different behavior.

Firstly, the top left figure shows the happy baby likes to turn around (meaning to explore), to move to caregiver (accepting his/her caregiver) or to stay still. In contrast, the sad baby always chooses to shout (feeling uncomfortable) or to move away from caregiver (resisting or avoiding contact to his/her caregiver). This could be derived from the top right figure.

The happy baby seems livelier and may have a secure attachment, while the sad baby looks like more timid and may have an insecure attachment. This remarkable contrast is an effective evidence for the confirmation.

The occurrence probability of each action, obtained from averaging statistics of the actions, also confirms this consequence. In fact, the data in the two lower figures have obviously similar appearance to the data in the higher figures. Moreover, the data of the lower figures are not like the numbers of occurrence in the higher figures, they consider the contribution from all possible actions, since they are statistic result. Thus the non-dominant actions are calculated as well.

4.4.2 Unable to determine attachment type

From previous analysis, we now ensure that different kind of caregivers will heavily influences the virtual baby they take care of. Moreover, the happy baby seems to possess a secure attachment and the sad baby may possess one of the three insecure attachment types.

However, from what we observed, we could not determine accurate attachment types of the two virtual babies.

The reason is that both two babies can hardly react to the event of caregiver’s leaving, caregiver’s return or stranger’s presence. In other words, when we expect the baby acts differently between two consecutive episodes in the simplified version of ‘Strange Situation Procedure’, the baby does not change his/her behavior.

This conclusion could be deduced from results of one arbitrary execution in second stage (stage of simplified ‘strange situation procedure’).

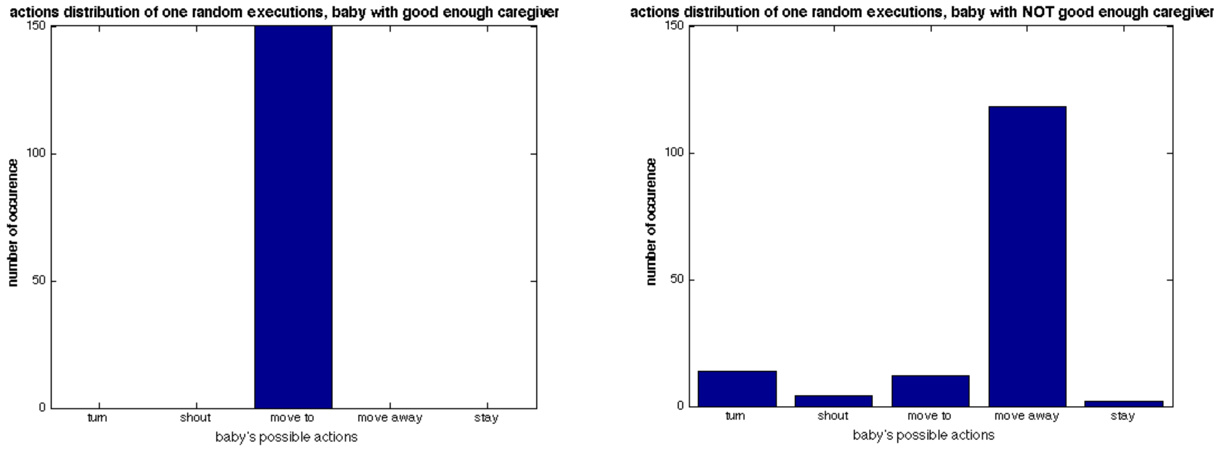


Figure 4.3 Behavior of two babies in second stage (testing stage). The unsensitivity to change of external siganls is already observable for both babies. The clearer figures are provided in Appendix.

These two pictures show the behavior of two virtual babies in the second stage. Obviously, no matter the happy baby or the sad baby, only one action is performed in most of time. For the happy baby (left picture), the situation is extreme. He/she will always choose to move to the caregiver, whatever the situation is.

These abnormal behaviors are not what we expect during design of experiment and tell us the determination of attachment type is not feasible in such situation.

More importantly, these odd observations show that our designed system must have drawbacks, which need to be found and corrected.

4.5 Analysis of the simulating system

Reasoning about determining attachment type in previous subsection indicates that our built system needs to be reviewed, because a learned virtual baby likes to choose a same action in the testing stage. In other words, this is a problem of sensitivity, the sensitivity of this simulating system regarding to external signals.

Intuitively, the first aspect to consider is the parameter configuration. This built system has a relatively sophisticated structure and a big number of parameters. Most of the default values of these parameters are derived from the original idea of arousal-based model in his work (Hiolle, A et al. 2012). However, as the structure is changed, these values may be not valid any more. Moreover, the new parameters need to be checked as well.

A deep analysis on parameter configuration is therefore reasonable. In purpose of accomplishing this analysis, a list of comparisons are designed and performed.

4.5.1 Comparison on learning rule of the perceptron

The perceptron is the decision-making neural network, and consequently one of the most precious pieces of the system. In this neural network, the learning phase decides how the virtual baby will 'grow up'. Thus, trying to improve the learning phase of perceptron by modify the learning rule is a natural idea.

The original learning rule is a modified version of delta rule:

$$\Delta W_{ij} = \alpha * feedback_i * distribution_i * input_j$$

Detailed explication of this formula is located at section 3.4.6.

Using $'feedback'$ and $'input'$ is natural and reasonable, since one is feedback and the other is input. However, instead of using variable $'distribution'$, we may have an alternative choice. Specifically, this $'distribution'$ variable plays the role of output vector in a standard delta rule. So the alternative choice is that we use the output results of the perceptron, in place of variable $'distribution'$.

Consequently, the learning rule will be modified to:

$$\Delta W_{ij} = \alpha * feedback_i * output_i * input_j$$

4.5.1.1 Result of comparison

We repeat the experiment with both two learning rules. What we observe this time is the two-dimensional weight vector of the perceptron after learning.

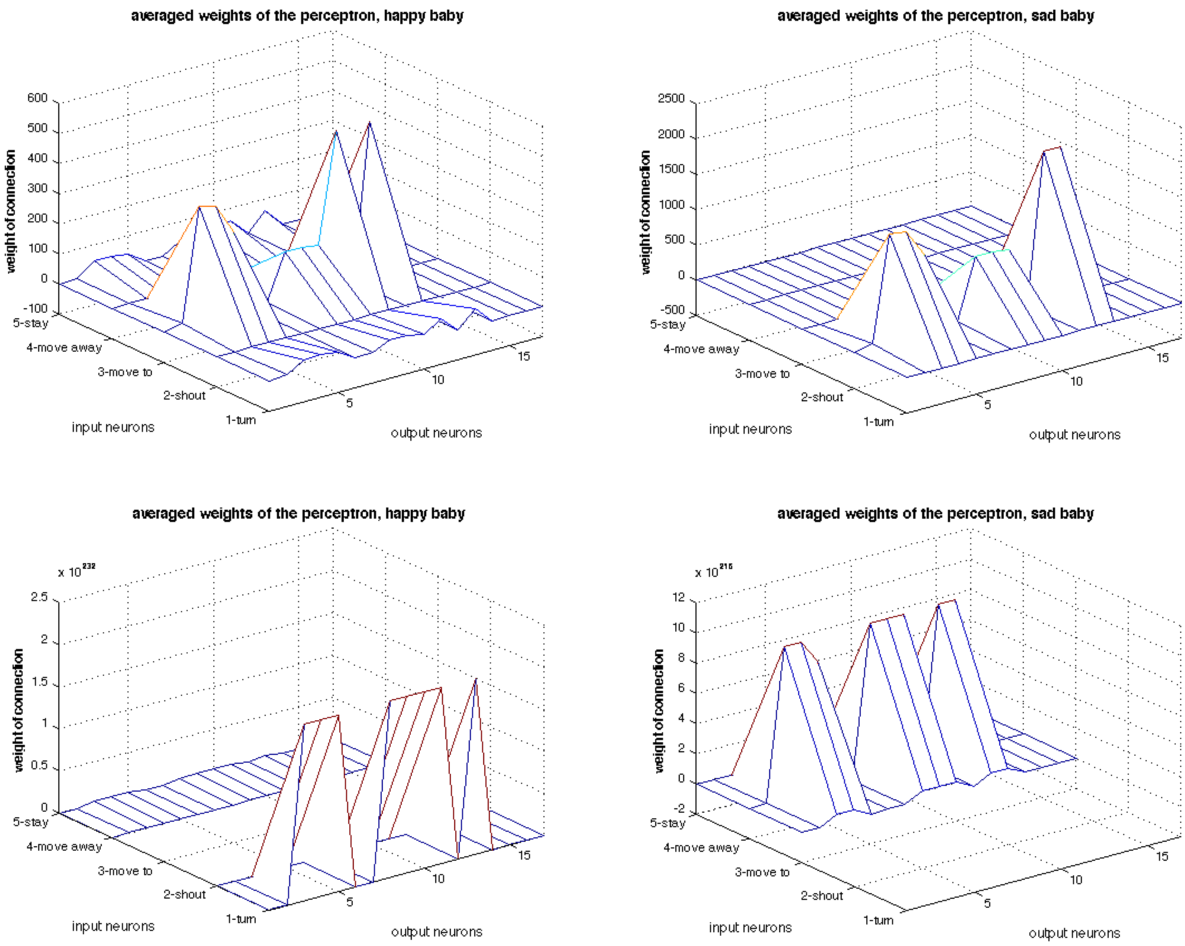


Figure 4.4 The weight matrix for two babies (left and right) in two conditions (upper and lower). The order of magnitude for original condition (upper) is much more reasonable. We must also notice the missing parts in modified condition (lower).

The first row shows the weight vectors in standard system for happy baby and sad baby, while second row shows the weight vectors in modified system for two babies.

There are two obvious differences. The first one is the magnitude of weights, indicated at the z-axis. For the standard system, magnitude of the highest weight is at an order of 3, in case of both two babies. In contrast, for the modified system, magnitude of the highest weight is at an extremely high order.

In addition, the figures of the lower row have some missing part. This is also the second difference. By inspecting the values of these missing parts in Matlab, we could find they are 'Inf'. In fact, when we use the output vector in learning rule, the increase of the output vector will then be proportional to itself. Consequently this output vector will increase exponentially and soon exceed the upper bound of data type 'double'.

Easily, we could conclude the original learning rule is more appropriate, although the modified version is more similar to the standard delta rule.

4.5.2 Comparison on calculation of feedback to perceptron

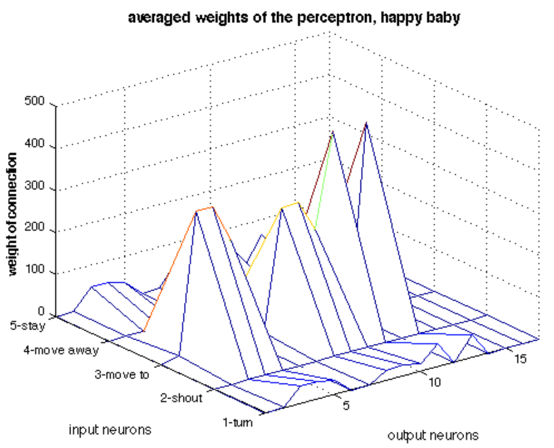
Besides the learning rule, there is another piece of the system influencing greatly the learning phase of the perceptron. It is the calculation of feedback to perceptron. Difference in calculation of feedback will directly modify the increase of weights in perceptron.

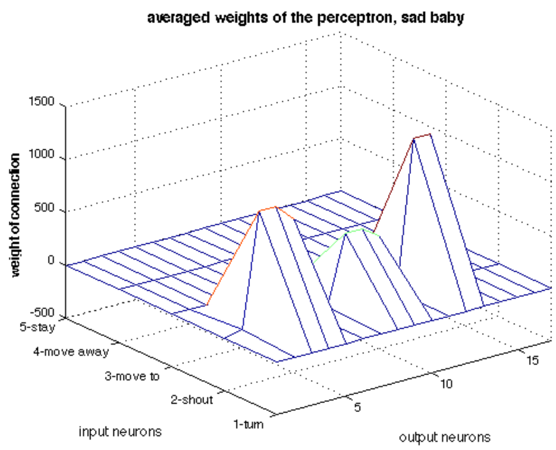
In the standard system, the feedback vector is a weight vector multiplying an influence factor, as introduced in section 3.4.5.2. The influence factor is how the chosen action for baby is appropriate. The weight vector contains a value of 3 for the chosen action and a value of 1 for other actions.

