

Fast, low resource, head detection and tracking for interactive applications

M. Perreira Da Silva^{*♦}, V. Courboulay[♦], A. Prigent[♦] and P. Estraillier[♦]

[♦]L3i
(France)

ABSTRACT

This paper presents a real time, low resource, head tracking system. This system is used for a broad range of applications, the simplest being the control of a car in an arcade racing game. Another use of this system is the improvement of the gameplay of an adventure game. A more advanced application is the detection of the player's attentional state using a simple attention model in an attention aware game framework. This state is then used to adapt the game unfolding in order to enhance user's experience and improve the game attentional attractiveness. The experiments conducted on these different games showed that even if using head as a simple input device for explicit game control can improve the player's immersion, its full potential can only be exploited when adapting or building new gameplay.

Keywords: *Head-based interaction, gameplay, low resource, low cost, head tracking*

1. Introduction

Providing good human-computer interactions is an important point in the development of interactive software (Barr, Noble, & Biddle, 2007). In the field of games, it is a way to increase immersion in the virtual world of the player. Classical interactions with mouse, keyboard or gamepad, are limited in comparison with the reality of graphics displayed. Indeed, a growing trend is to concentrate on new kinds of interfaces between the player and the virtual world. For example, some approaches are using a headpiece device to detect head movements in order to change the game camera direction. Another example is the use of the interactions between human bodies as a core gameplay element ("Teaser of the EPIDEMIK exhibition - Cité des sciences et de l'industrie (Paris, France)," pas de date). In this paper, we describe a

This is a draft of the final paper that can be retrieved from : <http://www.psychology.org/>

* Corresponding Author:
Matthieu Perreira Da Silva
L3i, Université de La Rochelle,
Av Michel Crépeau, 17042
La Rochelle, France
mperreir@univ-lr.fr

fast low cost head tracking system which is used to experiment new kinds of gameplay and improve the player's experience.

We present four prototype applications that have been developed or modified in order to use head direction as input. These applications have an increasing level of integration of head tracking in the gameplay. The first two are a Tetris and a racing game where head direction is simply used as an input to directly control the game. The third application is an adventure game in which user's head direction helps modifying the game scenario in order to try to make the game more immersive and fun to play. The last one is a pedagogical game in which information about the user's attention (derived from head direction) is used to adapt the game unfolding in order to refocus the player's attention. It is used as a tool for pedo-psychiatrists working with children with autism in the pedo-psychiatric hospital of La Rochelle, targeting the objective of improving children's attention.

In chapter 2, we describe the low cost head tracking system that we have developed. In chapter 3, we show how this system can be used as an explicit way to control a game. Then in chapter 4, we describe a more advanced software architecture that uses gaze tracking in order to detect the players' attentional state and update the game unfolding accordingly.

2. Low cost, fast head tracking

Many face detection algorithms have been published over the last 15 years. A survey of many of these systems can be found in (Yang, Kriegman, & Ahuja, 2002) and (Hjelmås, 2001). Face or head tracking is also a very active research area. See for example (Erik Murphy-Chutorian & Mohan Manubhai Trivedi, 2009) for a recent survey of the existing systems. However, many of these systems are conceived for performing either face detection or tracking and are not real-time capable. Recent work (E. Murphy-Chutorian & M.M. Trivedi, 2008) proposes a robust and real time system (30fps) for head detection and tracking, but this system uses the GPU in order to be real time. As a consequence, the system uses most of the CPU and GPU resources of the computer and no interactive application can be run in parallel.

To overcome these limitations, we propose a low resource head detection and tracking algorithm which allows running an interactive application (even a 3D game) in parallel on the same computer.

2.1. Constraints

As our system is designed to be used by a wide range of applications and users (from educational games for children with autism to adventure games for "common gamers"), some constraints have emerged:

- non invasive material: users should be able to forget the presence of the tracking device and concentrate on the interactive application;
- low cost: the system must use affordable, off the shelf, hardware;
- single user: the system should detect and track only one person at a time. If multiple persons are present in front of the camera, only one of them will be tracked.
- recordable information: the algorithm should output tracking information that can be easily stored or exploited by another algorithm (ex: attention model).
- standard computer: the system should run smoothly on a standard mid-end computer, with no additional or specific hardware.
- unconstrained environment: our algorithm should be able to work without any additional lighting.

The system is based on a low cost (IEEE-1394 or USB) webcam connected to a standard computer. Despite its low cost, this type of camera captures video frames of size 640x480 at 30 frames per second which are suitable characteristics for both accurate face features localization and efficient face features tracking. Depending on the application, the system can use a color or grayscale camera. The choice of a grayscale camera instead of a more common color camera will be driven by the environment of the aimed application. Indeed, in some environments (mainly indoor), the amount of light available is often quite low. As grayscale cameras usually have better sensitivity and have a higher image quality (as they don't use Bayer filters) they are more suitable for these environments. Grayscale cameras may also be used with infra-red light and optics that don't have infra-red coating in order to improve the tracking performance by the use of a non invasive more frontal and uniform lighting. During our experiments we have only used (so far) color or gray scale cameras with no additional infrared lighting.

2.2. Architecture

The tracking algorithm is built upon four modules which interoperate together in order to provide a fast and robust face tracking system. The algorithm contains two branches: the first one for face detection and the second for face tracking. At run time, the choice between the two branches is made according to a confidence threshold, evaluated in the *face checking and localization* module. In the following paragraphs, we detail the algorithms used by each modules of Figure 1.

