



EÖTVÖS LORÁND UNIVERSITY
FACULTY OF INFORMATICS
DEPARTMENT OF SOFTWARE
TECHNOLOGY AND METHODOLOGY

Analytical tools for the evaluation of progress and entertainment level during educational games

Máté CSÁKVÁRI

Software Information Technology MSc

1. semester

Author

Zoltán TŐSÉR

Software Information Technology MSc

3. semester

Author

Dr. habil. András LŐRINCZ

Head senior researcher

Supervisor

Budapest, 2012

Contents

1	Theses	1
2	Introduction	3
3	Methods	5
3.1	Handorino	5
3.1.1	The frame game: a version of Ms. Pac-Man	5
3.1.2	The mini games	7
3.2	Constrained Local Models	8
3.2.1	Principal Component Analysis	11
3.2.2	Point Distribution Model	11
3.2.3	Formalization of Constrained Local Models	12
3.2.4	Expectation Maximization for CLM	12
3.3	Preprocessing of the data	14
3.4	Hierarchical CLM	14
3.5	Databases	15
3.5.1	Cohn-Kanade Facial Expression Database	15
3.6	Tests and Participants	16
3.7	Procedure	17
4	Results	18
4.1	Cumulated results	18
4.2	Behavioral studies	19
4.3	Performance distribution	22
4.4	Correlation analysis	24
5	Discussion	27
5.1	The effect of the frame game on performance	27
5.2	Identifying different strategies during mini games	28
5.3	Real-time feedback and frustration detection	28

5.4	Correlation analysis	29
5.4.1	Covariance matrix	29
5.4.2	Principal component analysis and independent component analysis	29
6	Future work	31
7	Conclusion	32
A	Face tracking software	33
A.1	Overview	33
A.2	Structure of the C++ software	34
A.3	Models and training examples	34
B	Structure of the 3D Pac-Man game	37
B.1	Overview	37
B.2	The layered architecture	37
B.3	Subsystems	39
B.3.1	The multi-agent system	39
B.3.2	The mini games	39
B.3.3	The logging system	40
B.3.4	Networking	41

Abstract

Playful learning is an important technique for making children motivated in developing their cognitive skills. We have studied a set of educational mini games teaching inductive reasoning. The game has been embedded into a frame game that may change the stress level and the entertainment level. Logs of the games were collected and children were monitored during the game using webcams. The collected videos were analyzed in the context of the game. We developed and used a constrained local model in C++ for the analysis of head pose, facial expression, and eye movement variations. Preliminary results are presented about the diverse behaviours of the children.

Chapter 1

Theses

We have worked out the software components and the methodology for the estimation of facial expressions within context. We can track facial expressions in real time and can provide real time feedback during task execution. Our work is limited to educational games, but it enables the learning of individual facial expressions and the association to the hidden emotions of the user, including the basic emotions, such as anger, happiness, surprise, etc., and the non-basic emotions, such as doubt, curious or gambling behaviors among many others. Below, we summarize our particular statements that can serve as starting points for a large body of future research:

Thesis 1: Frame game creates sub Gaussian distribution

Frame game embedding of the development of inductive reasoning for young children changes the characteristics of distribution. We have found the expected broadening of the distribution of children's performance when frame games were added. We found that the distribution from approximately Gaussian with mini games only changes to sub Gaussian of approximately square like shape when the mini games are embedded into the frame game. In turn, we claim that performances in mini games and in frame games are not independent. Instead, those, who like the mini games probably dislike the frame game and vice versa. This hypothesis will be checked by questionnaires.

Thesis 2: Better strategies can be identified during the mini games

There is a large variety of performances over the different development tasks on inductive reasoning. Special cases of this variety may indicate talents or special needs. The strategies of solving these tasks are of special interests. We have found that those children who click more show better performance. We hypothesize that these children

are willing to reconsider their decisions and are more willing to correct their solutions. Further data mining is required to support this assumption.

Thesis 3: Computerized Internet based testing and development can be improved by detecting frustration that makes real time feedback possible

Some of the inductive reasoning developmental tasks are too hard for some children. Such case gives rise to disappointment and frustration and they are clearly shown in the videos that we took during the lessons. We suggest to use feedback and help the child under such conditions, alike to 'face-to-face' developments when the teacher can interact with the child. Our software tools enable the detection of the uncertainty of the child and, in turn, the task dependent help could be provided. As a particular example, we found a special distribution in one of the tasks, which frustrated the boys and was relatively easy for girls.

Chapter 2

Introduction

Automatic detection of human emotions based on visual information and human–computer collaboration have been in the focus of research interest. However, a number of challenges are still present before this technology could enter our everyday life. Common devices of households, such as laptops and smart phones, refrigerators, washing machines, etc., usually do not come with high quality cameras, and even worse, real-time analysis with high computational power is a must for timely responses. On the other hand, in a WiFi’ed home it is sufficient to have connection to *a* computer, or – under proper bandwidth conditions – to the cloud. Our European ICTLabs project titled ‘Playful learning on the cloud’ targets this case via mobile devices.

Beyond the hardware constraints there are algorithmic and software requirements, too. Emotions are tricky, since they come in a large individual variety and may also depend on culture. Furthermore,

- emotion understanding is hard for some of us, namely for autistic people, easier for males and females are best at detecting and interpreting such socially relevant signaling.
- emotion understanding is poor if only facial expressions are considered, body talk, prosody help us to disambiguate this type of information.

Having such limitations, we designed a special method and developed a software that can be used for benchmarking. We can gain additional information if we collect emotional signs covering a larger time interval *in context*. Context here means that we are trying to restrict the observable external world of the user to the computer and we generate the content on the screen. If this content is such that it produce emotional reactions of different kinds then we can interpret the emotional ‘output’ of the user. This way we might be able to characterize the emotional state of the user under conditions when we do not know the context.

We chose the educational games as our specific benchmark problem. In order to study this problem we developed a number of software tools:

- Developments:**
1. We have adapted the *Molnár mini game set* that makes use of Klauer's method on the development of inductive reasoning and implemented these paper-pencil games to the Internet.
 2. We took part in the Java adaptation of the Ms. Pac-Man like frame game called Handorino that can be used over the Internet, too.
 3. We took part in the development of a C++ code capable of real time facial expression estimation.
 4. We extended this software by an iris monitoring part in order to follow eye movements during task solving.
 5. We used this software for real time feedback for the children. The software measured the head pose of the child and animated an avatar when the child turned away from the computer screen.
 6. We developed the software for the Internet based management of starting and finishing the frame game – mini game system.

Tests: We cooperated with researchers of Szeged University in test execution, bug fixing during the tests, and data collection.

Evaluation: We started to evaluate the wealth of information provided by webcam. Our theses contain these findings.

This work is organized as follows. First, we describe our methods, including the algorithms and basic components of facial expression estimation and games, respectively. Details of the software components will be presented elsewhere. This method chapter is followed by the description of our results. The section on discussion contains some considerations on future works since we think that this is the beginning of a beautiful 'projectship'.

Acknowledgments

We are grateful to a number of people, who contributed to this work, including Tamás Nagy for leading the C++ development of the constrained local model, András Dudás for his thoughtful planning and supervision on the development of the Handorino game and the mini games, Dr. Gyöngyvér Molnár, Attila Rausch, Csaba Dobó for their careful work at the Arany János School of Szeged and to Ágnes Kissné-Gera, the director of the school as well as to the teachers and system administrator of this school for their special care and efforts during the execution of the tests.

Chapter 3

Methods

In this section we describe the methods used in the experiments. First, we present the game types that we developed. There are two of them, a frame game that can leverage excitement in the game and a set of mini games that serve the training of inductive reasoning.

Previous studies showed the strong positive effect of the mini games on inductive reasoning. Here, we were interested in the effect of the frame game. We used different tests to meet our goals. Before starting the games a pre-test was taken by the children. Players were organized into two groups of similar performance distribution. Then one of the two groups played both the frame game and the mini games, whereas the other group played the mini games only. After the training session a post test was administered.

We logged the events of the games, utilized an avatar to help attention, and measured head pose and facial expression changes during game play. Pose variations and facial expressions were tracked by our software. The software was able to track eye motion as well. Here, we present the first results, the wealth of collected information requires further considerations.

3.1 Handorino

Handorino is an educational game aimed at young children. It consists of two components, the frame game and various mini games.

3.1.1 The frame game: a version of Ms. Pac-Man

The video-game Pac-Man was first released in 1979, and reached immense success. It is considered to be one of the most popular video games to date [11].

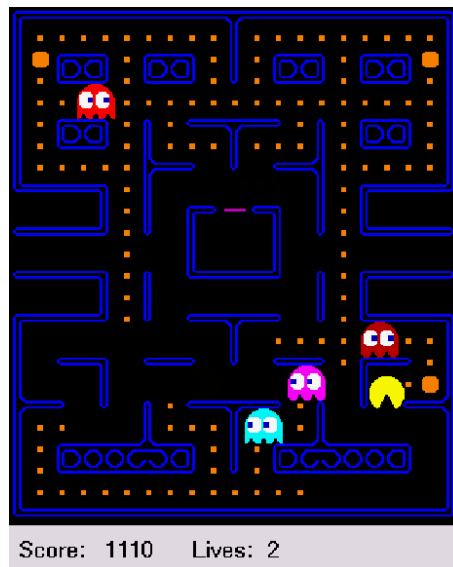


Figure 3.1: A snapshot of the Pac-Man game

The player maneuvers Pac-Man in a maze (see Fig. 3.1), while Pac-Man *eats* the dots in the maze. In this particular maze there are 174 dots, each one is worth 10 points. A level is finished when all the dots are eaten. To make things more difficult, there are also four *ghosts* in the maze who try to catch Pac-Man, and if they succeed, Pac-Man loses a life. Initially, he has three lives, and gets an extra life after reaching 10,000 points.

