Expressive Textures: Synthetic and Video Avatars

by

Kenny Fei

Submitted in partial fulfilment of the requirements for the degree Magister Scientiae

in the Faculty of Natural and Agricultural Science

University of Pretoria

Pretoria

May 2002

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Expressive Textures: Synthetic and Video Avatars

By

Kenny Fei

Abstract

Avatars or virtual characters represent the users in various virtual environments and in variety of applications, giving the user an identity and a visual representation in the virtual environment. Avatars enabled the participant to interact with other avatars and manipulate other objects in the virtual world. When the avatar representing the participant interacts with other avatars, the avatar is required to show facial expressions and body expressions in order to make interaction more meaningful and realistic.

Extensive research has been done on facial animation for avatars. The aim of most of the approaches is to achieve a high degree of realism and this leads to the development of complex models of kinematics, muscle movement, movement of clothing as well as cognition and behavioural models. Image-based techniques and video avatars are also used for creating virtual humans. However, the complexity of the geometric and physically simulated avatar models used by the above methods make them unsuitable for use in distributed collaborative virtual environments that run on low bandwidth networks or over the Internet.

This thesis presents the overview of related research in synthetic and video avatars with the core focus in facial animation. It then presents the Expressive textures theoretical approach that uses texture manipulation to create facial animation of real and synthetic avatars. Furthermore, it presents the design and implementation of an interactive avatar creation tool that generates avatars using the expressive texture approach. The interactive avatar creation tool can create simple avatars, creating simplified models of virtual humans, animating the avatar‘s facial expression and supplementing it with simple body movements. Using the expressive texture method, the face of each avatar can be fine tuned and animated by manipulating the textures.

The interactive avatar creation tool aims at providing a fair amount of realism enabled avatar creation and animation suitable for low cost Tele-presence, Tele-conferencing, distributed virtual environment applications and computer games.

The interaction between avatars is demonstrated in a simple virtual environment for a social interaction application. In particular, a virtual pub environment is used where full body avatars, created via the interactive avatar creation tool, are inserted This allows the user to observe the interactions between the avatars, and interactions between the avatar and the virtual objects in the virtual environment.

Thesis supervisor: Prof. V. Lalioti
Department of Computer Science

Submitted in partial fulfilment of the requirements for the degree Magister Scientiae

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Expressive Textures: Synthetic and Video Avatars

Deur

Kenny Fei

Opsomming

Inkarnate* verskaf aan die verbruiker verskeie visuele omgewings en toepassings. Bg. Karakters veroorsaak dat die gebruiker kan identifiseer met die karakters en visuele verteenwoordiging in 'n virtuele omgewing kan plaasvind.

Uitgebreide navorsing is gedoen oor die gesigs-animasie van inkarnate. Die hoofdoel van hierdie navorsing en benaderinge is om 'n hoë graad van realisme te bewerkstellig. Dit het weer gelei tot die ontwikkeling van komplekse kinematiese modelle, spierbewegings, beweging van klere sowel as bewustheids en gedrags modelle. Verbeelding-gebaseerde tegnieke en video inkarnate word ook gebruik om menslike voorbeelde te ontwikkel. Die kompleksiteit van die geometrese en fisies gesimuleerde inkarnaat modelle maak dit egter ongeskik vir die algemene verspreiding van virtuele uitbeeldings wat op lae-bandwydte netwerke op die Internet plaasvind.

Hierdie tesis verteenwoordig 'n oorsig van die verwante navorsing wat gedoen is in sintetiese en video inkarnasie met die klem op gesigs animasie. Dit verskaf dan die teoretiese benadering wat tekstuur manupilasie gebruik om gesigs animasie van regte en kunsmatige inkarnate te bewerkstellig. Hierdie interaktiewe inkarnaat ontwikkelings instrument kan dan eenvoudige inkarnate tot stand bring en vereenvoudigde modelle van menslike inkarnate met gesigs uitdrukkings en eenvoudige liggaamsbewegings skep. Hierdie ekspresiewe tekstuur-metode het tot gevolg dat die gesig van elke inkarnaat baie fyn ingestel en uitgebeeld kan word.

---

* Inkarnate (avatars) of virtuele karakters (virtual characters) is 'n visuele manifestasie van 'n abstrakte konsep

Die bogenoemde interaktiewe inkarnaat ontwikkelings model het ten doel om 'n redelike mate van realisme daar te stel wat dit geskik sal maak vir lae-koste Tele-teenwoordighied, Tele-konferensies, verspreiding van verskeie omgewings toepassings asook vir rekenaarspeletjies. Die interaksie tussen inkarnate word gedemonstreer deur 'n eenvoudige virtuele, omgewing wat 'n sosiale interaksie toepassing het. Sodoende kan die gebruiker die interaksie tussen die inkarnate en die interaksie tussen die inkarnate en virtuele onderwerpe in 'n virtuele omgewing waarneem.

Tesis studieleier: Prof. V. Lalioti
Department Rekenaarwetenskap

Ingedien ter gedeeltelike vervulling van die vereistes van die graad Magister Scientia

Acknowledgements

I would like to thank the following people for their assistance and contributions, without which this research would not have been possible:

- Prof. V. Lalioti for her guidance, assistance and support as my supervisor
- UP for the use of their facilities
- NRF for financial support

I would also like to thank everyone that supported me during this time and had endless patience listening to my thoughts and ideas and encouraging me when I needed it the most.

List of Figures

Figure 2.1: A 2.5D video avatar interacting with the user in the shared virtual world.

Page. 8.

Figure 2.2: An example of a Stereo camera that is used in immersive environment (Triclops stereo camera, developed by Point Grey Research Inc.).

Page. 21.

Figure 2.3: The datagloves are used in capturing motion of the actor's hands.

Page. 32.

Figure 2.4: The exoskeleton is used to capture full body motion, but it is rigid and has limited body movement for actors. Data is captured in real time.

Page. 32.

Figure 2.5: The left image showed optical sensors on Mr. Tiger Woods' body, tracking his golfing body motion for a computer game. The right top image shows the pre-set-up area for capturing body motion. The right bottom image is an example of a digital camera, Eagle, developed by Motion Analysis Corporation.

Page. 32.

Figure 2.6: Star Trak, wireless electromagnetic sensors.

Page. 33.

Figure 2.7: Retargetting process showed under Kinetix's character Studio R2.1, when the same motion is applied to characters of different sizes and the motion is re-adapted to each character without distorting the walking motion.

Page. 37.

Figure 2.8: Markers are placed on the actor's face for facial recognition.

Page. 40.

Figure 2.9: A standard cylindrical texture maps the front and side view of the user's face texture together.

Page. 44.

Figure 2.10: The left image indicates the head mesh before mapping. The right image shows the result from mapping the standard cylindrical texture onto the head mesh.

Page. 45.

Figure 2.11: The underlying human face muscles are simulated under Water's anatomically correct muscles approach.

Page. 49.

Figure 3.1: A synthetic face mesh with eye sockets and eyeballs, which can simulate the eye animation by rotating the eyeballs.

Page. 56.

Figure 3.2: Indicating the positions of the control points around the facial features, this marks the facial features to be mapped in the same corresponding region in the face mesh.