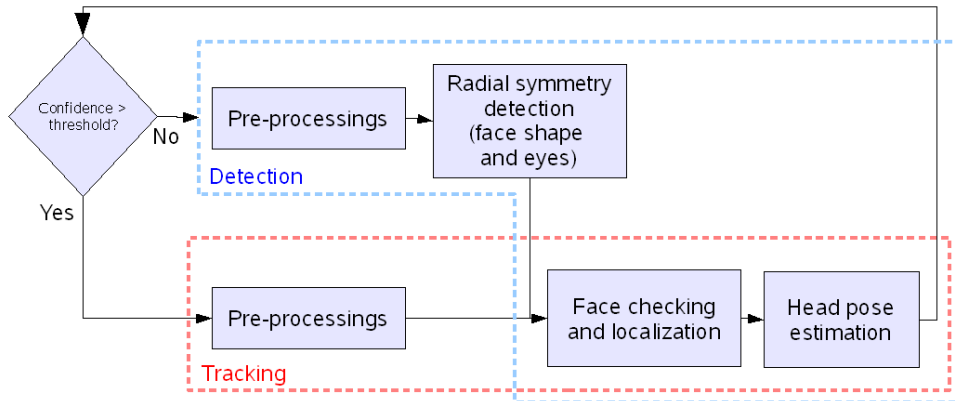


Figure 1. Architecture of the face tracking and pose estimation system.

Pre-processings

Before face or radial symmetry detection, the input image must be pre-processed in order to improve face and face feature detection performance. The main pre-processing steps are:

- image rescaling; the input image is rescaled so that low resolution data is available for radial symmetry detection algorithms;
- (in case of color camera) skin hue detection, using a commonly used histogram-back-projection (Zaqout, Zainuddin, & Baba, 2005)
- lighting correction, also called contrast normalization which consist in adapting each pixel intensity according to the local mean intensity of its surrounding pixel. As this task is naturally performed by human's retina, several complex models have been developed to mimic this processing (Beaudot, 1994).

Since our system needs to be real-time, we have chosen to approximate this retinal processing by a very simple model which consists in the following steps:

- For each image pixel, we build a weighted mean $M_{x,y}$ of its surrounding pixels. In our implementation we used first order integral filtering (Bartlett

Filter) in order to achieve fast filtering. Note that first order integral filters are an extension of the commonly used zero order integral filters (box filters). For more information about generalized integral images see (Derpanis, Leung, & Sizintsev, 2007).

- Then we calculate the normalized pixel intensity: $I_{x,y} = \frac{S_{x,y}}{M_{x,y} + A}$ with S the source image, I the normalized image, and A a normalization factor.

Figure 2 shows the result of our simple contrast normalization algorithm on a side lit scene.



Figure 2. a) Tracking result of a side lit scene. b) Source image after lightning correction. c) Result of face ovoid detection. d) Result of eyes detection.

Radial symmetry detection for face shape and eye detection

Once the image is pre-processed, we use a set of radial symmetry detectors in order to localize a candidate face region that will be further checked by the *face checking and localization* module. Once again our real-time constraint guided the choice of the algorithms we used.

Face ovoid shape is detected in a low resolution version of the input image (typically 160x120, since we don't need much precision for this step) using an optimized version of the Hough transform (Figure 2.c) whereas eyes are detected using an optimized version of the Loy and Zelinsky transform (Loy & Zelinsky, 2003) (Figure 2.d). In order to speed up both transforms, the following improvements have been made: only pixels with a gradient magnitude above a pre-defined threshold¹ are processed; the algorithms vote in only one accumulator for all radius and accumulators are smoothed only once at the end of the processing.

Using these two symmetry maps and a set of face geometry based rules, we define the face candidate area.

¹ defined empirically

Face checking and localization

This module serves two purposes:

- When called from the face detection branch, it checks if the face candidate area really contains a face and outputs the precise localization of this face.
- In the case of face tracking, it only finds the new position of the face.

In both cases, the module outputs a confidence value which reflects the similarity between the interface face model and the real face image.

Face checking and localization is based on the segmentation of the face candidate image into several blobs, determined as follow: the source frame is first filtered by a difference of Gaussian (DoG) filter; the result image is then adaptively thresholded. The resulting connected components (blobs) are then matched on a simple 2D face model in order to check and localize the face.

Once the face is localized, the 2D face model is deformed to adapt to the detected face features. These deformations are kept from one tracking step to the other in order to improve tracking performance. Since we are performing continuous tracking at 30 fps, changes from one image to the other are quite small, hence we do not use any kind of prediction (ex: Kalman filtering).

Head pose estimation.

Similarly to previous research we use triangle geometry for modeling a generic face model. For example, (Kaminski & Shavit, 2006) resolves equations derived from triangle geometry to compute head pose. Since we favor speed against accuracy, we use a faster and simpler direct approximation method based on side length ratios (Nikolaidis & Pitas, 2000).

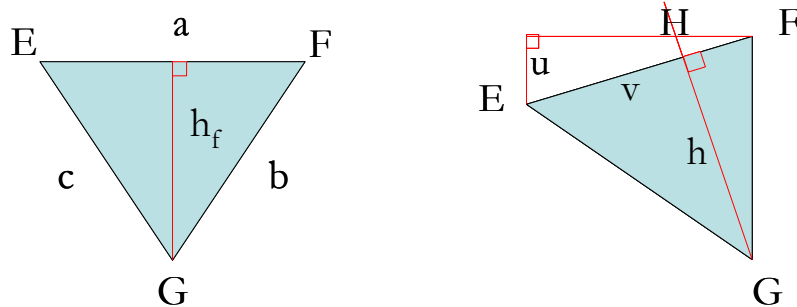


Figure 3. Features used for head pose estimation.

