Full Body Tracking Kendo Game

by

Silviu Dumitrescu, B.Sc.

Dissertation

Presented to the

University of Dublin, Trinity College

in fulfilment

of the requirements

for the Degree of

MASTER OF SCIENCE

University of Dublin, Trinity College

September 2008

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

____

Silviu Dumitrescu

September 9, 2008

Permission to Lend and/or Copy

Acknowledgments

I would like to thank my supervisor Gerard Lacey for all his support and guidance, to Pete Mc Nally for all of his animation help, to all the lecturers who shared their knowledge with us during the course, to Steven Collins who created this amazing course, to everyone in the class who worked so hard during this year for and everyone else who supported me during the hard times of the course.

SILVIU DUMITRESCU

University of Dublin, Trinity College
September 2008

2008

Full Body Tracking Kendo Game

Silviu Dumitrescu

September 2008

Contents

1. Introduction 7

2. State of the Art 9

2.1 Body Tracking 9

2.1.1 Single Camera Approaches 9

2.1.2 Multi-Camera Approaches 12

2.2 Head Tracking 18

2.2.1 Markerless Head Tracking Approaches 19

2.2.2 Head Tracking using Markers 27

3. Design 30

3.1 Sword Tracking 30

3.1.1 Colour Based Tracking 30

3.1.2 Wii Remote Tracking 31

3.2 Body Tracking 31

3.3 Head tracking 31

3.4 Playing Environment 32

3.5 The Game 33

3.5 System Diagram 34

4. Implementation 35

4.1 Sword Tracking 35

4.1.1 Colour Based Tracking 35

4.1.2 Wii Remote Tracking 36

4.2 Body Tracking 37

4.3 Head Tracking 38

4.4 The Game 39

5. Evaluation 44

6. Conclusions & Future Work 46

7. Bibliography 48

8. Appendices 51

1. Introduction

Thanks to next generation games consoles the player's immersion in 3D video games has been taken to a new level. A perfect example is the Nintendo Wii which has broken away from the conventional controller based gaming and introduces a new way of control in video games: the Wii remote or simply "Wiimote". The Wiimote is a wireless controller for the Nintendo Wii console that features motion sensing capabilities, which allows the user to interact with and manipulate items on screen via movement and pointing through the use of an accelerometer and optical sensor technology. Another feature is its expandability through the use of attachments such as the Wii nunchuk or classic controller. The Wii nunchuk is another controller which connects to the expansion port at the base of the Wii remote via a cable. The nunchuk adds an analog stick and two triggers to the Wiimote but it also features its own three axis accelerometer. The classic controller is as the name implies, a classic controller; which plugs into the expansion port of the Wiimote and it is mostly used for playing last generation games (Nintendo GameCube games) on the Nintendo Wii.

This new approach to controlling video games is very different to the usual controllers which used to feature a directional pad, analog sticks (both usually used for controlling movement in a game) and a few buttons (used for actions such as jump, crouch, shoot, reload, etc). Since the Wiimote is a Bluetooth device it can be easily connected to a Bluetooth enabled computer and all of its functions can be easily accessed. This innovation in interacting with video games has led developers to create all sorts of new ways to play them, such as the video game Wario Ware: Smooth Moves for the Nintendo Wii, where the game requires the player to hold the Wiimote in a variety of different ways ("The Umbrella" - holding the Wiimote vertically like one would hold an umbrella handle, "The Elephant" - holding the bottom end of the Wiimote against the nose, emulating an elephant's trunk, "The Samurai" – holding the Wiimote like a samurai holds his sword , etc) . Other examples such as Wii Sports, which features a collection of sports games (Boxing, Tennis, Golf, Baseball and Bowling) that work wonderfully with the Wiimote in creating a virtual tennis racket, bowling ball, baseball bat, golf club and boxing glove. Along with the built in vibration motor and speaker in the Wiimote, the player becomes immersed even more in the game in comparison with traditional controllers.

A different way of interacting with games through motion was created by Sony in 2002, for the Playstation 2, called the Eye Toy. The Eye Toy is essentially a webcam that is connected to the Playstation 2 or Playstation 3 and with the aid of computer vision it allows players to interact with games using motion, colour detection and sound through it's built in microphone. This was yet a movement in another direction in relation to controlling/interacting with games. Although similar to the Wiimote, in the way it changed how games are played; the games released for the Eye Toy required no interaction with any sort of physical controller but only used kinetic energy created by the human body to control the games. Just like the Nintendo Wii, some of the Eye Toy games are focused on physical effort to create more “exciting” work-out exercises which are just like real exercises due to the peripheral free nature of the games.

The aim of this project is to bring these two concepts together and design a 3D kendo fighting game with the aid of computer vision and optical sensor technology as explained above, that tracks the player’s body, head and sword, in order to create a 3D experience that immerses the player into the 3D world, thus bringing the concept of a virtual 3D world that much closer to reality.

As the player moves around, the 3D perspective view changes using Parallax according to the player’s moves (Parallax: an apparent displacement or difference of orientation of an object viewed along two different lines of sight (Wikipedia, 2008)). The player’s experience can be further enhanced by using polarized lenses in order to create a 3D viewing experience. The game’s goal is for the player to defeat the 3D character that is in front of him, in a sword fight. The game environment consists of a life size projection of the enemy character at roughly two meters away. To win, the player must use a combination of moves such as dodge the enemy attacks, block the enemy attacks and attack the enemy (the enemy has a similar set of moves). Once the health gauge of the player or enemy is completely drained the game is over and the player with some remaining health wins.

The tools required for the game are:

- Two wiimotes: one is in charge of tracking the head of the user while the other serves as a sword.
- One camera (can be any USB web camera) which is used to track the movement of the sword and the body of the player
- A “sword”/light saber which acts as the player’s weapon and is tracked by the camera.
- A pair of glasses or/with a set of two infra-red LEDs attached that will be used for the head tracking.
- A Bluetooth enabled computer.

The rest of the paper is organized as follows: Section 2 discusses the latest in head and body tracking techniques, Section 3 discusses the design choices of the system, Section 4 shows how the system was implemented, Section 5 looks at what the system achieved/did not achieve followed by conclusions and future work in Section 6.

2. State of the Art

2.1 Body Tracking

This section is dived into two subsections: approaches using monocular vision and approaches using stereo or multiple cameras.

2.1.1 Single Camera Approaches

Single camera approaches are preferred over multi-camera approaches mainly because the computation needed for multiple cameras is more substantial (thus making the real-time requirement harder to achieve) and it is also a more difficult task that cannot be reproduced by anyone in their home (not everyone has two or more cameras in the house).

Background Subtraction, Silhouette Extraction, Colour approach

The approach in this sub-section focuses on single camera background subtraction, silhouette extraction and using colour information of the body in order to perform the tracking.

In (Chih-Chang Yu, 2007) automated body tracking is achieved from a single camera view. This is accomplished by extracting a 2D model (a silhouette) of the human body from 11 joint points which include: head, shoulders, hip, elbows, knees, hands and feet (Figure 1 explains the segmentation of the body). Firstly Yu et al, performs background subtraction in order to identify the moving body. The difference image is then subjected to a few closing/opening operations in order to clean the image.

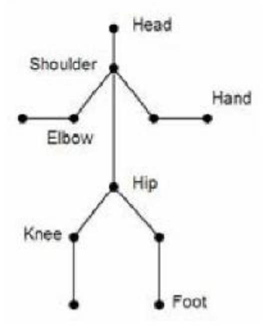


Figure 1 - 2D segmentation of the human body model (Chih-Chang Yu, 2007)

The torso and head are the first parts to be extracted from the difference image and that is done by using negative minimum curvature (NMC) based cuts generation. The head and torso are easier to extract as the head can easily be found in the image by looking for a round shape, and both head and torso can be treated as near-rigid objects. Once the head is extracted the torso can be found with the help of the head coordinates and connectivity (two points belong to the same region if a straight line can be drawn between them without intersecting any silhouette margins). By using connectivity and drawing a line from the center of the head to the contour margin then this line will most likely pass through the torso (Figure 2a). An ellipse is then fit around this region and the major and minor axes of the torso are found (Figure 2b).

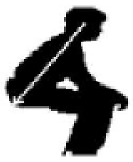


Figure 2 - (a) connectivity from the head

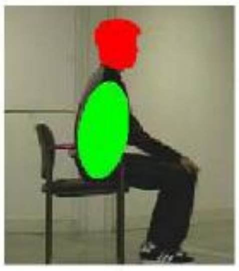


Figure 2 – (b) extracted head and torso

Once the torso is found, the shoulder and hip are also easily found because they would be positioned at the very top and very bottom of the ellipse that surrounds the torso. Hands and feet are the extremities of the body (also the head but that is already found) and are classified in regards to their distance to the shoulders. For elbows and knees Yu et al, classify them as two different sets: hand-elbow-shoulder and foot-knee-hip; then using other points found in the previous steps, connectivity and human kinematic constraints (so that joins only move/rotate in the possible ways). The approach taken by Yu et al is a promising one for segmentating and tracking the body as it only requires one camera and it is very fast. But their implementation does not work if the human is carrying any objects which are required in the Full Body Tracking Kendo Game project (the player must wield a "sword") also the Kendo project does not require identification of any particular body parts, it only needs to know the silhouette of the body so that the 3D character can hit it.

Other approaches

This sub-section contains approaches that use methods other than background subtraction, silhouette extraction or colour information in order to track the body using a single camera.

(Wren, 1997) is a paper referenced by many authors due to its age but also due to the fact that is has been successfully applied to areas such as real time reading of sign language and vision driven avatars. Pfinder uses a priori knowledge about the human body in order to detect /recover from errors and like most tracking systems requires an almost static background working space. The body is created with the help of contours and exploitation of spatial and colour information (expressed in YUV space) of the pixels, blobs are created which map onto specific body parts (Figure 3).

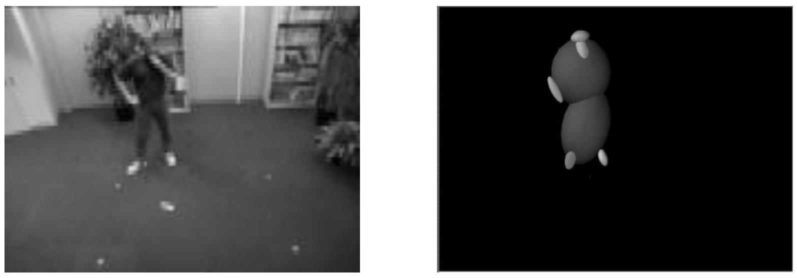


Figure 3 - (a) Video Input (b) 2D representation of the blob

The scene is assumed to be almost static and is regarded as a texture with YUV colour information. The update loop now recognizes the body model and the scene so it must calculate the changes such as: check if a pixel corresponds to a certain blob, whether a piece of the background has changed (moving an object around the scene), detect any subtle changes in lighting/shadows and ultimately assign pixels to either foreground blobs or background texture. The shadowing is resolved by checking the Y value of a pixel, and if it is high then it need not be classified as a shadow, but if it is very dark it can be a possible shadow. This is done by normalizing the chrominance value by the brightness value $U^* = U/Y$ and $V^* = V/Y$. Subtle changes in lighting will also be ignored due to this normalization.

The blob creation steps are: 1) initialize the scene with no person in it, 2) if the scene detects huge changes in it that are not result of noise, then it starts gets a contour silhouette of the new data and creates different coloured blobs according to the different body parts detected. If a blob cannot be created or is lost due to occlusions it gets deleted and when it is detected again then it gets re-added to the body model. The finding of body parts is done in two ways: a contour analysis which can return parts found in a single frame and class analysis which substitutes for contours when they fail to detect parts such as the hands being in front of the body (and therefore creating no contours).

Wren et al developed a body tracking system that works in real time that also allows for small changes in the environment and lighting/shadow conditions; but due to the fact that hardware has changed significantly since the time of the writing of the paper, (i.e. higher resolution cameras) it is unknown if their implementation is suitable for integration with the 3D resource demanding application at hand.

(Nicholas R. Howe, 1999)'s approach to 3D human body tracking uses a single camera and Bayesian probability for the reconstruction of the body in 3D. This approach also uses prior knowledge in the form of a training set in order to track the human body parts/joints in 2D. The tracking is done on a 2D video stream in order to detect the body parts. A body is split up into 14 parts and the tracking algorithm's purpose is to minimize the error between the image data and the projected body parts. Each body part is found by computing a weighted average or the last few previous frames. Since only one camera view is available and occlusions will always happen, Nicholas et al use support maps as described in (J. R. Bergen, 1992). Once the 2D positions of the body parts are computed they can be passed on to the 3D model reconstruction.

The 3D reconstruction takes in the positions of 20 tracked body points (Figure 4) in 2D and must assume the correct depth of each of them. From a pre-learned set of 3D motions it is determined if a reconstruction is possible; possible motions are a mixture of Gaussian probabilities in a high dimensional space. Bayesian probability is applied to the pre-learned training set in order to compute the probabilities of prior different 3D motions. Then the 20 tracked body points are tracked in 3D each frame. This system can only detect short motions and the short motions (called snippets) are composed of 11 consecutive image frames. A rendering function then maps the 3D snipped onto the 2D image coordinate system by scaling, translating, and rotating it.

The system created by Nicholas et al, works quite well on short video sequences and in the case of severe self occlusion it still creates a decent reconstruction. The tracking however requires human assistance in its initialization stage, where the model needs to be overlaid on the 2D image in the first frame, thus making it not completely human assistance free.

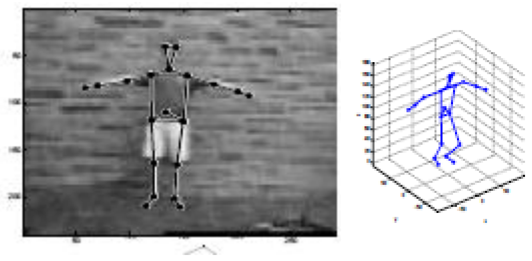


Figure 4 - 2D image and its reconstruction

2.1.2 Multi-Camera Approaches

Multiple camera approaches benefit from disambiguation when the single view becomes confused; having more cameras means seeing the scene from different views therefore self occlusions can be fixed and depth can also be extracted (unlike from a single view).