There are four power-up items in the corners of the maze, called *power dots* (worth 40 points). After Pac-Man eats a power dot, the ghosts turn blue for a short period (15 seconds), they slow down and try to escape from Pac-Man. During this time, Pac-Man is able to eat them, which is worth 200, 400, 800 and 1600 points, consecutively. The point values are reset to 200 each time another power dot is eaten, so the player would want to eat all four ghosts per power dot. If a ghost is eaten, he remains hurry back to the center of the maze where the ghost is reborn. At certain intervals, a fruit appears near the center of the maze and remains there for a while. Eating this fruit is worth 100 points.

In the original version of Pac-Man, ghosts move on a complex but deterministic route, so it is possible to learn a deterministic action sequence that does not require any observations. Many such *patterns* were found by enthusiastic players. In most of Pac-Man's sequels, most notably in *Ms. Pac-Man*, randomness was added to the movement of the ghosts. This way, there is no single optimal action sequence, observations are necessary for optimal decision making. In other respects, game play was mostly unchanged.

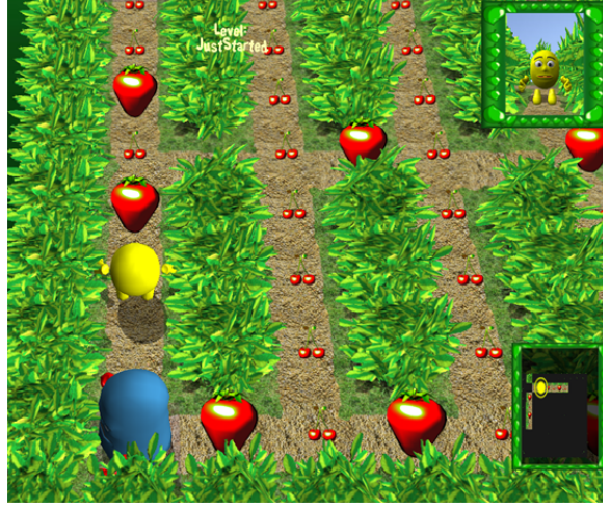


Figure 3.2: A snapshot of our 3D Pac-Man game. Ghost: blue, Pac-Man: yellow. There are two insets. Inset at the top-right corner: facial expression of the (artificial) partner. Inset at the bottom-right corner: part of the labyrinth already seen while moving along the trajectory.

In our implementation, ghosts moved randomly in 20% of the time and straight towards Pac-Man in the remaining 80%, but ghosts may not turn back. To emphasize the presence of randomness, we shall refer to our implementation as a Ms. Pac-Man-clone.

The frame game is a 3D version of the Ms. Pac-Man game. This choice was made in order to include human level intelligence based on reinforcement learning [9]. We changed the traditional visual form of Pac-Man: walls were bushes, dots were cherries and power-dots were strawberries. The number of ghosts could be changed, as well as many other parameters that can influence the level of expertise required to succeed.

Feedback can be provided during the frame game of Handorino: there is a little figure in the top right corner of the screen (Fig. 3.2) that could show facial expression, could utilize body talk, and so on. In the present game, we used this little figure to provide feedback about the pose of the user: the more pose variations were detected the more red this little figure became and the color was relaxed to yellow if pose variation was below threshold.

3.1.2 The mini games

Mini games were developed at the University of Szeged by Gyöngyvér Molnár [7]. The Szeged group led by Benő Csapó is employing Online Diagnostic Assessment System (ODAS) in Hungary. As ODAS is still in experimental phase, no real feedback

information is available yet at the main assessment domains (reading, mathematics and science). However, several online assessments have been carried out to measure the achievements on an inductive reasoning test (see, [2]). Therefore, the development of inductive reasoning by a teaching game was piloted, as a model for further similar computer games at other domains. We developed the Internet version of the original paper tests of Molnár [7] in a way that it could be included into the frame game.

Training is based on Klauer's theory of inductive reasoning [4, 5] and consists of 120 learning tasks integrated into a game, which can be solved through inductive reasoning. To verify the hypothetical assumptions a four-week pilot study was implemented. First grade students constituted the experimental and control group.

The mini games are categorised as follows [7]:

- Generalization
 - Class formation
 - Class expansion
 - Finding common attributes
- Discrimination
- Cross-classification
- Recognizing relations
 - Series completion
 - Series ordering
 - Simple analogy
- Discriminating relations
- System formation

An example mini game is shown in the Figure series 3.3-3.5. There were 120 training problems. One quarter of the problems were manipulative requiring e.g. colorful building blocks, Dienes' logical set, matches, and so on [7].

3.2 Constrained Local Models

In order to enable shape recognition for parametric models, the prepared (also called annotated) database is inevitable for the optimization of the parameters called learning. In our approach, a sample means an image annotated with landmark points. These landmark points (also called features) are selected so that they characterize the given

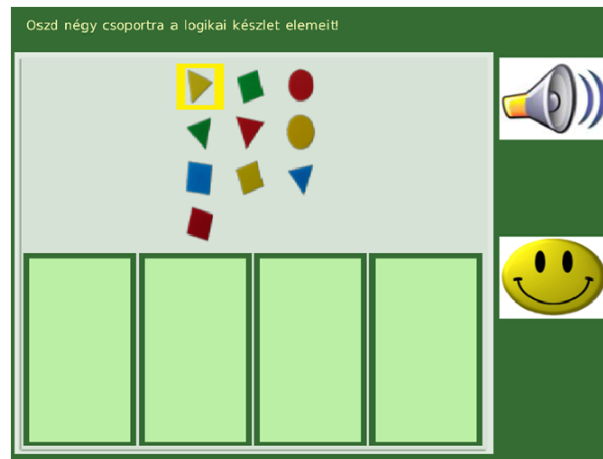


Figure 3.3: A sample mini game. The task is to order the geometric figures according to their features. In some of the tasks multiple solutions are present. The task is told vocally and can be rehearsed by clicking on the knob representing the loud speaker.

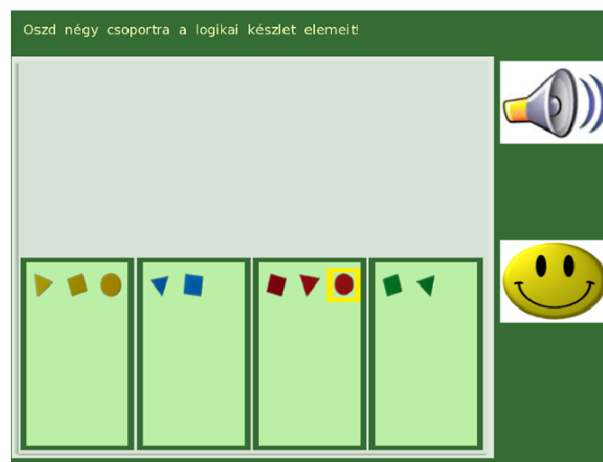


Figure 3.4: The sample mini game of Fig. 3.3 after proper ordering.



Figure 3.5: Feedback upon proper solution of sample mini game of Fig. 3.3.

shape well. For example, a landmark may be the outer eye corner, the end of the eyebrow, the intersections of the eyelids and the iris etc.

This chapter describes the constrained local models (CLM) that includes point distribution models (PDM), which is based on principal component analysis (PCA). First, we estimate 3D landmark positions on face images using the CLM method. We describe the details of this technique in this Section.

CLM methods are generative parametric models for person-independent face alignment. We developed a *C++* software that makes use of the 3D CLM method. The shape model is defined by a 3D mesh and in particular the 3D vertex locations of the mesh, called landmark points. Consider the shape of a 3D CLM as the coordinates of 3D vertices of the M landmark points:

$$\mathbf{x} = (x_1, y_1, z_1, \dots, x_M, y_M, z_M)^T, \quad (3.1)$$

or, $\mathbf{x} = (\mathbf{x}_1^T, \dots, \mathbf{x}_M^T)^T$, where $\mathbf{x}_i = (x_i, y_i, z_i)^T$. We have N samples: $\{\mathbf{x}^{(n)}\}_{n=1}^N$. CLM models assume that – apart from the global transformations; scale, rotation, and translation – all $\{\mathbf{x}^{(n)}\}_{n=1}^N$ can be approximated by means of the linear principal component analysis (PCA) forming the PCA subspace.

Next, we describe the PCA method utilized by the 3D Point Distribution Model (PDM) and the CLM itself. Then we elaborate on the estimation of the positions of the landmarks that applies expectation maximization.

3.2.1 Principal Component Analysis

Assume that sample points $\mathbf{x}(t) \in \mathbb{R}^D, t = 1, \dots, T$ are given. The task is to find the d -dimensional subspace $\mathbf{A} \in \mathbb{R}^{D \times d}$ such that

$$f(\mathbf{A}) = E \left[\|\mathbf{x} - \text{proj}_{\mathbf{A}}(\mathbf{x})\|^2 \right] \quad (3.2)$$

is minimal. Here $E[\cdot]$ denotes the expectation value taken according to the underlying distribution of the samples. The d pieces of the eigenvectors belonging to the largest eigenvalues of matrix $\mathbf{A}$ form the best estimation for Gaussian noise:

$$\mathbf{\Lambda} = \text{diag}(\lambda_1; \dots; \lambda_d) \in \mathbb{R}^{d \times d}, \quad \mathbf{\Phi} = \begin{bmatrix} \mathbf{\Phi}_1 \\ \vdots \\ \mathbf{\Phi}_M \end{bmatrix} \in \mathbb{R}^{3M \times d}, \quad (3.3)$$

where $\mathbf{\Phi}_i$ ($i = 1, \dots, M$) is a $3 \times d$ piece in $\mathbf{\Phi}$ that correspond to the landmarks and columns of $\mathbf{\Phi}$ are eigenvectors of matrix $\mathbf{A}$.