Here, the rejected actions receive also a feedback since the weight vector distributes a non-null weight to them. So how the system will evolve if we delete the weights for rejected actions is a very attractive topic. Thus we will build a modified copy of system by changing the weight vector in feedback calculation.

Concisely, the weight for chosen action will decrease to 1 and the weight for rejected actions will be 0. So if the influence vector is 2.5 and the chosen action is the second action ('to shout'), then the new feedback is [0; 2.5; 0; 0; 0], while the old one is [2.5; 7.5; 2.5; 2.5; 2.5].

Like in the previous comparison, the designed experiment will be repeated with original system and modified system respectively. The observed parameters are the two-dimensional weight matrix of perceptron and the statistic probability of possible actions.





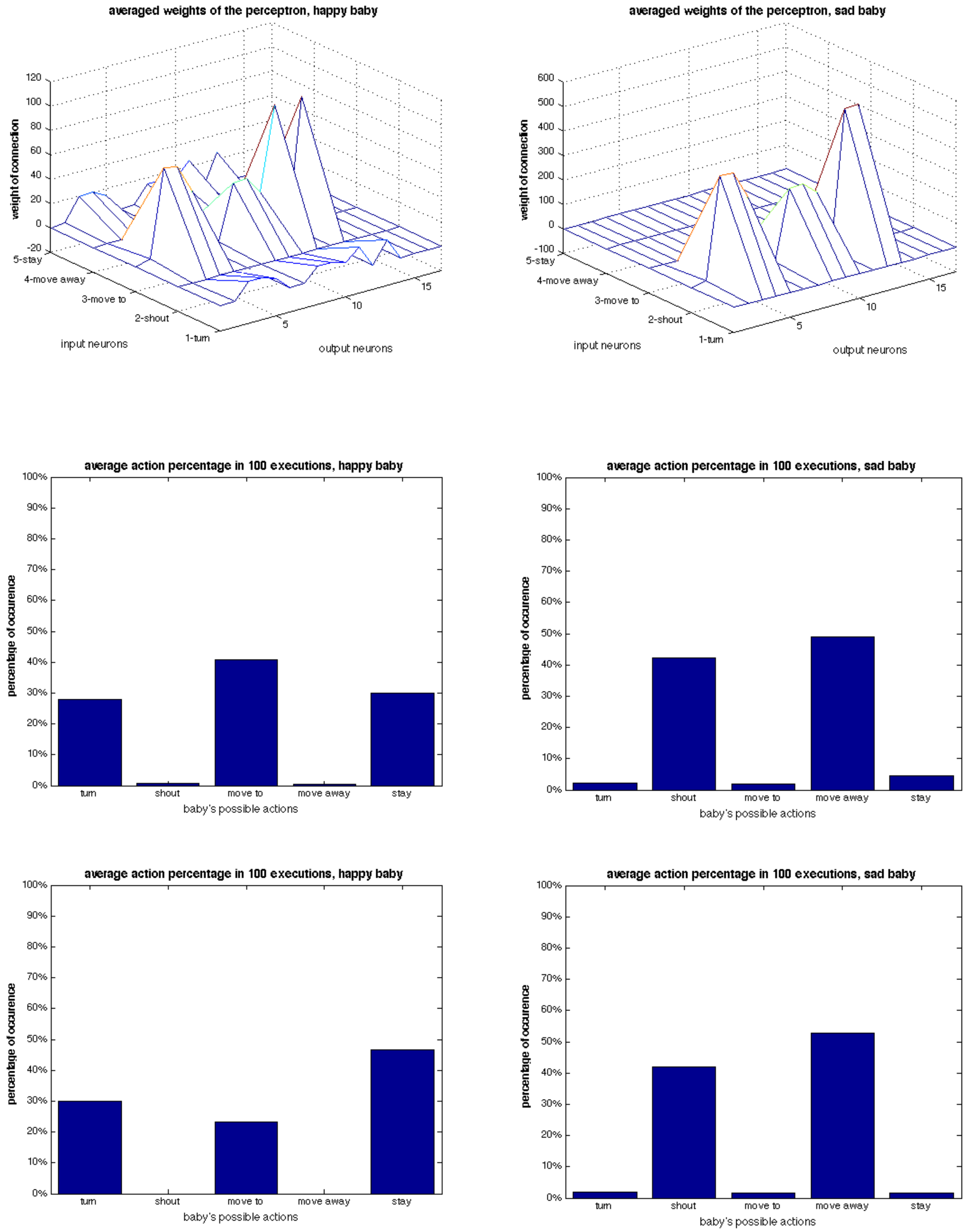


Figure 4.5 The weight matrix and statistical behavior for both babies (left and right) in different conditions. The difference between tested conditions are not evident. Larger figures are provided in Appendix.

The first group of four mesh pictures shows the weight matrices of original system and modified system, respectively to happy baby and sad baby. Evidently, we could observe that the forms of weight matrices have not changed critically when modifying the system. Only the weight matrix of happy baby in modified system is slightly more undulating than the matrix of happy baby in original system.

The second group, consisting of statistical information about action's occurrence, also indicates there is little difference between original and modified systems. Only the happy baby in modified system moves less frequently to caregiver and becomes more interesting in staying still. The relationship between happy baby and caregiver becomes a little more insecure.

After all, we now understand the modification on feedback calculation has no evident effect. One possible reason is the chosen action has always a dominant weight to the rejected actions in calculation of feedback.

4.5.3 Analysis on input preprocessing

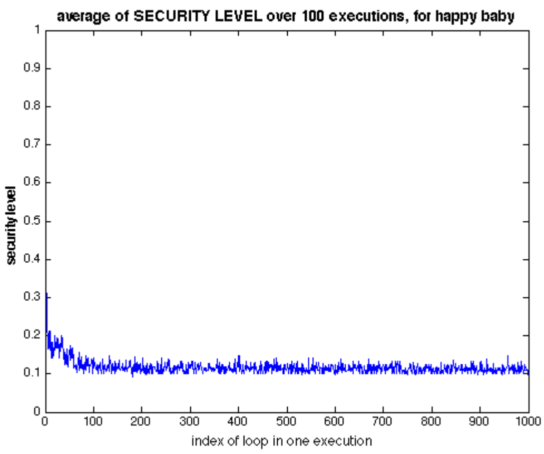
Learning rule of the perceptron is just one important aspect of trying to improve the system. In addition to this aspect, we could try improvement on the preprocessing stage of the input signals as well.

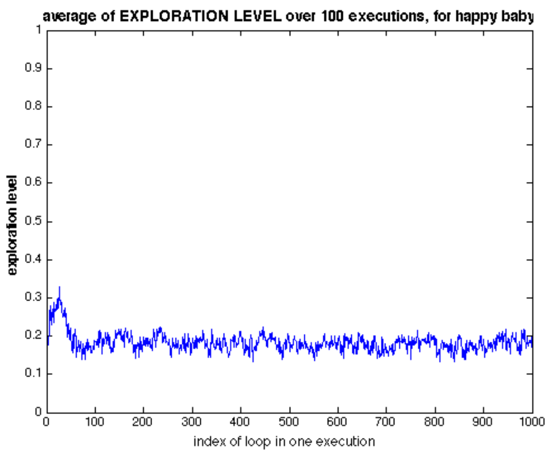
The touching and distance signals will be discretized and employed directly. But the environment and face recognition have a similar prepossessing procedure. We will try whether improvement is possible on these two signals.

There is one interesting characteristic of those two signals: strong attractors exist in the Hopfield network used in preprocessing procedure. Concisely, there is one strong attractor (face of the caregiver) for face recognition and four strong attractors (north, south, west and east environments) for environments.

To understand how these strong attractors influence the calculations in preprocessing of the environment and face recognition signals is therefore crucial.

In purpose of evaluating the influence of strong attractors, two directions of study could attempt. One direction is the number of strong attractors in the Hopfield network; the other is the degree of the strong attractors, i.e. the times of storage of a strong attractor.





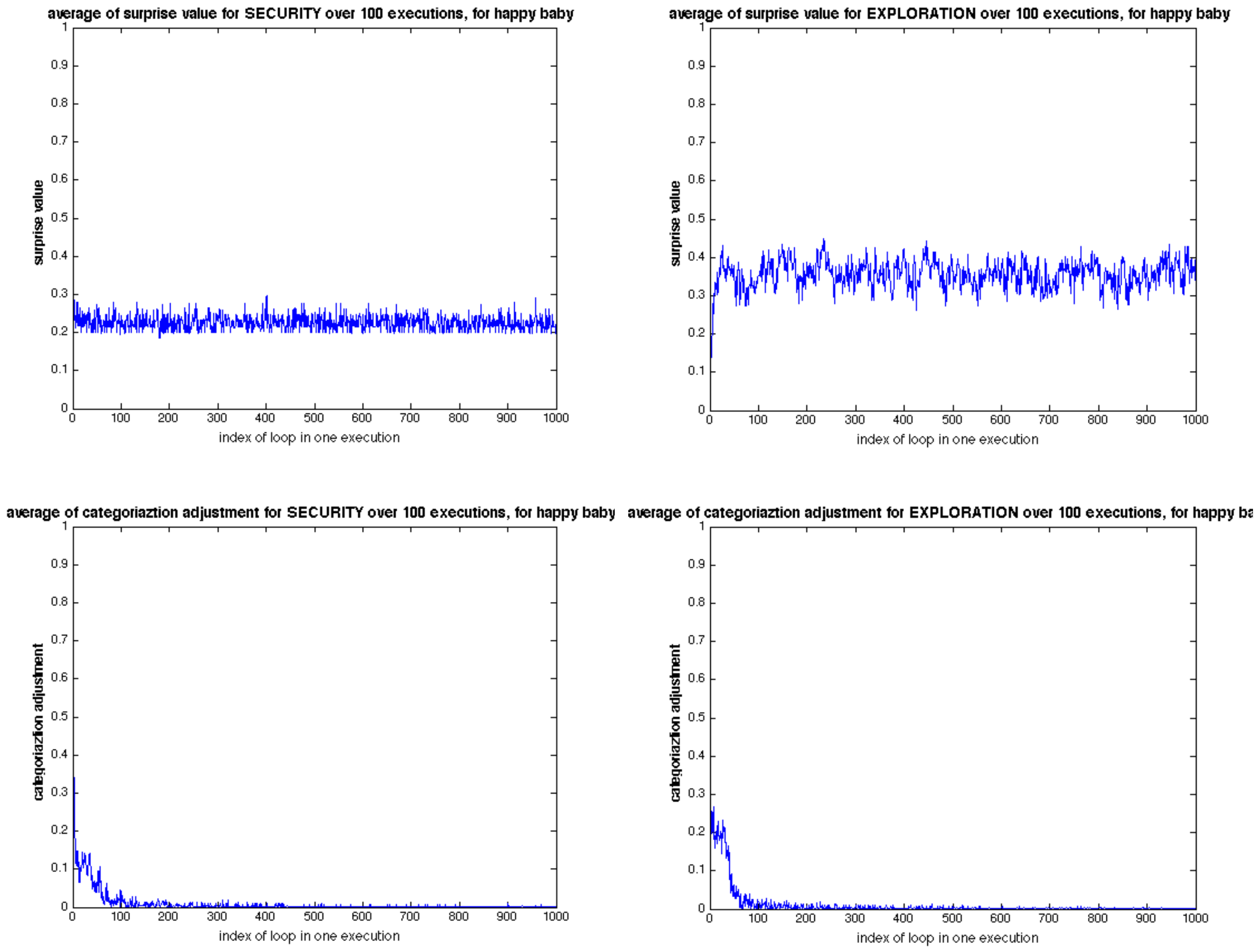


Figure 4.6 The Security Level, Exploration Level, two Surprise values and two Categorization Adjustments in first stage (learning stage) for the happy baby. Larger figures are provided in Appendix.

Fortunately, both these two directions could be attempted in our standard system configuration.