Background Subtraction, Silhouette Extraction, Colour approaches

The approaches in this sub-section focus on body tracking methods such as background subtraction, silhouette extraction and using colour information of the body in order to improve the accuracy of the system.

In (Kehl, 2005) full body pose tracking is performed using stochastic sampling. The volumetric representation of a body is extracted from silhouettes from multiple image frames. Then this data is applied to an articulated 3D model using SMD (stochastic meta descent).

Kehl et al, (Kehl, 2005), use a voxel representation in order to reproduce a 3D reconstruction or the body. Multiple frames are concurrently being captured from multiple cameras with foreground and background subtraction applied to each. The 3D shape is given by the intersection of the projection cones given by the foreground mask. Background subtraction is performed against each current frame against a “clear” background image (an empty static image of the background) which yields the image in Figure 5.

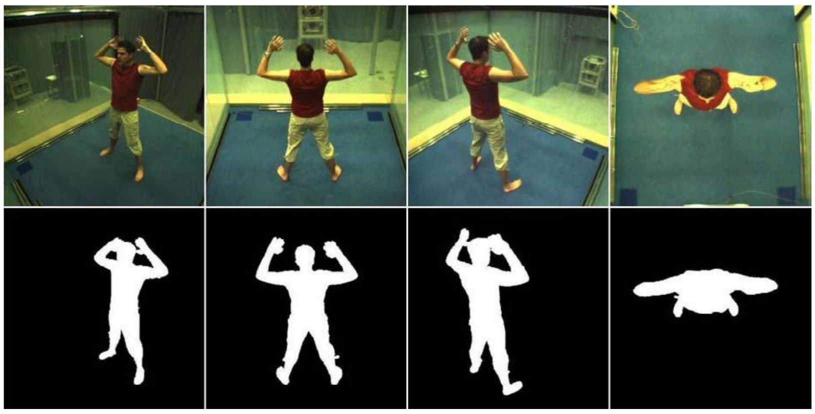


Figure 5 - Foreground/Background Subtraction from 4 Views (Kehl, 2005)

The volumetric representation of the person is created from the intersection of the back projection cones of the foreground masks. Since voxel operations are too expensive, instead of projecting each voxel into the image plane of each camera, Kehl et al do the opposite. They keep a look-up table for each camera view and store a list for each pixel that points to all voxels that project onto that pixel. To speed things up each voxel corresponds to a bitmask where each bit b_i is equal to 1 if its projection lies in the foreground of the camera i and 0 if it lies in the background. A voxel can then be tested if it belongs to the subject if the corresponding bitmask contains all 1's. The 3D reconstruction is done by going pixel by pixel through all the segmented binary images (Figure 1). During the reconstruction stage if a pixel in the current view changed its value from the previous frame, then the corresponding bit b_i for all voxels in the reference list is then updated with the new value. If the reconstruction is not correct due to partial occlusion or overlapping body parts Kehl et al, use colour matching to correct the problem by assigning groups of voxels to regions of similar colour. A depth buffer is used detect occlusions in the person along with a four component representation for the colour (where three components represent the coordinates of the voxel and the fourth is the number of views in which the voxel appears). In order to keep computational costs down all voxels and camera images are only gone through once and the reconstruction is only updated not fully recalculated.

The tracking takes only one frame to initialize and that requires the person to stay in the "Da Vinci" pose. Then an objective function is used to map the voxels to the corresponding vertices of the 3D model (Figure 6). Since there are a large number of vertices in the 3D model (22000) they may not map 1:1 to the voxels. So a subset $T$of tracking vertices t_i is picked. If for each t_i, the closest voxel v_i is needed which gives the following objective function:

$$f(x) = \sum_T ||t_i - v_i|| \qquad (1)$$

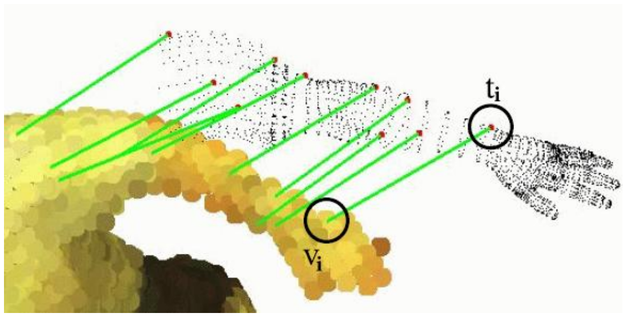


Figure 6 - Objective function of the arm – the green lines represent the assignment of tracking vertices (Kehl, 2005)

The core computation that the paper is based on is the Stochastic Meta Descent (SMD) gradient with local step size adaptation which converges quickly and scales up excellently. The reason for using stochastic sampling of the tracking vertices (rather than deterministic sampling which uses a fixed set), means that a smaller set size can be used, therefore resulting in a speed increase on each iteration. A more detailed explanation of SMD is provided in (M. Bray, 2004).

In Figure 7 the results of SMD with stochastic and deterministic sampling of the points is compared. It can be seen that due to the fixed size of the sampling set, the deterministic sampling looses track at the point where the right arm is missing a large piece. One way of fixing that is to increase the number of samples; but with stochastic sampling, because the sampling set is more sporadic, there is a higher chance that some points will fall in the area of the arm which is missing the large piece. Stochastic sampling has a higher success rate than deterministic sampling when it comes to incomplete data and also because the set size is smaller the computation is also faster.

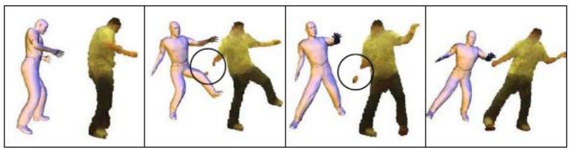
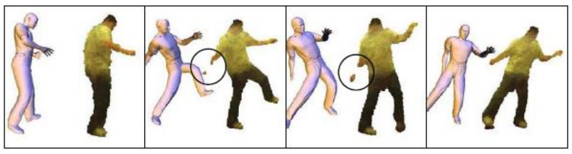



Figure 7a - SMD with stochastic sampling

Figure 7b - SMD with deterministic sampling

Due to the fact that the objective function $f(x)$ in (1) is not very accurate at distinguishing between body parts that are very close to each other and sometimes may stick them to wrong part of the body, Kehl et al also introduced colour based 3D segmentation. Each vertex of the 3D model is assigned a colour at the initialisation stage, after that for each frame the colour is updated according to the last assigned voxel. As a conclusion Kehl et al have devised a fast method that tracks the full human body in different poses in all 3 dimensions. Their algorithm manages to analyse roughly one frame per second. While this is acceptable for offline application recording such as motion capture for animations in movies / games, it does not suit the project at hand that requires real time body tracking at minimum 30 frames per second using only one camera as well.

In (Jing-Feng Li, 2006) an augmented reality system that tracks the upper body of a person without the aid of markers in 3D is proposed. As many other tracking systems, they also use multiple cameras and a model-based representation of the upper body. The image frames come from a "Miniature Stereo Machine" (Yunde Jia, 2003), which is a stereo vision apparatus that can capture stereo images in real-time and calculate the depth maps for those images at rates of 30-50 fps.

The first approach is to compute the background subtraction and depth map, using the Miniature Stereovision Machine, then in order to determine the head position, face detection using Haar-like features is applied. Once the face is detected a colour-histogram is used to find the hands also, which are very similar in colour to the face. The hands are detected using a modified version of the K-Means clustering algorithm, which clusters similar colours together (such as the face and hands). Finally the torso, shoulders and elbows are calculated using kinematic constraints in the body model. The 3D model applied to the upper human body is made up of 8 control points: head, shoulders, elbows, torso and hands (Figure 8). Jing-Feng et al also use common ratios of the human body parts in order to find other body parts once the head and hands have been found (ratios i.e. $W_{TORSO} = 1.4W_{HEAD}, H_{TORSO} = 4H_{HEAD}$ where W and H are the width and height)

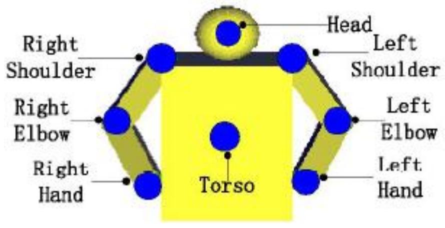


Figure 8 - 3D model applied to the upper human body

In order to cater for the fact that when using colour based background subtraction, variances in light can give inaccurate readings, the background subtraction is applied on both the colour images but also the depth maps (Figure 9). The algorithm responsible for the background subtraction checks each pixel and if the depth of that pixel is less than that of the background then this pixel is part of the foreground.

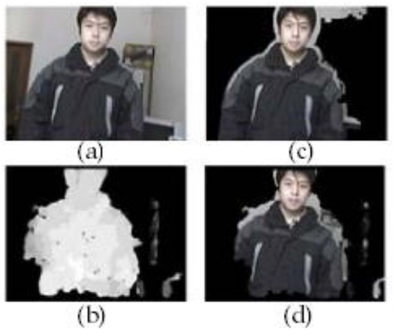


Figure 9 - (a) colour image (b) depth map (c) colour foreground (d) depth based foreground

When the scene gets overly complicated Jing et al, apply a colour histogram based particle filtering algorithm (Stauffer, 1999) to the images to improve the detection of hands and head. In this paper Jing et al, created a 3D human body tracking system which runs at frames of around 25fps, which would be enough for most real time applications but it does not suit the project at hand due to the fact that it is more of a static body tracker and it does not track a full human body (from the waist down).

In (Colombo C., 2001) a stereo vision approach is taken to detect the human body in 3D and then based on the pose of the body a 3D model is created that replicates that pose. The image processing is used to only identify the head, hands, and feet from the binary images. At the image level Colombo et al use a very common technique in human body tracking, background subtraction, which separates the foreground from the background and creates a silhouette of the human body.

The colour based tracking algorithm used to determine the different body parts makes certain assumptions such as: the pants and shoes colours are expected to be different; the only visible skin parts are expected to be the hands and face. The algorithm contains two phases; one that separates background/foreground and one that detects regions that belong to colours classes of interest (face, hands, shoes, pants, etc). Once a point is found from the foreground, a region growing algorithm is used to create the connected human body shape. Next the contour is examined to detect any short range curves which may belong to either head or limbs. This is also where occlusions and lost body parts are corrected using heuristics and stereo triangulation.

Colombo et al use a method known as "symbolic stereo" in order to identify the 3D locations (X, Y, and Z) of the head, hands, and feet based on their 2D (X, Y) locations from the stereo images. Once the 3D positions of the body parts are extracted, they need to be applied to the 3D model using inverse kinematics. The model needs to have the correct number joints and degrees –of-freedom (DOF) and the 3D model Colombo et al is restricted to 8 joints, 9 rigid parts, and 17 DOFs.

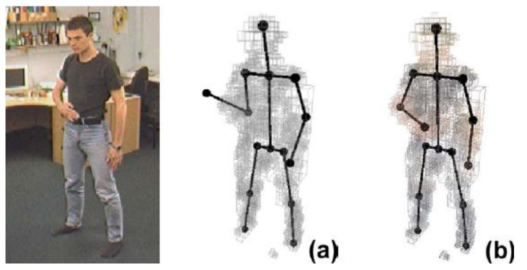
The system proposed by Colombo et al is a fast one, running at roughly 30 fps and with a robust error correction for occlusions, but it is very constraining when it comes to detecting the body (initial body pose plus heterogeneous colours are required for different body parts).

Other Approaches

This sub-section contains approaches that use methods other than background subtraction, silhouette extraction or colour information in order to track the body.

In (Caillette F., 2004) real-time body tracking using coloured voxels and 3D blobs is achieved with the aid of multiple camera views. Caillette et al, reconstruct a 3D voxel representation of the human body instead of using the 2D images acquired from all the cameras used. The 3D reconstruction is done by using visual-hull that is explained in (Szeliski., 1990) and states that the volume of interest is the intersection of the 3D projection of its silhouettes. The system they have proposed can attain real-time processing speeds because it does not compute any binary segmentation on the image frames but instead compute the distance to the background model. Based on this distance voxels are assigned, discarded, or marked as belonging to the foreground model.

One way Caillette et al, managed to distinguish between ambiguous poses and self occlusions is to add colour information to each voxel (Figure 10). The tracking is performed by applying the 3D voxels to a kinematic model with blobs attached to its bones. The blobs are generated dynamically as the tracking loop iterates (Figure 11) and if the blob has no voxels assigned to it (total occlusion occurred) then the blob is removed until the occlusion if removed.



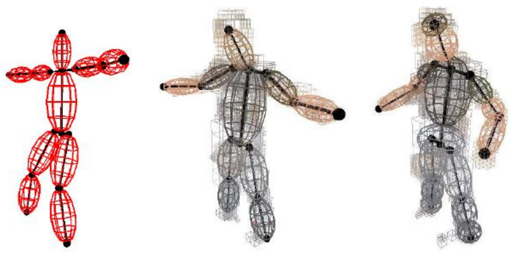


Figure 10 - (a) voxels fitted without colour information (b) with colour information.

Figure 11 - Dynamic blob reconfiguration

Caillette et al did manage to create a “real-time” 3D human tracking method at only 15fps which might be enough for some interactive applications but in the field of games the ideal frame rate is of 30-60fps and the use of multiple cameras is not feasible when integrating the body tracking into the finished product.

In (D.M. Gavrila, 1996) a multi view approach to multiple human bodies tracking in 3D is taken. Their main aim is to recover/track poses and recognize movement patterns. The extracted poses are then applied to a 3D model that has the usual human body constraints (joints, rigid parts, DOFs). Gavrila and Davis like to be able to detect and match more complicated and diverse human poses/motions than other systems (dancing, non-parallel to the camera walking, etc). Like (Wren, 1997) some a priori knowledge of the human body model is assumed in relation to the kinematic and shaper properties of the body.