Continuous values for pan, tilt, roll as well as x, y and z position are processed using simple triangle geometry. The considered EFG triangle is built from the respective position of left eye, right eye and nose tip. We hypothesize that for a frontal view, this triangle is isosceles.

$$a = b = c \quad \text{et} \quad h_f = \frac{a\sqrt{3}}{2}$$

When the face is rotated we can determine the following values easily:

$$\begin{aligned} yaw &= \frac{v - \frac{a}{2}}{a} & \text{with} \quad v &= \frac{\|\overrightarrow{EH}\|}{\|\overrightarrow{EF}\|} = \frac{\overrightarrow{EF} \cdot \overrightarrow{EG}}{\|\overrightarrow{EF}\|^2} \\ pitch &= \frac{h}{h_f} \\ roll &= \frac{u}{a} \end{aligned}$$

2.3. Performance and robustness

Processing time

We measured the processing time of the algorithm on a laptop PC equipped with a 1,83GHz *Intel Core Duo* processor. We obtained the following mean processing times:

Algorithm	Processing time per frame	CPU load
Face detection ²	30 ms	50% (100% of one core)
Face tracking	16 ms	25% (50% of one core)

Table 1. Processing time of the detection / tracking algorithms.

Processing time include all processing steps, from pre-processing to head pose estimation. Since image capture is done at 30fps and the algorithm is using only one of the two processor cores the algorithm is fast enough to enable running a game in parallel on a standard middle-end computer. Please note that we have not used any GPU acceleration in order not to slow down any 3D graphics in the target interactive application.

Robustness

As can be seen from the tracking example shown on Figure 4, the algorithm can handle a broad range of head orientation and distance, as well as different people. The

² first detection or detection after tracking failure

contrast normalization step also allows the algorithm to run under different illumination conditions.



Figure 4. Result of the tracking algorithm for different face distances and orientation.

However, this algorithm is designed to be fast, as a consequence tracking performances still need to be improved under some lightning conditions (backlit scene, directional lighting casting hard shadows, etc.) and head pose (Figure 6). When using a color camera and skin color detection algorithms (as in (Schwerdt & Crowley, 2000) or (Séguier, 2004)) the face detection and tracking robustness are improved. This modification however prevents us from using infra-red light to improve the algorithm performances under poor lightning conditions. A possibility would be to add an infra-red lightning and adapt the current algorithm to this new lightning in order improve the robustness of the system.

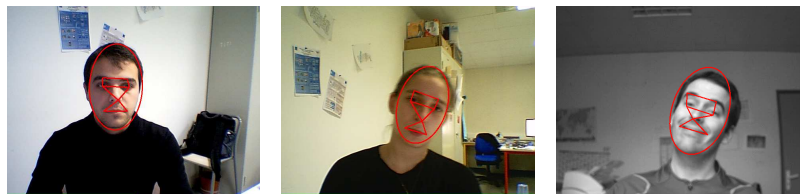


Figure 5. Result of the detection / tracking algorithm for different participants.

Lastly, the algorithm is not designed for occlusions handling: if the face is masked by any object, tracking is lost (but quickly recovered at the end of occlusion). Fortunately, occlusions are very rare when the system is used with interactive applications.

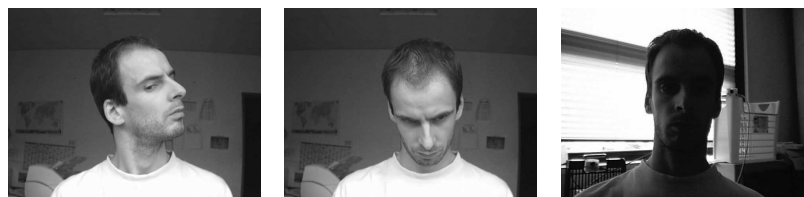


Figure 6. Examples of detection / tracking failure. Left and middle : face pan and tilt are too important. Right : backlighting is too strong.

2.4. Gaze tracking

Some of the applications described below require information about user's gaze. Since eyes are a feature extracted for our face model, it should be quite easy to use their position to (in conjunction with head orientation) estimate eye gaze. However, considering the large freedom of movement allowed to users, extracted faces can be as small as 48 x 57 pixels. In this case, it is difficult to compute an accurate and reliable estimate of gaze direction.

To overcome these problems, we propose to use head direction as an approximation for gaze direction. Although it may look simplistic, in practice, it provides enough information for the simple attention monitoring tasks described in chapter 4.

The algorithm described above has been integrated into several games / interactive applications. In the next chapter we describe the experiments conducted on these applications.

3. Explicit control of games via head tracking

3.1 Head as a simple input device

Our first experiments consisted in modifying two classic games: a Tetris and a racing game (which are both public domain Microsoft XNA Game Studio game samples) in order to connect them with our head tracking system. We chose these games because any casual gamer knows how to play them, which shortens the learning phase. Moreover, racing game players tend to naturally roll their head when the car is within curves; head movements seemed therefore good candidates for an alternative way to control the car. Once the games modified, we were able to use head movements like any other input device.

Figure 7 show screenshots of both applications. Video demonstrations are also available on YouTube³.