3.2.2 Point Distribution Model

The 3D point distribution model (PDM) describes non-rigid shape variations linearly and composes it with a global rigid transformation, placing the shape in the image frame:

$$\mathbf{x}_i(\mathbf{p}) = s\mathbf{P}\mathbf{R}(\bar{\mathbf{x}}_i + \mathbf{\Phi}_i\mathbf{q}) + \mathbf{t}, \quad (3.4)$$

where $i = 1, \dots, M$, $\mathbf{x}_i(\mathbf{p})$ denotes the 2D location of the i^{th} landmark subject to transformation $\mathbf{p}$, and $\mathbf{p} = \{s, \alpha, \beta, \gamma, \mathbf{q}, \mathbf{t}\}$ denotes the parameters of the model, which consist of a global scaling s , angles of rotation in three dimensions ($\mathbf{R} = \mathbf{R}_1(\alpha)\mathbf{R}_2(\beta)\mathbf{R}_3(\gamma)$), translation $\mathbf{t}$ and non-rigid transformation $\mathbf{q}$. Here $\bar{\mathbf{x}}_i$ is the mean location of the i^{th} landmark averaged over the database, i.e. $\bar{\mathbf{x}} = [\bar{\mathbf{x}}_1^T; \dots; \bar{\mathbf{x}}_M^T]$, $\bar{\mathbf{x}}_i = [\bar{x}_i, \bar{y}_i, \bar{z}_i]^T$, $\bar{x}_i = \frac{1}{N} \sum_{n=1}^N x_i^{(n)}$, and similarly, for $\bar{y}_i$ and $\bar{z}_i$. Matrix $\mathbf{\Phi}_i$ ($i = 1, \dots, M$) is a $3 \times d$ piece in $\mathbf{\Phi} \in \mathbb{R}^{3M \times d}$ and corresponds to the landmarks. Columns of $\mathbf{\Phi}$ form the orthogonal projection matrix of principal component analysis and its compression dimension is d . Finally, matrix $\mathbf{P}$ denotes the projection matrix to 2D:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (3.5)$$

and thus $\mathbf{x}_i(\mathbf{p}) \in \mathbb{R}^2 (\forall i)$.

Gaussian assumption on these parameters justify PCA on the $\mathbf{x}_i$ points and we get an estimate of the prior of the parameters:

$$p(\mathbf{p}) \propto N(\mathbf{q}; \mathbf{0}, \mathbf{\Lambda}), \quad (3.6)$$

that is CLM assumes a normal distribution with $\mathbf{0}$ mean and $\mathbf{\Lambda}$ variance for parameters $\mathbf{q}$. $\mathbf{\Lambda} = \text{diag}(\lambda_1; \dots; \lambda_d) \in \mathbb{R}^{d \times d}$ in (3.6) is provided by the PCA and the parameter vector assumes the form $\mathbf{p} = [s; \alpha; \beta; \gamma; \mathbf{t}; \mathbf{q}]$.

3.2.3 Formalization of Constrained Local Models

CLM is constrained through the PCA of PDM. It works with *local experts*, whose opinion is considered independent and are multiplied to each other:

$$J(\mathbf{p}) = p(\mathbf{p}) \prod_{i=1}^M p(l_i = 1 | \mathbf{x}_i(\mathbf{p}), \mathcal{I}), \quad (3.7)$$

where $l_i \in \{-1, 1\}$ is a stochastic variable, which is 1 (-1) if the i^{th} marker is (not) in its position, $p(l_i = 1 | \mathbf{x}_i(\mathbf{p}), \mathcal{I})$ is the probability that for image $\mathcal{I}$ and for marker position $\mathbf{x}_i$ determined by parameter $\mathbf{p}$, the i^{th} marker is in its position.

Local experts are built on the well known *multi-layer perceptrons* (MLPs). MLPs were trained on labeled samples of the neighborhoods of the landmarks (Fig. 3.6).

Local expert's response – that depend on the constraints of the PDM and the response map of the local expert in an appropriate neighborhood – can be used to express $J(\mathbf{p})$ in (3.7) (Fig. 3.7):

$$J(\mathbf{p}) = p(\mathbf{p}) \prod_{i=1}^M \sum_{\mathbf{y}_i \in \Psi_i} p(l_i = 1 | \mathbf{y}_i, \mathcal{I}) p(\mathbf{y}_i | \mathbf{x}_i) \quad (3.8)$$

where CLM assumes Gaussian noise that spoils the true position of the marker points. For the i^{th} marker point we write

$$\mathbf{y}_i = \mathbf{x}_i + \epsilon_i \quad (3.9)$$

$$\epsilon_i \sim \mathcal{N}(\mathbf{0}, \rho \mathbf{I}) \quad (3.10)$$

$$(3.11)$$

and ρ assumes the following form $\rho = \sum_{i=d+1}^{3M} \frac{\lambda_i}{(3M-d)}$ where λ_i is the i^{th} eigenvalue of $Cov(\mathbf{x})$, the covariance matrix of stochastic variable $\mathbf{x}$. These equation can be derived by Bayes'rule and via the tacit assumption [8] that $\sum_{\mathbf{y}_j \in \Psi_i} \mathcal{N}(\mathbf{y}_j; \mathbf{x}_i, \rho \mathbf{I})$ is a weak function of the parameters to be optimized.

3.2.4 Expectation Maximization for CLM

Expectation maximization (EM) can be used to find the (local) optimum. For the E-step, the probability of parameter $\mathbf{p}$ given the current parameter estimation $\mathbf{q}^{(t)}$ is

$$Q(\mathbf{p} | \mathbf{p}^{(t)}) = E_{q(y)} \left[-\log \left\{ p(\mathbf{p}) \prod_{i=1}^M p(l_i = 1, \mathbf{y}_i | \mathbf{x}_i(\mathbf{p}), \mathcal{I}) \right\} \right]$$



Figure 3.6: Landmarks on the face.

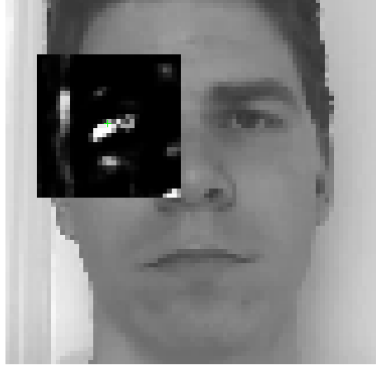


Figure 3.7: Response map for the right corner of the right eye.

where

$$q(y) = \prod_{i=1}^M p(\mathbf{y}_i | l_i = 1, \mathbf{x}_i(\mathbf{p}^{(t)}), \mathcal{I})$$

and thus

$$Q(\mathbf{p} | \mathbf{p}^{(t)}) \propto \|\mathbf{q}\|_{\Lambda^{-1}}^2 + \sum_{i=1}^M \sum_{\mathbf{y}_i \in \Psi_i} \frac{w_{\mathbf{y}_i}}{\rho} \|\mathbf{x}_i - \mathbf{y}_i\|^2$$

where Ψ_i is the set of pixel positions of the i^{th} response map and

$$w_{\mathbf{y}_i} = p(\mathbf{y}_i | l_i = 1, \mathbf{x}_i, \mathcal{I}) = \frac{p(l_i = 1 | \mathbf{y}_i, \mathcal{I}) \mathcal{N}(\mathbf{y}_i; \mathbf{x}_i, \rho \mathbf{I})}{\sum_{\mathbf{z}_i \in \Psi_i} p(l_i = 1 | \mathbf{z}_i, \mathcal{I}) \mathcal{N}(\mathbf{z}_i; \mathbf{x}_i, \rho \mathbf{I})} \quad (3.12)$$

For the parameter optimization M-step the task is to minimize $Q(\mathbf{p} | \mathbf{p}^{(t)})$, i.e.,

$$Q(\Delta \mathbf{p} | \mathbf{p}) = \|\mathbf{p} + \Delta \mathbf{p}\|_{\tilde{\Lambda}^{-1}}^2 + \sum_i \sum_{\mathbf{y}_i \in \Psi_i} \frac{w_{\mathbf{y}_i}}{\rho} \|\mathbf{x}_i + \mathbf{J}_i \Delta \mathbf{p} - \mathbf{y}_i\|^2 \rightarrow \min_{\Delta \mathbf{p}} \quad (3.13)$$

giving rise to

$$\Delta \mathbf{p} = - \left(\rho \tilde{\Lambda}^{-1} + \mathbf{J}^T \mathbf{J} \right)^{-1} \left(\rho \tilde{\Lambda}^{-1} \mathbf{p} - \mathbf{J}^T \mathbf{v} \right)$$

where $\mathbf{J}$ is the Jacobian of the cost function, $\tilde{\mathbf{\Lambda}} \in \mathbb{R}^{(6+d) \times (6+d)}$ is the diagonal matrix with the PDM eigenvalues and with leading zeros in its diagonal (the extension of the $\mathbf{\Lambda}$ diagonal matrix with as many leading 0s in its diagonal as many rigid parameters we have, which is 6 in our case), $\mathbf{v}$ is concatenated from the mean shift values $\mathbf{v} = [\mathbf{v}_1; \dots; \mathbf{v}_M]^T$ where

$$\mathbf{v}_i = \left(\sum_{\mathbf{y}_i \in \Psi_i} \frac{p(l_i = 1 | \mathbf{y}_i, \mathcal{I}) \mathcal{N}(\mathbf{x}_i; \mathbf{y}_i, \rho \mathbf{I})}{\sum_{\mathbf{z}_i \in \Psi_i} \pi_{\mathbf{z}_i} \mathcal{N}(\mathbf{x}_i; \mathbf{z}_i, \rho \mathbf{I})} \mathbf{y}_i \right) - \mathbf{x}_i. \quad (3.14)$$

and $\pi_{\mathbf{z}_i}$ is the shorthand for $p(l_i = 1 | \mathbf{z}_i; \mathcal{I})$.

3.3 Preprocessing of the data

We utilized different heuristics for CLM that we list below

Contrast normalization In real videos the range of pixel intensities can vary. It may occur that on the scale 0-to-255 pixel intensities of a large part of the face fall within the interval 245-255. Local contrast normalization methods were used to compensate for this problem

Procrustes transformation For any shape $\mathbf{x}$, Procrustes transformation applies translation, uniform scaling and rotation to match the reference shape $\mathbf{x}_r$ in Euclidean norm. The minimum of this cost function is called the Procrustes distance. We applied it in our 2D transformations.