For the 3D model representation of the human body, volumetric primitives are used to match the body parts (cylinders, spheres, ellipsoids, hyper-rectangles, see Figure 13). These shapes are obtained from the projections of the occluding contours in two orthogonal views, parallel to the zx and zy planes (the person facing the camera, and side-view of the person). The back projection of the 2D contours of a quadric gives the 3D occluding contours which are then searched in order to find the best-fitting quadric. Chamfer matching is used to measure between the fitted and back-projected 3D contours. A similarity measure is used which compares between the actual scene and the model view. Gavrila and Davis use a variation of the chamfer matching algorithm (directed chamfer matching) which gets the directed chamfer (Figure 12) distance between a sample point set T and the reference point set R obtained by summing the distances of each point in T to its nearest point in R (2):

$$DD(T,R) = \sum_{t \in T} dd(t,R) = \sum_{t \in T} min_{r \in R} ||t - r|| \qquad (2)$$

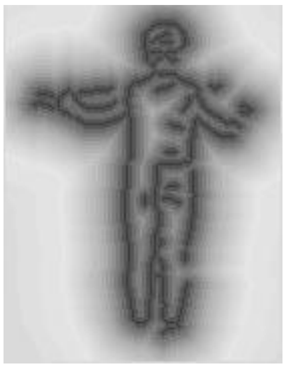


Figure 12 - chamfer image

In order to resolve the problem of not knowing how close the reference points set is to the sample set, an undirected normalized chamfer distance $\overline{D}(T,R)$ is used:

$$\overline{D}(T,R) = (\overline{D}\overline{D}(T,R) + \overline{D}\overline{D}(R,T))/2 \qquad (3)$$

The four cameras used can help in the disambiguation of poses, when one view does not provide with an accurate 3D pose, also the similarity measure described above is applied to all four views. A best-first search technique is used to iterate through the high-dimensional pose data. This is also where more than one human body can be detected (Figure 13).

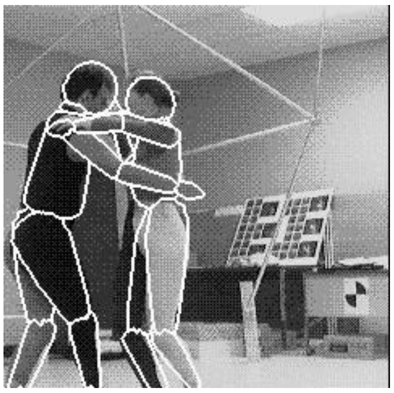


Figure 13 – Two human bodies along with volumetric primitives applied to different body parts

Gavrila and Davis proposed a system which tracks unconstrained human motion very accurately in 3D with the aid of segmentation and prediction. While having more than one body being tracked at the same time, something that is not universal in vision tracking systems, the system would more than likely not suit a fast paced action tracking system like one required for a game and also while being very accurate due to the multi view approach, having more than one camera would definitely put a constraint on the speed of the Kendo system due to the processing power required.

2.2 Head Tracking

This section contains two sub-sections: a sub-section which examines markerless head tracking approaches and a sub-section which examines an approach that uses markers to perform head tracking.

2.2.1 Markerless Head Tracking Approaches

This sub-section examines some markerless head tracking approaches that use either single or multiple cameras.

In (S. Ohayon, 2006) they propose a way to extract 3D position and orientation information of the head with the aid of a single camera. 3D feature points of the head are selected before the tracking begins and the head pose estimation is obtained by solving for a version of the "Perspective n Point" problem (it consists in the determination of the distance between the camera and a set of points well known in an object coordinate space). The system also handles self occlusions and error recovery.

As a first step a set of 3D points from the head are extracted from 2D features in the image, this represents the acquired model. By solving the "Perspective n Point" problem, the distances from the camera centre to the 3D head points corresponds to the recovered model. The head pose is obtained by finding the best rigid transformation which maps the points from the acquired model to the recovered model. The 3D head point set is of a size ≤ 20 and is acquired by viewing the head at different angles. Then these points are transformed into a head centered coordinate frame with $[0,0,0]$ as its origin (the head center). They represent the set $P_M^u = [X_M^u Y_M^u Z_M^u]$ as the set of 3D points for the acquired model. Now the 3D set of points of the acquired model are searched through the 2D image and once the 2D features (p_u) are matched to the 3D model points, the distances from the camera center to the 3D head points are calculated using the PnP problem. $P_R^u = [X_R^u Y_R^u Z_R^u]$ becomes the set of recovered head points and by matching the 3D model points P_M^u and the best rigid body transformation that maps between P_M^u and P_R^u they find out the rotation and translation of the head pose in a current frame. The system can have features initialized manually (the user selecting them) or automatically (the system uses natural occurring features).

Ohayon et al devised a 3D head tracking algorithm that detects head poses using camera pose estimation and it not only works for humans but for animals too (as proved in their experiments). The system can automatically pick features to track or they can be provided artificially by the user and features get occluded they can be tracked again once the occlusion is resolved due to the extrapolation of previous frames that is used in order to predict the 2D future positions of the features in the current frame. However the system manages to run at around 4 fps which is not feasible for a real-time implementation.

In (R. Lopez, 1995) two different head pose estimation methods are proposed: a feature point method and a template matching method. Monocular vision is used and both methods are applied to a 2D greyscale image. The texture of the head is obtained from a Cyberware Head Scanner and is basically "unwrapped" along with the 3D range data (Figure 14).

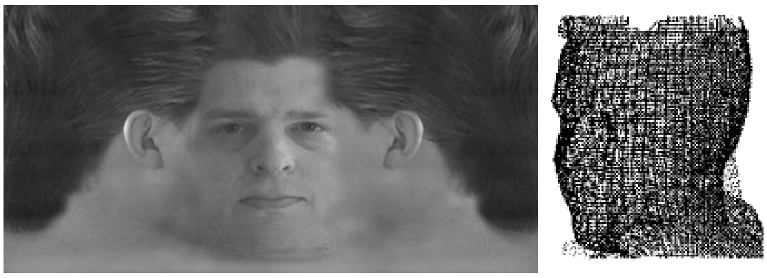
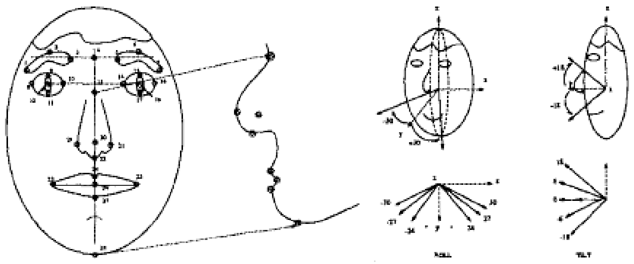


Figure 14 - Unwrapped texture (http://www.worldforge.org/project/newsletters/October2002/CreateHeadTexture) and range data both from Cyberware Scanner

For the feature point method Lopez and Huang use the diagram in Figure 15 to map 3 of the 28 most common points from 3D space to 2D image space.



(a) (b)

Figure 15 – (a) The 28 points used in feature point tracking, (b) Angles used in constructing the template database

The template matching method creates a database of templates that represent the head model in 3D at different poses and scales. Next, an algorithm is created that finds an association between the 2D image and the templates in the database.

Lopez and Huang devised two methods of estimating a human head pose in 3D. Of the two, the feature point method was unsurprisingly a lot more robust, quick and accurate. That is due to the fact that the template matching method is a lot more expensive as a result of the step that must build the database. Their approach cannot be applied to the project at hand as their input comes from a head scanner and not a standard camera.

In (Moreno F., 2002) real-time 3D head tracking is presented using an ellipse along with a trained colour histogram of the skin and hair samples. The colour histogram is used to detect the skin and hair samples and this is also responsible for the taking into account illumination changes in the scene. Stereo vision is then used to retrieve information about the head such as location in 3D, depth and scale. In order to update the colour histogram the current image is read in from the camera and checked against the last image to determine any differences; this computes the new position of the head also. The stereo images continuously updates the position of the head in 3D, in relation to its last know position from previous frames. The Kalman filter takes this information from the histogram and stereo vision and computes an estimate of the difference in scale and position of the head compared to the last estimate.

The colour histogram step is being constructed by starting with the detection of the human head (as done in their previous work (F. Moreno, 2001)). This returns an ellipse containing the initial position and scale of the head. Once the tracking begins and images are continuously pulled from the camera, the initial histogram is compared against the newly created one for the current image. In order to speed up the creation of the new histograms, whenever the ellipse from the current image intersect the ellipse from the previous image, the common pixels can be used in the new histogram and then just add the new ones.

The depth step extracts the head position from the left image of the stereo setup and by restricting the search algorithm to only those pixels inside the ellipse and the pixels on the right image that have a disparity centered on the previous disparity value, a real-time execution can be achieved. The “fusion” of the updated histogram and head position is done with a Kalman filter and the result is a state vector.

Moreno et al devised a system that tracks a human body in real-time (30Hz) using colour and depth information. Their tests were done on a video feed with a resolution of 160x120 pixels. For the project at hand a resolution of at least 320x240 will be used for which the computation timings are undetermined; and information on the rotation of the head is also required, something which is missing from Moreno et al’s approach.

(Schodl A, 1998) proposed a system that maps the human head (as a texture) under perspective projection onto a 3D model in order to track the face (Figure 16). The head’s texture is projected onto the 3D model and rotation and translation parameters are determined in order to perform the tracking. This approach is helpful because once the human head is modelled as a 3D object; it is then easy to obtain rotation information about it, unlike in (Yuan Li, 2006).

A 3D head model is rendered using OpenGL and the live video stream needs to be applied to that 3D model using the image gradient and derivatives of the rotation/translation parameters. The transformation from 3D to 2D is given by the transformation matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & f^{-1} & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot R_z R_y R_x \qquad (4)$$

Where R_x, R_y, R_zbeing the rotation matrices for the x, y, z axis, $-t_x, -t_y, -t_z$ being the translations along the axes and f is the focal length. Schodl et al came across problems such as when rotating the head around the y-axis, it has a similar effect to translating it around the x-axis with the only difference being a perspective distortion. That problem is solved parameterization of the model with uncorrelated features sets.

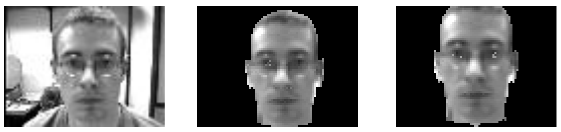


Figure 16 - (a) input image, (b) rendered model, (c) reprojection onto the model

While the system proposed by Schodl et al has its flaws such as incorrect measurement due to illumination changes/specular reflections, large errors when the rotation angle is too great, non real-time execution; it is still an interesting one because it retrieves information such as translation and rotation of the head both which are required for the project at hand.

(Terada K, 2005) implemented a system that tracks the human head in 3D using stereo vision and applying a particle filtering algorithm. This approach is similar to (Schodl A, 1998) in the way it creates a 3D model of the head. The particle filter algorithm is applied to a set of depth map images from the stereo cameras. The 3D head model is created by a polygon mesh that contains information of the human head such as skin and/or hair colour. This geometrical and colour information is used to determine the head position and orientation, with head position done first, followed by orientation. Terada et al assume a few things such as: focal length and camera separation (individual camera from the stereo setup) need to be knows for the depth map and observation model and also the geometrical and colour information of the head needs to be known. The observation model consists of a 3D head model which is a polygonal mesh with colour information that represents the human head (this model is overlaid on top of the actual live image Figure 17 and is extracted from a range scan of the actual human and it contains 50000 vertices); 3D surface observation which is extracting the 3D coordinates of a point from the depth map along with its colour information; computation of the position tracking; computation of the orientation estimation.

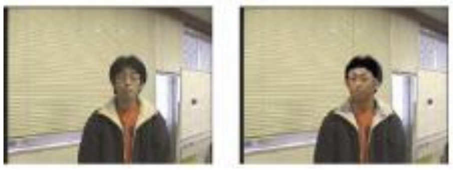


Figure 17 - actual image from camera – estimated pose using 3D head model

Terada et al devised a system that uses geometric and colour information from stereo vision with the help of depth maps, in order to estimate head position and orientation. The technique is a novel one but the non-real time usage and the head scan that means the 3D model of the head can only be used on a person that had their head scanned, makes this approach not feasible to use with the Kendo project.

The approach in (Rougier Caroline, 2006) is one of interest as it uses a single calibrated camera, and real-time 3D head tracking. As in many similar implementations the head is approximated by an ellipse in the camera view. To detect the head an algorithm is used that takes in anthropometric data of a human head, the 2D points in the camera plane that correspond to the head and the fundamental camera settings such focal length, coordinates of the principal point and distortion coefficients. This algorithm gives back the position of the head in the camera coordinate system which is then transformed to the world coordinate system.

Like in (Terada K, 2005) a particle filter is used to track the ellipse around the scene, using colour and gradient information. But unlike (Terada K, 2005) three different filters are used: the first filter checks if the head is almost in the same place (if very little movement was detected), the second filter is used if the first one fails and it looks for the head (approximate position) farther away, the third filter corrects the last knows position using a weak noise. From all of the positions of the ellipse a 3D trajectory is built which when analysed detects a fall.

The system created by Rougier et al has its advantages such as using monocular vision and being real-time but it is also prone to errors like losing the ellipse for the head at time, and since the ellipse must be initialized by hand at the beginning it is impossible to automatically detect the head again once it has lost track of it.

In (H. Kawanaka, 2006) another particle filtering head tracking approach is developed. This approach uses four stereo cameras in order to create a voxel representation of the head as unlike in 2D image tracking, where there is a need to constantly check the shape and size of the head while tracking, the head shape does not change. The voxel representation is created from the depth maps created by the four stereo cameras and in order to maintain a real-time operation the resolution of the 3D voxel shape (Figure 20) is kept low (voxel size = 6x6x6 where 1 voxel = 3.5mm).

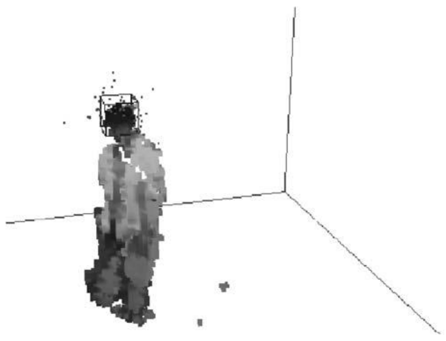


Figure 18 – Low resolution voxel representation of head and body

Colour information such as hair colour and skin colour is used to detect the head but since that is not enough, distance information from the depth maps is used to make a distinction between foreground and background objects. Particle filtering is used for the head tracking inside the 3D voxel space. Based on past and present observations of the tracked object, a Bayesian estimation is created which is then calculated as the likelihood function (that is in turn calculated from a location histogram and a colour histogram).