We conducted some experiments with a small set of five persons aged from 23 to 35 (all were casual gamers). Each person was shown how to play the game for 5 minutes. Then, they could play on their own for 15 more minutes. Lastly, they were interviewed

³ <http://www.youtube.com/watch?v=e8VGafkN4RQ> and <http://www.youtube.com/watch?v=91F9VnBa7Wo>

in order to know what they thought about this new kind of interaction. These experiments showed that:

- Gaze control is not suited to all kinds of games. Moving and rotating Tetris bloc with head movements is quite unintuitive and requires some training. On the contrary driving with gaze is quite easy to learn, and quite fun.
- Robustness is a key point for a good gaming experience. If tracking is lost during the game (because of extreme head positions or bad lightning condition as described in section 2.3), all immersion improvements are lost. However, the system need also to be fast, if the system's responsiveness is low immersion improvements are also lost.
- The gameplay of head controlled games needs to be adapted in order to be really intuitive. For example, gamers found it difficult to control the Tetris game with their head. Additionally, they obtained lower scores when controlling the game with their head than when controlling it using the keyboard. For the racing game, although head control was easier and fun, users sometimes had difficulties controlling throttle. A solution would be for example, mixing mouse (throttle) and head tracking (steering) in order to propose a more natural gameplay.

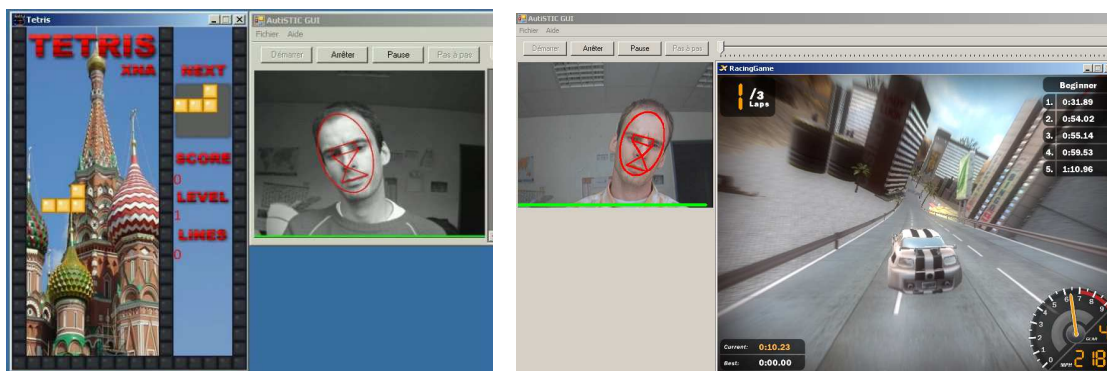


Figure 7. Left: screenshot of the gaze controlled Tetris game. Right screenshot of the gaze controlled racing game.

To further exploit the potential of gaze in games we have decided to add head controlled sequences in an adventure game prototype developed by students of the ENJMIN⁴ School: L3iLife. This time we don't replace an input modality by another. Instead, we introduce new gameplay elements which were created for head based interaction.

⁴ Ecole Nationale du Jeu et des Médias Interactifs Numériques

3.2 Head as a new gameplay element

We have developed an adventure game based on the visit of a virtual world that represents the computer science laboratory (L3i) of the University of La Rochelle. The player has to explore the laboratory by opening doors and visiting rooms. Figure 8 shows a screen capture of the prototype that has been developed with the Unreal Engine editor⁵.

The game concept is the following one. The player is a student that has a fixed delay to give a work to his teacher. He is in direct competition with an *evil* student that tries to prevent the player from reaching his goal and with a *little pest* that tries to steal his work and give her own work first to the teacher.

We propose to give to the player a maximum amount of interactivity while keeping a robust and interesting narrative framework. The approach of emergent narrative consists of a particular architecture that increases the player's freedom of action and produces a dynamic control of narrative quality. A challenge is, for example, to detect the player's behavior in order to dynamically modify the scenario.

Interested readers will find more details about the architecture of the game in (Champagnat, A. Prigent, & P. Estrailier, 2005) and (Perreira Da Silva, Vincent Courboulay, & Armelle Prigent, 2007).

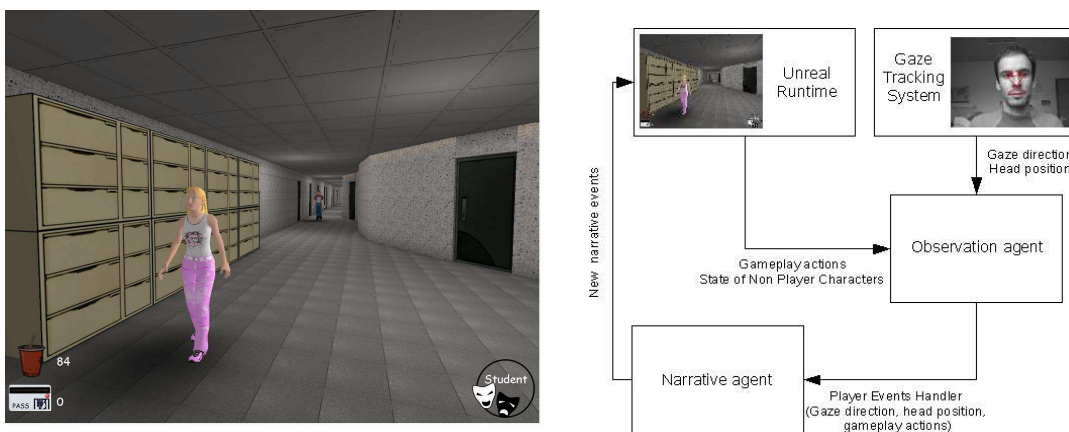


Figure 8. Left: A screen capture of the prototype game (L3iLife). Right: Overview of the narrative based game architecture

⁵ <http://www.unrealtechnology.com/>

Game / head interaction

During our first experiments (conducted with the same protocol and participants as in section 3.1), we only took into account a few explicit players' behavior:

Firstly, we focused on the interaction with the non player character of the *little pest*. She tries to steal the work of the player. The player can interact with the girl by doing a wink⁶ to the camera when he is in front of the girl. Then, the girl will give his work back to the player if she has stolen it to him.