Constraining to emotion space Our method takes advantage of learning from samples. Parameters of the distribution are also estimated from the samples. However, the distribution computed by varying the faces and the emotions is very different from the distribution computed by varying the emotions for a given face, i.e., by limiting the parameters to the emotion space and then computing the distribution for all faces with this constraint. The former method allows for transformations between faces, whereas the latter does not. The latter, however, requires the computation of the rigid for of the actual face in real time in order to ‘enter’ the space of emotions. This real time estimation was also applied

3.4 Hierarchical CLM

Eye motion provides information about the actual state of the mind, including intentions. Our software can apply CLM models in order by going from low resolution to higher resolutions locally. This way, we utilize the following step-by-step approach

Face localization We use Viola-Jones face detector as the first step to localize the face

Face CLM The CLM is applied for a face down-scaled to approximately 60 by 60 pixels. We find the position of the most important marker points. Precision is sub-pixel, tracking range as a function of head pose is larger than the detection range of the Viola Jones method.

Eye CLM We have developed CLMs for the eye region including the iris in order to estimate eye motions. Iris markers can be part of the face CLM or a smaller CLM that tracks the eye can be invoked during facial expression tracking.

3.5 Databases

In our simulations we used the Cohn-Kanade Extended Facial Expression (CK+) Database [6] and the BU-4DFE dataset [12].

3.5.1 Cohn-Kanade Facial Expression Database

This database was developed for automated facial image analysis and synthesis and for perceptual studies. The database is widely used to compare the performance of different models. The database contains 123 different subjects and 593 frontal image sequences. The original Cohn-Kanade Facial Expression Database distribution [3] had 486 FACS-coded sequences from 97 subjects. CK+ has 593 posed sequences with full FACS coding of the peak frames. A subset of action units were coded for presence or absence. To provide more ground-truth data for the original database, the RPI ISL research group manually re-annotated the original CK dataset [10] frame-by-frame. This dataset is referred to as the CK Enhanced Dataset.

The BU-4DFE Dynamic Facial Expression Database

We carefully marked the BU-4DFE dataset [12] the same way the CK+ dataset is marked. The BU-4DFE dataset is a high-resolution 3D dynamic facial expression database. It contains 3D video sequences of 101 different subjects, with a variety of ethnic/racial ancestries. Each subject was asked to perform six prototypic facial expressions (anger, disgust, happiness, fear, sadness, and surprise), therefore the database contains 606 3D facial expression sequences.

3.6 Tests and Participants

Three consecutive tests were employed. The first test was run before our studies. The goal was to measure the Internet based training of inductive reasoning and to compare it to paper and pencil tests. Results showed that the Internet based test gives rise to similar advances as the paper-pencil tests [1]. Software for the Internet based experiment was developed by Gergely Nagy and Judit Sebők of the Eötvös Loránd University during their MSc Thesis work at the Faculty of Informatics. They developed a Silverlight implementation.

The second test set was executed during the second semester of academic year 2011-2012. This test used mini games embedded into the Handorino frame game. We found the same development as before, but very little differences with and without the frame game. Results indicated that learning may have reached a (different) plateau for each child and we decided to repeat the experiment. The third test took part during the first semester of academic year 2012-2013 and was finished before the end of November. In this testing we implemented feedback about their head pose to the children during the game by applying an avatar (see above), so their attention could be better focused. Furthermore, we took detailed logs of performance that enabled a better statistics for the individual developments. We wanted to separate the effects of learning the set of mini games from the effect of transferring this knowledge to the post tests on inductive reasoning. Below, we detail the experimental groups for the experiments that involved game Handorino. The largest portion of this software was developed by András Dudás during his BSc studies at the Eötvös Loránd University.

In the experiment of year 2011/12 the experimental group had 123 first and second grade students, whereas the control group consisted of 137 students having similar background variables. One third (38 students) of the experimental group conducted game-based training in computer-based environment, while the remaining part performed face-to-face training.

In the experiment of year 2012/13 the experimental group had 181 first and second grade students, whereas the control group consisted of 172 students having similar background variables. One third (58 students) of the experimental group conducted game-based training in computer-based environment, while the remaining part performed face-to-face training. The software was modified and improved during the summer of 2012 by András Dudás working closely with one of us (Zoltán Tőser) and by other students, including Balázs Kollár. Experiments were run by researcher at Szeged. Their work was continuously monitored and supervised by one of us (Zoltán Tőser). Zoltán played a major role in fixing software bugs and remotely updating the software. This was also a test of the software. As a result, the software may be deployed at different schools as soon as it can be launched from a browser (work under

progress).

3.7 Procedure

The training execution was divided into 12 sessions, each lasting approximately 45 minutes. In the first and last sessions before and immediately after the training process test-based data collection took place to measure the cognitive dimension of the training-effect. The interval between the pre-test and the post-test was 8 weeks, the period in which the training was performed.

The program consists of 120 tasks (games) which can be solved through the application of appropriate inductive reasoning processes. In the face-to-face form of the training one quarter of the problems were manipulative requiring e.g. colorful building blocks, Dienes's logical set, matches, and so on [7]. Solution of the 120 computerized educational games required the handling of the computer mouse or a touchscreen. Children used headsets and training occurred in a high quality experimental classroom. Each game started by a description of the task and had one or two additional questions that could be required by the children for additional help during game playing.

In terms of the methods and work forms of the training, the students were given the training individually. In face-to-face environment there was a team of implementers, namely the class teachers, who introduced the activities to the children individually. In the computer-based training two facilitators were present in the class-room or in a class-room like computer-lab during the whole training, where students were trained individually, however at the same time in groups of 20. The cognitive dimension of the training-effect was measured by traditional test-based data collection in both situations. The test comprised 37 figural, non-verbal items (Cronbach $\alpha = .87$). Test construction and a precise alignment between the framework and the test ensured validity of the test [7].

Chapter 4

Results

In this section we show our results. First, we present cumulated information about the similarities and differences of the children. Then we show detailed behavioral studies about individual children. At the end of the section we present our result about the performance distribution difference. From now on, we denote the different mini game categories with the following color codes:

Generalization - Class formation	Red
Generalization - Class expansion	Orange
Generalization - Finding common attributes	Yellow
Discrimination	Light Green
Cross-classification	Green
Recognizing relations - Series completion	Cyan
Recognizing relations - Series ordering	Blue
Recognizing relations - Simple analogy	Dark Blue
Discriminating relations	Purple
System formation	Magenta

Figure 4.1: Color codes of the different mini game categories

4.1 Cumulated results

Figure 4.2 shows sample pie charts belonging to various children. Each color represents a mini game type in accordance with figure 4.1. The size of the segments are proportional to the time spent in the corresponding game type.

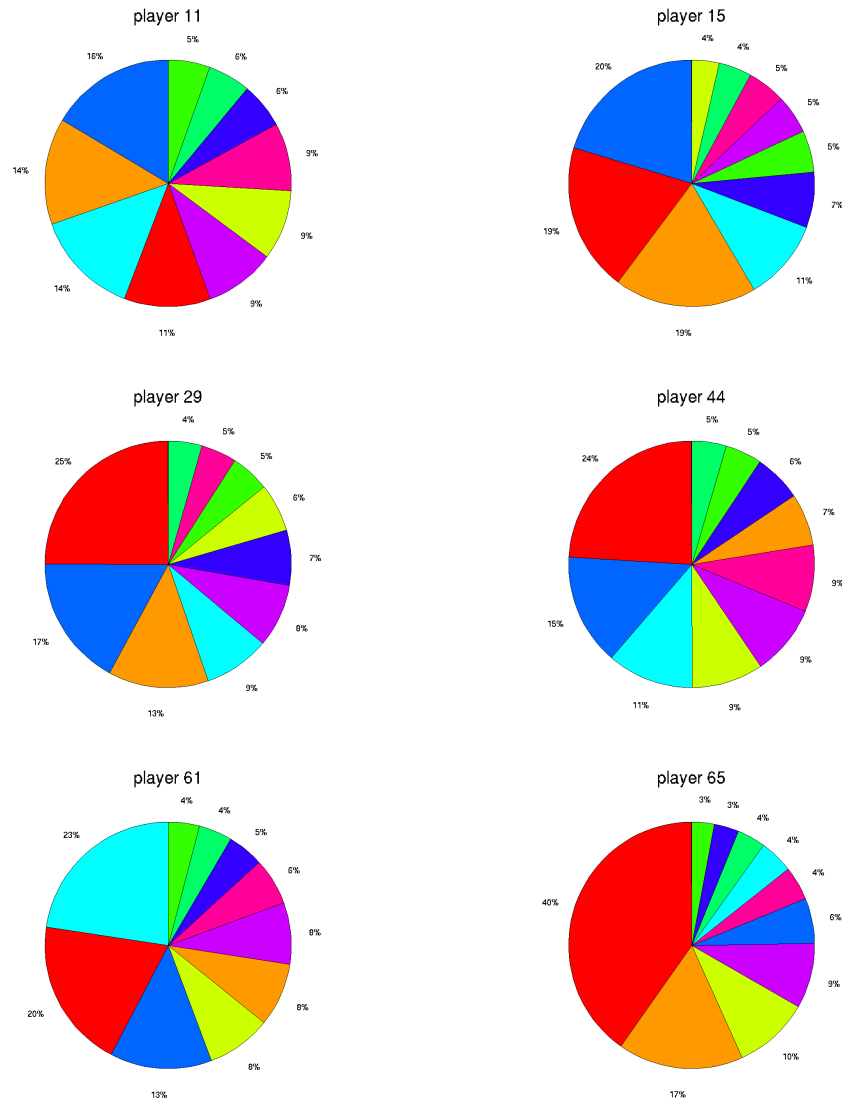
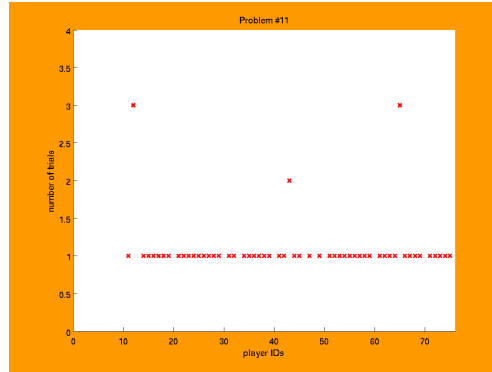


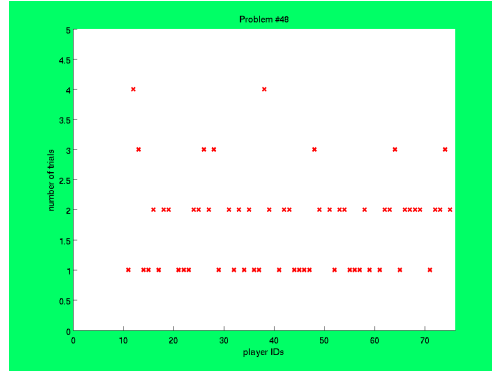
Figure 4.2: Time spent on different types of mini games: normalized data. Game types are color coded.