Kawanaka et al created a robust and fast system that tracks the head in real-time using four stereo cameras and a particle filtering algorithm for the head tracking. However the ease of use of the system and the hardware requirement is too great for it to be used on a large scale with consumer orientation in mind.

(Yuan Li, 2006) use a multi-state particle filtering technique in order to achieve real-time head tracking. Unlike (Terada K, 2005) this method uses a learned observation model along with a head observation model and a probabilistic tracking framework. The face observation model is trained with large data sets and the head observation model uses colour histograms and contour edges.

The multi-face observation model uses Haar-like features to determine where the face is in a frame and it covers $\pm 90°$ out-of-plane rotations and $\pm 45°$ in-plane rotations. Because it uses a probabilistic approximation of how likely an image patch resembles to a face, the face can also be tracked when occlusions or other factors that affect the quality of the image(such as camera vibration, cluttered background or illumination change) occur (Figure 18).

The head observation model uses colour histograms inside the head region (the head is modelled as an ellipse) and an intensity edge contour along the head.

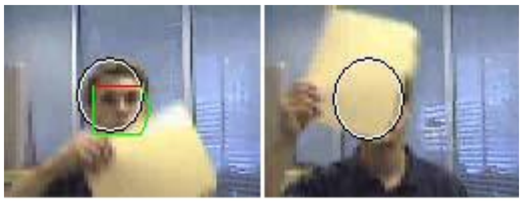


Figure 19 - Head tracking working even under severe occlusion

Li et al managed to create a head tracking method that not only tracks the head in a sequence of images but also detects a face pose even when the image is severely affected by a blur due to camera vibration, severe occlusion or illumination changes; all of this while maintaining a real-time execution. The head tracking under occlusion is an important detail for the project at hand as it is a likely scenario to occur due to the sword appearing in front of the head thus partially occluding it. However the face pose estimation may not provide a accurate orientation for the head which is needed for creating a correct 3D perspective view.

In (Nanda H., 2004) another ellipse based head tracking method is described. Their motivation is to efficiently track the human head in a cluttered environment and also when it is partially/completely occluded. The elliptical head tracker provides the position and depth of the head in a scene. The ellipses are detected and tracked using depth maps and chamfer distance then an edge detector is applied to the depth maps to determine the depth differences within the scene.

The head is modelled as a 2D vertical ellipse with an aspect ratio of 1.2, which is then just searched for within the scene until a match has been found. One limitation that arises from this method is that if another object that is not the head is present in the scene and it resembles an ellipse, this will cause a false positive match. A binary shape is overlaid on top of the gradient image and using the chamfer distance, the distance transform image can be extracted (Figure 19). When using the distance transform edge orientations are not taken into account, so a lot of false positive matches can occur when overlaying the binary shape on top of the gradient image. To fix this Nanda and Fujimura split the gradient image into multiple channels, with each channel being treated as a new image that holds orientation information of the edges (horizontal or vertical edges); then they apply the chamfer distance on the multiple channels of the image.

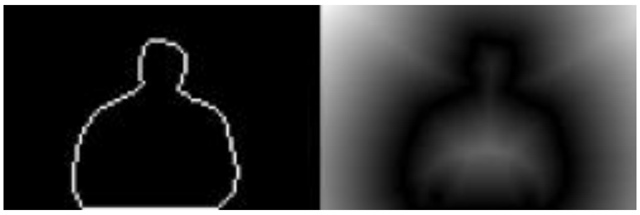


Figure 20 - Gradient Image,   Distance transform image

Head tracking is done by predicting where the head will be in the next frame. Given that it is known where the head position is at time t and time $t-1$, the position of the head at time $t+\ 1$ is predicted using constant velocity assumption.

Nanda and Fujihara have created a head tracker that successfully detects, tracks the human head in 90% of the time while also dealing with partial or complete occlusions. The only drawback is that it requires depth images as input which are usually obtained from multiple cameras but in their case were obtained by a sensor (greyscale images can be also be used but the detection/tracking accuracy is not as good).

(Xu X., 2005) features another particle filter approach to head tracking, using an ellipse for the head shape along with an intensity gradient for it and a colour histogram for the interior of the ellipse.

The particle filter based head tracking is done as in other research papers mentioned, where the head shape is modelled as an ellipse with its center at (x, y) and of size (H_x, H_y). The initial position of the ellipse can be either automatically initialized using a face detection method or manually initialized if the face is not completely visible. Then the head properties are described by a state vector S_t

$$S_t = \{x, y, X_v, Y_v, H_x, H_y, H_{vx}, H_{vy}\}$$

with (x, y) being the center of the ellipse, (X_v, Y_v) being the motion velocity of the head, $\left(H_x, H_y\right)$ being the lengths of the half axes of the ellipse and (H_{vx}, H_{vy}) being the scaling of the axes. Since particle filtering is a maximum estimation method based on past and present observations, a set with correctly weighted random samples of the state vector S_n, needs to be created; $\left\{s_t^{(n)}, \mu_t^{(n)} \middle| n = 1 \ldots N\right\}$ where $s_t^{(n)}$ is a sample of the state vector and $\mu_t^{(n)}$ is its given weight. There are three steps that need to be applied in order to perform the head tracking: a sample selection step which extracts N particles at time $t-1$ from the sample set based on their weights, in order to create a sample set at time t; a prediction step which propagates each sample according to the system model $S_t = AS_{t-1} + N_{t-1}$; and finally the update step which calculates two separate weights for each sample; one weight that is based on the intensity gradient and one weight that is based on the colour histogram. The two different weights are then averaged into one weight before being passed in the mean estimation.

Xu and Li's have proved that by combining the intensity gradient with the colour histogram a much more accurate system can be developed which can track a human head through a 360 degree turn (Figure 21) even in a cluttered background, something which the individual components cannot achieve (the colour based trackers can easily return false positives when items of similar colour with the object being tracked, are present in the scene and in edge based approaches can return false positives when the background is very cluttered). However the one thing that is not mentioned about this approach is its real-time performance which is essential for the project at hand.

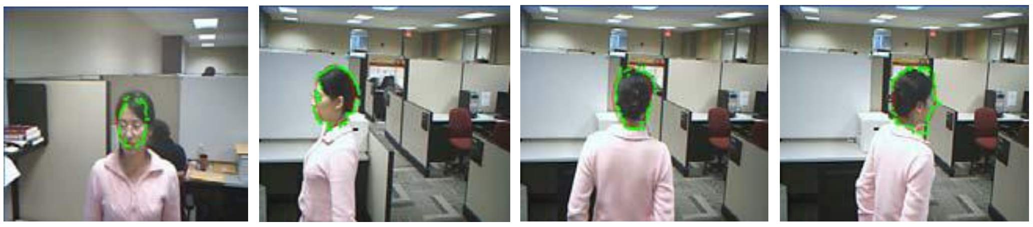


Figure 21 - 360 degree results of Xu and Li's implementation

(M. La Cascia, 1998)'s approach to head tracking is similar to (S. Ohayon, 2006)'s in the way that a texture map of the head (Figure 22) is extracted from the 2D image data, and is then applied to a 3D surface model. The 3D model of the head is being updated by image mosaicking. The system takes as input 2D video frames and outputs 3D head parameters along with a dynamic 2D texture map of the head all while solving for self occlusion, illumination changes, specular lights.

The 3D surface model which represents the head is assumed to be a cylinder with a 360 degree view but only 180 degrees will actually be mapped by the texture map. Only the part of the head visible to the camera will be mapped to the texture map, the non visible part is just drawn black (Figure 22). Over a number of frames the dynamic texture map that is applied to the 3D surface model, helps build a mosaic of reference textures which help in tracking the rotation of the head.

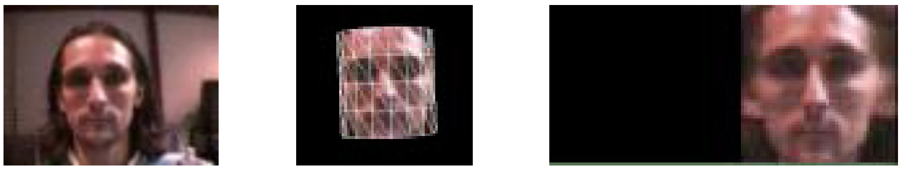


Figure 22 – Input image texture map and 3D model texture map image

There are three different coordinate systems which must be mapped in between; a (x, y, z) coordinate system for the 3D model (the cylinder), a (u, v) coordinate system for image space and a (s, t) coordinate system for the texture map. Since the mapping from image space to texture map space is not 1:1, a warping function is created to map to and from each coordinate system. The tracking is then performed in the texture map coordinate system. A confidence map for each pixel is created when mapping pixels from image space to texture map space due to perspective projection. An approximation of confidence can be obtained from the ratio of a triangle’s are in image space over the triangle’s area in texture map space.

Once the 3D head parameters, dynamic texture map and confidence map are obtained the image mosaicking can be performed. That is done by mixing the mosaic with each new frame and if a pixel in that frame has a higher confidence then it is replaced. The mosaicking improves the tracking and also allows for larger rotations of the head.

The method for head tracking developed by La Cascia et al is able to detect facial expressions along with head tracking but it has a few drawbacks which do not make it a right candidate for the project at hand. The initialization of the tracking must be done manually; error recovery is almost nonexistent, so when the tracked head is lost it must be selected again manually and the processing time is too slow for a real-time application because it contains such intensive operations (mostly due to the dynamic texture map creation).

2.2.2 Head Tracking using Markers

This sub-section focuses on a head tracking method that uses markers on the user’s head and the advantages it has over its markerless counterpart.

In (Lee, 2007) the approach to head tracking is done in a totally different way than all the head tracking papers discussed above. Johnny Chung Lee has taken existing technology which is usually used in a different way and built his own real-time head tracking method which effectively works at 100Hz, faster than any web camera out there. The only difference in his approach is that his is not a markerless head tracking approach; the user is required to wear two infra-red LEDs which are then tracked by a camera.

The basis of Johnny Lee’s idea came from the Nintendo Wii, a games console that uses as a controller a one handed pointing device that detects movement and orientation in three dimensions, the Wii remote (Figure 23). The Wii remote has a few components built-in such as an accelerometer, a vibration motor for force feedback in games; a small speaker, an expansion port through which different peripherals can be connected; a 16 Kilobyte EEPROM chip and the part that is responsible for the pointing device features: a camera with an infra-red block in front of it so it detects IR light only. The Wii remote camera is has a resolution of 1024x768 pixels and supports blob tracking of up to 4 blobs at 100Hz but unlike standard cameras the blob tracking is actually performed by hardware on board the Wii remote itself. Because the of the success of the Nintendo Wii and its broad installed base, the Wii remote is found in many homes around the world and because it is so available it makes for a great head tracking device for anyone that owns a Nintendo Wii (or even without a Wii being priced at around €50 it still makes for a nice cheap head tracking solution). The Wii remote is a Bluetooth device thus making it very easy to connect to a Bluetooth enabled computer and due to the phenomenal success of the Wii some people developed their own open source Wii remote libraries which can be used with a variety of programming languages so an official Nintendo Wii SDK is not needed and everyone can get coding right away.

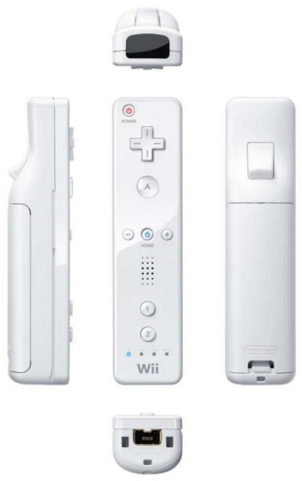


Figure 23 - The Nintendo Wii remote from different views

The Wii uses the remote as a pointing device, much like a mouse cursor with the help of a static sensor bar (Figure 24) which is usually placed above or below the monitor. The user then moves the Wii remote around and the camera reads the new positions of the IR LEDs (which do not change but the camera in the Wii remote changes thus making it appear as if the IR dots are moving).

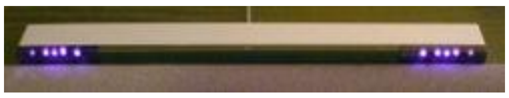


Figure 24 - Nintendo Wii Sensor Bar with highlighted IR LEDs

In order to perform head tracking using the Wii remote and IR LEDs Johnny Lee reverses their role; the Wii remote remains static on top or below the screen, while the IR LEDs move around along with the head of the user. The IR LEDs are placed on the user's head using a pair of glasses with two IR LEDs attached to the sides (Figure 25) for added mobility and comfort.

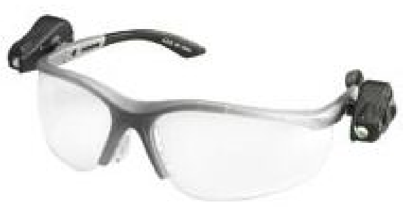


Figure 25 – Safety glasses with IR LEDs attached

The head tracking achieved by Johnny Lee is able to detect and track the (x, y, z) positions of the head. This is how it is performed: the Wii remote must be able to see two different blobs (the light emitted by the IR LEDs); that is because the Wii remote can obtain various information about those blobs such as (x, y) positions in the camera coordinate space, size of blobs from which other data can be derived such user's head distance to the camera (the Wii remote), and once that distance is found the x and y position of the head can be found next (the calculations will be explained in detail in the implementation of the head tracking into my system).

3. Design

In order to create a realistic experience for the user, the game must feature elements of body tracking; head tracking; sword tracking; which help in the immersion of the player into a virtual environment but also help in the creation of the game. The design section is made up of five sub-sections: sword tracking, body tracking, head tracking, the playing environment and the actual game.

3.1 Sword Tracking

The system needs to be able to detect and track the player's sword such that realistic moves can be performed (i.e player can only hit the 3D enemy character when the sword is close to the enemy, the sword is within viewing range of the camera and the player performs actual sword swings).