Secondly, we allow the player to protect himself against the *evil* student by looking down. Indeed, if the evil student arrives near the player, this one will put his head down and then the *evil* student will go on without stealing his work.



Figure 9. Gaze controlled sequence of L3iLife. Left: The evil student is about to catch the player. Middle: The player gaze down in order to show his submissiveness. Right: The evil student does not catch the player: game is not over!

During our first experiments we have observed that this new kind of interaction improves the players' immersion in the virtual world of the game. Moreover, it also increases its interest for the game because the gameplay is richer and the game more fun to play. These observations will certainly be confirmed once our framework will include the observation of implicit behavior as we will have more real-time feedback on the user's gaming experience.

We are currently working on the integration of more complex kinds of behavior into the interactive game framework. Gaze tracking could be used to observe the level of attention of the player. For example, if the player stops watching at the screen, a particular game action can be launched to refocus his attention. In L3iLife for example, in this kind of situation, the adaptive architecture can modify the unfolding of events,

⁶ Since the player's eyes are already localized by our face tracking algorithm, blinks are detected by a simple frame subtraction.

and make the *evil* student run after the player to steal his work. It is a stressing action that can bring some interest back to the player.

This implicit behavior system is not yet implemented in L3iLife, but we conducted some similar experiments with an educational game aimed at autistic children. We describe this system in the next section.

4. Implicit control of games via head tracking

Usually, computer software has a very low understanding of what the user is actually doing in front of the screen. It can only collect information from its mouse and keyboard inputs. It has no idea of whether the user is focused on what it is displaying, it doesn't even know if the user is in front of the screen.

One of the first steps to making computer software more context aware is to make it attention aware. In (Roda & Thomas, 2006) attention aware systems are defined as «Systems capable of supporting human attentional processes.»

From a functional point of view, it means that these systems are thought as being able to provide information to the user according to his estimated attentional state. According to (Roda & Thomas, 2006), one of the best ways to estimate user's attention is using gaze direction. Consequently, building good adaptive attention aware system requires estimating reliably user's gaze.

Many high accuracy commercial gaze tracking systems are currently available (Abdallahi Ould Mohamed, Perreira Da Silva, & Vincent Courboulay, 2007). Most of them use dedicated or costly hardware in order to estimate user's gaze with the desired precision. Moreover, these systems are designed to be used with cooperative users (most of the time disabled users or attention studies participants) in a constrained environment. A choice needs to be made between gaze tracking precision, equipment cost and user's freedom of movement. As our attention aware system only needs to know whether the user watches the screen or not, we propose to use head pose as an estimator for gaze direction.

But before closing this, let's have a closer look on what attention is.

4.1 Attention

Attention is historically defined as follows (James, 1890):

«Everyone knows what attention is. It is the taking possession by the mind in clear and vivid form, of one out of what seem several simultaneously possible objects or trains of thought...It implies withdrawal from some things in order to deal effectively with others»

Thus, attention is the cognitive process of selectively concentrating on one thing while ignoring other things. In spite of this single definition, there are several types of attention (A. Ould Mohamed, V. Courboulay, K. Sehaba, & Menard, 2006): awakening, selective attention, maintained attention, shared attention, internal or external absent-mindedness and vigilance. For an interactive task, we are mainly interested in selective and maintained attention. The analysis of the first one allows knowing whether people are involved in the activity. The second one enables us to assess the success of the application.

It has been proven that the same functional brain areas were activated for attention and eye movements (Corbetta et al., 1998). Consequently, the best attention marker we can measure is undoubtedly eyes and gaze behavior. A major indicator concerning another type of attention, vigilance, named PERCLOS (Dinges & Powell, 1998) is also using such markers. (Horvitz, Kadie, Paek, & Hovel, 2003) presents different models for integrating attention into an Attentional User Interface. But the models are mainly focus on attention, without any clear link with gaze. On the contrary, (Peters, Pelachaud, Bevacqua, Mancini, & Poggi, 2005), presents how gaze direction can be used in order to improve the behavioural plausibility of an Embodied Conversational Agent. But the models described have only been used in a simulated environment.

In continuity of such studies, we based our markers on gaze behavior, approximated by head pose, to determine selective and maintained attention. A weak hypothesis is that a person involved in an interesting task focuses his/her eyes on the salient aspect of the application (screen, avatar, car, enemy, text...) and directs his/her face to the output device of the interactive application (screen). Nevertheless, if a person does not watch the screen, it does not necessarily mean that he/she is inattentive; he/she can be speaking with someone else about the content of the screen (Kaplan & Hafner,

2006). However, the more time users spend not watching the screen, the more probable is their inattention. Consequently, we have decided to adopt the following solution: if the user does not watch the screen during a time t , we conclude to inattention. In the following subsection, we present how t is determined. If inattention is detected, we inform the application.

A simple model of human inattention

The goal of this model is to define the delay after which the application tries to refocus the user on the activity. Actually, we easily understand that in this case, an interactive application does not have to react the same way if people play chess, role player game or a car race. Until now, this aspect of the game was only directed by the time during which nothing was done on the paddle or the keyboard.