Page. 57.

Figure 3.3: The face mesh with the dots indicating the control points.

Page. 57.

Figure 3.4: The Face Image in pixel co-ordinate system.

Page. 59.

Figure 3.5: Texture mapping co-ordinate system, the texture co-ordinates lie in the texture mapping area will not cause the texture to be tiled or clamped.

Page. 59.

Figure 3.6: A still Photo image (left) is first scaled to the correct size, and then texture mapped onto the face mesh using the adjusted texture co-ordinates (right).

Page. 61.

Figure 3.7: The overall process of texture mapping video images to the face mesh, first the captured video image is mapped onto the face mesh using default texture co-ordinates values, then the texture co-ordinates are adjusted to eliminate the distortion.

Page. 62.

Figure 3.8: The texture mapping adjustment process.

Page. 64.

Figure 3.9: The eyes on top are at neutral position, while the eyes at the bottom shows the effect of moving the upper texture co-ordinates closer to the lower texture co-ordinates that map the eyelid texture to simulate eye close.

Page. 67.

Figure 3.10: The face image's right eye is merged with the image of the face with closed eyes.

Page. 67.

Figure 3.11: The facial expressions animated using expressive textures

Page. 67.

Figure 3.12: Image of the skin layer with a hole created by the masking that appears over the pupil image

Page. 69.

Figure 3.13: The shifting of the pupil texture simulates the rotation of the real eye.

Page. 70.

Figure 4.1: The complex hand model (Right) is re-modelled with fewer polygons, which result in a simplified hand model (Left).

Page. 74.

Figure 4.2: Tree hierarchy of the female model

Page. 75.

Figure 4.3: Tree hierarchy of the male model

Page. 75.

Figure 4.4: The female body model prototype modelled in 3D studio Max R2.5.

Page. 76.

Figure 4.5: The male body model prototype modelled in 3D studio Max R2.5.

Page. 77.

Figure 4.6: The hand model rotation simulating wrist movement.

Page. 78.

Figure 4.7: The rotation of the arm around its pivot caused the hand moved with the arm, while remain connected with the arm.

Page. 79.

Figure 4.8: The adjusted pivots of all upper body parts for upper body animation.

Page. 80.

Figure 4.9: The use-case diagram of the state machine.

Page. 85.

Figure 4.10: The state diagram of the state machine.

Page. 86.

Figure 4.11: (Continue) The state diagram of the state machine.

Page. 87.

Figure 4.12: Internal state diagram of the talking interaction process.

Page. 88.

Figure 4.13: Internal state diagram of the drinking interaction process.

Page. 89.

Figure 5.1: A simple method using matrix transformation stack to perform the drawing of the female avatar.

Page. 94.

Figure 5.2: The side view of the modelling transformation sequences for a bending body limb.

Page. 95.

Figure 5.3: The avatar is watching television.

Page. 96.

Figure 5.4: The left image shows the avatar drinking a can of beer and the right image the avatar is listening to the music from the jukebox.

Page. 96.

Figure 5.5: The avatars talking to each other.

Page. 97.

Figure 5.6: The wire-frame view of the floor in the virtual pub environment, which is divided into a grid to detected space availability.

Page. 99.

Figure 5.7: The wire-frame view of the virtual pub environment with all the virtual objects.

Page. 100.

Figure 5.8: The virtual pub environment without textures.

Page. 101.

Figure 5.9: A single texture containing all the textures of the virtual pub environment and the virtual objects.

Page. 102.

Figure 5.10: The result of the virtual pub environment and objects texture mapped with the single combined texture.

Page. 103.

Figure 5.11: The avatar creation tool.

Page. 105.

Figure 5.12: The social interaction application with one of the avatar selected for interaction, this was indicated by the bounding box.

Page. 106.

Figure 5.13: The actions for the avatars were situated at the top panel in the GUI and the interaction tasks are situated at the bottom panel in the GUI.

Page. 107.

Figure A.1: The avatar creation tool prompted for an input file containing the avatar texture mapping data and texture images used for mapping the avatar from the user.

Page. 115.

Figure A.2: After the input file is entered at the prompt, the tool begin to initialise and loads the textures of the graphical user interface and texture images of the avatar.

Page. 115.

Figure A.3: Once the avatar creation tool completed the initialisation, the social interaction application commences with its initialisation.

Page. 117.

Figure A.4: The GUI of the avatar creation tool with the controls numbered for easier identification.

Page. 118.

Figure A.5: The pop-up menu of the avatar creation tool.

Page. 120.

Figure A.6: The selection controls of the social interaction application GUI.

Page. 123.

Contents

ACKNOWLEDGEMENTS i

LIST OF FIGURES ii

INTRODUCTION 1

2. Related Work 4

2.1. Approach to Avatars 4

2.1.1. Video Avatars 5

2.1.1.1. 2D video avatars 6

2.1.1.2. 2.5D video avatars 7

2.1.1.3. 3D video avatars 8

2.1.2. Synthetic Avatar 9

2.1.2.1. Pure avatars 10

2.1.2.2. Autonomous characters 11

2.1.2.3. Guided Avatars 13

2.1.2.4. Interactive perceptive avatars 13

2.1.2.5. Networked real-time synthetic avatars 15

2.2. Applications of Avatars 15

2.2.1. Applications of Video avatars 15

2.2.2. Application of Synthetic avatars 16

2.3. Creation of Avatars 16

2.3.1. Constructing the body and head mesh of the avatar 18

2.3.2. Texture mapping of the avatar 29

2.3.2.1. Texture extraction 29

2.3.2.2. Real-time texture-fitting 30

2.3.3. The body movement of the avatar 30

2.3.3.1. Motion sensory devices 31

2.3.3.2. Motion Control 34

2.3.3.3. Avatar motion and Virtual world 35

2.3.3.4. Retargetting motion 36

2.4. Facial expression and emotions in Avatars 38
2.4.1. Designing facial expressions in avatars 39
2.4.2. Texture Mapping the face mesh 41
2.4.2.1. Face Texture extraction 42
2.4.2.2. Markers removal from texture 42
2.4.2.3. Determining texture mapping 43
2.4.3. Facial Animation in Avatars 46
2.5. Discussion 51

3. Expressive Textures 54
3.1. Face model 54
3.2. Texture Mapping 58
3.2.1. Video Images 61
3.2.2. Texture Mapping Adjustment 62
3.3. Facial Expressions 64
3.4. Eye Animation 68
3.5. Summary 70

4. Social Interaction 72
4.1. Full body Avatar 73
4.1.1. Body Creation 73
4.1.2. Body Animation 77
4.2. Interaction 81
4.3. State Machine 83
4.4. Sounds 89
4.5. Summary 90

5. Design and Implementation 91
5.1. Avatar Generation 91
5.1.1. Body Creation 92
5.1.2. Animating Body motion 94

5.2. Virtual Environment Generation 97
5.2.1. Virtual Pub Environment and objects creation 97
5.2.1.1. Virtual Pub Environment 98
5.2.1.2. Virtual objects 99
5.2.2. Texture mapping virtual environment and objects 100
5.3. Avatar creation tool 103
5.4. Social Interaction application 105
5.5. Summary 108

6. CONCLUSIONS AND FUTURE WORK 109
6.1. Expressive Textures 109
6.2. Conclusions and Future work 111

APPENDIX: Avatar creation tool and Social interaction Application 114
A.1. Initialisation of the avatar creation tool and social interaction application 114
A.2. Avatar creation tool 117
A.3. Social interaction application 122
A.4. Summary 125

BIBLIOGRAPHY 126

Chapter 1

"The human face is capable of an astonishing array of expressions: some of them declare unmistakable emotions, others are very subtle."