3.1.1 Colour Based Tracking

The player's sword is composed of two parts: a sword adapter for a Wii remote which safely and comfortably holds the Wii remote in its handle and features a "light saber" design with 22 ultra bright LED's fitted in a tube that is attached to the Wii remote handle (Figure 26). The GloSword is powered by 3xAA batteries and features an on/off button on the handle.

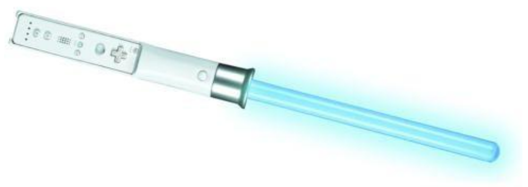


Figure 26 - The GloSword created by Gamexperts, used at the player's sword

For this project, the most important feature of the GloSword is the 22 ultra bright LEDs which emit a distinct blue light. This light will then be used as the input colour for the colour based tracking. Because the light is so distinct, it is highly unlikely that the system will get false positives from the background or the clothes of the player.

3.1.2 Wii Remote Tracking

The motion sensing capabilities of the Wii remote are used to detect movement in the actions of the player's sword. A Wii remote (Figure 27) contains a 3g[1] accelerometer which measures acceleration in the 3 axes and also tilt (roll, yaw and pitch). Because the accelerometer operates at 100Hz any movement of the Wii remote, no matter how subtle, is detected in real-time and transmitted wirelessly to the computer via Bluetooth. The Wii remote will be used to detect actions of the player, such as whether the player is blocking or attacking.

3.2 Body Tracking

The player's body needs to be tracked as it moves around the playing area (Figure 30), in order to determine if the player is within range of the camera. If the player moves out of the viewing range of the camera the game needs to be paused, and then resumed when the body is detected again. Body tracking is always susceptible to errors due to external factors such as background noise, changes in illumination, incorrect foreground/background segmentation so an appropriate method must be chosen and when it loses track of the body it must automatically re-acquire it fast and without any human interaction.

A Logitech QuickCam 500 USB camera with a resolution of 320x240 pixels that captures video at 30 frames per second is used to detect and track the body; but since the same camera is also used to track the sword the system must still be able to compute all this information in real-time. The body position is used to determine if the 3D enemy character can hit the player so it must be very accurate also the player must be able to move freely around the playing area without being constrained by any further equipment such as markers to aid in the tracking of the body.

Figure 27 - Wii remote that is inserted into the GloSword

3.3 Head tracking

Head tracking involves determining where the head of the player is, in three dimensions, in relation to the camera. These readings must be very accurate and real-time because the head tracking is what actually changes the 3D perspective in the game; so if the head tracking is incorrect or too slow it will break the immersion of the player into the game. The head tracking is responsible for what the player sees on the screen and that controls

[1]g or g-force = measurement of an object's acceleration

the camera inside the game; so for example if the player moves away from the screen, everything in the game seems further away like in real life; if the player's head is turned left or right, the view in the game also changes allowing the player to view more of the scene, much like a window into another world.

After considerable research into a markerless approach to head tracking, a non marker-free approach has been chosen due to its extraordinary response time, ease of implementation and incredible pin-point accuracy. This approach adds a very subtle piece of equipment over the user's head, a pair of glasses (Figure 28) that act as a placeholder for two infra-red LEDs which are tracked by the Wii remote.

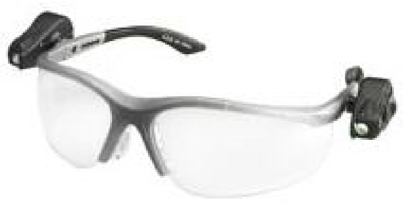


Figure 28 - Glasses used for head tracking with IR LEDs on the sides

A second Wii remote is placed under the USB camera on top of the screen which tracks the two IR LEDs on the head of the player. The amazing feature about this approach is that in computation terms, the cost of the head tracking is almost nil due to the fact that all of the tracking/detection computation regarding the IR LEDs is done by hardware onboard the Wii remote (there is no need to pass in images and track the two LEDs since the Wii remote can only see infra-red lights and the two IR LEDs will be the only IR lights in the scene (Figure 29)); the computer simply receives the data and does not need to compute it. By alleviating this computation, the computer can spend more resources on the more expensive image processing data.

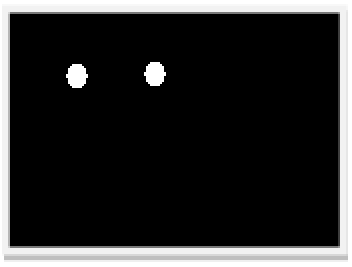


Figure 29 - What the Wii remote sees - the white dots represent the IR light detected

The IR glasses can obtain (x, y, z) position information of the user's head. These values are used to manipulate the camera in the 3D environment and also since the z value represents depth (how far the player is from the screen) that will be used to determine if the player is within hitting range.

3.4 Playing Environment

The playing environment represents the area in which the player may operate. This area is defined by the viewing range of the USB camera and the Wii remote performing the head tracking. If the player moves outside this boundary the game will pause until he/she returns inside the designated area (Figure 44). The component setup is as follows: a large screen preferably large enough to project a life size image on (the one used here is 2x3 metres), a projector with a resolution of a minimum of 1024x768 pixels (Figure 31 item 6.), a USB camera which must be placed on top of the screen and in the middle of it too (Figure 31 item 3.), a Wii remote which must be placed beneath the USB camera (Figure 31 item 4.), two IR LEDs which must be placed on the user's head that must be at least 8 cm apart (the IR LED glasses Figure 31 item 7.), a blue light source (the GloSword) and a 2nd Wii remote both which serve as the player's sword (Figure 31 item 2.).

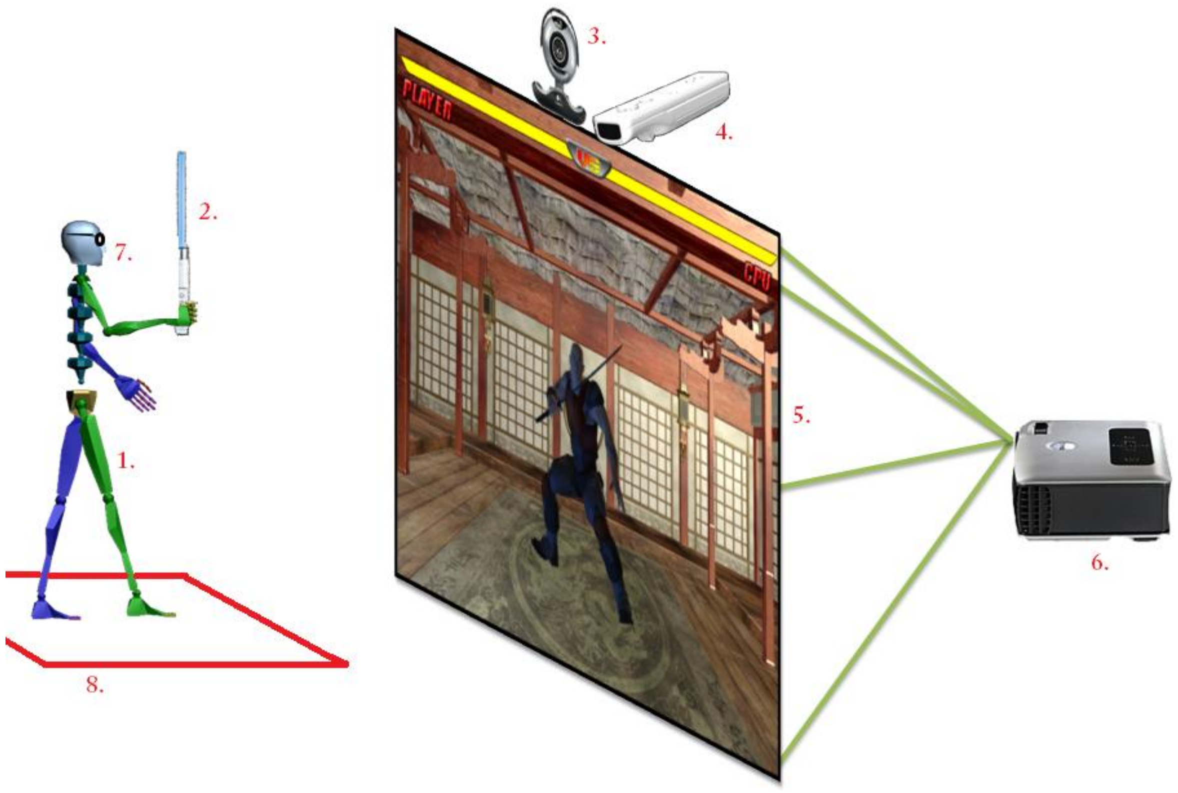


Figure 30 - Component setup: 1. The user, 2. GloSword+Wii remote, 3. USB Camera, 4. 2nd Wii remote, 5. Large screen, 6. Projector, 7. IR glasses, 8. Playing area.

3.5 The Game

In order to demonstrate the three different types of tracking and the real-time requirement a short game must be created that implements all these components. The game must be able to combine these elements in such a way that the player becomes immersed in it. In order to add to the immersion factor sounds and force feedback will also be implemented in the game.

In the game the player will have a first-person view of the 3D environment and he/she will be fighting a 3D enemy character. The purpose of the game is to defeat the enemy in a sword fight. Both the player and the enemy start with full health bars, whenever one inflicts damage upon the other, the one who got hit will have some health deducted from his/hers health bar; whoever's health bar reaches zero first, then that is the loser. The player will need to be able to attack, defend, and dodge enemy attacks while the enemy will also need to be able to perform these actions.

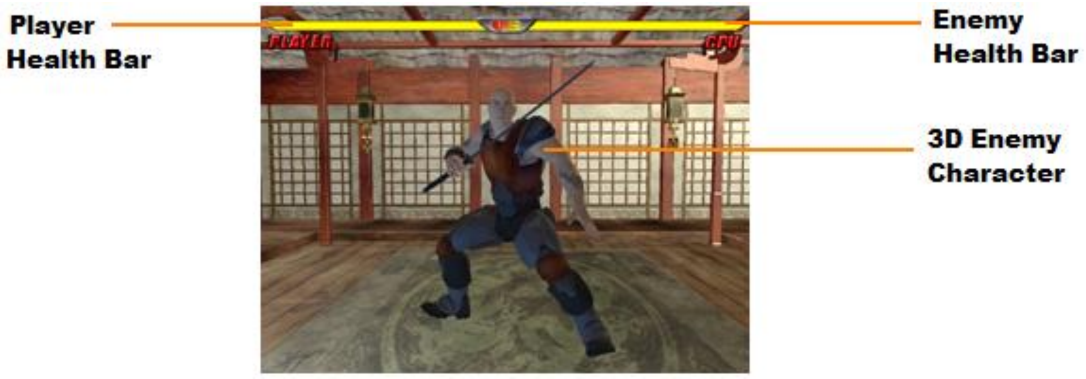


Figure 31 - The game that will be displayed on the large screen

3.5 System Diagram

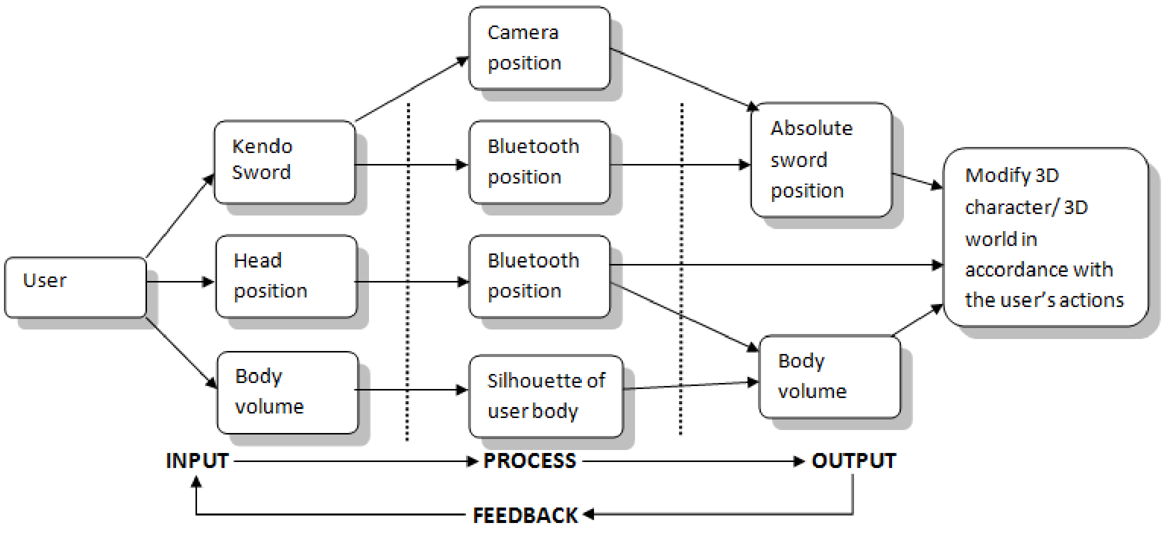


Figure 32 - System diagram that represents the different inputs-outputs of the system and the process they all undergo.

The system diagram shows the flow of the data from its input state through the processing stages, to the output state and after which it is passed into the input state again. The inputs are the three key tracking objects: sword, body and head; these inputs are then examined by having their positions calculated; based on those positions the game is then updated and the output becomes the input for next iteration of the system.

4. Implementation

This section describes in detail what design decisions actually made it into the final system and the methods used in implementing those decisions. The full system features different types of tracking and a game loop which must be continuously rendered at 60fps in order to preserve the real-time requirement of current games. These different types of tracking have to be merged into the one robust system that is able to compute and update them in real-time; this section explains how that is achieved.

The implementation section is divided into four sub-sections sword tracking, body tracking, head tracking and the game.

4.1 Sword Tracking

The player's sword is tracked using vision and also using the Wii remote's acceleration/tilt values; vision checks where the sword is (position and rotation in two dimensions) and the Wii remote can check rotations in three dimensions and also acceleration. If the sword is not detected by the camera the game is paused and then resumed when it comes into the view again.