We based our model of what could be named inattention on two parameters:

- the type of application;
- time spent using the application.

The last parameter depends itself on two factors:

- a natural tiredness after a long time
- a disinterest more frequent during the very first moments spent using the application than once attention is focused, this time corresponds to the delay of *immersion*.

Once the parameters are defined, we propose the following model in order to define the time after which the application try to refocus the player who does not look at the screen.

Potential of attention

As we mentioned, potential of attention depends mainly on two parameters, tiredness and involvement. We have decided to model arousal (the opposite of tiredness), or potential of attention, by a sigmoid curve parameterized by a couple of real number β_1 and β_2 . β_2 represents the delay after which the first signs of fatigue will appear and β_1 is correlated to the speed of apparition of tiredness (Figure 10).

$$P_{arousal} = \frac{\exp^{-\beta_1 t + \beta_2}}{1 + \exp^{-\beta_1 t + \beta_2}} \quad \text{where } \beta_1 \text{ and } \beta_2 \text{ are two real parameters.}$$

For the second parameter, we have once again modeled involvement, or interest probability, by a sigmoid. We started from the fact that activity is a priori fairly

interesting, but if the person is involved after a certain time ruled by α_2 , we can consider that interest is appearing at a speed correlated to α_1 (Figure 10).

$$P_{interest} = \frac{1}{1 + \exp^{-\alpha_1 t + \alpha_2}}$$

For our global model of potential of attention, we couple both previous models in order to obtain:

$$P_{attention} = P_{interest} \times P_{arousal}$$

Delay of inattention

Once this model is defined, we are able to determine the time after which the software has to react (if the person still does not look at the screen). Here, it is an arbitrary threshold γ determined by experience, which characterizes each application. The more the application requires attention, the higher this coefficient is. Under the hypothesis that attention follows an exponential decay, the equation we have adopted is an exponential function which models the time after which attention reaches its activity switching threshold.

$$D_{game} = \exp^{\gamma(t) \times P_{attention}}$$

γ is a function of time because we have estimated that it can exist several tempo in an application (intensive, stress, reflection, action ...).

As a conclusion, we can summarize our model of inattention by the two following steps (Figure 11):

- depending on the time elapsed from the beginning of the application, we estimate the potential of attention $P_{attention}(t)$;
- depending on this potential and the application, we estimate the delay $D_{\gamma(t)}(P_{attention}(t))$ after which the software has to refocus the inattentive player.

Please note that this model was validated on only five children and still need to be validated on more people. Nevertheless, the first tests are quite promising.

This model of inattention has been used in conjunction with our real time gaze tracking system in an educational game. This game is part of a project called the *AutiSTIC Project* which we describe in the following section.

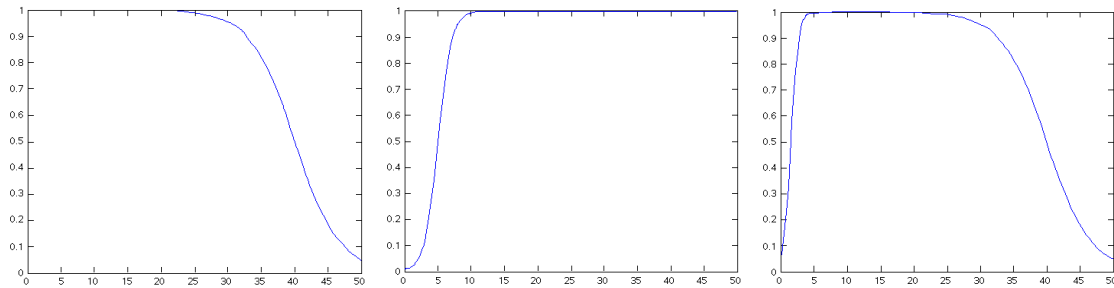


Figure 10. Left: model of tiredness evolution. Middle: model of interest probability. Right: model of potential of attention. Abscissa represents time in minutes. ($\alpha_1=1$, $\alpha_2=3$, $\beta_1=0.3$, $\beta_2=12$)

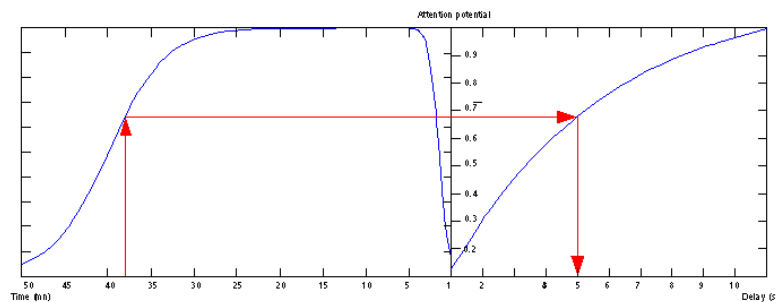


Figure 11. Curves used to determine the delay of software interaction.

4.2. The AutiSTIC Project

The AutiSTIC Project tends towards implementing a system that can help autistic children during the rehabilitation process. The role of such a system is to provide the child with personalized activities in the form of educational games. During a session, the system collects through various devices (camera, touch screen, mouse, and keyboard) actions and attention, in order to understand her/his behaviour and responds to it, in real time, by adequate actions considering directives. These directives concern rupture, avoidance, stereotype gestures... For instance, the system may attract attention by displaying an image on the screen, or by launching a characteristic music. It is impossible to generalize activities or reaction without precaution; we have to favour the adaptability of a system to take into account the specificity of persons. It is important to locate and interpret carefully these intrinsic behaviours as eye and gaze orientation, in order to help him/her to rehabilitate. We do not have to perturb the child when he/she is working; misinterpreting his/her attention may deeply perturb him/her because of his/her monotropism (Murray & Lawson, 2005).