John Raynes [100]

Introduction

People seek meaning through conversation, learning through interaction, and feel the emotions that others experience and feel through facial expression. During a conversation, facial expression can tell the respondent the emotion of the speaker and eye movement can indicate the abstract social value of a person. (In the western cultures, People who use frequent eye contact are perceived as more attentive, friendly, co-operative, confident, mature, and sincere than those who avoid using eye contacts. However, for the eastern cultures e.g. Japanese culture, direct eye contact is perceived as impolite and aggressive.).
When people interact with objects and other people, interaction is only possible if the objects and other people can be seen or touched by the respondent (the respondent sees the reaction of the other person through his/her facial expression). Therefore, facial expression forms a crucial part in the communication and interaction process.

Expressive texture avatars allows people to interact and communicate with other users, computer generated characters and objects in the virtual environment with lesser computations and using fewer system resources. The expressive texture avatar represents the user in the virtual environment, so that other users can identify the user from his/hers avatar representation. The expressive texture avatar can animate basic

facial expressions and use body motions together with facial animation to show emotional states and action.

By using the avatar creation tool, the face texture of the avatar can be adapted for different face images. The social interaction application shows the social interaction between different expressive avatars, which allow the user to observe the changes in the emotional state of each avatar under interaction in a synthetic environment.

Thesis focus

The work presented in this thesis is a theoretical approach and the software tool that enables users to create and animate facial expressions of avatars. The approach is to use a generic mesh onto which any face image, either synthetic or from a real person can be texture mapped, and by manipulation of the texture co-ordinates a variety of facial expressions is achieved. A software tool has been implemented to support the theoretical approach. The thesis presents expressive texture approach and highlights how this approach is different from the traditional facial animation approaches that were followed. It then discusses the process of avatar creation with a main focus in facial animation and discusses briefly body animation of the expressive avatar. It then highlights how virtual social interactions are achieved in the virtual environment.

Thesis layout

Chapter 2 discusses the classification of different approaches to and the distinct differences between them. It gives an overview of applications that make use of avatars, of the avatar generation process from a broad perspective, and of how facial expressions are created in avatars and the various facial animation approaches over the years.

In chapter 3, the Expressive Textures approach for animating facial expressions is presented. It discusses how the approach uses textures of images of both the synthetic faces and the real faces captured from video, and map these images onto a face mesh of the avatar. It presents an interactive way of fine-tuning and adjusting the underlined model, which allows a more realistic mapping of a special facial image. Furthermore, it presents how facial expressions are created and animated by texture manipulation.

Chapter 4 looks into the process of creating a simple full-body avatar suitable for a synthetic social environment and in how to animate the upper body movements of the avatar associated with facial emotions and other simple actions. It further discusses the interaction process between different avatars and their virtual environment, and discusses briefly the effects of combining sounds in the virtual environment. It then highlights how the interaction process between all the avatars in the synthetic social environment is co-ordinated using a simple state machine.

Chapter 5 discusses implementation of the Expressive Texture theoretical approach into the avatar creation tool and discusses the modelling that was required for the interactive virtual pub environment. It then discusses briefly how the model structure of the avatar is represented in OpenGL. It looks into the interactive virtual environment and how it simulates a real-world social environment, which allows the user to observe the social interaction between avatars, and the emotional changes of avatars caused by the social interaction.

Chapter 6 provides conclusions of the dissertation and discusses future work with regard to the avatar creation tool and the social interaction application.

In the appendix of this thesis the initialisation and complete function of all the controls in the GUI of the avatar creation tool and the social interaction application, is explained

Chapter 2

Related Work

The first section of this chapter discusses the classification of different approaches to avatar, and the distinct differences between them. The second section gives an overview of a few applications that make use of avatars. Section 2.3 discusses the avatar generation process from a broad perspective, highlights how different researchers generated the avatars, from creating the avatar to animating the avatars motion, and how do these processes differ in implementation. Section 2.4 discusses how facial expressions are created in avatars and various facial animation approaches over the years.

2.1. Approach to Avatars

The term "Avatar" in Hindu mythology means the descent of god or goddess to earth in bodily form, this is a human body represents god on earth [102].

In Virtual environments, "Avatar" means the user's bodily incarnation in the virtual world. This means an avatar is an entity in a virtual world that represent a human user or a software agent capable of interacting with other entities and the virtual environment [103][104]. When participants enter the virtual environment, they want to interact with the objects and other participants in this virtual world. However, interaction is only possible if the users can view other users or themselves in this virtual world knowing where they are and what they are looking at in the virtual environment. Therefore, avatars represented the users and the virtual characters they interacted within this virtual world. The avatar can be implemented as virtual human (computer generated human

like characters in the virtual world), synthetic avatars (computer generated entity in the virtual world), and video avatars (video generated entity in the virtual world). A Virtual human is a form of synthetic avatar, which has a detailed geometric model that closely resembled the human body.

Avatars can be divided into two distinct types by their implementation and controls, namely video and synthetic avatars. Implementation involved obtaining input data, how the avatar is represented in the application, in terms of modelling, texturing, etc. Controls involved how can the user controlled this avatar in the application.

2.1.1 Video Avatar

As the term "Video Avatar" indicated, the core build up of this avatar is based on video input. They use video capture as the input for building the avatar, therefore the representation of this avatar will be the users themselves and human-like. To acquire the images, the person stands on a turntable and place in front of a blue screen at a distance from the camera. Alternatively, a video camera can be placed in front of the user in the immersive projection display to record the user's image [46].

The image taken by the video camera is captured by the graphics workstation. The user's figure is segmented from the background by comparing the background image without the user to the captured image, or use "Chroma keying" to segment a clear image of the user. "Chroma keying" involved setting up a blue screen behind the user, so that the background of the video images becomes transparent when the chroma key values are set up correctly.

In order to obtain the user's position, electromagnetic sensor is used to track the user's position in the immersive projection display. The user's segmented video image is superimposed onto the three-dimension position in the virtual world. The video image is rendered as video texture in real-time.
The video avatar can be placed at the correct three-dimensional position, however it cannot represented three-dimensional body motion, because it is a two-dimensional

plane image. For example, if the participant points at an object in the virtual world, the accurate information about the index fingertip cannot be transmitted to the other user.

In order to represent the three-dimension body movements, it is necessary to make the video avatar using a three-dimension model. A stereo camera will be required to generate a video avatar that incorporates a three-dimension model. Distance of the user from the camera position can be measured using the triangulation algorithm. So resolution and depth of the image can be calculated based on the distance of the user from the camera.