4.1.1 Colour Based Tracking

In order to detect the player's sword, a colour based detection/tracking method is implemented. That is because it is not computationally expensive and due to the fact that the sword is of a very distinct colour, it cannot be easily misinterpreted by the camera. The colour is detected by thresholding. An input image examined and for each pixel that exhibits a higher RGB value than the threshold, that pixel's original RGB values are copied into the new image and if the pixel's RGB values are below the threshold value then they are set to black (Figure 33b).

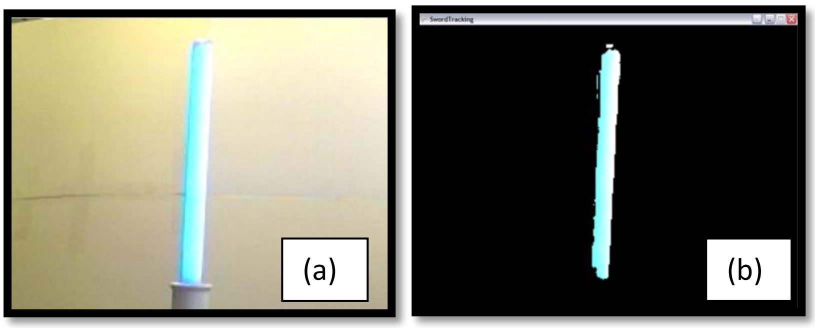


Figure 33 - Colour based tracking: (a) input image; (b) detected colours

Next, a minimum bounding box is created that surrounds the detected sword colours (Figure 34). With the aid of the bounding more data can be obtained about the sword (in two dimensions (x, y)) such as width, length, area, angle and position. The bounding box is created using the $minX, maxX, minY, maxY$ coordinates of the detected sword (Figure 33b).

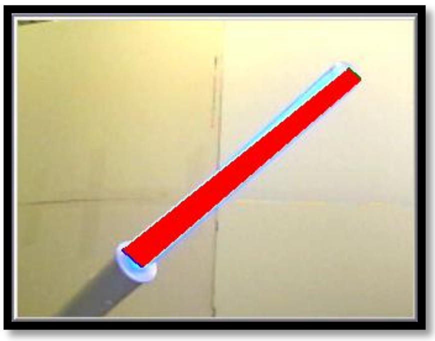


Figure 34 - Minimum bounding box of the sword

Because this method is sensitive to illumination changes, if a sudden big change in lighting conditions occurs, the tracking will not be as accurate. One way used to increase the efficiency of the tracking algorithm is to perform the bounding box calculations for the body tracking, at the same time as the sword bounding box. This eliminates the need to iterate through the whole image again while looking for the $minX, maxX, minY, maxY$ of the body bounding box. All these computations are performed in the ImageProcessing class (Figure 34).

4.1.2 Wii Remote Tracking

Tracking of the sword is also done with the Wii remote that is inside the GloSword handle. Thanks to the built-in accelerometer the Wii remote can read acceleration and tilt values in real-time to a very high degree of accuracy (Figure 35).

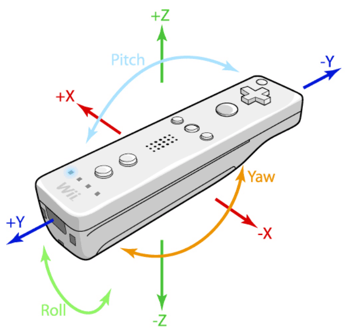


Figure 35 - Wii remote axes for acceleration and tilt

The tilt information is used to determine where the sword is facing on the three axes; needed in order to know if the player is blocking. Blocking can be of three types: the player holding the sword vertically in front of him/her, holding the sword horizontally left and right (Figure 36). To check if the player is blocking in any of the three ways, a simple check is done on the Wii remote tilt values and if it is within the allowed range then the player is blocking.

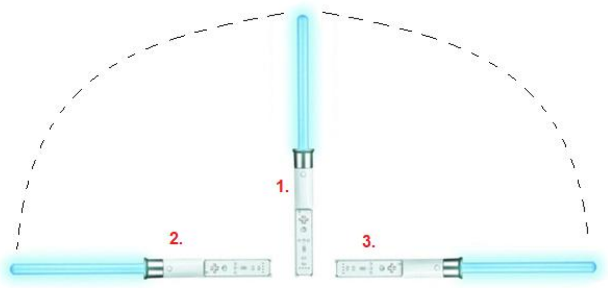


Figure 36 - The three different types of blocking: 1. left horizontal, 2. vertical, 3. right horizontal blocking

The acceleration values are used to determine if the player has performed an attack. Acceleration values usually range from 0 to a value that is determined by the velocity of the Wii remote. If a player takes a swing, acceleration values in the three axes are returned, and if the sum of the values is equal of higher to the threshold value, then the player has performed an attack. The Wii remote calculations are performed in the KendoGame class (Figure 34), along with all other Wii remote operations.

4.2 Body Tracking

The body detection/tracking is required to determine if the player is within the playing area and also if the enemy can hit player. As seen in (Colombo C., 2001), (Kehl, 2005), (Jing-Feng Li, 2006), (Chih-Chang Yu, 2007) body extraction from a scene is performed by using background subtraction. Once the background subtraction has been performed what is left is the silhouette of the body. Like any background subtraction methods the scene must be empty when initialized, then the first frame that is captured by the camera is subtracted from any subsequent frames. This will result in any new objects (the body), which were not in the scene at the first frame, to be detected. Next a way to track the body is required and in order to do that Freeman chain contours are used on the difference image. Before the chains are applied, the difference image is converted to greyscale and has standard image operations applied to it (erode, dilate, etc.). This will surround the body in the scene (Figure 37). An axis aligned bounding box is then created from the chain contours which tracks the body around the scene in two dimensions. The bounding box is created in a similar way to the bounding box of the sword but it is not a minimum bounding box so it requires only two points to be created $(minX, minY)$ for top left corner and $(maxX, maxY)$ for the bottom right corner.

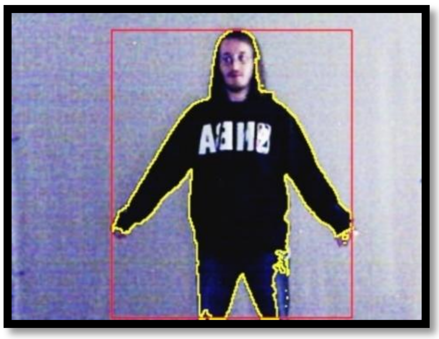


Figure 37 - Body tracking: yellow represents the Freeman chains and red represents the bounding box

Using the bounding box it is possible to find out what the (x, y) position of the body is, if the body is still in viewing range of the camera, and if not the game is paused and then resumed when the body enters the view again. The body tracking is also performed in the ImageProcessing class.

One important detail to note about the ImageProcessing class is that it runs on its own thread. This was done because the camera runs at a fixed frame rate (15 or 30 fps) and the application must run at 60fps+ in order to run in real-time. If the ImageProcessing and game loop is executed at the same time then the game is reduced to update at the speed of the camera (which is too slow for a fast action paced game). With the image processing calculations running on their own thread, the game can update at its own speed.

4.3 Head Tracking

Head tracking is required in order to change what the player sees on the screen and creates an immersion into the game. The 3D perspective view must change according to the head movement of the player. (Lee, 2007) created a way to obtain 3D information of the human head by placing two infra-red LEDs on the user's head and a Wii remote on top of the screen which tracks the head on the (x, y, z) axes. That method is used here because it is very robust; it does not require multiple cameras, it is virtually free to compute (all the tracking of the IR LEDs is done by hardware on-board the Wii remote) and it is also very sensitive even to the smallest head movements. There are three values that need to be tracked in relation to the head: headDistance (the distance from the player's head to the Wii remote on top of the screen, z axis), headPosX (position of the head on the horizontal x axis) and headPosY (position of the head on the vertical y axis).

In order track the head, the first variable that must be recovered is the headDistance, which is the distance from the player's head to the screen (Wii remote). (Lee, 2007) uses Pythagoras to solve for the distance of the head to the screen (Figure 39). For that, need to know the values of "dotDistance" and "Angle", this can be easily calculated as dotDistance is simply the distance between the two IR LEDs and Angle is the 45° field of view of the Wii remote multiplied by half the pointDistance ($pointDistance = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ where x_1, x_2 are the x coordinates of the two IR LEDs and y_1, y_2 are the y coordinates of the two IR LEDs). The reason why pointDistance is halved and why dotdistance must be halved as well is because we are only interested in the isosceles triangle in Figure 38 (the shaded triangle). Now that dotDistance and Angle are known, Pythagoras's theorem states that $\tan(angle) = \frac{opposite}{adjacent}$. That means $headDistance = \frac{dotDistance\ /2}{\tan(Angle)}$. Once the z position (headDistance) of the head is obtained, the x and y positions are recovered in a similar fashion.

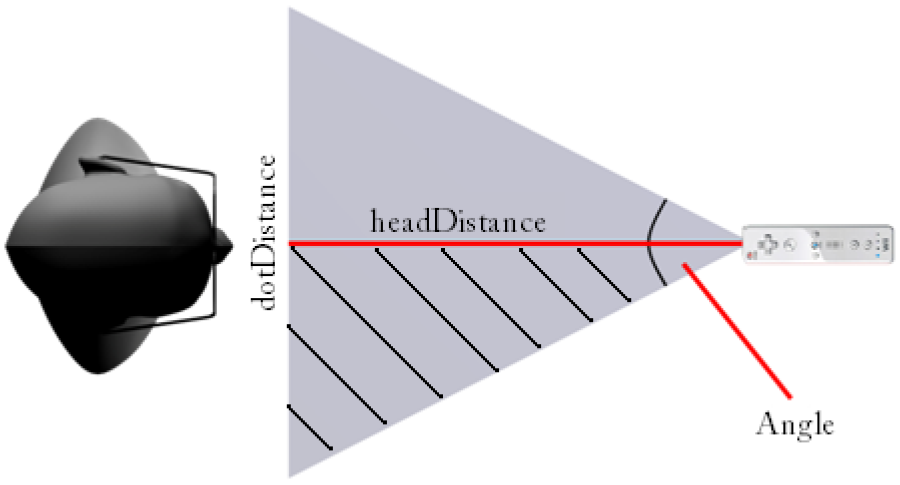


Figure 38 - Calculating the head distance to the Wii remote

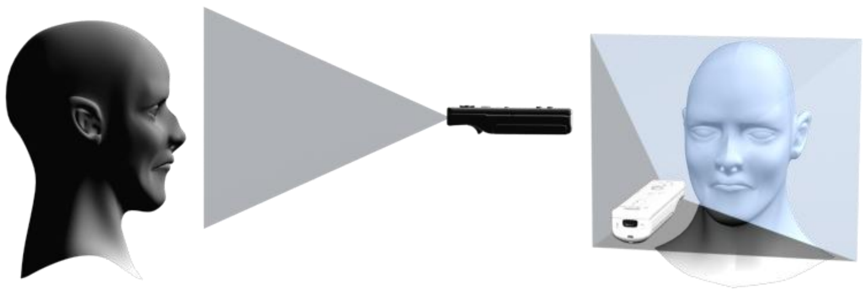


Figure 39 - Wii remote 45° degree field of view

After calculating the x, y, z positions of the head, these values are used to create the correct 3D perspective based on where the player is looking. If the player turns his/her head left, right, up or down the camera inside the 3D environment also rotates left, right, up or down in order to create the illusion that the player is looking through a window into another room. If the player moves closer and further from the screen the camera inside the 3D environment also moves in and out.

The head distance determines a few other factors in the game. For example if the head distance goes over a certain value, because the body is also assumed be directly below the head, then the player is outside hitting range; meaning that the player is too far away from the enemy character and he/she cannot hit him but also the enemy character cannot hit the player since he/she is too far away. All computation for the head tracking is done inside the KendoGame class along with any other Wii remote operations.

4.4 The Game

The game is the visual representation of the different types of tracking, brings all other additional factors together and it provides an immersive experience for the user. The sword fighting game is comprised of a 3D environment and a 3D enemy character. In order to fit the sword fighting theme, the 3D environment is modelled as a dojo (Figure 40a) and the enemy is an animated 3D model (Figure 40b).

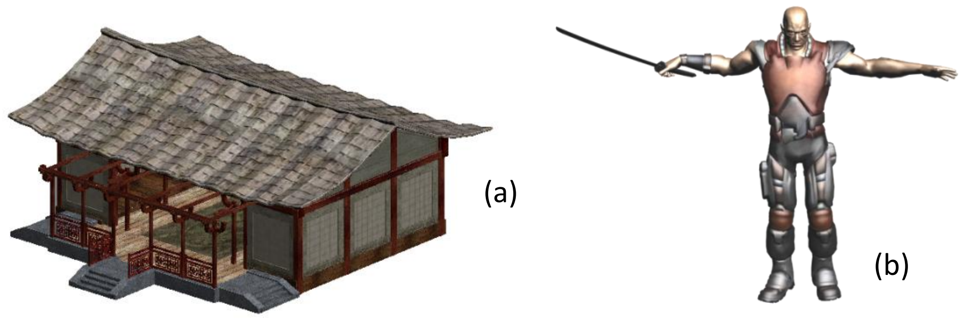


Figure 40 - 3D models used in the game: (a) the dojo environment; (b) the enemy character

The fight between player and enemy character happens inside the dojo. The game is based on a few rules: the player must fight the enemy character in a sword fight; the player can either attack the enemy (if the enemy character is not blocking he will lose a piece of health, else no damage is done to him), the player can block enemy attacks by performing a block (as in Figure 36), or the player can step back to dodge the enemy attacks (know how far away from the screen the player is thanks to the head tracking); the enemy can also attack the player (if the player is not blocking he/she will lose a piece of health, else no damage is done to the player); he can block the player's attack.