4.2 Behavioral studies

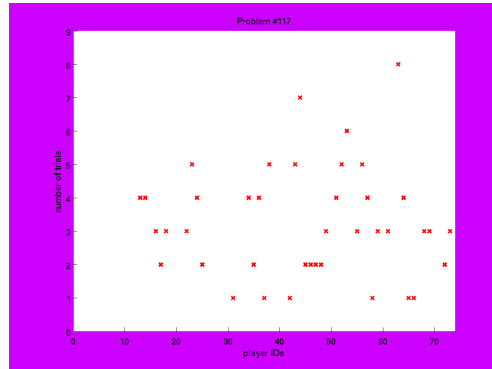
Figure 4.3 shows three types of mini games. The games were categorized by the distribution of total number of trials needed to solve that game. The three categories are easy, medium and hard games.



(a)



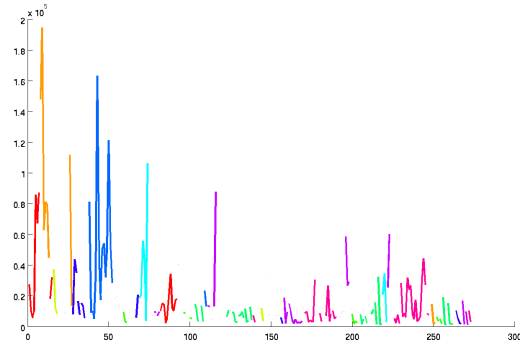
(b)



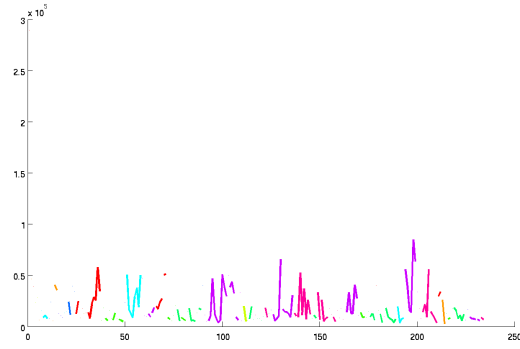
(c)

Figure 4.3: Typical game statistics: (a) easy games with uneven distributions, (b): medium hard games with random trials, (c): hard games with broad distributions of trials. The horizontal axis denotes the IDs of the children, and the vertical axis shows the number of times the children tried to solve the given mini game instance. The background is colored according to 4.1

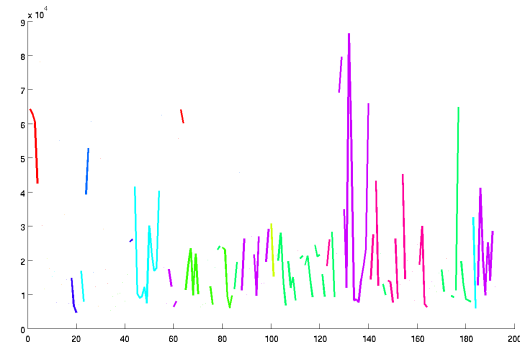
During the experiments, when a child could not solve a mini game on the first



(a)



(b)



(c)

Figure 4.4: Developments from game to game: games of the same game type have identical colors. Each figure belongs to a given child. The horizontal axis denotes different occasions the child tried to solve a mini game, and the vertical axis shows the time spent. Consecutive data points are connected if they belong the same mini game instance.

attempt, she is given the same instance the next time she plays a mini game. Figure 4.4 shows the times and attempts required to solve a mini game. The end point of a continuous line stands for a successful attempt.

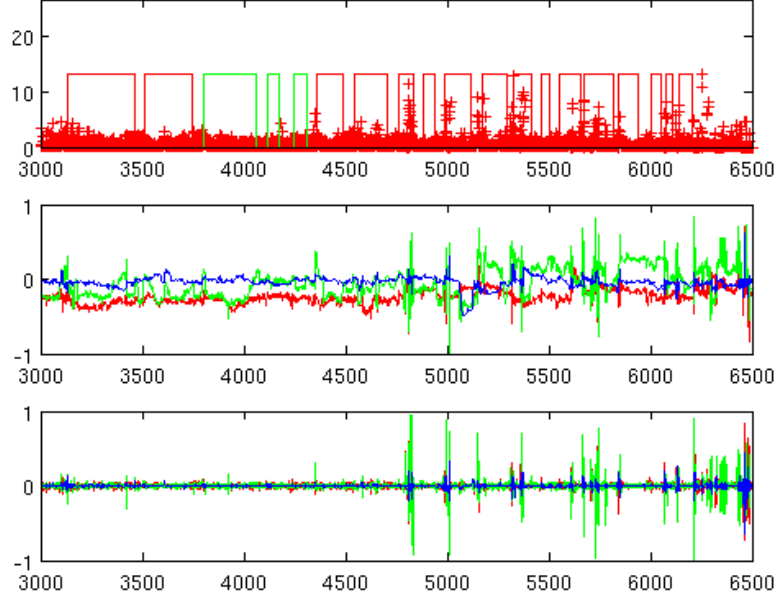


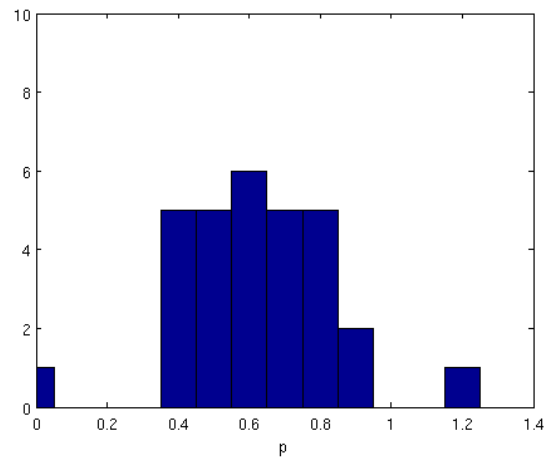
Figure 4.5: Eye movement difference and head pose time series: The top plot represents eye movement variations as a function of frame numbers. The rectangular areas refer to mini game intervals. Green/red colors refer to successful/unsuccessful mini game attempts, respectively. The central plot represents the actual head pose direction measured in radians. Head pose is described as three rotations named pitch(red), roll(green) and yaw(blue). The bottom plot shows the head pose differences between each consecutive frame.

4.3 Performance distribution

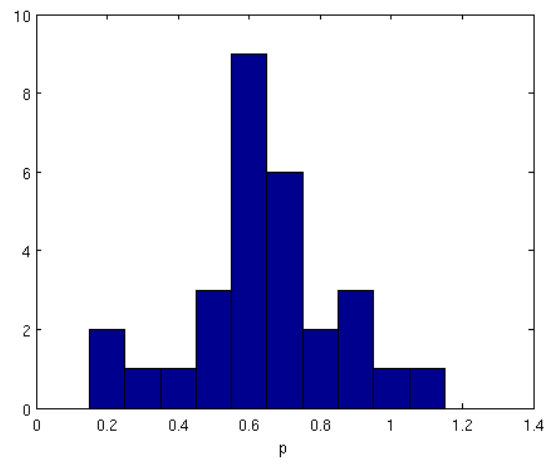
Let $f(child, games) = \sum_{g \in games} \chi(child \text{ solved } g \text{ on the first attempt})$. The performance of the children was measured by computing f on the first and the last third of the mini games and taking their quotient:

$$p(child) = \frac{f(child, games_{81...120})}{f(child, games_{1...40})} \quad (4.1)$$

Figure 4.6 shows the distribution of p in two groups: the children who played the



(a)



(b)

Figure 4.6: Performance distribution difference: (a) children who played the Handorino frame game, (b): children without the frame game

frame game and the ones who played only mini games.

4.4 Correlation analysis

Let $\tau(p, g_i)$ denote the time spent of player p on the i th instance of mini game g . Furthermore, let

$$\Phi_{g,p} = \frac{\sum_{i=1}^{|attempts_{p,g}|} \tau(p, g_i)}{\max_q \sum_{i=1}^{|attempts_{q,g}|} \tau(q, g_i)} \quad (4.2)$$

Figure 4.7 shows the Hinton diagram of the covariance matrix of Φ .