Video avatars represented the users or participants in the virtual world, allowed the users to communicated and interacted with other users in the virtual environment.
A video avatar usually appears, as a 3D-wax character in the virtual environment, their action and position is dependent to the participant's action and position.
It provides high quality, easily recognisable people for virtual environments.
Because the video images of people are generally static when they are recorded from the camera, they appear rather like statue, but provide significant information about where the other users are located in the virtual environment and where they are looking at.

The control and input methods for different types of video avatars are very similar, but their display/representation differs. Therefore, video avatar can be classified according to their dimension:

- ♦ 2D video avatars
- ♦ 2.5D video avatars
- ♦ 3D video avatars

2.1.1.1. 2D video avatars

They are implemented by capturing a remote virtual reality participant from a single real-time video. Then sent these video images to the system, so that the position and rotation of the user can be obtained by analysing the images and tracking devices [46].

The virtual-reality application used these images as texture, and mapped it onto a polygon, which is placed at the right translation and rotation according to the tracker information of the remote user. The image of the user is separated from the rest of the image by chroma keying.

This kind of video avatar is a texture mapped on a flat polygon; the background-cut-out made the avatar appeared reasonably immersed in the virtual environment. The limitation of this kind of avatar is that when viewing from avatar at angle the avatar appeared to be flat from the view-point of other users.

2.1.1.2. 2.5D video avatars

2.5D video avatars (Figure 2.1.) are not flat like 2D video avatars, they have depth, which is limited, and required a more complex texturing algorithm than 2D avatar.
Unlike 2D video avatars, 2.5D video avatars required one or more cameras to captured the video images of the user, this will depended on the approach taken. The 2.5D video avatars can be implemented in varied methods [46].

The first approach is that the video image is projected onto a static head model using a projective texture matrix. This allows the view from the camera to be mapped back onto the head model from the sensor position and orientation. Using this approach, only the head of the avatar can be displayed onto the screen output.

The second approach is to use two two-dimension images to represent the avatar; but not all the depth cues are supported by this approach. The depth cues supported are convergence, binocular disparity, horizontal motion parallax, and occlusion.

The third approach used the image from the video, and calculated the distance of all the pixels of the captured image to generate the image depth. The three-dimension position of each pixel is calculated in the virtual world co-ordinate system from the depth image. Connecting each pixel with the triangular meshes can generate a surface model. This creates a video avatar in 2.5 dimension that has a surface model for the front side, which

is generated from the texture mapping of the segmented user image onto the surface model [46]. Although the video avatar used a surface model for the front side, there is no shape at its other aspects. The image of the avatar also becomes distorted as the participant moved away from the camera position or is viewed at a different angle. By placing more cameras around the user's area of movement, the camera closest to the user will be used to capture the texture image of the user. This method of camera switching represents a three-dimensional shape according to the other user's viewpoint. This approach was created from a developed method that constructed a three-dimension structure using multiple cameras [47][48].

The limitation of this approach is that it cannot be used in real-time, as large amount of calculations is required for camera switching between several cameras.

Figure 2.1. A 2.5D video avatar interacting with the user in the shared virtual world [46].

2.1.1.3 3D video avatars

This approach was developed by Takaai Akimoto et al. [39], which involved more processing and calculations, than the other types of video avatars mentioned previously. Therefore, only the head of the user can be generated using this approach, but the body of the avatar can be computer generated.

By obtaining the two images of the user's head a front and side view, 3D video avatars are generated. From these two images, the facial features are extracted and calculated according to the facial profile. This determined the position of the facial features (eyes, nose, and mouth) within the head image. Then the values obtained are use to modified the vertices of the generic head mesh and it's facial features to match the head of the

user. The front and side-view images of the user are texture mapped on to the modified head mesh of the user.

The result of this approach is very realistic, but it cannot extract the facial features for some users' head correctly. Processing the extracted data require large amount of time (recognition, transmission and synthesised the facial expression) and the frame-rate is low, which made this approach not feasible for real-time collaborative environment.

2.1.2 Synthetic Avatar

The term "Synthetic Avatar" referred to a computer-generated entity or character in a virtual world. These characters are rendered in real time and performed actions without direct user control. They often interacted with the user who is exploring the Virtual Environment or represented the user's presence in the virtual world.

It is not necessarily that all synthetic avatars are represented in a human form, some avatars appear in computer-generated autonomy, which included animals, insects, robots, and machines, and even characters that can only existed in our creative imagination. They are not necessarily in a virtual human form (human-like) and take independent actions, which include interaction and reaction to the user presence or response. Alternatively, they take dependent actions when representing the user. The user drives the actions and interaction of the synthetic avatars.

Since the form that synthetic avatars may appear varies in different applications, the most complex form of synthetic avatars is virtual human representation. They demanded tracking and other high level mechanisms to animate with maximal facility and minimal input. Therefore, classifications can be made based on their motion control methods and their interactions as proposed by Magnenat-Thalmann and Thalmann [26][4][6].

The participant should be able to animated his/her virtual-avatar representation in real-time, however the avatar control is not straightforward, the complexity of virtual human representation needed a large number of degrees of freedom to be tracked. In addition, interaction with the environment increases this difficulty even more.

Therefore, the human control should use higher level mechanisms for animating the representation with maximal facility and with minimal input. The motion control of avatars increased in complexity, it will be more appropriate to used a classification that base on body motion and facial animation:

- Pure avatars
- Autonomous characters
- Guided avatars
- Interactive-perceptive avatars
- Network real-time synthetic avatars

2.1.2.1 Pure avatars

They are in the form of virtual human, which required a technique call, "real-time rotoscopy" [5]. This consisted of recording input data from a virtual reality (VR) device in real-time and applied these same data to the avatar in the virtual environment concurrently.

The body can be animated using sensors like Polhemus Fastrack or Flock of Birds (sensors attached to the user's arms, legs and body to feed the user's motion as input values to the system). Therefore the body and face of the avatar would look natural and the animation would correlate to the actual body and face.

For example, when the user/animator opens his/her fingers about 2cm, then the avatar should also open its fingers by 2cm. When immersive interaction is required by an application, then the complete body of the avatar should have the same movements as the user's body. This can be achieved by using many sensors to track the every degree of freedom. Molet et al [40] suggested that a minimum of 14 sensors are required to managed a correct and natural-looking posture, but with Semwal et al. [38] 's close-form algorithm only 10 sensors are required to approximated the body posture.

The video sequence of the user's face can be texture mapped onto the avatar's face continuously. This will require the user to be in front of the camera, so that the image of the user from the head to shoulders, or even the whole body can be captured by the camera. The users are allowed to move freely in front of the camera without distorting

or losing the facial images required for texturing the avatars. A simple and fast algorithm is needed to determine the bounding box of the user's face within the image. The algorithm requires a view between the head to shoulders and a static background. Position detection occurs when the algorithm compares each image with the background's original image. Since the background is static, the user's movement causes changes in the image. The position changed in the image can be obtained by calculations from the algorithm. It is possible to analyse the images and extract the changes on the set of facial parameters (Eyes, eye brows and lips positions) that represent a facial expression. The camera in front of the user digitises the video images of the user. Detail measurements of the changes in the facial feature are required for the accurate recognition and analysis process, to determine facial expressions from the video sequences.