The feedback from the user's actions is displayed on-screen, so when the player lands a successful hit on the enemy, "Hit" is displayed on-screen for a short period of time (Figure 41); if the player attempts to hit the enemy while he is blocking "Block" is displayed on-screen. Whoever's health bar reaches 0 first, that is the loser. The enemy has three different action states he can be in: idle – meaning he is not attacking nor he is blocking (so this is the best time to attack him); attacking – meaning that if the player is within striking range and he/she is not blocking, then the enemy landed a successful hit and some health is deducted from the health bar of the player and a blocking state – which means the enemy cannot be damaged at all until he switches state. There are two other states, hit – in which the enemy character has been hit by the player and a hit animation is played and a death state – which is only entered when the player has defeated the enemy character.

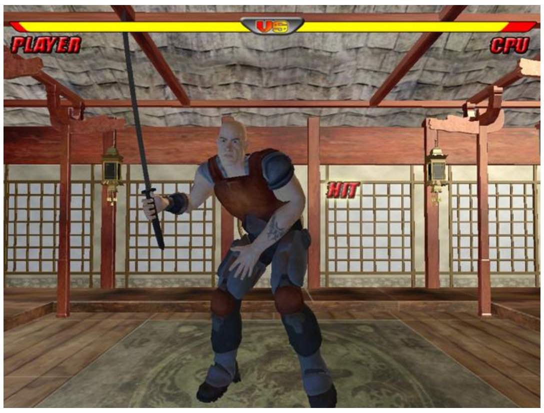


Figure 41 - Enemy character being successfully hit by the player

Because artificial intelligence is outside the scope of this project the enemy's states are randomly switched and only when he is hit by the player does he either try to retaliate or block from further damage.

To create the game, the XNA Game Studio 2.0 IDE (Microsoft) was used and for the playback of the enemy animation the SkinningSample (Microsoft, 2007) was adapted. Because the SkinningSample contained the very basic of animation, the 3D model needed to be rigged with a biped skeleton (Figure 42) in order to be able to add new animations to it. The skeleton was applied to the model mesh in 3D Studio Max and once that was done, the animations can be created. Motion capture data was used to create the different animations of the enemy character. Once the animation data is saved to the model it can be played back in XNA using the SkinningSample animation libraries.

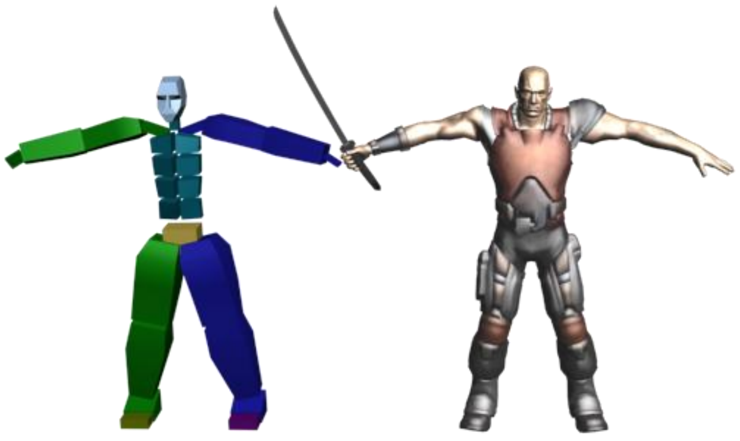


Figure 42 - Biped skeleton and model mesh

In order to create an even more immersive experience for the player, two more aspects are added to the game: sounds and force feedback. The sounds add a more realistic feel to the game, such as when the player hits the enemy character and he is blocking a clash of swords sound is heard; or when the enemy attacks and he performs a martial art "yell" (which is also a good indication to the player that an attack is incoming). The force feedback is basically a vibration coming from the rotor inside the Wii remote in the GloSword's handle. A quick "jolt" can be felt by the player when an attack which does not miss is performed (either hit or block will cause the Wii remote to vibrate) and also when the player is hit by the enemy.

The game logic is performed in the KendoGame class which is also responsible for rendering the game.

Setup Images

These are images of the fully completed system in action.

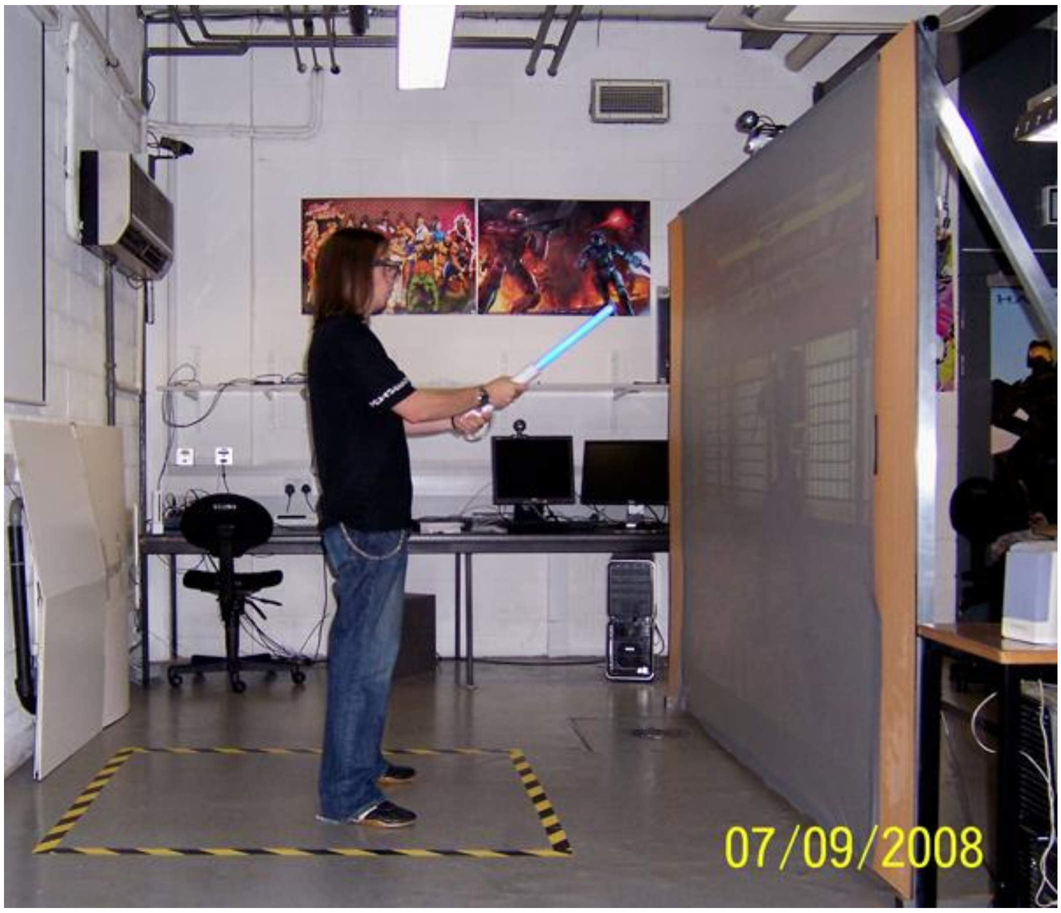


Figure 43 – The Full Body Tracking Kendo Game setup and myself testing the game

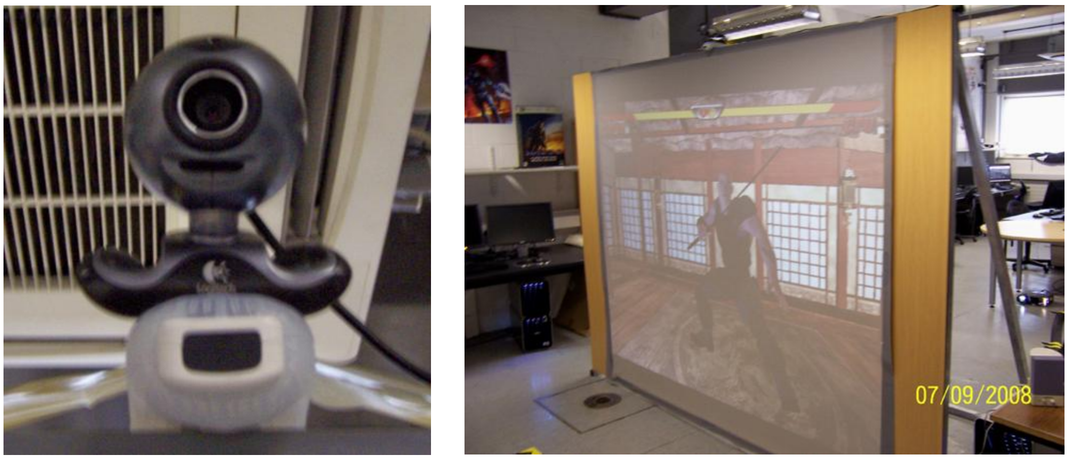


Figure 44 - Close up of USB camera, Wii remote and back projection screen

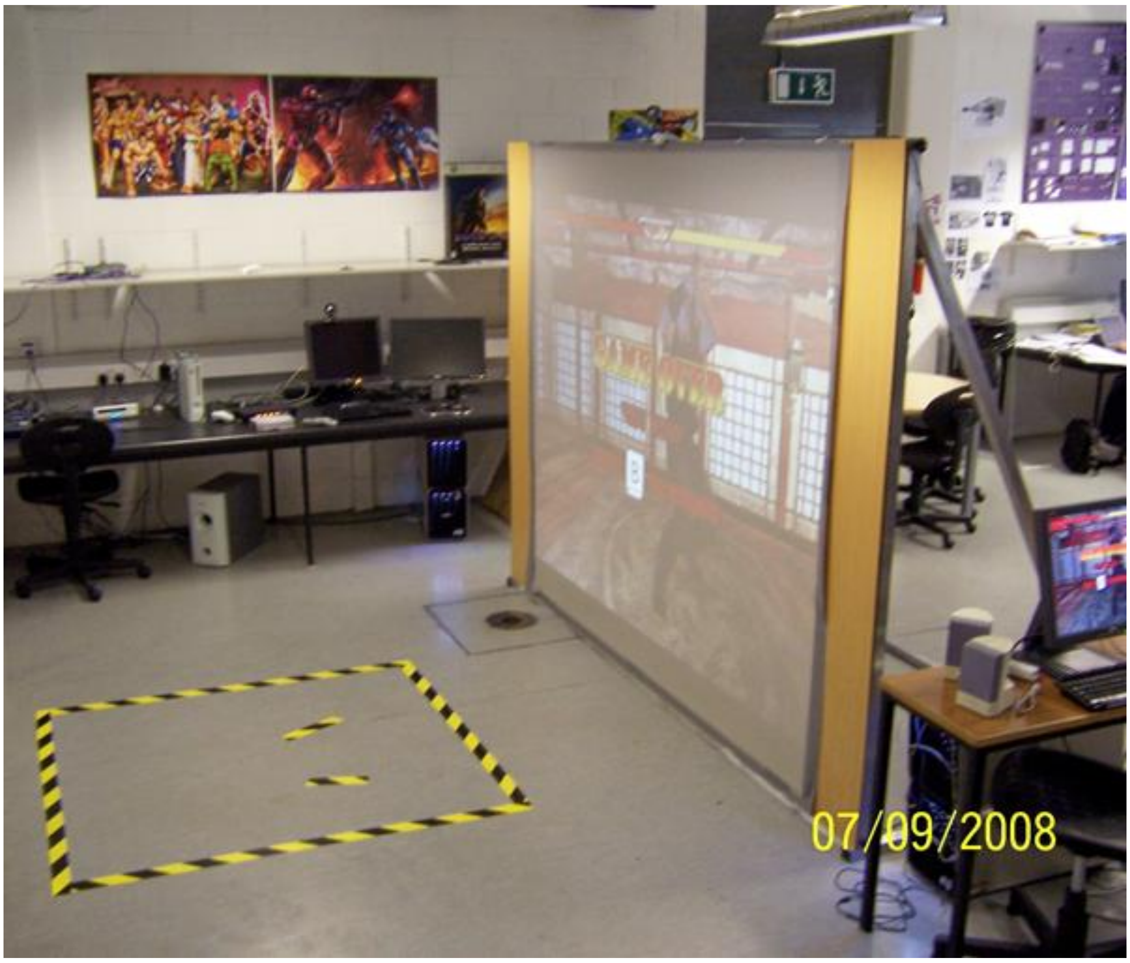


Figure 45 - Entire Kendo Game setup - playing area, screen, USB camera and Wii remote

5. Evaluation

Given the set of goals described in the introduction, this section will examine those goals and compare them to the end result.

A full body tracking kendo game was proposed which needed to implement three types of tracking: sword, body and head tracking but also a game to encompass and demonstrate these methods. The sword tracking is implemented using both vision and data from an IMU (the Wii remote). The vision part of the sword tracking detects and tracks a special type of sword (the GloSword, Figure 26) using colour based tracking and because the tracking depends on a blue light it also works well under small changes in illumination. The second part of the sword tracking is done using the acceleration/tilt parameters of the Wii remote which gives highly accurate, highly responsive feedback based on the actions of the player. From these two types of sword tracking, various information about the sword can be obtained such as: rotation in the (x, y, z) axes from the Wii remote, acceleration in the (x, y, z) axes from the Wii remote, size, position and rotation in the (x, y) axes using vision.

Body tracking uses vision, background subtraction and silhouette extraction in order to determine where the body is so that it can be tracked. The body tracking technique tracks the body in two dimensions (x, y) but a third dimension z can be obtained from the head tracking (body distance to the screen is assumed to be equal to the head distance).

The head tracking achieved is not marker free but that is fair trade off when taking into account the accuracy and the computation resources used for this method. Head tracking is achieved on the (x, y, z) axes in real-time with the aid of two head mounted infra-red LEDs and a Wii remote which tracks the IR LEDs in the three dimensions. The computation for the tracking of the IR LEDs is completely done by hardware on-board the Wii remote so the computer must only interpret the data from the Wii remote.

The game itself features 2D tracking from vision combined with 3D tracking of the head using the Wii remote, but by also combining the tracking with other factors such as sounds, force feedback, large screen size, a truly immersive experience is created for the player. The full application runs in real-time at over 60 fps which is an achievement, considering all the tracking involved and also the actual processing of the 3D models + animations.