Specifically, in the standard configuration, the one strong attractor (caregiver's face) in the face recognition's neural networks and the four strong attractors (north, south, west and east environments) in the environment's neural networks are a very effective contrast. The former case stands for singularity and the latter case represents the multiplicity. Therefore, what we need to do is to study the difference between the measurements about face recognition and the measurements about environment. Concisely, to compare the security level to the exploration level, to compare the surprise values of two signals and to compare the categorization adjustment of two signals. This is the research about number of strong attractor.

The other research direction, the degree of strong attractors, could be achieved by observing the evolvement of the six measurements in the preprocessing procedure on function of time. The reason is simple: in the standard configuration of simulating system, the number of storage of a strong attractor is proportional to the number of loops in execution, by factor of a probability.

After all, we know understand that both two studies could be done in one experiment. So the designed experiment has been repeated once more and the average value sequences of the six measurements (security level, exploration level, two surprise values and two categorization adjustments) are recorded for analysis.

The six pictures in figure group above show the evolvements of these six measurements on function of time. The three data in left column are successively the security level, the surprise value for security and the categorization adjustment for security. Relatively, the three in right column are the exploration level, the surprise value for exploration and the categorization adjustment for exploration.

4.5.3.1 Influence of number of strong attractor

For this research, we need to compare the data in left column to the data in right column. The data in left column have just one strong attractor, while the data in right column have multiple strong attractors.

Evidently, we observe higher values in the right column than in the left column. It means adding strong attractor will increase the measurements. In addition to this difference on level of value, we observe a larger fluctuation in the right column than in the left. This indicates that more strong attractors make the system more sensitive. And higher sensitivity is just right what we expect for improvement.

In order of a deeper analysis, we will concentrate on the surprise value and categorization adjustment, because the security level and exploration level are just the average of corresponding surprise value and categorization adjustment. Moreover, between the surprise value and categorization adjustment, the former measurement is much more precious, since the latter one decreases to near zero very soon after launch of experiment. The analysis for this decrease is in following section.

So now we focus on the second row of the picture group. Evidently, the surprise value on the right side is much more fluctuating than the one on the left side. In fact, in situation of left side, the Hopfield network used for calculating surprise value has just one strong attractor, and the basin of attraction of this only strong attractor will expand larger and larger. This extension is an irreversible transform as the strong attractor has been stored again and again in the network. In contrast, in case of right side, the four strong attractors will also be stored again and again in the Hopfield network and their basin of attraction will expand as well, but the basins of different strong attractors will conflict with each other. This confliction on frontier of basins makes the retrieved pattern of a state unstable. And it is this confliction and this instability that complicate the dynamics of the Hopfield network and implies the fluctuation.

In summary, if we want the system to be more sensitive, we need to let the neural networks have more strong attractors. In other words, we need to let the baby have more precious memories, like mother, father, his/her room and so on.

4.5.3.2 Influence of degree of strong attractor

For this research, we will observe how the measurements evolve on function of time. In our experiment, we use the index of loop to present time. The x-axis of the figures is indeed an axis of index of loops.

In the six measurements, the two measurements about the categorization adjustment expose an interesting phenomenon that the value of categorization adjustment goes down to near zero very quickly after launch of experiment. Normally, the value of measurement will stay close to zero after nearly 100 loops, as shown on the pictures of the third row.

This phenomenon is in real unfavorable in our experiment, because categorization adjustment could not help us to assess the environment and face recognition signals any more. Thus we need to identify why this phenomenon appears and to find out a solution.

Fortunately, if we observe the variations of the weight vectors of the applied self-organized map, we could notice that after nearly 100 loops, the map has converged to a stable structure. If we continue to store the strong attractors, the map will have little modification, and consequently the categorization adjustment will be really low, i.e. near to zero. This is the reason of generating the described phenomenon.

In order to avoid this phenomenon, we should not store the strong attractors with too many times. But it is contradictory to the fact that the strong attractors must be stored many times as they play the role of precious memories.

One possible solution to solve this dilemma is to add a small noise to the strong attractor during storage. This noise could be seen as the cognitive deviation. For example, the appearance of caregiver has right to change, like to wear a hat on head. So to store the exact strong attractor into network every time is not very real and this small change could be modeled as a random noise.

In other hand, if we use a real sensor for capturing the four input signals, there will be noises added to the received strong attractors as well. This also proves to add a small noise during storage is reasonable and feasible.

However, capacity of storage with this method is not clear, we are not sure that we could store the strong attractors (with noise) as many times as we desire. Therefore, this topic about capacity of storage could be a future work to accomplish.

4.5.4 Conclusion

In the above subsections, we performed two comparisons on learning stage of the decision-making perceptron and two analyses on influence of the strong attractors in the preprocessing of signals. From the results of these comparisons and analyses, we could obtain some useful conclusions.

Firstly, the current configuration of the perceptron is moderate to give instructive results for distinguishing different attachment types. However, if we want to have accurate authentication of the attachment types, we need to increase the sensibility of the simulating system to the external signals. Two possible solutions are proposed. One solution is to apply more strong attractors in Hopfield networks, in order to make the surprise values more fluctuating. The other one is to add small noise on strong attractors during storage, in purpose of restoring the normal functionality of categorization adjustment.

5 Evaluation and conclusion

5.1 Reviewing the built simulating system

The built simulating system introduced in whole chapter 3 is the core piece of this project. We expect this system could simulate how an infant learns and decides to act with respect to different environment. The design of this built system is inspired by the idea of arousal-based model (Hiolle, A et al. 2012).

However, the arousal-based model is applied directly to a Sony AIBO robot, but in our situation we will just build a system running in virtual environment. Thus we have to change the format of the input signals. Moreover, arousal-based model uses a deterministic method to choose the wanted action but this method could not adapt to a learning system. Hence we should also use a neural network instead of this deterministic method, and define the learning rule of this neural network. After this, we have to evaluate the influence of caregiver's action since this influence will appear in learning stage of decision-making network.

Because of these modifications, the built system is largely different to the arousal-based model, although the idea of arousal level has been kept in our new system.

Besides the task of simulation mentioned above, this system has other two targets:

i. Simulated virtual babies with different kind of caregiver will have different learning and decision-making mechanism. And this difference should be authenticable.

ii. The different learning and decision-making mechanisms could be related to different attachment types in attachment theory.

Evidently, the second target is based on the first one and consequently more complicated.

5.2 Achievements

An experiment has been designed, with help of the concept of 'Strange Situation Procedure'. And the built system has been tested by this experiment. Unfortunately, the results of experiment indicate the second target listed above is not accomplished. Concisely, the completion of these two targets is:

i. We could successfully identify different simulated virtual babies cared by different kind of caregivers. In addition, the virtual baby cared by a good enough mother seems have a secure attachment, while the virtual baby cared by a not good enough mother seems have an insecure attachment.

ii. We observe that the virtual baby cared by a good enough mother seems have a secure attachment, while the virtual baby cared by a not good enough mother seems have an insecure attachment. This means a basic identification of attachment types could be done. For more accurate identification, it is feasible once improvements on system are made.

This experiment shows that the conception of the built system is successful, while limits also exist at the same time. More efforts on improving this designed system are required.

5.3 Analysis of system

Some attempts have been done in order to analyze and improve the system.

5.3.1 Decision-making network

Concerning the part of perceptron, which performs decision-making, two comparisons are designed. The first one compare the original learning rule to a new one, while the new rule uses the output vector of perceptron instead of the normalized distribution vector. This attempt failed, the new learning rule has a critical drawback: the weights of perceptron increase exponentially and soon exceed the upper bound of variable in Matlab.

The second comparison focuses on the feedback vector used in the learning rule of perceptron. This feedback vector is in fact the evaluated influence of the interaction between baby and caregiver. The original method of calculating feedback will distribute to every possible action a non-null value, but the selected action will much bigger than others. In contrast, the new method will only give the selected action a non-null value, and give other rejected actions a zero. Results of experiment indicate these two methods having little difference. The fact that the weight of selected action is always dominant may be a possible reason to the shown indifference.

5.3.2 Preprocessing of input signals

The preprocessing procedure of the input signals is another important part of the built system, because the idea of arousal level incarnates in this part. In addition, this part applies the strong attractors as the precious memories in calculation.

Two analyses have been done on the topic of how the strong attractors influence the system. Some useful conclusions come out.

At first, when we assess the influence of number of strong attractors, we find multiple strong attractors could make the dynamics of the applied Hopfield networks more complicated, and then let the processed signals be more fluctuating. Here, a bit more fluctuation is not a bad thing, because the system is not enough sensitive to external signals and fluctuation equals to sensitivity. Thus this first analysis encourages us to use more strong attractors.

The second analysis focuses on the influence of degree of strong attractors. By observing the evolvement of measurements in the preprocessing procedure, we find to store a strong attractor too many times in its exact form will degrade the system. The most appropriate solution to this problem is to add a small noise to the strong attractors during storage.

5.4 Conclusions

Good results and bad results exist at the same time in the experiment for testing our built system. We could not therefore declare that our conception is complete and perfect, there are still many places needing improvement. But we should also recognize that our system is developed in correct direction, as basic identifications succeed. Improvement is the next step we should carry on in this research.

Some comparisons and analyses have been done in purpose of helping to understand this system better. Fortunately there are already two potential improvements discovered during the analyses. The validity of these improvements needs to be confirmed in future works.

5.5 Future work

A number of directions could be considered for continuing this work in future:

i. There are still many parameters that may have a more effective value. We could perform more comparisons on these parameters in order to find a better configuration of our simulating system.

ii. Only three caregiver's action templates are used in our experiments. Some different templates could be conceived and be tested in experiments, to see whether other templates give a better result.

iii. The found potential improvements need to be tested. If they are valid, we could implement these improvements and assess the new system.

iv. A theoretical and mathematical analysis of the system is favorable. If we understand the profound principle of this system, the improvements will be easier.

Bibliography

Ainsworth, M. D. S., Blehar, M. C., Waters, E., & Wall, S. (1978). *Patterns of attachment: A psychological study of the strange situation*. Lawrence Erlbaum.

Bohland, J. W., & Minai, A. A. (2001). Efficient associative memory using small-world architecture. *Neurocomputing*, *38*, 489-496.

Bowlby, J. (1969). Attachment and loss. *Attachment, vol*, *1*, 1982.

Calcraft, L., Adams, R. G., & Davey, N. (2006). Gaussian and exponential architectures in small-world associative memories. In *Procs of the European Symposium on Artificial Neural Networks, ESANN'06*.

Calcraft, L., Adams, R., & Davey, N. (2007). Optimal connection strategies in one-and two-dimensional associative memory models. *Proceedings of the 7th Intl. Wksp. on Information Processing in Cells and Tissues, Oxford*.

Doidge, N. (2007). *The brain that changes itself: Stories of personal triumph from the frontiers of brain science*. Penguin. com.

Edalat, A. (2013) *Capacity of strong attractor patterns to model behavioural and cognitive prototypes.* In the Proceedings of The Neural Information Processing Systems (NIPS) 2013 (unpublished)

Edalat, A., & Mancinelli, F. (2013). Strong Attractors of Hopfield Neural Networks to Model Attachment Types and Behavioural Patterns. In*Proceedings of International Joint Conference on Neural Networks (IJCNN)*.

Hebb, D. O. (2002). *The organization of behavior: A neuropsychological theory*. Psychology Press.

Herz, J., Krogh, A., & Palmer, R. G. (1991). Introduction to the theory of neural computation. *Lecture Notes Volume in the Santa Fe Institute Studies in the Sciences of Complexity. Cambridge, MA, Perseus Books*.

Hiolle, A., Bard, K. A., & Canamero, L. (2009, September). Assessing human reactions to different robot attachment profiles. In *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on* (pp. 251-256). IEEE.

Hiolle, A., & Canamero, L. Conscientious Caretaking for Autonomous Robots An Arousal-Based Model of Exploratory Behavior.

Hiolle, A., & Canamero, L. (2007). Developing sensorimotor associations through attachment bonds.

Hiolle, A., & Canamero, L. (2008). Why Should You Care?-An Arousal-Based Model of Exploratory Behavior For Autonomous Robot. In *ALIFE* (pp. 242-248).

Hiolle, A., Canamero, L., & Blanchard, A. J. (2007). Learning to interact with the caretaker: A developmental approach. In *Affective Computing and Intelligent Interaction* (pp. 422-433). Springer Berlin Heidelberg.