2.1.2.2 Autonomous characters

Autonomous characters (agent-based) are synthetic actors that embody the capabilities of the agents [32]. They are driven by agent technology and therefore have their own internal goals and ambitions, and they are able to demonstrated behaviour. An agent is an autonomous software object that is capable of making decisions and act to satisfy it's internal goals, based upon its perceived environment.

Autonomous characters interact with other participants can increase the real-time interaction with the environment and this will increase the sense of presence for the real-people (users).

The users do not guide autonomous characters. Autonomous characters require sufficient behaviours to act autonomously to complete their objective. This requires the appropriate mechanisms for interaction and building behaviours for motion.
Agent-based autonomous characters have the ability to sense their environment, perceive the objects and other avatars in the environment. They can react on that environment based on their perceived information in order to fulfil their objective,

change their own behaviour, and communicate to other avatars in the same virtual environment.

If more complex coding and AI are used in the system, then these virtual agents can be provided with the ability to adapt and evolve in the virtual environment.
Natural environment is modelled in the virtual environment with autonomous characters modelling the entity that researchers want to study. This enables the researchers to simulate the behaviour of people, animals, plants, insects and bacteria, which gives a better understanding of the environment they lived in.

Renault et al. [29], first introduced the concept of virtual vision as main information channel between the avatar and their virtual environment. Avatars perceived their environment through a small window of the rendered environment simulated their point of view. By accessing the pixel depth values and their own position in the virtual environment, the avatar can locate the objects in their virtual world, which are visible to them. Autonomous characters' facial expressions can be animated spontaneously depending on the perception and emotional state of the avatars.

Agent technology is the only method to animate and empower a Synthetic Avatar. Some methods like pre-scripting and programmed behaviours are used, but interactions are predictable and repetitive. Other methods utilise the aspects of agent relating to internal goals and decision-making, but the autonomous character has complete access to virtual environment database (knowledge of the character). Therefore, the agent can made decisions on the aspects of the environment, which it cannot sensed using it's own knowledge. This is commonly used in computer games where the computer opponent can know the position where the player is in the game environment.

The advantages of the agent-based Synthetic avatars is that they are unpredictable and their interactions are rich cause the virtual environment to be more realistic. If the design is adaptive, then the behaviour of the avatar can be changed based on the users' actions or a change in the virtual environment, creating more interesting, unique and

memorable experiences. This increases computations and processing, as decision cycles of the avatar increase when it determines its option based on the sensed information. Better algorithms are currently been developed to overcome these problems and improve the performance of the virtual system.

2.1.2.3 Guided Avatars

These are users driven but guided avatar's motions do not corresponded directly to the users' motions. An example is when an animator is animating an avatar that is not in the form of a virtual human. The animator can use Dataglove to give input values for an alien avatar's jaws movement, so the animator's movement does not correspond to the alien avatar's hand movement. Guided Avatars are very common in Computer Puppetry [66].

Guided avatars are based on the concept of real-time direct metaphor. The participants updated avatars' positions using the input devices. The input device computes the incremental change in the avatars' position. This enables the users to select a set of pre-defined facial expressions or movements from the menu or the graphical interface. In this way, the facial feature changes can be stored as values in the application, which animated a specific facial expression. When simulating a walking avatar, instead of using a real-person with sensors attached to feed input values of the walking motion to the avatar. Sensory information of the walking motion can be obtained using the instantaneous velocity of motion, to computed the joint angles for the whole body, then used gesture with a Dataglove or SpaceBall as the input device.

2.1.2.4 Interactive perceptive avatars

Interactive perceptive avatars are autonomous characters that perceive and communicate with of other avatars and real people interactively. Using postures and other indications of how people feel can do non-verbal communication. Postures provide a mean of communication by defining the positions of the arm and leg, and body angle.

As for communication between avatars, behaviour of the avatars would depend on the emotional state of the avatar, and the facial expression and speech depend on the type of avatars used in the virtual environment. Communication between real people and virtual avatars requires a means for the avatar to sense the presents of real people and their environment. Real people are aware of the presence and actions of the avatar through displays or VR – tools (head-mounted displays) but the difficulty is to allow an avatar to sense or feel the behaviour of real people. This requires some form of gestures and facial expression recognition to enable the interaction between real people and avatars.

Emering et al. [18] produced a combat engagement example between a real person and an autonomous character. This demonstrated gesture recognition by the avatar. The motion of the real person is captured using the flock of birds. The system identified the gestures and transmitted the information to the autonomous avatar, which decided and reacted according to the information of the real person's attitude that was received from the virtual reality sensors.

The same principle can be applied to facial communication between real person and avatar, but facial recognition based upon the video sequence captured by the camera. Mase and Pentland [15] used optical flow and principal direction analysis for reading lips. This model was further refined by Essan et al. [13]. Waters and Terzopoulos [17] animated faces by estimating the muscle contraction parameters from video sequence using "snakes"[5]. This is first developed by Kass et al. [68], active contour method that fitted a contour on a given image. This allowed the fitting of the boundary points of maximum contrast close to the pre-defined rough contour. To get correspondence between points from pictures and points on a generic model, this has a defined number. Some people have used external markers and lipstick on the real face to obtained a more robust recognition of facial movements and expressions [8][9]. Li et al.[11] used a Candid model for 3D-motion estimation for model-based image coding. Azarbayejani et al.[1] used a Kalman filter to retrieved motion parameters restricted to head-motion and orientation. Real-time performances are not featured in many of these methods, but Pandzic et al.[12] developed a fast method that used on a "soft mask" (a set of points on

the facial image that the user adjusted interactively). The method allowed the avatar to imitate a real person's expressions through recognition with real-time performance. Alternatively, this method can be further implemented to enable the avatar perceived the real person's facial expressions.

2.1.2.5 Networked real-time synthetic avatars

Avatars are important in a networked virtual environment, because they represented the user in the virtual environment, enabled the user to interact with other users, avatars and the surrounding environment in this networked virtual environment. For example, the VLNet (Virtual Life Network) system [42] supports a networked-shared VE (Virtual Environment) that allows multiple users to interact with each other and their surroundings in real-time. In this network, it can also include other types of avatars that have been mentioned earlier, using autonomous characters to represent users that are not connected to the network or to represent a background character. This allows asynchronous co-operation between distant partners.

2.2. Applications of Avatars

There are various applications for different types of avatars that we might not even noticed in our daily life. More recently, large number of TV advertising companies use 3D avatars to advertised a product. We can see these kinds of avatars every day, and this is one of the applications where avatars are implemented for marketing products.
The applications for avatars can be divided into applications that implement video avatars and applications that implemented synthetic avatars.

2.2.1 Applications of Video avatars

Video avatars allow distant remote users to communicate and perform non-sophisticate interactions to each other, by representing the users in the same virtual world.

Apart from presenting a participant in networked-virtual environments, video avatar can also be used to improvise acting. This was demonstrated with the experiments done by M. Hirose et al. [46], in which video avatar was used to create an improvised acting

scenario between remote users. This experiment was performed at VR Culture Forum held in Yakushima; the conference hall in Yakushima and the immersive projection display CABIN [78] at the University of Tokyo were connected via two 128 kbps ISDN lines. One line transmitted the video image, and the other line was used for telephone voice and the computer data transmission.