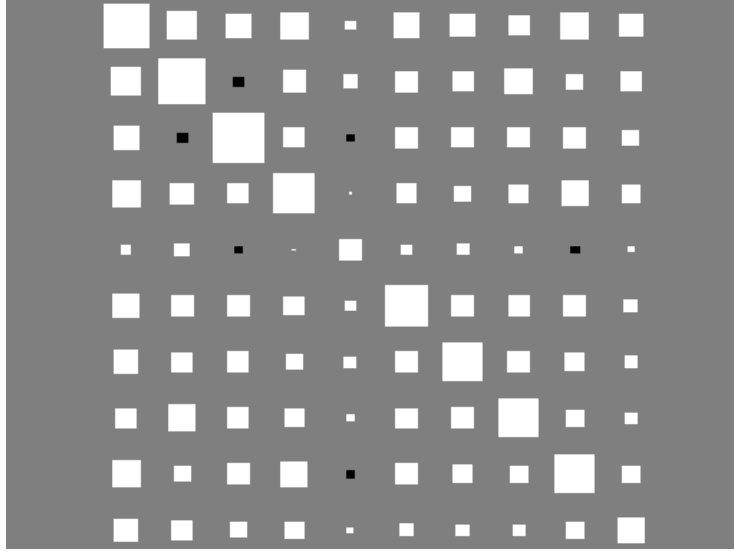


Figure 4.7: The Hinton diagram of $cov(\Phi)$ as defined in formula 4.2. Each (possible degenerate) square represents an element of the matrix. The square's size are proportional to the absolute value of the elements. Negative values have black squares, positive values have white ones. The order of game types is the same as on figure 4.1.

We analysed the $cov(\Phi)$ matrix using principal component analysis. The compression errors as a function of the number of principal components used for compression are shown on figure 4.8. Figure 4.9 shows the Hinton diagram of the principal components.

We also run independent component analysis on $cov(\Phi)$, the results are shown on figure 4.10.

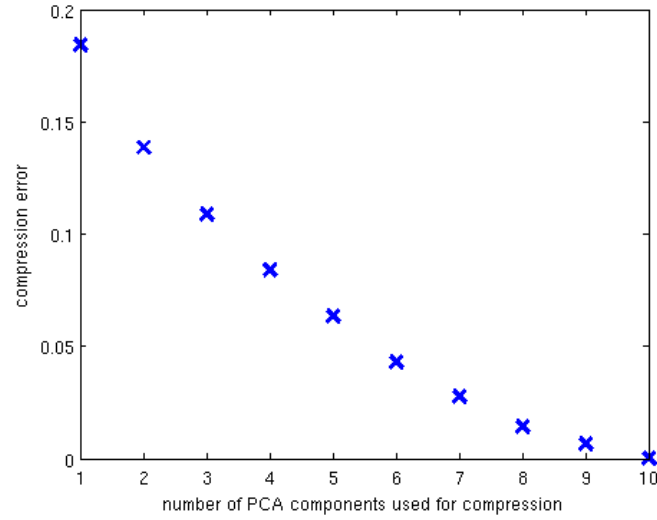


Figure 4.8: The compression errors in function of the number of principal components used for compression. A compression error for n principal components is $\sum_{i=n+1}^{10} \lambda_i$, where the λ values are the eigenvalues returned by the PCA procedure.

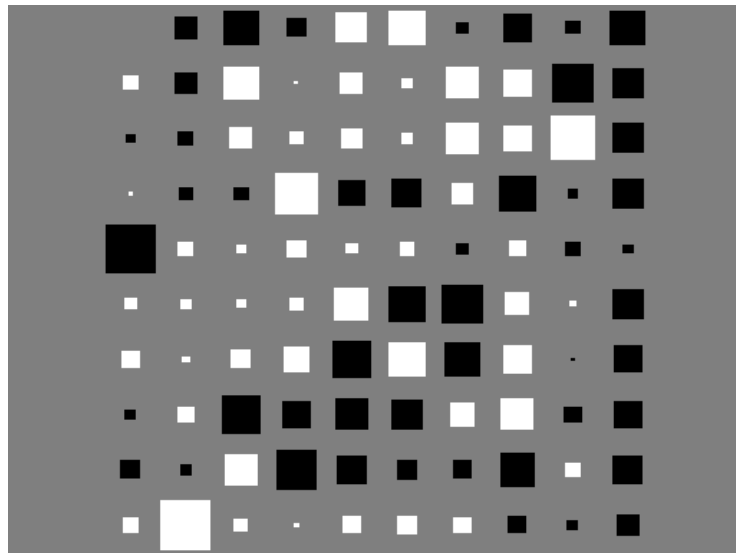


Figure 4.9: The Hinton diagram of the eigenvectors returned by the PCA procedure. Each columns corresponds to a principal component. The vectors are ordered, the rightmost one belongs to the largest eigenvalue.

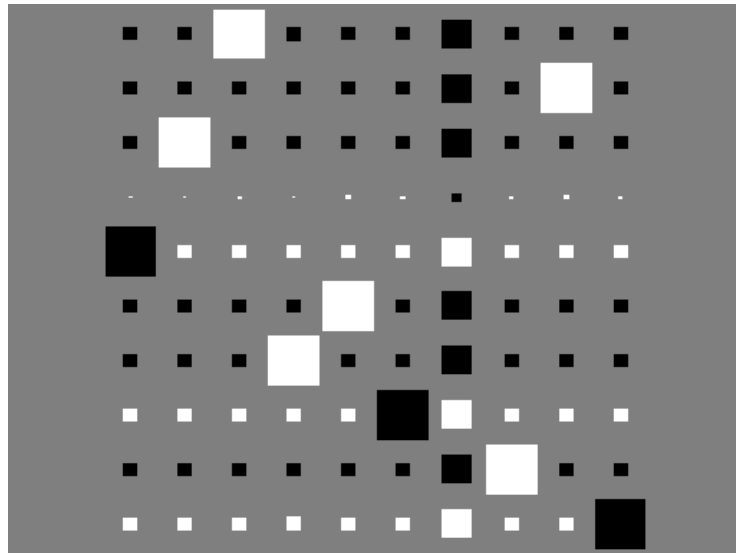


Figure 4.10: Hinton diagram of ICA components of $cov(\Phi)$

Chapter 5

Discussion

5.1 The effect of the frame game on performance

One of the main aspects of this experiment was to find out how the Handorino game affects the children's performance. Prior to evaluating the collected data, we expected that the frame game would motivate most of the children to focus on solving the inductive reasoning tests. This turned out to be false.

Formula 4.1 was defined in order to measure the development of the children. Because some of them tended to guess randomly when they became frustrated, we tried to minimize the effect of this phenomena. Also, the children showed a large variety in the time to come up with a solution. Larger values may not necessarily be worse, they could also originate in different underlying strategies. Because of these reasons, we took neither time, nor attempts other than the first one into account.

With constant inductive test difficulty, a p value of 1 means that over time, a child does not get better in solving mini games. As shown in figure 4.6, most of the p values were below 1, which could indicate that the children got worse. However, the inductive tests tended to get more difficult over time. We interpret this result as children being unable to keep up with the growing difficulty of the tests.

A more interesting aspect of figure 4.6 is the difference in distributions of p between the children with and without the frame game. The charts show that the frame game turns the approximately Gaussian distribution into a sub Gaussian one. In other words, it homogenizes participants.

The exact reason for this is currently unknown, it requires further investigation. Our hypothesis is that the performance in the frame games and the mini games are not independent: children who liked the frame game disliked the mini games and vice versa. We will continue to work on this subject in the future.

5.2 Identifying different strategies during mini games

Children showed many types of different strategies during inductive reasoning tests. Some of them were more successful than others. In order to create an effective recommendation system, we have to identify these strategies and compare them to each other.

The problem with evaluating strategies is that they do not perform homogeneously on all mini games. Some problems are easier to solve with certain strategies, while others may prove to be more difficult. For example, guessing can lead to relatively low solving time in small-size problem spaces, but it will fail to scale well to larger spaces. Therefore, we introduced a classification of the mini games. Figure 4.3 shows representative examples of the three categories, easy, medium and hard. We consider a game easy if almost everybody solves it on the first trial. Hard problems have a large deviation of trials. The intermediate problems are said to be of medium difficulty.

Easy games introduced bias in strategy comparisons, therefore we discarded them for this kind of evaluation. During analyzing the recorded video files from the experiments, we found that children who invested more time into making their final decisions, tended to be more successful. We found a connection between the average number of mouse clicks and performance on medium and hard games: children who clicked more solved more difficult problems with fewer trials.

Let

$$snc(child, games) = \sum_{g \in games} \frac{\sum_{a \in attempts(c, g)} clicks(a)}{\max_{c' \in children} \sum_{a \in attempts(c', g)} clicks(a)}$$

We hypothesize that this characterizes the willingness to reconsider a solution.

We found a correlation between the performance on hard and medium games and *snc* with correlation coefficients of 0.6710 and 0.6787, respectively. This shows that it is possible to infer strategies from different statistical properties using analytical tools.

5.3 Real-time feedback and frustration detection

During our experiments, we found that consecutive unsuccessful attempts of solving an inductive reasoning test made children frustrated. This caused a further drop in their performance. Usually, a teacher's help was required to resolve this state. In order to prevent this situation from occurring, frustration detection and real time feedback is required. We discuss these problems separately.

The lack of real time feedback caused several problems in our previous experiments. The video logs of these experiments have shown that children's frustration

during games can be characterized by rapid head pose and eye movement variations. Figure 4.5 shows that after a few unsuccessful attempts to solve an inductive reasoning test, the child’s head pose started to vary constantly. This allows us to give estimations on the frustration level based on head pose and eye movement informations.

However, in order to use this information effectively and prevent the previously described self-amplifying process, we have to do it in real time. We took part in the development of a CLM-based C++ software for robust face tracking. We used this software together with the Handorino game as a proof-of-concept implementation to give the children real time feedback on their on estimated focus level.

5.4 Correlation analysis

5.4.1 Covariance matrix

Figure 4.7 shows the Hinton-diagram of the covariance matrix of Φ defined in formula 4.2. $cov(\Phi)_{i,j}$ is the covariance between the random variables corresponding to the appropriate mini game types. We have three remarks about the matrix:

- The fifth mini game type (which is cross-classification) has a low covariance with all the other game types. This may mean that the performance of children in the cross-classification games do not depend on their performance on other games. The reason for this may be a different strategy in these games, this is a subject of further analysis.
- The variance of the cross-classification game (0.0076) is the lowest of all. This may confirm our previous suspicion: if the majority of children follow a guessing strategy in cross-classification, it would result in a lower variance than other mini game types.
- The covariance between the second and third types (two generalization subtypes, class expansion and finding common attributes) is negative. This may be caused by the different logical operations required to solve games of these types.