Hiolle, A., Cañamero, L., Davila-Ross, M., & Bard, K. A. (2012). Eliciting caregiving behavior in dyadic human-robot attachment-like interactions. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *2*(1), 3.

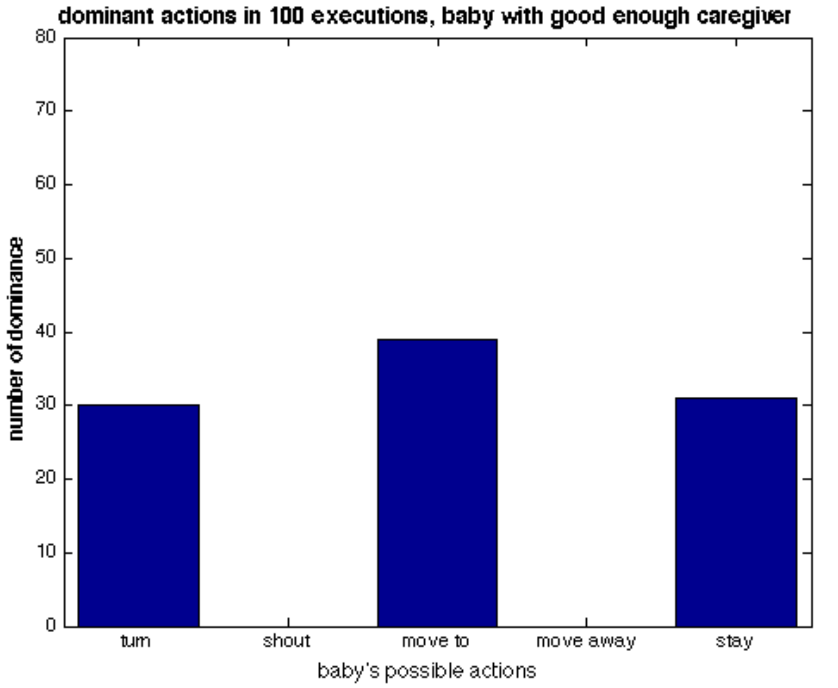
Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological cybernetics*, *43*(1), 59-69.

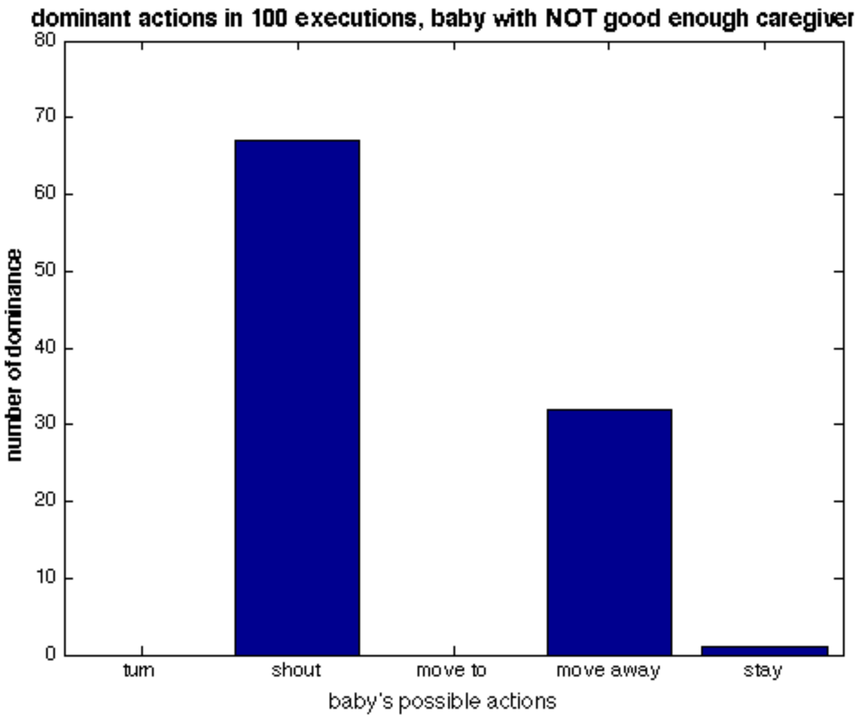
Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of ‘small-world’networks. *nature*, *393*(6684), 440-442.

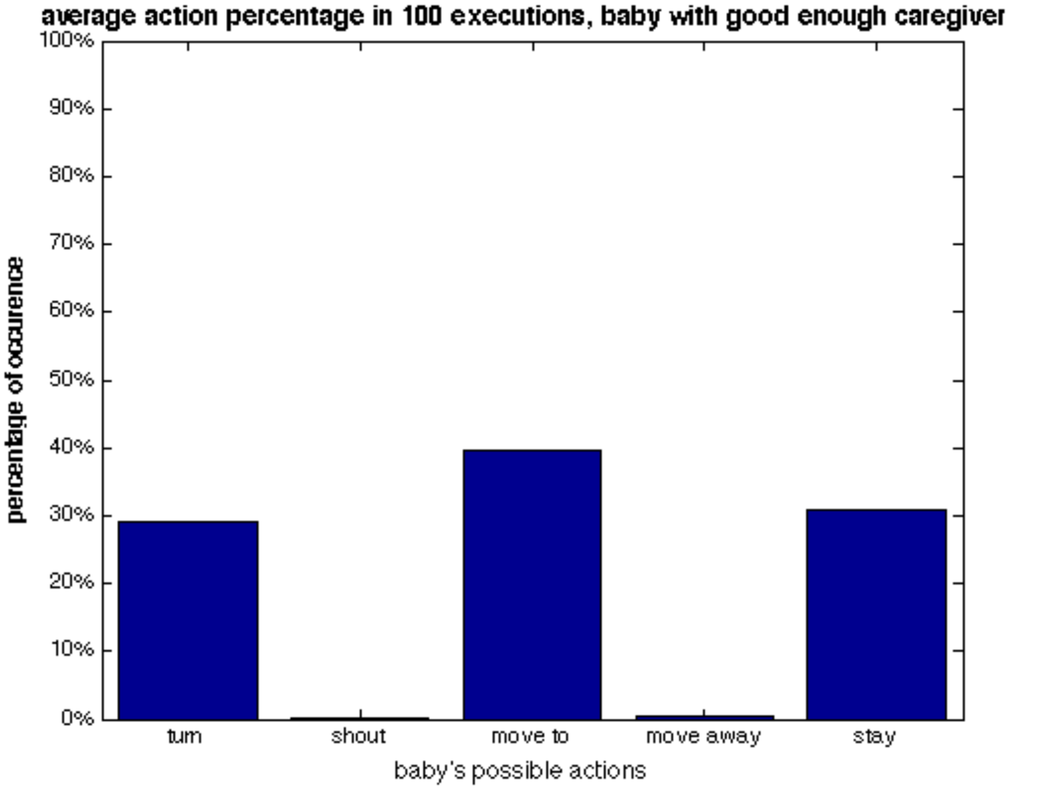
Appendix

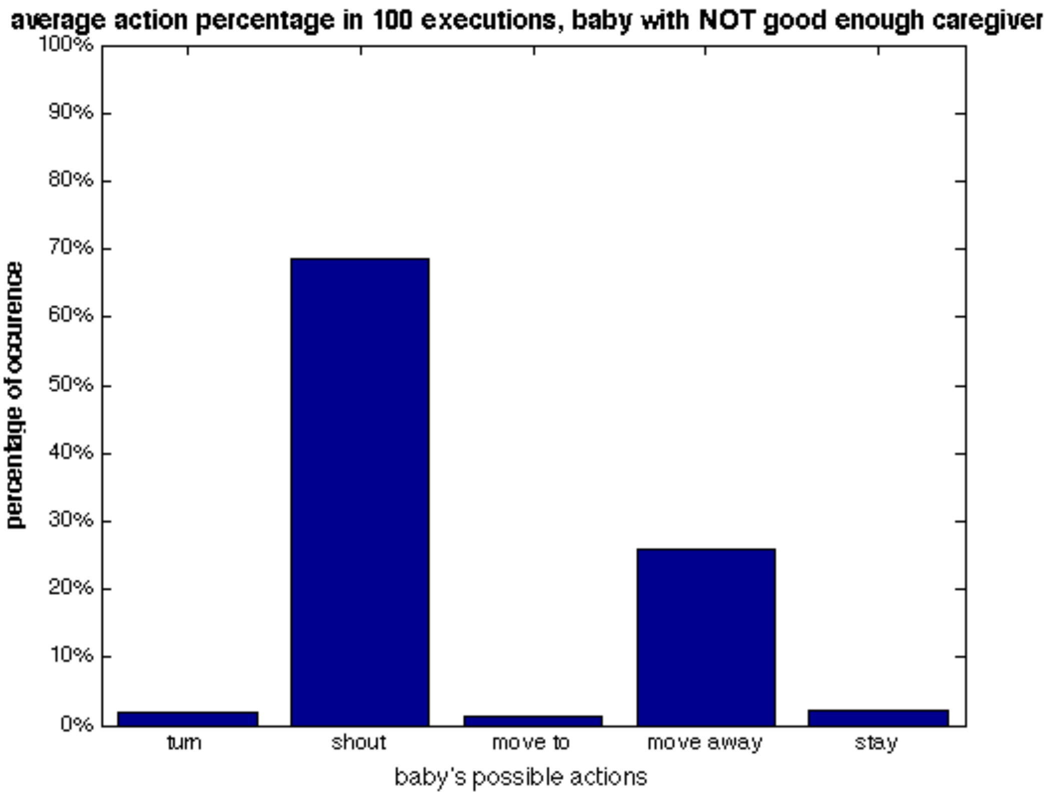
In this section, larger figures showing results of the experiments will be posted here, in the same order as they are in the chapter 4.

- First group of figures (page 33, 4 figures)