In the conference hall, the dancer played out her part on the stage without the stage setting. Her performance was filmed by the video camera and the image was transmitted to the CABIN at the University of Tokyo. The video avatar was generated and superimposed on the virtual world displayed in the CABIN. The user in the CABIN communicated with the video avatar of the dancer in the three-dimensional virtual world, and created improvised acting created by remote users.

Video avatar's faces are also used in video conferencing, where distant users communicated to each other [94].

2.2.2 Application of Synthetic avatars

The roles of Synthetic actors in virtual environment may include teacher in an educational virtual environment [64], tour guide in the virtual world [23], companion, assistant, entertainer, opponent in computer games [90], presenter [67][33], and various other roles.

Beside the above mentioned roles for synthetic avatars, they are also found in TV advertising, films [34][35][36][69] or films as they are merged into captured video of the real world [25], and even helped to stimulated many other types of interactive applications [65].

The Virtual Shopping Mall example (business application):

For an application described an interactive shopping experience enhancement in dynamic interactive virtual mall environments.

In traditional retail outlets, the goal is to present to the visitor an aesthetic environment that displayed the product in an inviting, stimulating and cost-effective manner.

In shopping online via web page, there is a lack of interaction, the user only sees the image of products and the given information. This form of shopping is cost-effective for the company, but it makes shopping experiences dull and boring.
In a physical store, the environment is static; as such, broad common denominators of decor and theme needed to be factored into the floor presentation of merchandise.
By contrast, in a dynamic virtual mall synthetic environment, the aesthetics of presentation and the attributes of the interactive experience are all dynamic variables that can be directly driven by the personality attributes of a shopper's intelligent avatar. As a representation of the shopper's personality traits, demographic features, and aesthetic and emotional preferences, the avatar's personality matrix dynamically shaped the appearance and affordances of the synthetic-shopping environment. The user is represented by the synthetic avatar, while the shop assistant can represented a participant working for the company selling the products, or using an autonomous character as the virtual shop assistant. Therefore, users can shopped interactively online to various shops in the mall.

With the improvement of computer hardware and networks, and approaches that strive to achieve realism in avatars, this would expand the number of possible applications for avatars in future.

2.3. Creation of Avatars

Over the years there are various attempts and approaches developed to generate highly realistic avatars, some approaches are complex in implementation and require specialised hardware, but can achieve highly realistic results, while other approaches generate fairly realistic avatar with simple implementation and standard hardware. This
Section discusses the avatar generation process in general and some approaches implemented by other researchers in more detail.

When modelling/building the avatar, it is better to separate the creation process to manageable parts:

- Constructing the mesh of the avatar.
- Texture mapping of the avatar

- The body movement and kinematics of the avatar.

2.3.1. Constructing the body and head mesh of the avatar

The body mesh of the avatar can be constructed differently according to the form of the avatar. For example, the body mesh of a comical/non-humanoid avatar [64][65] is simpler than that of a virtual human [26][27].
Firstly, this is because the comical/non-humanoid avatar has a simplified body and limbs, which can be constructed by deforming basic geometric shapes and group them together. Secondly, the comical avatar would more likely to have less joints than a virtual human. Therefore, when using hierarchical modelling the avatar would have less leaf nodes, the complexity of each joint and the movement of the body are lower than the virtual human model. Thirdly, these kinds of avatars mostly have simple facial features created on the head mesh, which allowed facial animation to be fast and simple.

The body structure of the avatar must be modelled using the hierarchical approach. This means that the avatar's fingers are modelled first then the rest of the hand and so on.
Then linked the fingers to the hand, to the arm, to the body. The same applies to the other limbs. The reason for linking the body parts in this manner is, because when the arm is rotated the hand should be rotated with it. Therefore a hierarchical movement is established, which the base object (e.g. hand) inherits the movement and rotation from the upper object (e.g. arm). The other advantages of modelling the body using hierarchical approach is that restricted movement (angles between joints) applied to the upper object will automatically be applied to the inherited object. Therefore, animating the avatar's body motion would be less problematic as the body parts will remain in the correct position and not moved out of place during the animation of the avatar.

When constructing a virtual human and if realism is not the main issue or speed of the application is important, then one can created the body of the avatar using 3D geometric shapes to represented the limbs and body [65]. Triangle meshes can be used to model the head of the avatar. In this way, the avatar had a detail head with a simplified body, which required less system resources.

In constructing realistic avatars, T. K. Capin et al [41], defined an articulated structure that simulated the human skeleton. A 3D articulated hierarchy of joints represented the skeleton, each with realistic maximum and minimum limits. The skeleton is encapsulated with geometrical, topological, and inertial characteristics of different body limbs.
The body structure had a fixed topology template of joints and different body instances. These are created by scaling body limbs globally, as well as applying frontal, high and low lateral scaling, or specifying spine origin ratio between lower and upper body parts [61].

Magnenat-Thalmann et al. [27] attached this skeleton, with a second layer that consisted of blobs called "metaballs" to represented muscle and skin. This method's main advantage is that it permitted the entire human body to be covered with only a small number of blobs. Then the body is divided into 17 parts: head, neck, upper torso, lower torso, hip, left and right upper arm, lower arm, hand, upper leg, lower leg, and foot. Because of their complexity, head, hands and feet are not represented with blobs, but with triangle meshes instead [49].

For the other parts, a cross-sectional table is used for deformation. This cross-sectional table is created only once for each body by dividing each body part into a number of cross-sections and computing the outermost intersection points with the blobs.
These points represented the skin contour and are stored in the body description file. During runtime the skin contour is attached to the skeleton, and at each step is interpolated around the link depending on the joint angles. From this interpolated, skin contour the deformation component created the new body triangle mesh.

There are different parameter sets for defining virtual human postures and faces:

Global Positioning Domain Parameters:

These are the global position and orientation values of particular observable points on the body, in the body co-ordinate system. Possible choices are: top of head, back of neck, mid-clavicle, shoulders, elbow, wrist, hip, knee, ankle, bottom of mid-toe.

Joint Angle Domain Parameters:
These parameters comprise the joint angles defined above, connecting different body parts.

Hand and Finger Parameters:
The hand is capable of performing complicated motions and there are at least fifteen joints in the hand, not counting the carpal part [44]. Using hand joints almost doubles the total number of degrees of freedom and therefore separated the hand parameters from those of other body parts.

Face Parameters:
The face is generally represented differently than the other parts of the body. It is a polygon mesh model with defined regions and Free Form Deformations modelling the muscle action [60]. It can be controlled on several levels. On the lowest level, an extensive set of Minimal Perceptible Actions (MPAs), closely related to muscle actions and similar to FACS Action Units, can be directly controlled. There are 65 MPAs, and they described the facial expression completely. On a higher level, phonemes and/or facial expressions can be controlled spatially and temporally. On the highest level, complete animation scripts can be the input defining speech and emotion over time. Algorithms existed to mapped texture on such facial model.

Depending on the type of virtual reality application and the types of avatars used, there are various approaches to generate the avatars' head to display expressions.