In the application context, our architecture aims to bring flexibility and modularity in the individualized rehabilitation of children with autism. In the next section we present a part of our platform which observes children and analyses their visual attention.

The hello hidden game

This interactive game, which was developed within the context of the AutiSTIC project, is characterized by simplicity in order to not perturb the autistic child, often sensitive to complex environments. This is why the game has a static background, contains few objects and is easy to use. Nevertheless, each object has several behaviors in order to allow the control of execution according to the behavior of each child. The goal of the game is to allow autistic children to reach by interactive manipulation the competences of perception, motor function, spatial and temporal representation. Figure 12 shows the game interface of "hello hidden". This game allows using one or more balls of various colors on the screen. These balls appear in a frame that is displayed in fullscreen.

There are two kinds of balls:

- The small ball, called cursor, which the children can handle by applying a pressure on a touch screen.
- Big balls of various colors which remain motionless. The interaction between the cursor ball and the big balls contains two possibilities: Either the cursor ball disappears when it becomes near big ball and reappears when it goes away, or it stops its progression when it arrives at the periphery of a big ball and joins with it.

The objectives of this game vary according to various situations which occur. In the case of the presence of only the cursor ball, its elementary use allows the child to establish a relation between a direct motor action (to touch the screen) and the effect produced (movement of the object). In the case where one or more big balls are present on the screen and the cursor ball disappears when it moves near big ball, the objective is to analyze the capacity of the child to represent hidden objects and to act to make the cursor visible.

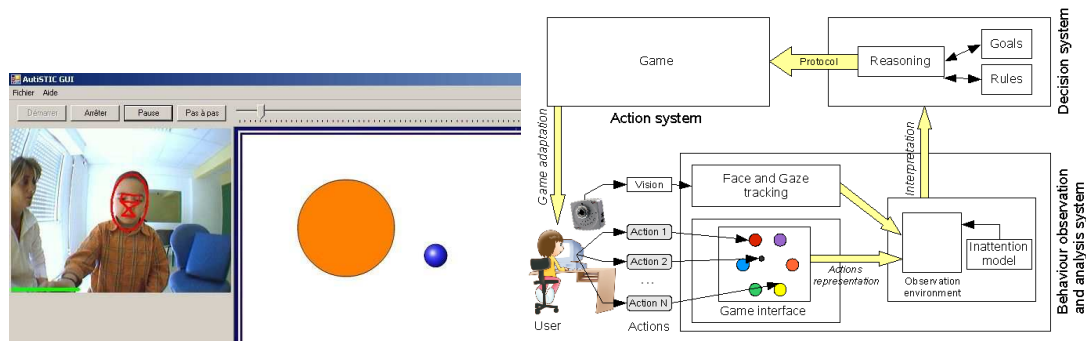


Figure 12. Left: Screenshot of the "hello-hidden" game. Right: General architecture of the adaptive software platform.

More details about the architecture of this interactive application can be found in (Karim Sehaba, Vincent Courboulay, & Pascal Estraillier, 2006). In this application, head orientation is used to estimate whether the child is watching the screen or not. When he is not watching the screen, we use our inattention model to decide when to play a sound to refocus him.

Results

Our platform can use, in real time, information concerning the explicit behavior of the child (mouse and keyboard events) and his/her implicit behavior (movements of the face, gaze...) in order to estimate the degree of carelessness of the child and to adapt the activity as well as possible. We use these annotations, done using video annotation software (L3iAnnote), in order to make automatic measures and to validate our model of inattention. Our software, installed both in academic and in medical sites, has allowed us to confront our model and the reality of children in front of a computer. Preliminary results obtained with children who took part to our study (three) are very interesting and promising, but the number of participants should be significantly increased and the model also needs to be validated on people without autism. Yet, it is very difficult to find children for the experimentations, because of the reserve of parents and the small number of children in the medical structure. Currently, we have tested our model on ten ordinary players in our laboratory, and the first results confirm those obtained with children with autism. We will continue the workshops with the same children by leaving this time the initiative of the stimuli to the computer, in order to analyze their behavior.

5. Conclusion

In this article we have presented a low resource real-time head tracking system. This system provides a fast and low cost solution for integrating approximated user's gaze into interactive applications. Its precision is not as good as gaze tracking devices and its robustness is not as good as slower head tracking systems. However, it provides a good tradeoff between cost, resource consumption, precision, robustness and freedom of movement. Ideally, one would like a more precise and robust system; but providing a precise system with great freedom of movement or a more robust one with low resource consumption are still unsolved issues.

This system has been successfully integrated into four different games: a Tetris game, a racing game, an adventure game (L3iLife) and an educational game part of the AutiSTIC Project. The experiments we conducted on these different games showed that even if using head as a simple input device for explicit game control can improve the player's immersion, its full potential can only be exploited when adapting or building new gameplay. This can be accomplished by the following ways:

- Do not try to use full head based control. Mix *classical* input devices and head control in order to provide the best ergonomics.
- Insert head controlled sequences in games only when necessary. For example in L3iLife, head controlled sequences were launched only in defined situations (interaction with non-player character).
- Do not change the way the game is controlled, but use head tracking as a gaze tracking approximation in order to monitor user's attentional state. These data can then be used to modify the game's unfolding.