Limitations

A limitation of the vision based sword tracking is that there is no adaptation for high variance in illumination changes (if the scene illumination abruptly changes from dark to very bright the tracking will lose in accuracy and detect false positives due to bright lights that might resemble the lights from the sword). The body tracking method used is susceptible to errors when noticeable changes in illumination occur in the scene, or the background is very cluttered. This is because background subtraction is used to determine the silhouette of the body and when the initial background is subtracted from each new frame that has a different illumination; the difference image will contain background information also (instead of foreground only). The head tracking method does not have the parallax effect implemented but gives the illusion of 3D by rotating the camera about a point (the centre of the screen).

6. Conclusions & Future Work

A Full Body Tracking Kendo Game was proposed that implements the latest HCI technologies in a way such that the player of the game becomes immersed in the 3D world. When dealing with vision applications most of the time there is a need for real-time online execution of code and this applies especially to games. This is a daunting task, as in order to extract relevant, accurate information from a single 2D image, there is a substantial computational expense due to the sheer number of pixels in an image (a 320*240 image has 76800 pixels).

In order to achieve real-time execution, the sword and body tracking are performed simultaneously. The head tracking approach is simple and requires minimal computations due to the fact that the tracking is done by hardware on-board the Wii remote. Unlike other approaches to head tracking the method used here is very robust, requires no manual initialization, can automatically recover if the tracked head is lost (due to occlusions) and it is accurate to the smallest head movement.

A game is created that simulates a sword fight between player and a computer opponent and given the elements of the game such as: a physical object that represents the sword and vibrates when the sword comes in contact with an object in the 3D world, life size projection of the 3D world, and sounds; a new way of HCI is created that immerses the player into a virtual world.

There is always room for improvement when it comes to vision. One of the areas of the project which could benefit from future work is the sword and body tracking. Noise and illumination changes are both factors which affect the performance of the colour based tracking of the sword and the body tracking. Illumination changes do not affect the sword tracking as much as the body tracking; because of the background subtraction, if a large change in lighting occurs the difference image will result in a completely different image that what it should have been, given the normally lighted background.

Another improvement which can be done to the body tracking is the removal of the sword from the extracted silhouette. At the moment because the silhouette is extracted from the difference image (difference between an image of the empty background and images with the body in them) both body and sword appear in it.

Improved Wii remote sword detection using the Nintendo Wii Motion Plus (Nintendo, 2008). The Wii Motion Plus is a peripheral, that has not been released as of yet, which connects to a standard Wii remote. It features a far more precise accelerometer than the one in the standard Wii remote which actually mimics 1:1 any arm movement of the user (as opposed to just reading tilt and acceleration values). Using this device it would be possible to register different swing motions and would also be possible to determine the position of the Wii remote in three dimensions thus reducing some of the vision based computations in tracking the sword.

Extracting a 3D body volume from an image. Or at least a 2.5D volume can be defined by extracting the silhouette of the body from a 2D image and the depth information from the head tracking (distance to screen). Then using collision detection between this volume and the enemy character's sword, a more accurate and realistic fighting sequence can be created.

7. Bibliography

" 3D real-time head tracking fusing color histograms and stereovision" [Conference] / auth. Moreno F. Tarrida A., Andrade-Cetto J., Sanfeliu A. // Pattern Recognition, 2002. Proceedings. 16th International Conference on. - [s.l.] : Inst. de Robotica i Informatica Ind., CSIC, Barcelona, Spain, 2002. - Vol. I. - pp. 368-371.

" Bayesian reconstruction of 3d human motion from single-camera video" [Conference] / auth. Nicholas R. Howe Michael E. Leventon, William T. Freeman. - 1999.

" Pfinder: real-time tracking of the human body" [Conference] / auth. Wren C.R. Azarbayejani, A. Darrell, T. Pentland, A.P. // Pattern Analysis and Machine Intelligence, IEEE Transactions on. - [s.l.] : Media Lab., MIT, Cambridge, MA, USA, 1997. - Vol. 19.

" Real-time tracking and reproduction of 3D human body motion" [Conference] / auth. Colombo C. Del Bimbo A., Valli A. // Image Analysis and Processing, 2001. Proceedings. 11th International Conference on. - [s.l.] : Dipartimento di Sistemi e Informatica Via Santa Marta 3,I-50139 Firenze, ITALY, 2001.

"3D HEAD POSE COMPUTATION FROM 2D IMAGES: TEMPLATES VERSUS FEATURES" [Conference] / auth. R. Lopez Thomas S. Huang // Image Processing, 1995. Proceedings., International Conference on. - [s.l.] : University of Illinois at Urbana-Champaign - Beckman Institute 405 N. Mathews Ave., Urbana, IL 61801, 1995. - Vol. 2. - pp. 599-602.

"3D human head tracking using hypothesized polygon model" [Conference] / auth. Terada K . Oba A., Ito A. // Systems, Man and Cybernetics, 2005 IEEE International Conference on. - 2005. - Vol. II.

"3-D model-based tracking of humans in action: a multi-view approach" [Conference] / auth. D.M. Gavrila L.S. Davis // in Proc. IEEE Computer Vision and Pattern Recognition. - San Francisco : Computer Vision Laboratory, CfAR University of Maryland, College Park, MD 20742, U.S.A., 1996.

"A Miniature Stereo Vision Machine for Real-Time Dense Depth Mapping" [Conference] / auth. Yunde Jia Yihua Xu, Wanchun Liu, Cong Yang, Yuwen Zhu, Xiaoxun Zhang, and Luping An // ICVS 2003. - [s.l.] : Department of Computer Science and Engineering, Beijing Institute of Technology, Beijing 100081, P. R. China, 2003.

"A Real-Time 3D Human Body Tracking and Modeling System" [Conference] / auth. Jing-Feng Li Yi-Hua Xu, Yang Chen, Yun-De Jia // Image Processing, 2006 IEEE International Conference on. - [s.l.] : Dept. of Comput. Sci. & Eng., Beijing Inst. of Technol, 2006.

"A robust elliptical head tracker" [Conference] / auth. Nanda H. Fujimura K. // Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on. - 2004. - pp. 469- 474.

"Adaptive background mixture models for real-time tracking" [Conference] / auth. Stauffer C. Grimson, W.E.L // Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on. - [s.l.] : Artificial Intelligence Lab., MIT, Cambridge, MA, 1999.

"Automatic Human Body Tracking and Modeling from Monocular Video Sequences," [Conference] / auth. Chih-Chang Yu Jenq-Neng Hwang, Gang-Feng Ho, Chaur-Heh Hsieh // Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on. - 2007.

"Full Body Tracking from Multiple Views Using Stochastic Sampling" [Journal] / auth. Kehl R. Bray, M. Van Gool, L. Comput. Vision Lab., Zuerich, Switzerland // Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. - 2005.

"Head tracking using a textured polygonal model" [Conference] / auth. Schodl A Haro A, Essa I. A.. - [s.l.] : College of Computing, GVU Center Georgia Institute of Technology Atlanta, GA 30332-0280, U. S. A., 1998.

"Head Tracking Using Particle Filter with Intensity Gradient and Color Histogram" [Conference] / auth. Xu X. Li B. // Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on. - [s.l.] : Center for Cognitive Ubiquitous Computing Dept. of Computer Science & Engineering Arizona Statue University Tempe, AZ, U.S.A., 2005. - pp. 888-891.

"Head Tracking via Robust Registration in Texture Map Images" [Conference] / auth. M. La Cascia J. Isidoro,S. Sclaroff // IEEE Conf. on Computer Vision and Pattern Recognition. - [s.l.] : Computer Science Department Boston University Boston, MA 02215, 1998.

"Hierarchical model-" [Conference] / auth. J. R. Bergen P. Anandan, K. J. Hanna, and R. Hingorani // European Conference on Computer Vision. - 1992. - pp. 237-252.

"Human Head Tracking in Three Dimensional Voxel Space" [Conference] / auth. H. Kawanaka H. Fujiyoshi, Y. Iwahori // Pattern Recognition, 2006. ICPR 2006. 18th International Conference on. - 2006. - Vol. III. - pp. 826-829.

"Localization of human faces fusing color segmentation and depth from stereo" [Conference] / auth. F. Moreno J. Andrade-Cetto, and A. Sanfeliu // ETFA. - 2001.

"Monocular 3D Head Tracking to Detect Falls of Elderly People" [Conference] / auth. Rougier Caroline Meunier Jean, St-Arnaud Alain, Rousseau Jacqueline // Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE . - 2006.

"Real-time markerless human body tracking using colored voxels and 3D blobs" [Conference] / auth. Caillette F. Howard T. // Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on. - 2004.

"Real-time octree generation from rotating objects" [Conference] / auth. Szeliski. R.. - [s.l.] : Technical Report 90/12, Digital Equipment Corporation, 1990.

"Robust 3D Head Tracking Using Camera Pose Estimation" [Conference] / auth. S. Ohayon E. Rivlin // Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06). - [s.l.] : Computer Science Department Israel Institute of Technology Haifa, Israel 32000, 2006.

"Robust Head Tracking Based on a Multi-State Particle Filter" [Conference] / auth. Yuan Li Haizhou Ai, Chang Huang, Shihong Lao // Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on. - 2006.

“3D Hand Tracking by Rapid Stochastic Gradient Descent using a Skining Model" [Journal] / auth. M. Bray E. Koller-Meier, P. M¨uller, L. Van Gool. - [s.l.] : CVMP, 2004.

Introducing Wii MotionPlus, Nintendo's upcoming accessory for the revolutionary Wii Remote [Online] / auth. Nintendo // Nintendo. - 2008. - http://www.nintendo.com/whatsnew/detail/eMMuRj_N6vntHPDycCJAKWhEO9zBvyPH.

Johnny Chung lee - Projects - Wii [Online] / auth. Lee Johnny Chung. - 2007. - http://www.cs.cmu.edu/~johnny/projects/wii/.

Parallax - Wikipedia, the free encyclopedia [Online] / auth. Wikipedia // Wikipedia. - 2008. - http://en.wikipedia.org/wiki/Parallax.

XNA Creators Club Online - skinned model [Online] / auth. Microsoft // XNA Creators Club. - 2007. - http://creators.xna.com/en-us/sample/skinnedmodel.

XNA Game Studio 2.0 [Online] / auth. Microsoft // XNA Creators Club Online. - http://creators.xna.com/.

8. Appendices

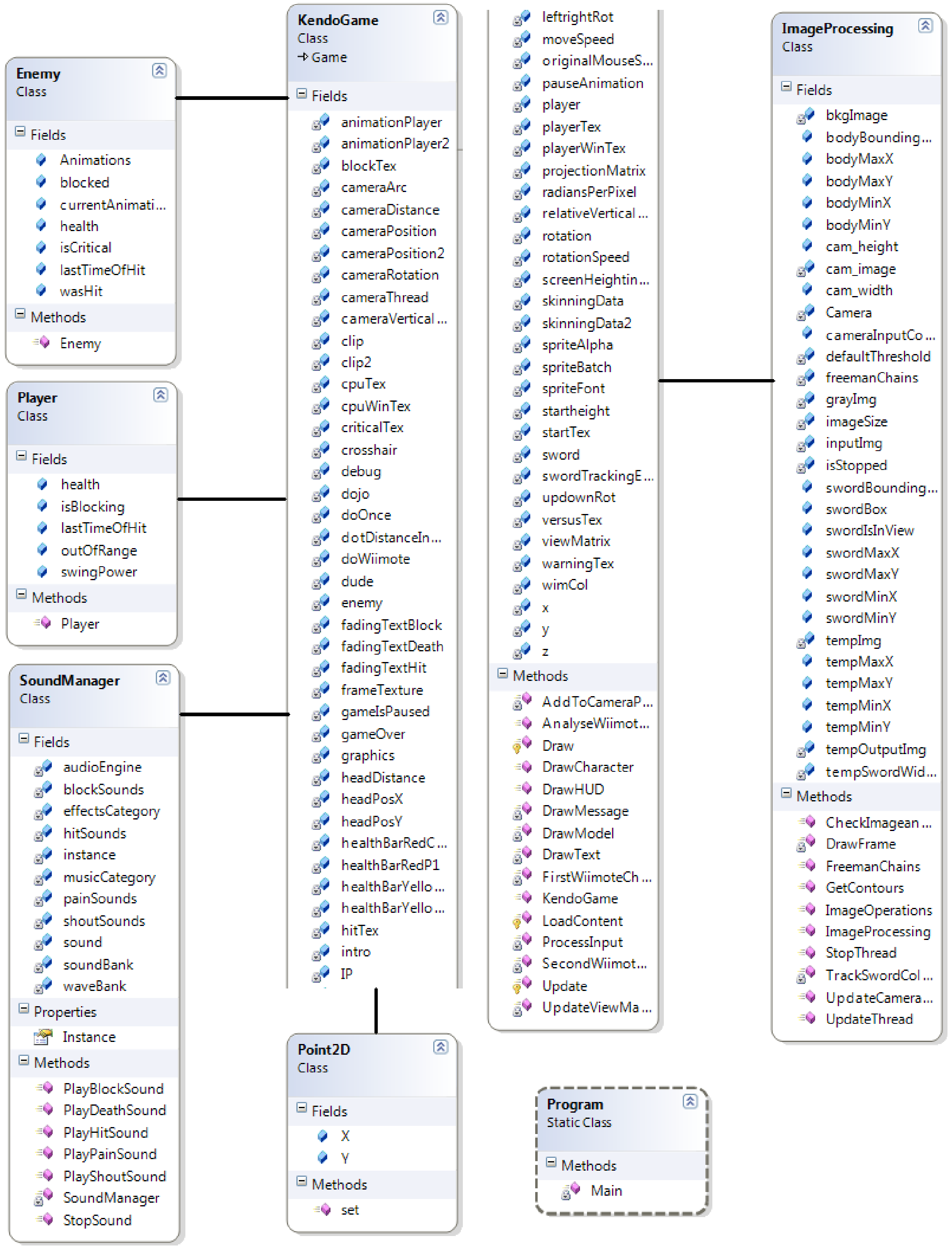


Figure 46 - Class Diagram of the full system: Enemy and Player classes contain information about the enemy and player, SoundManager class deals with the sounds in the game, ImageProcessing class handles all the sword/body tracking, KendoGame class is the main class where the game logic, head tracking and all other classes are merged, Program class starts the application.