In an immersive environment (e.g. CAVE), stereo cameras (Figure 2.2.) are placed around the participant to captured the images of the participant. These images are mapped to a flat polygon or projected to a generic mesh, but these images are static.

This indicated that the face of the avatar is simple and expressions are created by capturing the emotions of the participant and display them onto the avatar [46].

Figure 2.2. An example of a Stereo camera that is used in immersive environment (Triclops stereo camera, developed by Point Grey Research Inc.) [46].

This approach required some tracking devices or video analysis algorithm to tracked the orientation of the participant's face and it continuously texture mapped the video sequence of the user's face onto the face mesh of the avatar [2]. The user must be positioned in front of the camera, so that the camera can captured the image from his/her head to shoulders and possibly the entire body of the user. A fast, simple image-analysis algorithm is used to find the bounding box of the user's face within the image. The algorithm performed image analysis separated the head of the user from the background and projected the user's face to a simple face mesh. Only frontal projection is possible if one camera is used, as the front image of the user is available.

Deformation can be noticed when the facial images are projected onto a generic face-mesh [42]. For example, if the user's face is elongated and the image captured is mapped to a flat face mesh, then the user's face image will be stretched during the mapping process to cover contours along the lower jaw. To prevent less or no deformation on the images, the alternative approach is to create a generic face mesh at run time to fit the captured facial image [39][43]. This involved more processing, because the shape of the face and facial features are extracted from the image first, the shape is reconstructed and then the generic mesh is modified to match the information extracted from the image.

There are various methods for shape reconstruction to get a detail shape, but this is time-consuming, required a sophisticated equipment or complex algorithm. It can be divided into *Shape reconstruction* and *Structured Shape Reconstruction*.

There are a few methods in *Shape reconstruction*, which can get a detailed range data for a face.

Stripe Generator, is a form of structured light-camera range digitizer. A light striper with a camera and stripe pattern generator can be used for face reconstruction. The advantage of this compared to laser scanners is that it is cheaper. Stripe pattern is taken from the camera and projected on the three-dimensional object's surface. Three-dimensional shape can be calculated with information of the camera and projector positions and stripe pattern. Proesmans et al. [70] displayed a good dynamic 3D shape using a slide projector by a frame-by-frame reconstruction of the video.

Plaster Model, Magnenat-Thalmann et al. [69] used plaster models in the real world and selected vertices and facets that are marked on the models, which are digitised by taking photographs from various angles. From this method, high resolution can be obtained from any regions of the model, but the reconstruction process required a mesh drawn on the face, which made it time consuming.

Laser Scanning, In range image vision system, scanners produced range images. The range to the visible surface of the object in the scene is known for each pixel in the image. The spatial location is determined for a large number of points on this surface. Lee et al. [45] digitised facial geometry, using scanning range sensors, but this method based on 3D digitising required a powerful workstation and specialised, expensive hardware. In addition to geometric 3D information, the textural information can also be obtained to created a realistic model of the view face [85][86]. It is important to remove any markers on the actor's face during scanner, as the result of the face mesh would have an irregular surface due to the markers. Although the models are very realistic, these models cannot be animated as they are, because the vertices of their mesh of

There are various methods for shape reconstruction to get a detail shape, but this is time-consuming, required a sophisticated equipment or complex algorithm. It can be divided into *Shape reconstruction* and *Structured Shape Reconstruction*.

There are a few methods in *Shape reconstruction*, which can get a detailed range data for a face.

Stripe Generator, is a form of structured light-camera range digitizer. A light striper with a camera and stripe pattern generator can be used for face reconstruction. The advantage of this compared to laser scanners is that it is cheaper. Stripe pattern is taken from the camera and projected on the three-dimensional object's surface. Three-dimensional shape can be calculated with information of the camera and projector positions and stripe pattern. Proesmans et al. [70] displayed a good dynamic 3D shape using a slide projector by a frame-by-frame reconstruction of the video.

Plaster Model, Magnenat-Thalmann et al. [69] used plaster models in the real world and selected vertices and facets that are marked on the models, which are digitised by taking photographs from various angles. From this method, high resolution can be obtained from any regions of the model, but the reconstruction process required a mesh drawn on the face, which made it time consuming.

Laser Scanning, In range image vision system, scanners produced range images. The range to the visible surface of the object in the scene is known for each pixel in the image. The spatial location is determined for a large number of points on this surface. Lee et al. [45] digitised facial geometry, using scanning range sensors, but this method based on 3D digitising required a powerful workstation and specialised, expensive hardware. In addition to geometric 3D information, the textural information can also be obtained to created a realistic model of the view face [85][86]. It is important to remove any markers on the actor's face during scanner, as the result of the face mesh would have an irregular surface due to the markers. Although the models are very realistic, these models cannot be animated as they are, because the vertices of their mesh of

triangles do not corresponded to the physical ones of the face. In fact, the distribution of the triangles depended on the 3D data acquisition/estimation process.

One possible solution to this problem could be the adjustment of a sub-set of the vertices, chosen as the closest ones to be used as the key-points of the face. Once all such points are optimally relocated, their animation rules, and the consequent motion of all the other vertices, can be defined based on the displacements of the essential features. This will required the definition of different animation rules for each model, as the mesh geometry may vary from model to model. An example of commercial 3D digitizer based on laser-light scanning, is Cyberware Colour Digitizer.

Lighting Switch Photometry, computed the normal vectors for extracting shapes of static objects [71] or a human face in motion [72], by using three or more light sources. This method assumed that the reflectance map is lambertian. The normal vector can be computed at the points where three incident light sources illuminate using Lighting Switch Photometry. The limitation for this method is that computing accurately the normal vector is difficult, particularly at the point where the intensity of the radiance is small. One example is shadowed regions.

In *Stereoscopy, the* correspondence at certain characteristic points can be established by a distance measurement method. This method used the geometric relation over stereo images to recover the surface depth, which resulted in sparse spatial data. Fua and Leclerc [73] used it in texture areas by weighting the stereo component most strongly for textured image areas and the shading component most strongly for texture-less areas.

Most of the methods in Shape Reconstruction are focused on recovering a good shape, the limitation is that structured information is missing, and only shape can be obtained from these methods. Structured Shape Reconstruction involved getting a structured shape for animation. The common approach is modified the generic mesh with structured information such as the eyes, mouth, eyebrows, nose, etc.

These can be classified into methods using range data and without using range data.

With Range Data

In these methods, to made the model suitable for animation, structural information must be added to a set of three-dimensional points.

Using photogrammetric techniques, precise geometry of the head mesh can be created by the image [7]. Grids are drawn on the actor's face to marked positions on the face for modelling and animation. However these images used to construct the head/face mesh can no longer be used as a valid texture map for the subject. To overcome the grid problems, several methods have been proposed for modelling the face photogrammetrically without the used of grid [74][57]. These approaches used a small predetermined set of features to deformed the generic face mesh to the particular face being modelled, and offered no mechanism to further improved the fit. The result from this approach performed poorly on faces with unusual features or other significant deviations from the normal face.

Warping Kernels, is method by Williams [55]. This method used a Cyberware digitizer to reconstruct the head and applied warpware to animated the model. A set of warping kernels is distributed around the face, each of which is a Hanning (cosine) window. This is scaled to 1.0 in the centre, and diminishing smoothly to 0.0 at the edge.