Lastly, we have proposed a simple model of inattention, which provides an original solution for processing the attentional state of person from its eye or head tracking data.

6. Acknowledgments

This work has been partly funded by Orange Foundation and French Poitou-Charentes County.

The authors would also like to thank doctor Mr. D. Lambert Head of Department of Child Psychiatry of La Rochelle hospital (France) and his team, in particular: Mr. V. Gabet for their useful advices regarding the rehabilitation methods dedicated to children with autism.

7. References

- Barr, P., Noble, J., & Biddle, R. (2007). Video game values: Human-computer interaction and games. *Interacting with Computers*, 19(2), 180-195.
- Beaudot, W. (1994). The neural information in the vertebrate retina: a melting pot of ideas for artificial vision. Phd dissertation, TIRF Laboratory, Grenoble, France.
- Champagnat, R., Prigent, A., & Estrailier, P. (2005). Scenario building based on formal methods and adaptative execution. ISAGA 2005, Atlanta, Georgia.
- Corbetta, M., Akbudak, E., Conturo, T. E., Snyder, A. Z., Ollinger, J. M., Drury, H. A., et al. (1998). A common network of functional areas for attention and eye movements. *Neuron*, 21(4), 761-773.
- Derpanis, K. G., Leung, E. T., & Sizintsev, M. (2007). *Fast scale-space feature representation by generalized integral images* (No. CSE-2007-01). York University.
- Dinges, D. M., & Powell, J. (1998). *Evaluation of techniques for ocular measurement as an index of fatigue and the basis for alertness management* (DOT HS 808 762). Final report for the USDOT, National Highway Traffic Safety Administration(NHTSA).
- Hjelmås, E. (2001). Face Detection: A Survey. *Computer Vision and Image Understanding*, 83(3), 236-274.
- Horvitz, E., Kadie, C., Paek, T., & Hovel, D. (2003). Models of attention in computing and communication: from principles to applications. *Commun. ACM*, 46(3), 52-59.
- James, J. (1890). *The Principles of Psychology*. H. U. Press (Éd.), Cambridge, Massachusetts, (Vol. 1, pp. 403-404).

- Kaminski, J. Y. T., & Shavit, A. (2006, February). Head Orientation and Gaze Detection from a Single Image. Presented at the International Conference of Computer Vision Theory and Applications. Setúbal, Portugal.
- Kaplan, F., & Hafner, V. V. (2006). The challenges of joint attention. *Interaction Studies*, 7(2), 129-134.
- Loy, G., & Zelinsky, A. (2003). Fast Radial Symmetry for Detecting Points of Interest. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8), 959 - 973.
- Murphy-Chutorian, E., & Trivedi, M. (2008, June). HyHOPE: Hybrid Head Orientation and Position Estimation for vision-based driver head tracking. Intelligent Vehicles Symposium. Eindhoven, The Netherlands.
- Murphy-Chutorian, E., & Trivedi, M. M. (2009). Head Pose Estimation in Computer Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4), 607-626.
- Murray, D. L., & Lawson, W. (2005). Attention, monotropism and the diagnostic criteria for autism. *Autism*, 9, 139-156.
- Nikolaidis, A., & Pitas, I. (2000). Facial feature extraction and pose determination. *Pattern Recognition*, 33(11), 1783-1791.
- Ould Mohamed, A., Courboulay, V., Sehaba, K., & Menard, M. (2006). Attention analysis in interactive software for children with autism. *Assets '06: Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility* (p. 133—140). New York, NY, USA: ACM Press.
- Ould Mohamed, A., Perreira Da Silva, M., & Courboulay, V. (2007). A history of eye gaze tracking (L3i Technical Report No. 215967). La Rochelle: Laboratoire Informatique Image Interaction.
- Perreira Da Silva, M., Courboulay, V., & Prigent, A. (2007, September). Gameplay experience based on a gaze tracking system. Presented at COGAIN 2007 - Gaze based Creativity, Interacting with Games and On-line Communities, De Montfort

University, Leicester, UK.

- Peters, C., Pelachaud, C., Bevacqua, E., Mancini, M., & Poggi, I. (2005). A model of attention and interest using Gaze behaviour. In J. G. Carbonell and J. Siekmann (Eds.), *Intelligent Virtual Agents* (pp. 229–240). Berlin / Heidelberg : Springer.
- Roda, C., & Thomas, J. (2006). Attention Aware Systems: Theories, Applications, and Research Agenda. *Computers in Human Behavior*, 22, 557-587.
- Schwerdt, K., & Crowley, J. L. (2000). Robust Face Tracking Using Color. *FG '00: Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000* (p. 90). Washington, DC, USA: IEEE Computer Society.
- Séguier, R. (2004, September). A very fast adaptive face detection system. Presented at the International Conference on Visualization, Imaging, and Image Processing (VIIP). Marbella, Spain.
- Sehaba, K., Courboulay, V., & Estrailier, P. (2006). Observation and analysis of behaviour of autistic children using an interactive system. *Technology and Disability*, 18(4), 181-188.
- EPIDEMIK exhibition - Cité des sciences et de l'industrie (Paris, France). November , 2008, <http://www.cite-sciences.fr/epidemik>.
- Yang, M., Kriegman, D. J., & Ahuja, N. (2002). Detecting Faces in Images: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1), 34-58.
- Zaqout, I., Zainuddin, R., & Baba, S. (2005). Pixel-based skin color detection technique. *MG&V*, 14(1), 61-70.