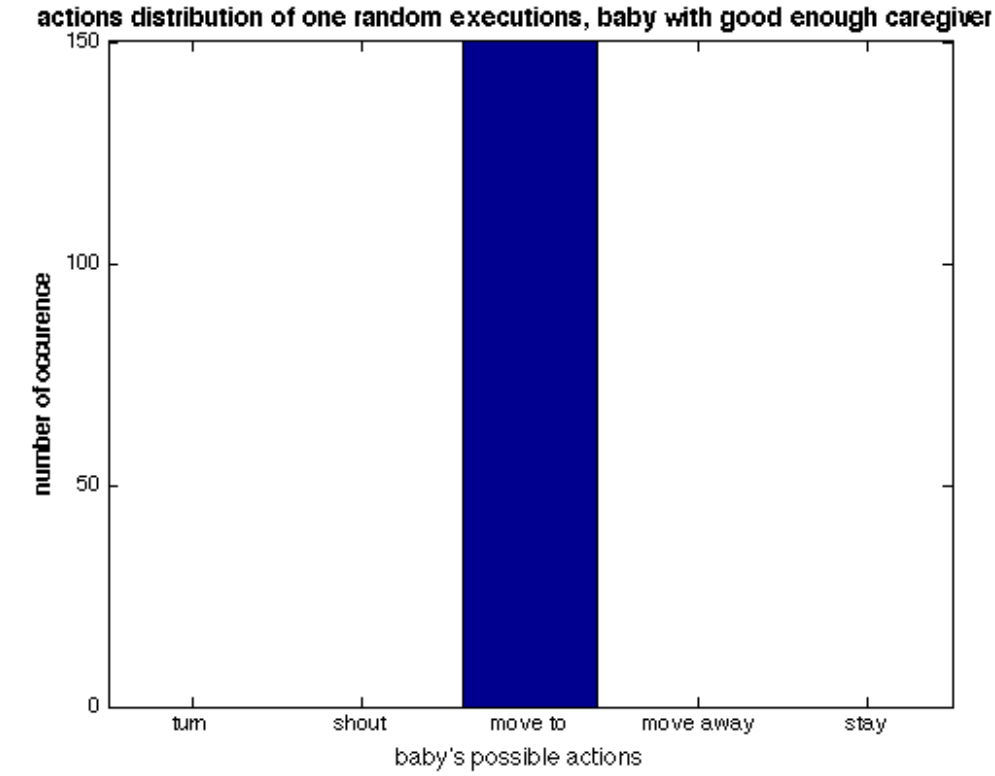


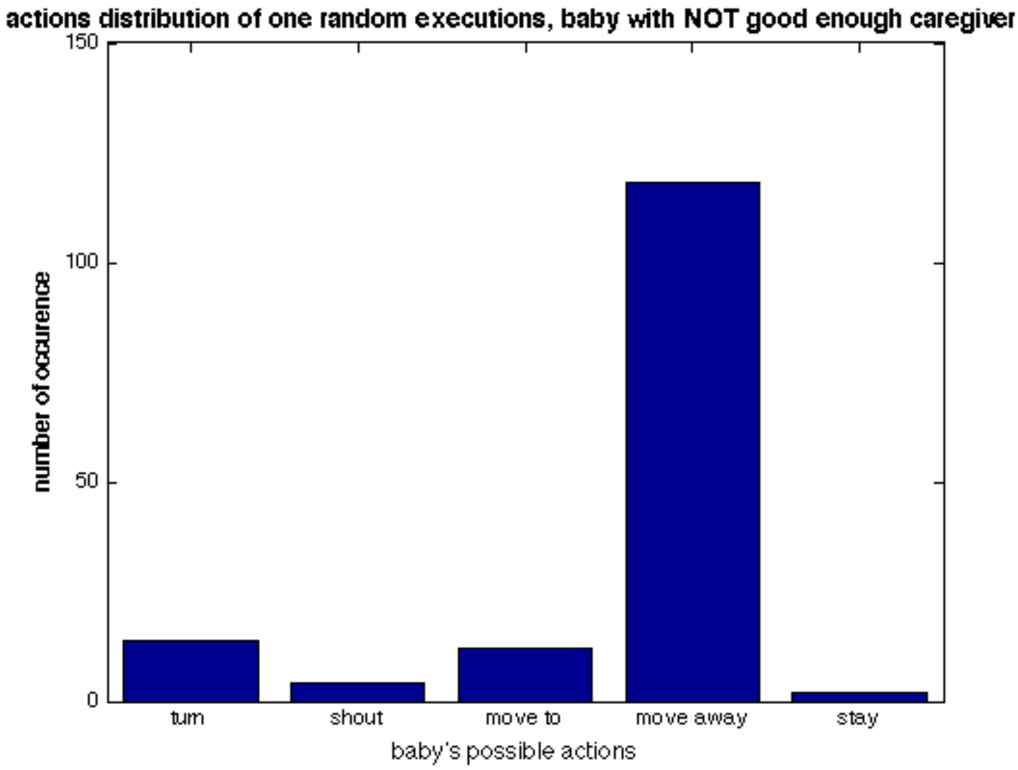



average action percentage in 100 executions, baby with good enough caregiver
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
percentage of occurence
turn
shout
move to
move away
stay
baby's possible actions

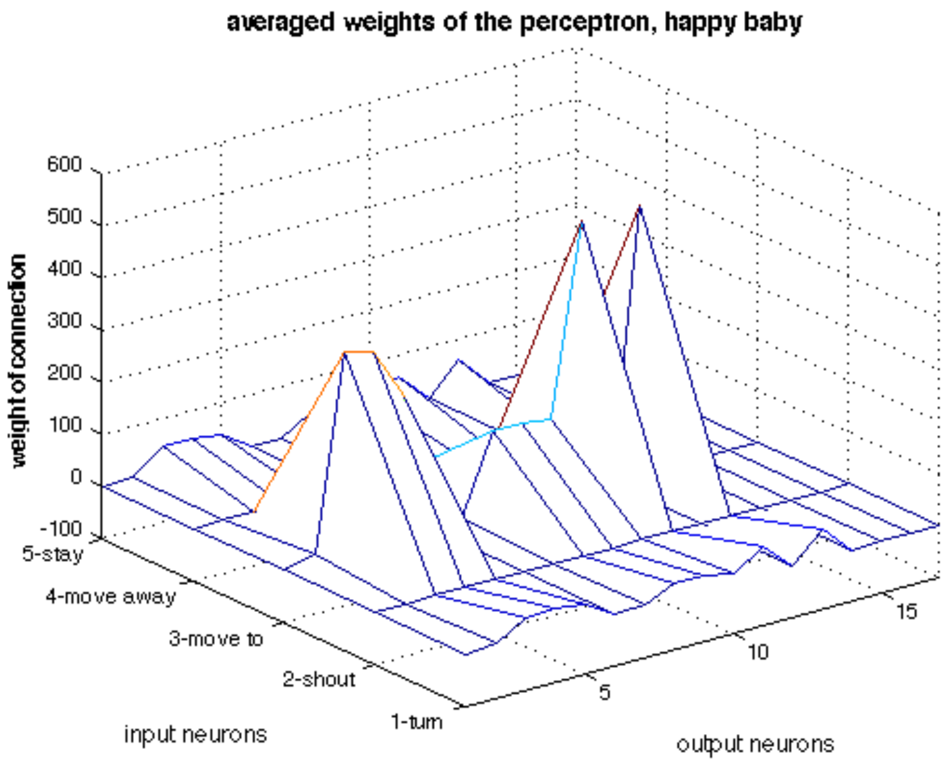

average action percentage in 100 executions, baby with NOT good enough caregiver
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
percentage of occurence
turn
shout
move to
move away
stay
baby's possible actions

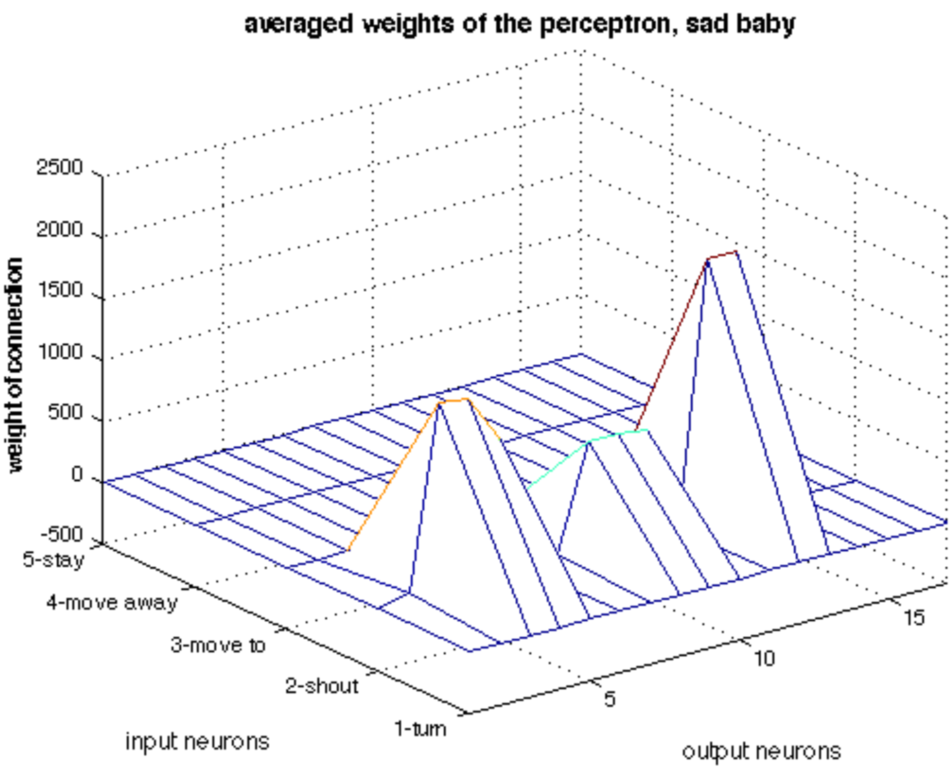
- Second group (page 35, 2 figures)

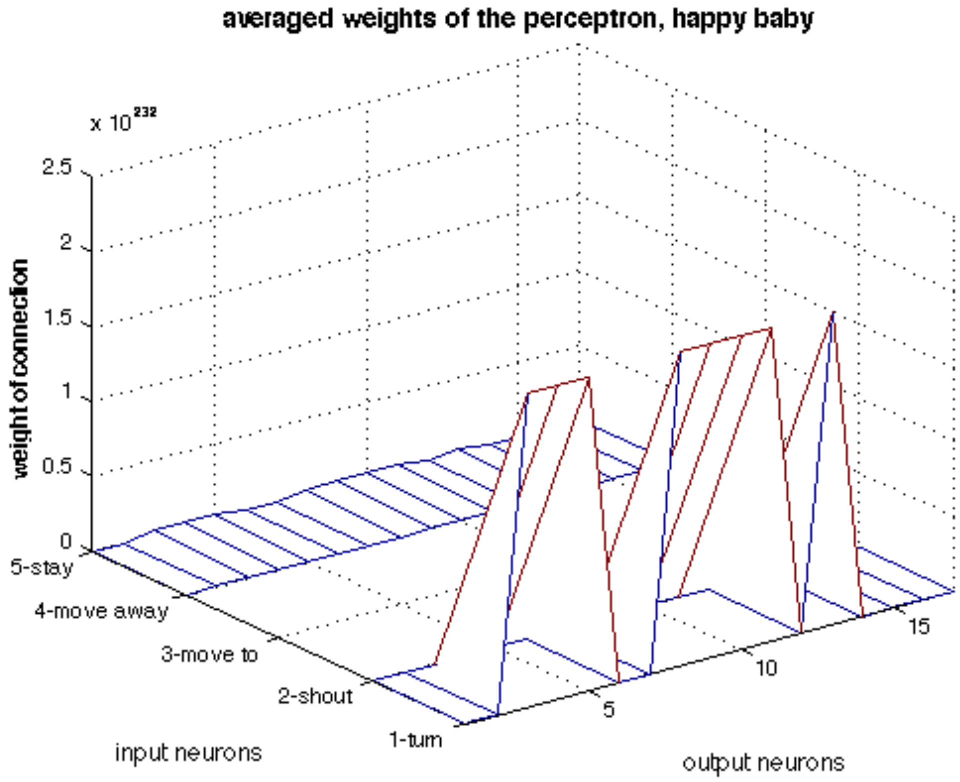


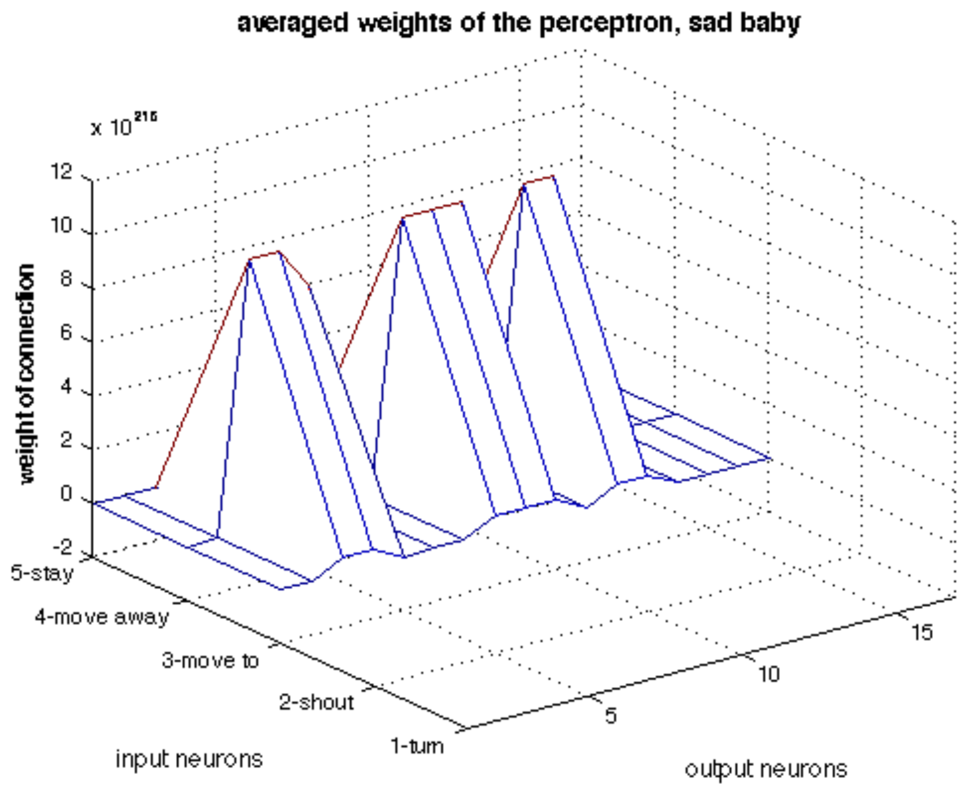


- Third group (page 36, 4 figures)