5.4.2 Principal component analysis and independent component analysis

We used principal component analysis on the Φ matrix. The eigenvectors of $cov(\Phi)$ are shown on figure 4.9. The interpretation of the PCA basis requires further research.

The compression errors, shown in figure 4.8, do not show a rapid decrease anywhere, they decrease almost linearly.

Let G be a function which measures dependency between random variables. This may be mutual information, association etc. We chose covariance in section 4.4. If the ICA components would have been block-diagonal, it would mean that the input can be well clustered. With covariance, this is not the case. By using other functions, this could change. This is a subject of future work.

Chapter 6

Future work

Children participating in the experiments solved a pre-test before and a post-test after the experiment, as described in section 3.7. The post-tests are currently being evaluated. The most important task to do in the future is to compare the results with the pre-tests and make the necessary conclusions.

It would also be useful to visualize the development of the children. To do this, we have to find an appropriate measurement of performance, and possibly embed it into a lower dimensional space (by using principal component analysis, locally linear embedding etc.).

The performance of the children can also be examined with correlation analysis methods described in section 4.4, but with different parameters. First of all, we could use another measurement of performance besides Φ . The normalization into the $[0, 1]$ interval may also be replaced to use a method less sensible to outliers. Also, the measurement of dependency between the random variables described in the aforementioned section could be something other than covariance, like mutual information. If the result of the independent component analysis is block-diagonal up to a permutation with any of these methods, it may provide a way to cluster the mini game types.

Chapter 7

Conclusion

In this paper we studied the effects of the Handorino game on children playing inductive reasoning tests called mini games. We analyzed their strategies and behavior. We found that the frame game has a significant effect on children in the long run. It changes the Gaussian distribution of their performance to a sub-Gaussian. The exact nature of this phenomenon is yet to be researched.

We classified mini games into three categories based on their difficulty. We analyzed several statistical features of the data collected in the experiments. We discovered correlation between a certain gameplay strategy and performance in the mini games. The strategy in question is characterized by a high number of mouse clicks per game. We hypothesize that these children frequently reconsider and correct their solutions which leads to a better overall performance.

We characterized frustration level of the children using head pose and eye movement information only, and showed that real time feedback is possible.

These results open up the possibility of improving the mini games and the strategies of children. By applying real time feedback in case of frustration, the efficiency of computer based playful learning might be improved.

Appendix A

Face tracking software

A.1 Overview

This software provides an API for efficient real time CLM-based face tracking. This is based on the methods described in Section 3.2. The system can be divided into two separate parts:

- Training (Matlab)
- Tracking (C++)

During face tracking, we use pre-trained PDM and landmark point classifiers. These trainings are done in Matlab and the models are saved in .mat files for further use. Our C++ CLM software parses and uses these models. Details of training is described in Section A.3 and the tracking procedure in Section A.2.

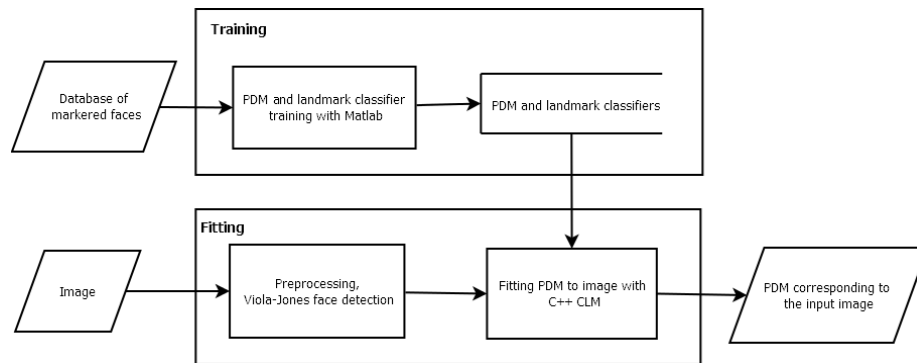


Figure A.1: Flowchart of the fitting and training processes. Shows the relationship between the two parts of the system, and the outline of the individual processes.

A.2 Structure of the C++ software

The software is designed to support replacing components. This includes changing the following components:

- landmark point classifiers
- PDM
- CLM algorithm

The trained PDM and the classifiers are contained in .mat files saved from Matlab with a pre-defined structure format. This software loads it's required data at startup from the model file directly with no explicit conversion required and thus can use different models. With the model loaded, the tracking itself is quite straightforward. As seen on Figure A.1, as an input image comes we run a Viola-Jones face detector which gives us a bounding box of the face, then we put the mean shape of the PDM in the bounding box as an initialization of the fitting algorithm. Numerous preprocessing steps can be included here, which can be changed. For example we can use contrast-limited adaptive histogram equalization or adaptive gain control on the face images for both fitting and training procedure. This partially resolves the problem of different lightings and contrasts. Most of our models currently use a multi-layer perceptron based algorithm for landmark point classification. The exhaustive search for landmark points uses a fast convolution based implementation which gives a significant boost to speed. The software makes extensive use of the OpenCV library for both rendering and algorithm implementations. Currently only face PDM fitting is available in our tracking software, iris PDM fitting is done in Matlab at the moment. The next step in the development is integrating this feature into the software.

A.3 Models and training examples

Training is done in Matlab. We have a complex framework for training arbitrary PDMs and classifiers on markeded databases. Most of the training is done on the BU-4DFE database described in Section 3.5. The PDM and the landmark point classifiers are trained on the same dataset. We have face PDMs with different number of landmarks, scale and PCA parameters. We also have an iris model which was trained on a different database consisting of approximately 700 training samples for each eye. These iris samples are quite low resolution images, thus the fitting can be really fast for this model. However despite the low resolution, precision is high. Figure A.3 shows a fitted face and iris PDM.

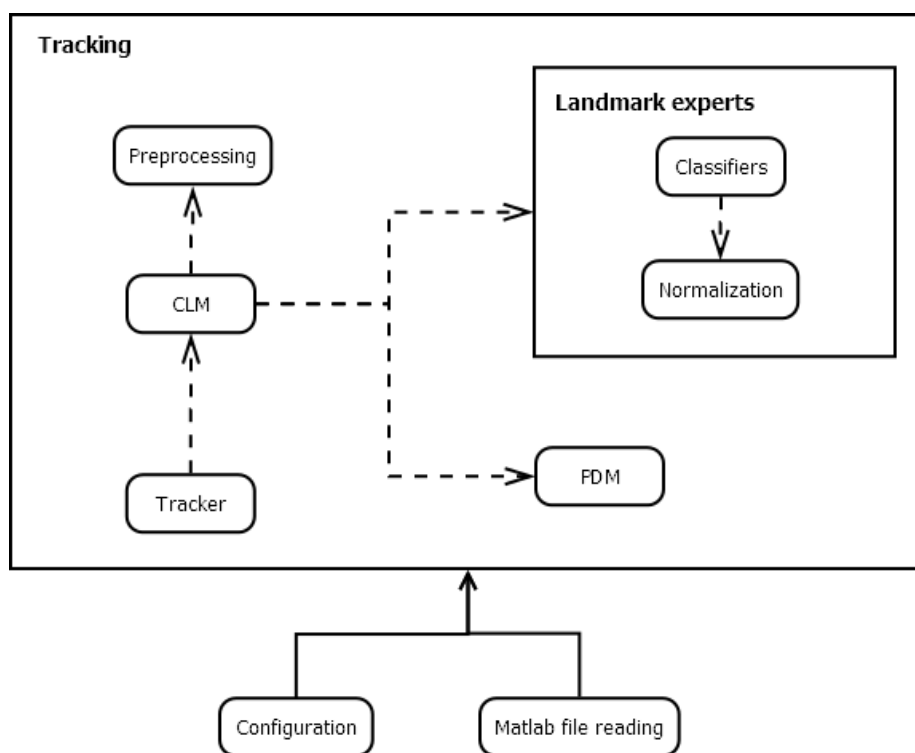
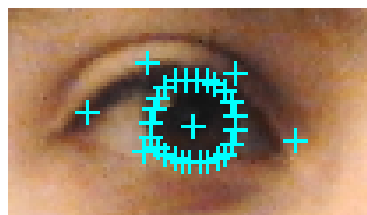


Figure A.2: Basic structure of the components



(a) Face PDM.



(b) Iris PDM.

Figure A.3: Face (a) and iris (b) PDM. The face PDM consists of 49 landmark points which is our regular model. The iris PDM uses 25 points.

Appendix B

Structure of the 3D Pac-Man game

B.1 Overview

Handorino, the 3D Pac-Man game was implemented in the Knapsack 3D game engine. The lead designer of Knapsack was András Dudás, working closely with Zoltán Tőser. This game engine itself is based on the PACSAGE multi-agent system, also designed by András Dudás.

Knapsack was designed to be a distributed, modular, multi-player 3D game engine. It supports various desktop and mobile input devices, but in the experiments only the keyboard and the mouse were used.

B.2 The layered architecture

Figure B.1 shows the architecture of the software. It consists of four different layers, with sharp boundaries between them. Each layer relies on functionalities provided by lower layers.

- The lowest layer contains the backing libraries: TAJTEEK and XStream. TAJTEEK is a general Java software packages which provides syntactic sugars, algorithms, generic collections, converters, parallel utilities etc. XStream is a Java library which provides efficient and customizable XML serialization and deserialization of Java objects.
- The second layer is highly customizable the multi-agent system.