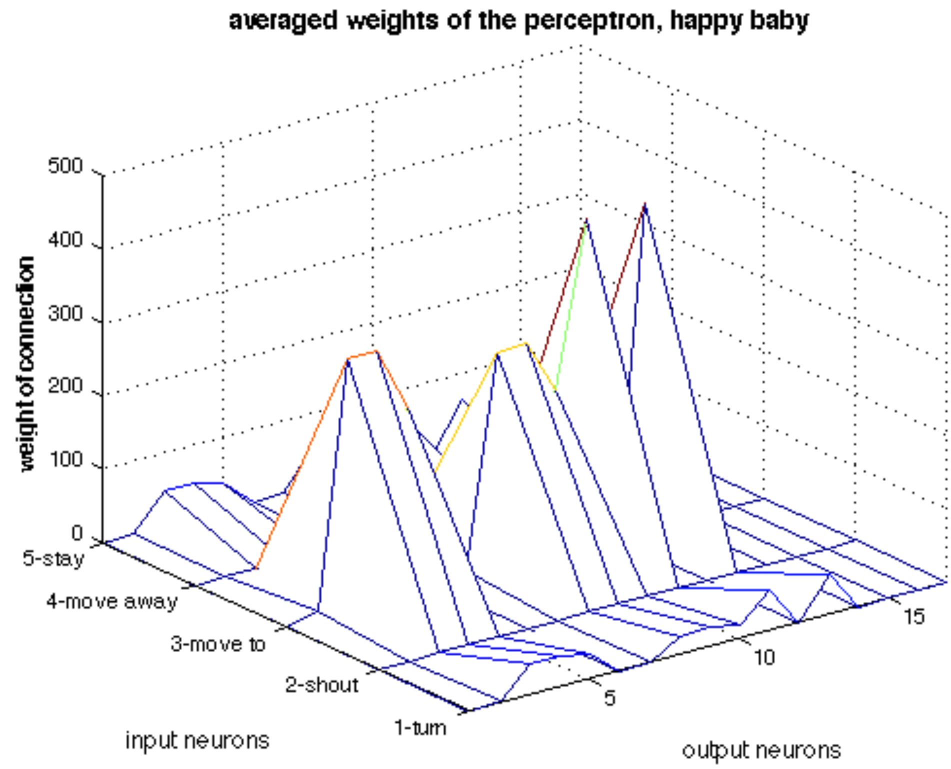


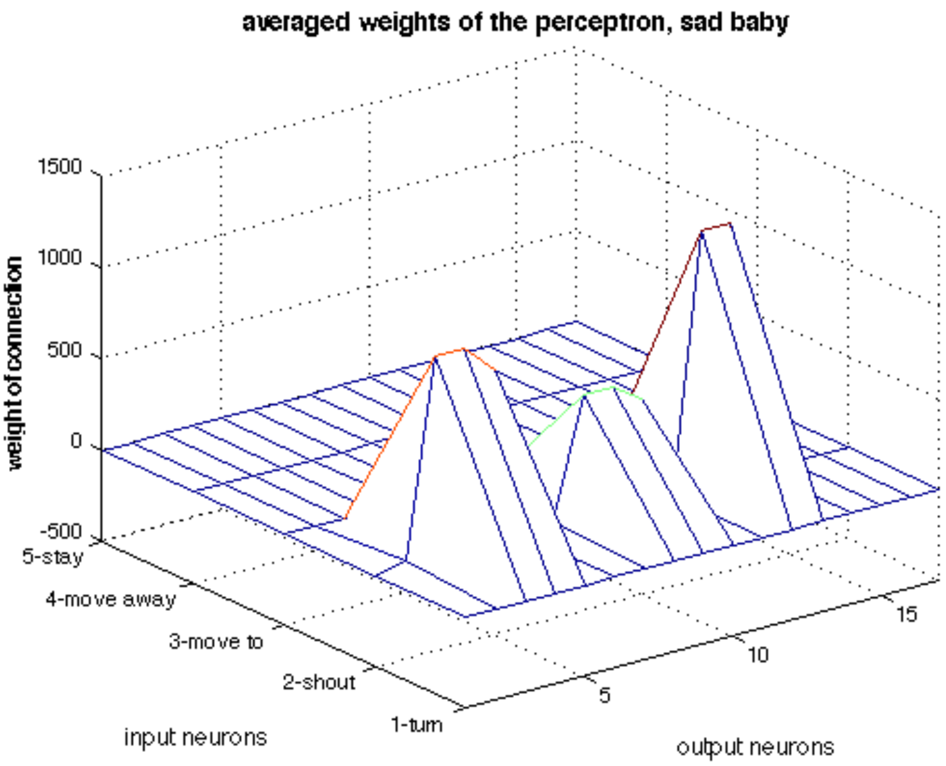


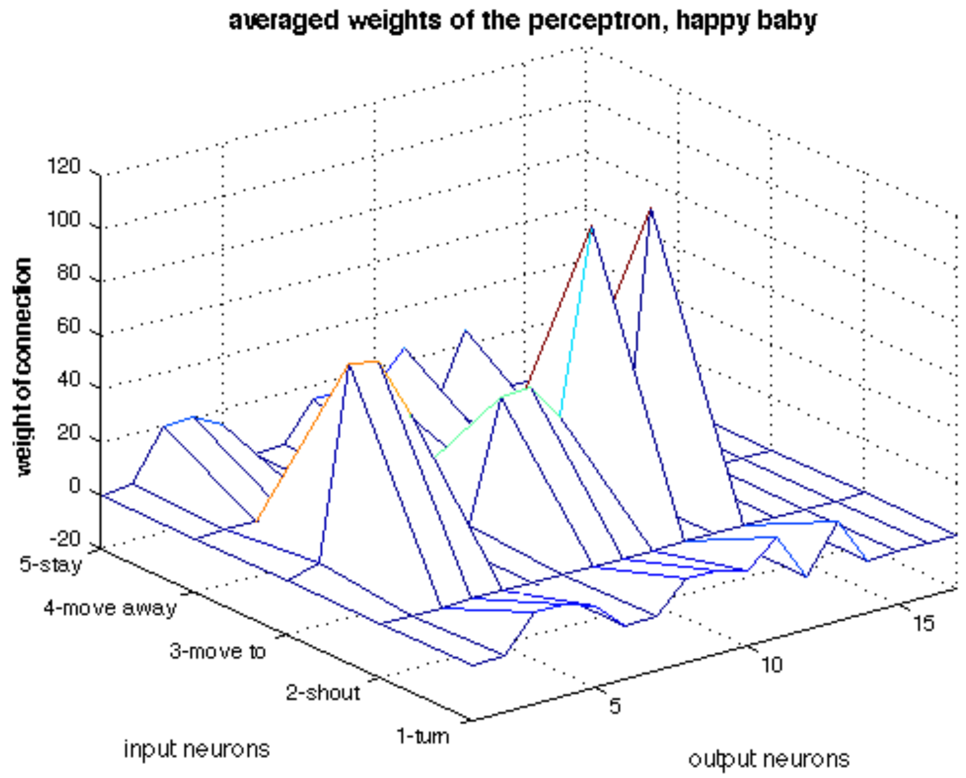

averaged weights of the perceptron, happy baby
x 10^232
2.5
2
1.5
1
0.5
0
weight of connection
5-stay
4-move away
3-move to
2-shout
1-turn
input neurons
5
10
15
output neurons

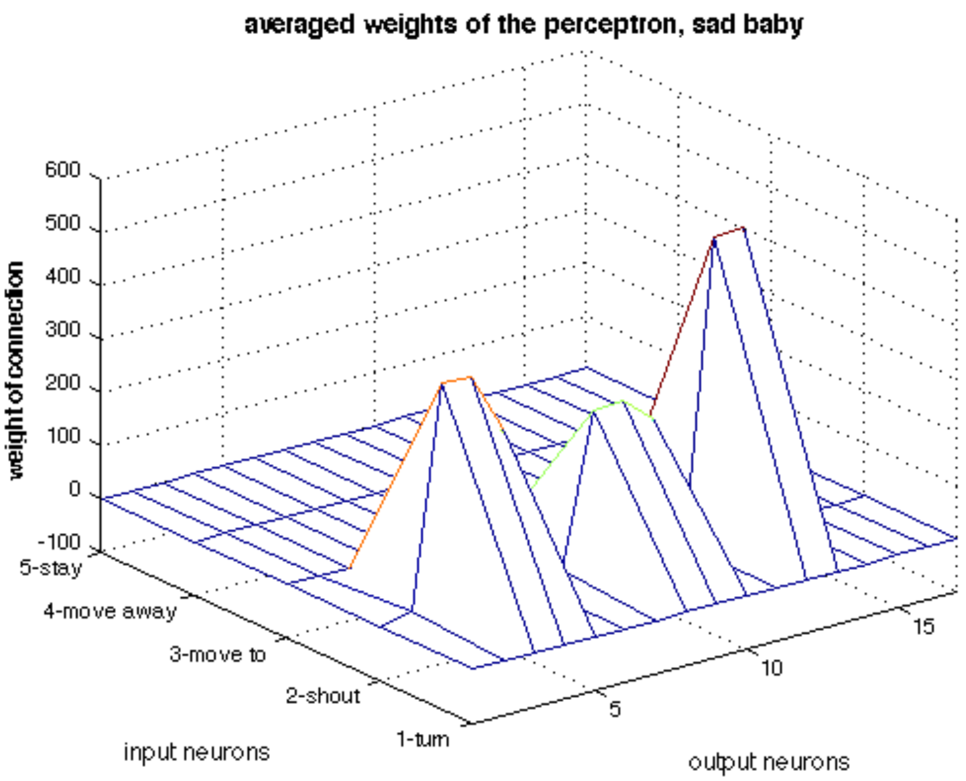

averaged weights of the perceptron, sad baby
x 10^215
12
10
8
6
4
2
0
-2
weight of connection
5-stay
4-move away
3-move to
2-shout
1-turn
input neurons
5
10
15
output neurons

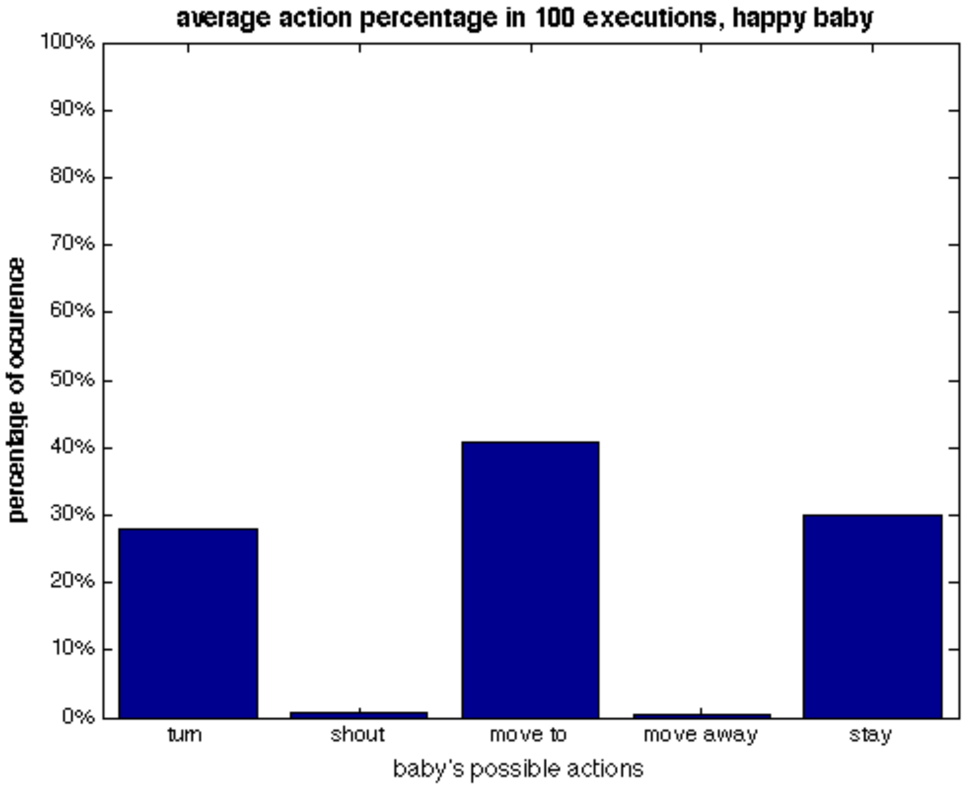
- Fourth group (page 37-38, 8 figures)

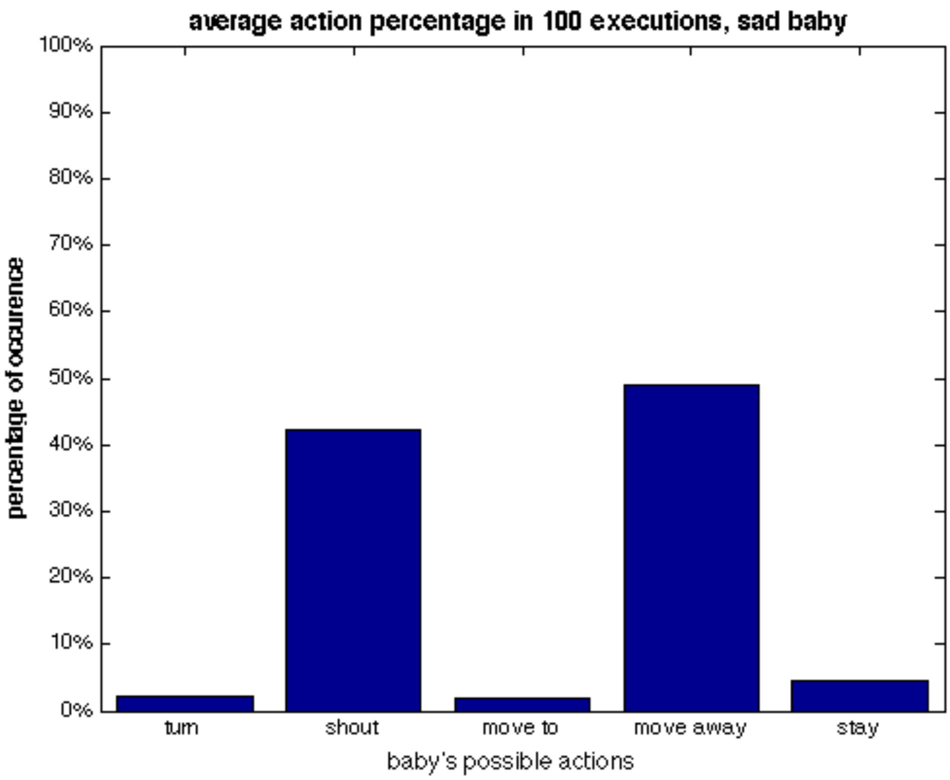


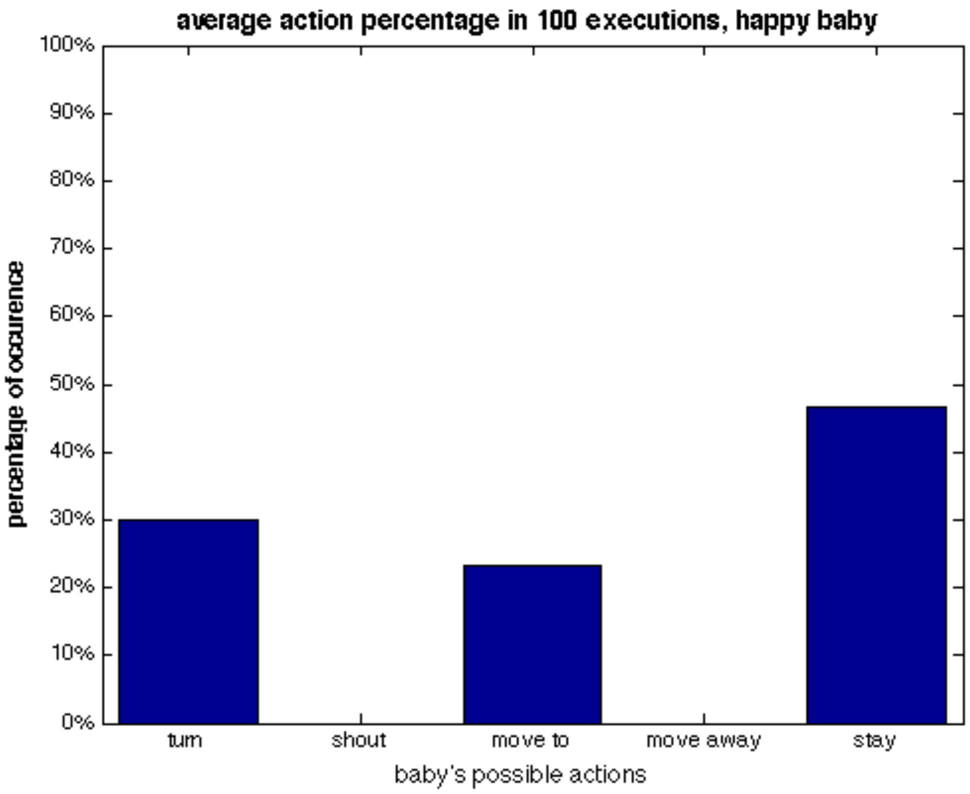


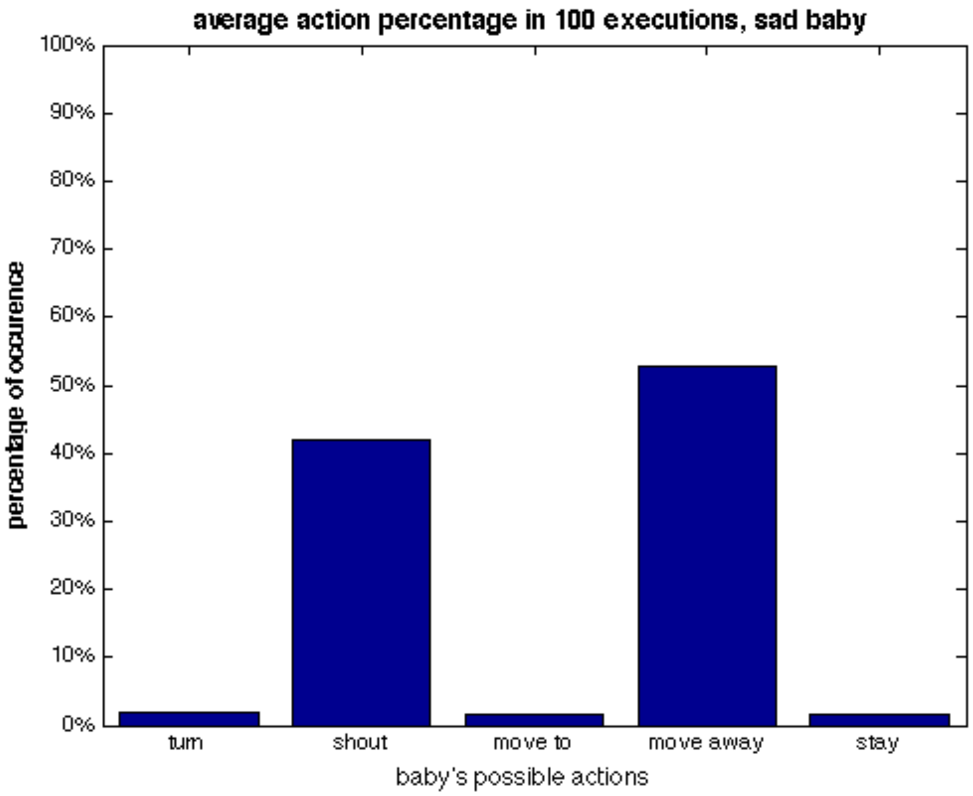

averaged weights of the perceptron, happy baby
120
100
80
60
40
20
0
-20
weight of connection
5-stay
4-move away
3-move to
2-shout
1-turn
input neurons
5
10
15
output neurons


averaged weights of the perceptron, sad baby
600
500
400
300
200
100
0
-100
weight of connection
5-stay
4-move away
3-move to
2-shout
1-turn
input neurons
5
10
15
output neurons

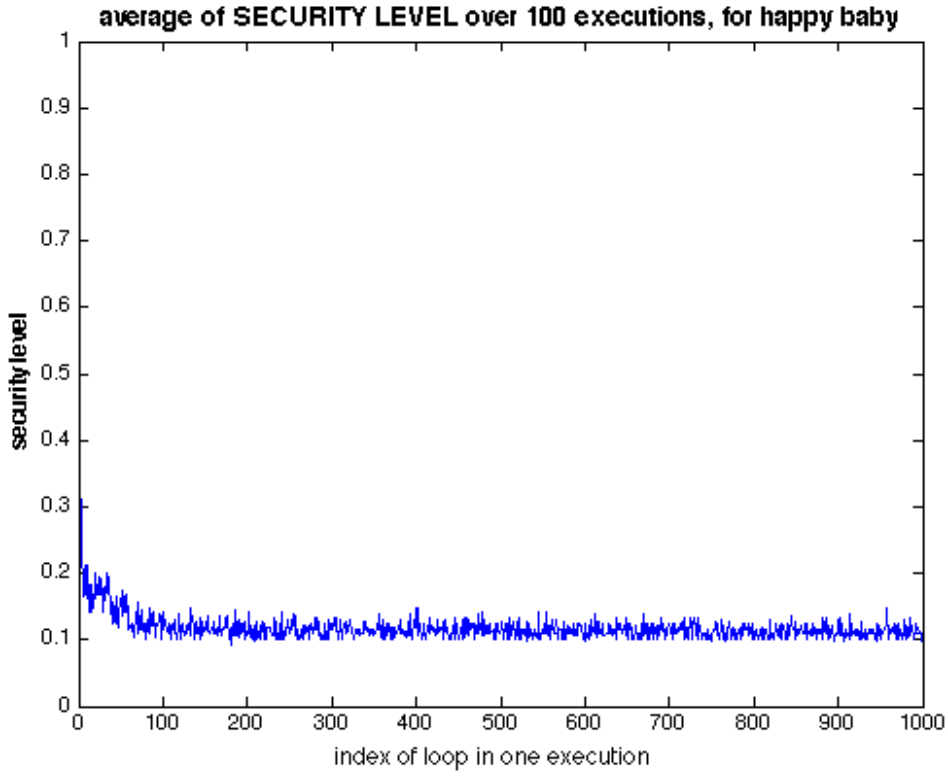

average action percentage in 100 executions, happy baby
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
percentage of occurence
turn
shout
move to
move away
stay
baby's possible actions

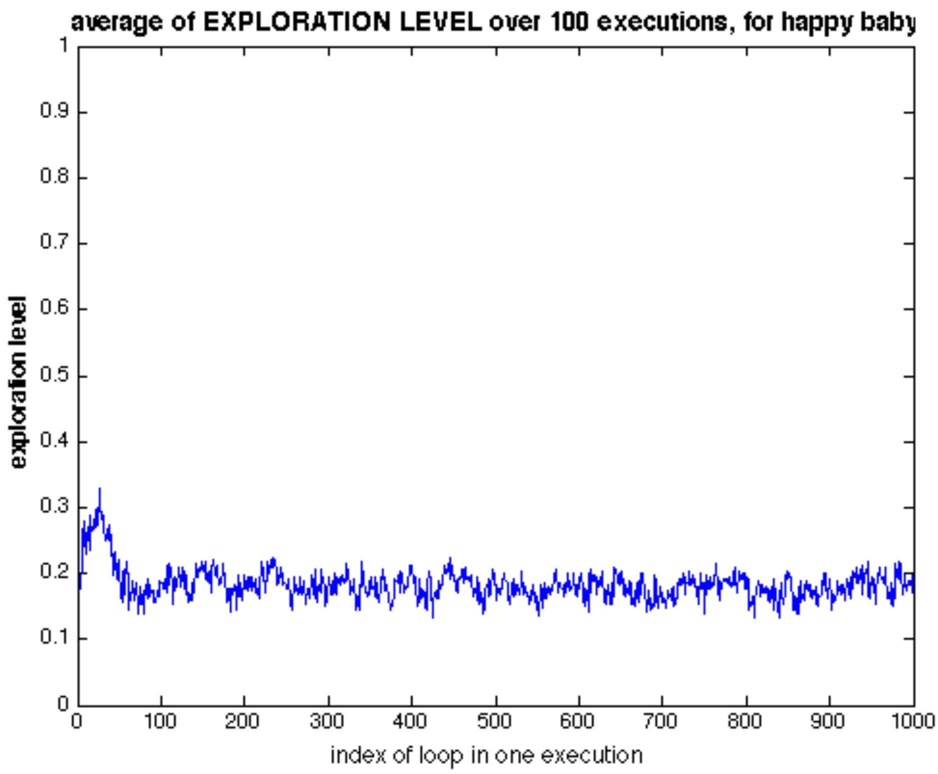

average action percentage in 100 executions, sad baby
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
percentage of occurence
turn
shout
move to
move away
stay
baby's possible actions