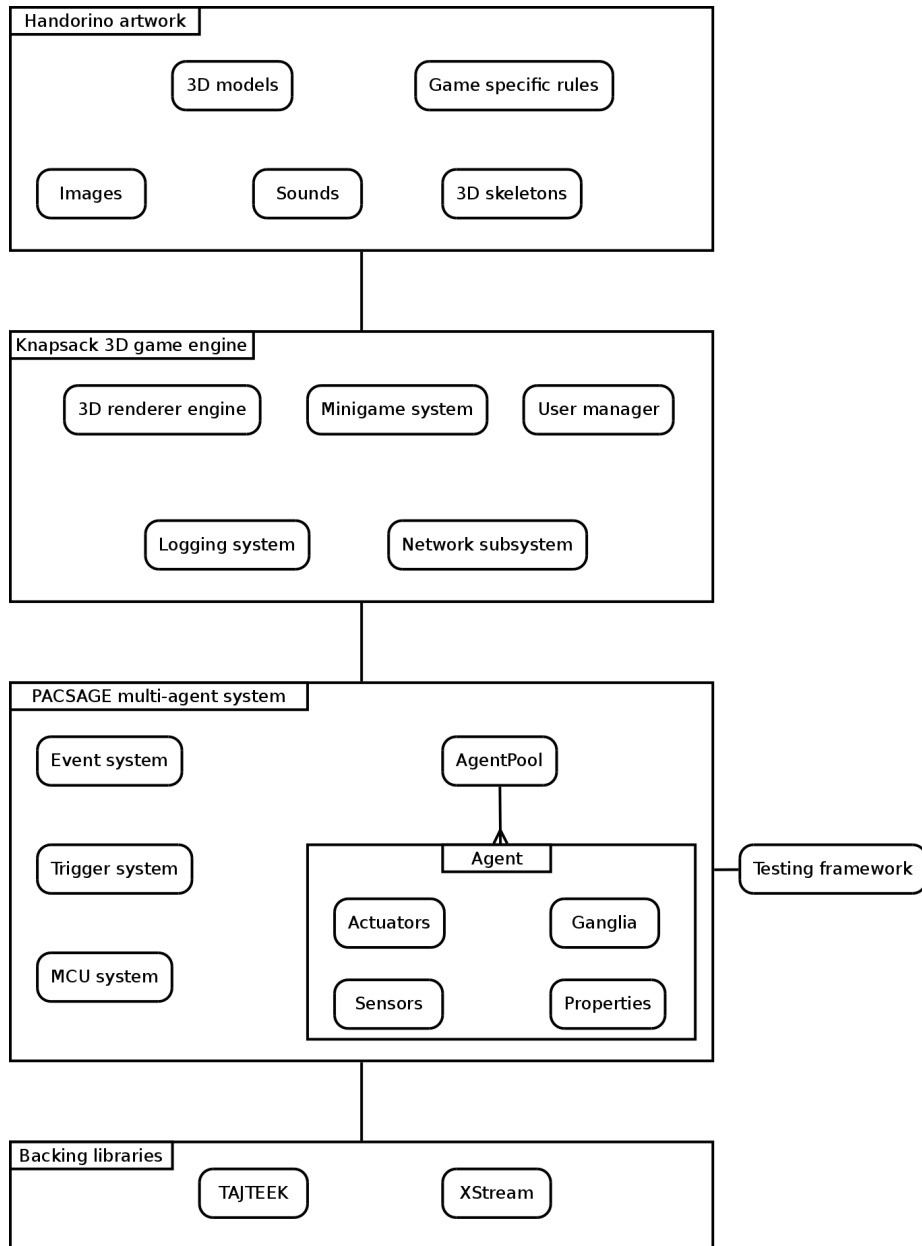


Figure B.1: The architecture of the 3D pac-man game

- The third layer, which is the highest layer that contains software, provides an implementation of the world in the multi-agent model, adds 3D graphical rendering and networking capabilities.
- The fourth layer contains the Handorino artwork. This consists of images, 3D models and skeletons, animations and sounds. The rules of the world specific for

the pac-man game are also in this layer.

B.3 Subsystems

This section describes several subsystems of the Handorino game. The list is not complete, only the more interesting ones were chosen.

B.3.1 The multi-agent system

The multi-agent system contains agents, which are located in the agent pool. From the individual agents' perspective, the agent pool is the "world". Each agent has four important components:

- Properties, which are simple or composite features of the agent. Other agents' properties may not be accessed directly, they may be read with sensors and modified with actuators (and in exceptional cases, with ganglia too).
- Actuators, which are used for changing the values of one or several properties. The actuators' target may be the initiating agent or others ones too.
- Sensors, whose role is twofold: they are used for accessing the properties of other agents, and they also provide a notifying mechanism, to execute callbacks when a certain event occurs in the agent pool.
- Ganglia. These are used for executing "extra-world" tasks, e.g. initializing a database connection, creating a new agent in the pool. Everything is considered extra-world that may not be expressed through the manipulation of the properties of certain agents.

Everything visible on the screen is an agent: not only the pac-men, the ghosts and the fruits, but also the floors and the walls. This makes it possible to customize and fine-tune implemented agent worlds using rules. The multi-agent system relies heavily on XML serialization, all agents are contained in XML form, when a pool is inactive.

It should be noted that agents themselves are passive components: they cannot react to sensor notifications or initiate actuator calls. Active behaviour is implemented by their ganglia: each ganglion can subscribe to sensors, launch control threads and call actuators, and also participate in extra-world processes.

B.3.2 The mini games

The mini games are modeled in the agent world. Each mini game instance has a mini game agent. When a user enters mini game mode, an agent selects the next mini game

instance for this player with a strategy pattern. The actual selection takes place in a ganglion, as database manipulation uses extra-world information. After the mini game instance is selected, the agent spawner ganglion creates a new agent instance of the appropriate agent type and initializes its properties.

The 3D renderer renders the screen based on the properties of the player's mini game agent. The options in an odd-one-out game and the current selection index are examples of mini game properties.

During mini game mode, the player indirectly manipulates the properties of her mini game agent. In the default use case, the mini games are controlled with the mouse (in contrary to the pac-man, which is controlled with the keyboard). However, due to current and future needs, they may also be controlled with the keyboard. To avoid confusing the users, the pac-man and mini game keyboard controls use the same buttons. However, generating different actuator calls from the same input source was difficult due to implementation constraints. The morphological computation unit (MCU) system was created to solve this problem. This subsystem is not described here.

When a player generates an input event on the client side, the keyboard or mouse controller notifies the server. It is here where the control flow enters intra-world, as the server creates an actuator call from the client notification. After the actuator call is finished, the sensors of the renderer receive a notification about it. Based on this information, it queries other necessary properties and refreshes the screen.

B.3.3 The logging system

Logging in a distributed system such as Handorino implicitly implies solving two problems:

- guaranteeing the consistency of the final log file
- handling failures during logging

As the system is distributed and log events are produced on every node, they should be merged, preferably run-time, to avoid long shutdowns. When a client connects to the server, their clocks are synchronized. Because a typical player session is about half an hour long, further synchronization is unnecessary.

After generating a log event, they are timestamped on the client. After this, delays in actually sending the events to the server and writing them to the disk are not very important. The server receives the messages on several threads, generates an object ready to be written out to XML, and a single thread handles the actual I/O operations.

The second problem is more difficult to solve. If a client has a noisy or unstable network connection, sending a log event may hold up the receiving thread for a long

time. Also, generating the XML data from log events may produce exceptions in case of invalid data. Handorino uses state-of-the art concurrent methods and solutions to guarantee that errors during logging are handled gracefully and different network nodes do not interfere with each other.

B.3.4 Networking

Handorino uses Java RMI for high-level network communication. The multi-agent system does not have a concept of networking, this is handled in the game engine layer.

The RMI roles of the participants are sometimes reversed, for example during an observer pattern: in this situation, the game client is the RMI server and the game server is the RMI client.

This causes problems when a game client is behind a router with network address translation (NAT), as when the router is not configured appropriately, the game client would not be able to generate RMI skeletons with a correct IP address (only with ones in its local area network).

To solve this problem, a high-level network protocol was designed, which works even in these situations by using the reverse connection for game server-game client communication.

Bibliography

- [1] B. Csapó, A. Lőrincz, and G. Molnár. *Innovative Assessment Technologies in Educational Games Designed for Young Students*, chapter Assessment in Game-Based Learning, Part 2, pages 235–254. Springer, 2012.
- [2] B. Csapó, G. Molnár, and K. Tóth. *The transition to computer-based assessment. New approaches to skills assessment and implications for large-scale testing*, chapter Comparing paper-and-pencil and online assessment of reasoning skills. A pilot study for introducing electronic testing in large-scale assessment in Hungary, pages 113–118. Luxemburg: Office for Official Publications of the European Communities, 2009.
- [3] T. Kanade, J. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. In *IEEE International Conference on Automatic Face and Gesture Recognition (FG'00)*, pages 46–53, 2000.
- [4] J. K. Klauer and G. D. Phye. Inductive reasoning. a training approach. *Review of Educational Research*, 2008.
- [5] K. J. Klauer. *Denktraining für Kinder I*. Hogrefe, Göttingen, Germany, 1989.
- [6] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression. In *3rd IEEE Workshop on CVPR for Human Communicative Behavior Analysis (CVPR4HB10)*, pages 94–101, 2010.
- [7] G. Molnár. Playful fostering of 6– to 8-year old students’ inductive reasoning. *Thinking Skills and Creativity*, 6(2):91–99, 2011.
- [8] J. M. Saragih, S. Lucey, and J. F. Cohn. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, 91(2):200–215, 2011.

- [9] I. Szita and A. Lőrincz. Learning to play using low-complexity rule-based policies: Illustrations through Ms. Pac-Man. *Journal of Artificial Intelligence Research*, 30:659–684, 2007.
- [10] Yan Tong, Wenhui Liao, and Qiang Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(10):1683–1699, 2007.
- [11] Wikipedia. Pac-Man — Wikipedia, the free encyclopedia, 2012. [Online; accessed 20-November-2012].
- [12] Lijun Yin, Xiaochen Chen, Yi Sun, Tony Worm, and Michael Reale. A high-resolution 3d dynamic facial expression database. In *8th IEEE International Conference on Automatic Face Gesture Recognition*, pages 1–6. IEEE, 2008. Tracking Number: 66.