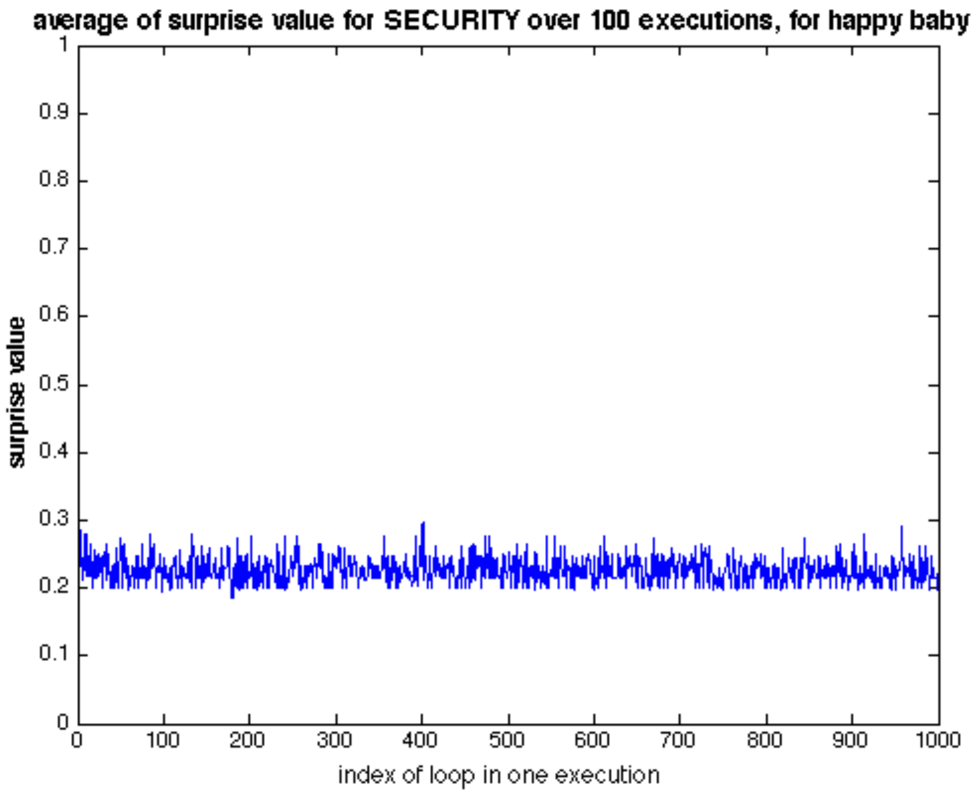

average action percentage in 100 executions, happy baby
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
percentage of occurence
turn
shout
move to
move away
stay
baby's possible actions

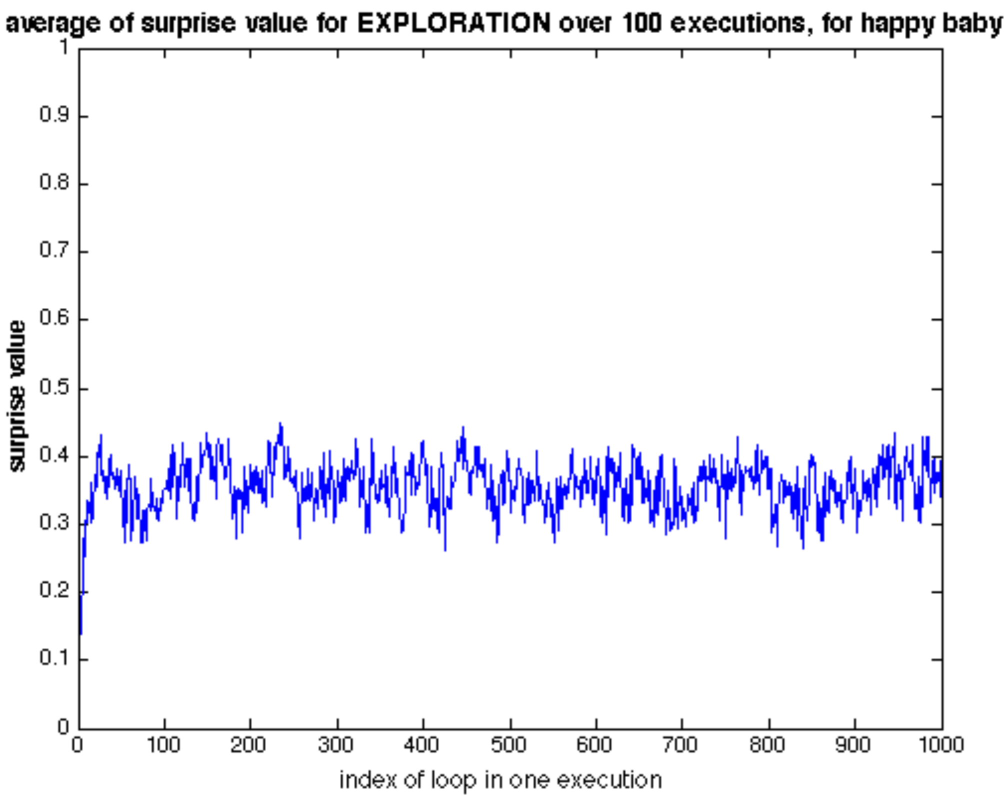

average action percentage in 100 executions, sad baby
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
percentage of occurence
turn
shout
move to
move away
stay
baby's possible actions

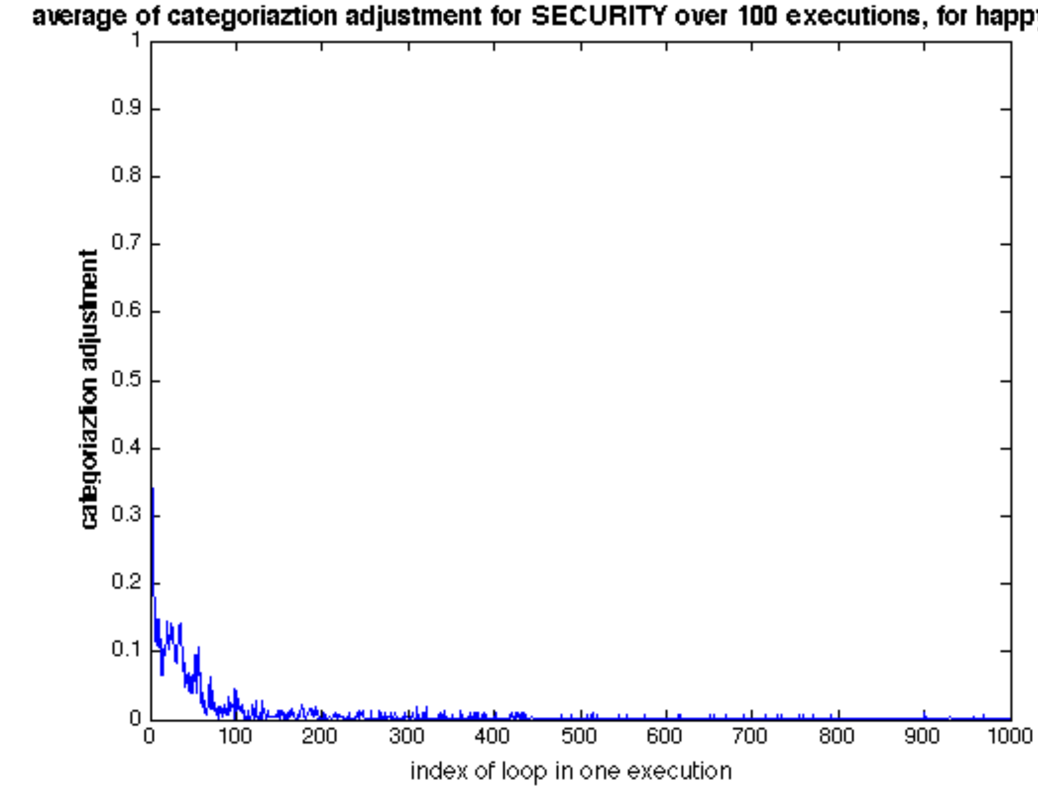
- Fifth group (page 39-40, 6 figures)

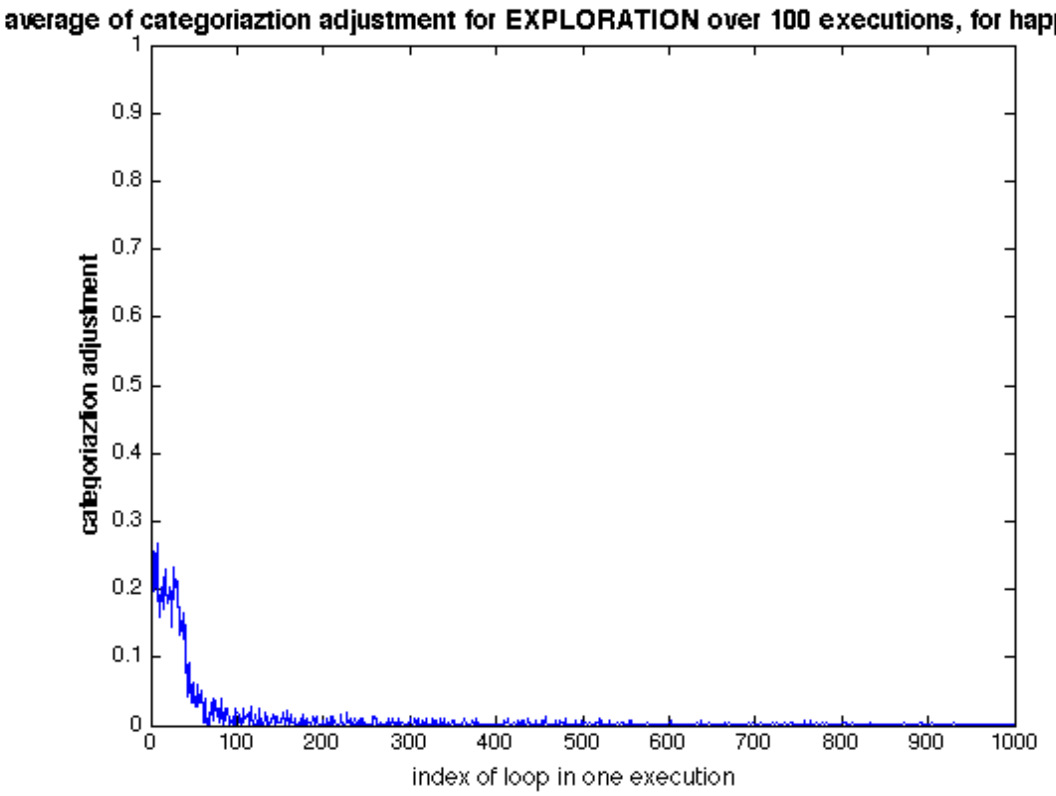





average of surprise value for SECURITY over 100 executions, for happy baby
surprise value
index of loop in one execution
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
100
200
300
400
500
600
700
800
900
1000


average of surprise value for EXPLORATION over 100 executions, for happy baby
surprise value
index of loop in one execution
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
100
200
300
400
500
600
700
800
900
1000


average of categoriaztion adjustment for SECURITY over 100 executions, for happy baby
categoriaztion adjustment
index of loop in one execution
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
100
200
300
400
500
600
700
800
900
1000


average of categoriaztion adjustment for EXPLORATION over 100 executions, for happy ba
categoriaztion adjustment
index of loop in one execution
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
100
200
300
400
500
600
700
800
900
1000