In Mesh Adaptation, Lee et al. [45] started with a structured facial mesh and developed algorithms that reconstructed automatically the functional models of the heads of people from laser-scanned range and reflection data. After the large arrays of data acquired by the scanner are obtained, these data is reduced into a parsimonious geometric model of the face that can be animated efficiently. The generic face is adapted according to the data. When the feature based matching technique completed the mesh fitting process, the algorithm samples the range image at the location of the nodes of the face mesh to

captured the facial geometry. The node positions also provided the texture-mapping co-ordinates that allowed the full resolution colour image to be mapped onto the triangles.

Without Range Data

Methods that are based on the three-dimension digitisation to obtain a range data often required a specialised high-cost hardware. The better and low cost approach is to reconstruct the two-dimension information to generate a three-dimensional object. Two commonly used methods are reconstruction method with feature points which modified a generic mesh model after feature detection method, and interactive deformation method which modified or generated a surface employing deformation.

The performances of methods that reconstructed a face shape from few pictures of a face [39][57][74] are faster in general. In this method, a generic model in 3D is provided in advance, and a limited number of feature points are detected automatically or interactively on two or more orthogonal pictures. The other points on the generic model are modified by a special function. Then 3D points are calculated by just combining several 2D co-ordinates.

An interactive method obtained a few points and Delaunay triangulation for the conformation of the face and texture mapping, in Kurihara and Arai 's approach [57]. The result is good, but the trade-off is that only few points are available, to modified the generic mesh model. If the shape of the generic mesh model is very different from the person's head, the results from few modification points would not looked similar to the person's head, which made texture mapping deformed or stretched the image slightly. To increase accuracy, more input points for modifying the generic mesh model must be available.

Ip and Yin [74] developed a similar approach as Akimoto et al. [39]. Both of these approaches detected the feature points automatically using dynamic template matching or LMCT (Local Maximum-Curvature Tracking). This checked for concave and convex points on the side profile of the face, and a simple filtering method obtained the interior

points. The method is automatic, but not very robust. For some people who have Mongoloid face it works well, but not for others.

In interactive deformation, by using an interactive tool, Magnenat-Thalmann et al. [27] generated a polygon mesh surface for creating figures. The operations performed included creation of primitives, selection, local deformations and global deformations. This method was time-consuming, but it is the only possible way to digitise a historical personage, whose pictorial or other source is not available and is useful when creating new characters.

Once the face mesh generation is completed, the image with the user's expression is mapped to the face mesh [39]. Instead of transmitting the series of image frames containing the facial expression, the later approach is to developed the head mesh with facial features (eyes, mouth, lips, teeth, tongue, ear) created in the head. An analysis from the video input determined the change in facial features, and extracted the set of parameters that described the facial expression. The facial features in the head mesh are modified according to this change. For example, if the eye in the video input changed its orientation (instead of looking straight, the eye looks up), the eye in the head mesh rotated upwards to match the orientation.
T. K. Capin et al. [42] described a "soft mask"- a set of points on the image of the face adjusted interactively by the user for the recognition process. Detailed measurement is required for accurate recognition and analysis of facial expressions from the video sequences, but it is computation expensive to perform these measurements. Therefore, decreasing the number of facial features to be extracted from the video reduced the computations and recognition of the facial features that relied mainly on colour-sample identification and edge detection.

The set of extracted parameters includes:

- Horizontal head rotation
- Vertical head rotation
- Head inclination

- Aperture of the eyes
- Horizontal position of the iris
- Eyebrow elevation
- Distance between the eyebrows
- Jaw rotation
- Mouth aperture
- Mouth stretch/squeeze

The extracted parameters are translated into minimal perceptible actions, which is sent to a facial animation engine that performed the facial animation. Many techniques allowed the parameterisation of expressions and expression feature components (eyes, lips, eyebrows, etc), deformation of facial models. The parameters are set in a way that each feature-component-animation is constraint by these parameters.
For example, the eyebrows can slide up, down, arch at the end of the eyebrow, etc. Parameters are set, so that if the eyebrows slide up the value is '+1', at neutral position is zero and slide down the value is '–1'.
Early work on this field was done by Ekman [101] who developed the Facial Action Coding System, FACS. Magnenat-Thalmann et al [56] have developed the Abstract Muscle Action System, AMA. Recently, an important development is ISO backed protocol: the MPEG-4 Synthetic/Natural Hybrid Coding (SNHC) scheme. This protocol defined the parameters for facial definition (FDP) and animation (FAP).

In the previous approaches, a generic face mesh is adapted to an individual's face from the images. Then constructing facial animation is simple as they are built in directly in the generic head model by defining the head mesh deformations. These approaches have trade-off between real-time rendition capabilities and realism. The face model may end up being an oversimplified, unrealistic head model for the avatar.

Instead of making the generic mesh specified to a given person, Valente and Dugelay [37] begin from person-dependent data (range and texture image) that corresponded to a neutral facial expression. These data is processed to made them suitable for a general

analysis framework. The head mesh has no separated primitives for the eyeballs, the image used for texturing the face is static initially and the face model is a plane surface. Although the head mesh did not have facial features pre-created in the model, it is still possible to animate the facial expressions. Facial expressions can be achieved from different levels of implementation (vertices, texture co-ordinates, and texture image) and simple deformations on the wireframe vertices and other image manipulation techniques. This method will be further discusses in more details under the section "Facial expressions".

The most difficult aspect of facial animation is constructing a believable 3D facial model that is realistic and flexible. Physically based muscle models, developed by Lee [45], Terzopolulos [50][51], and Waters [16], are fairly realistic, but the trade-off is they required large amount of processing for generating the face mesh and they are computations intensive. Deformations of geometry and texturing [52] are commonly applied when the avatar is used in a virtual environment over the network, because their speed performances are better.

Many systems tried to construct a facial model that resembled the user. Guenter et al [52] used Cyberware scans with complex texture mapping, while Escher and Magnenat-Thalmann [21] fitted a generic mesh model to a specific face via control points obtained from the two camera views.

The limitations of scanning the actor's face to generated a face mesh are:

- If the markers on the actors face is not removed during the scan, dumps will be created on the face mesh.
- The mesh did not have an opening for the mouth.
- The face mesh resulted from a scan has too many polygons, which made animation more difficult as the number of control points is large.

The approaches discussed above are a few methods for creating an avatar's head mesh.

In simplified synthetic avatars, the clothes of the avatar are modelled as part of the body mesh and texture with different images of clothing materials.

In realistic synthetic avatars, the clothes are modelled separately [62]. P. Volino et al. [63] had developed a system for creating clothes for synthetic avatars. The creation process for clothes of avatar is similar to a real tailor designing clothes in real-life. The clothing is mostly created using the B-splines, because folds can be ceased on clothing and simulating clothing can be done easily. The cloth panels are design in two-dimension initially, and the texture is defined for each one. Then the seams are defined. The clothes are placed around the avatar's body in third dimension and the seaming lines are closed, wrapping around the avatar's body. Seaming is performed on the clothing model and the panels are merged together to form the garment. Collision detection algorithms and law of physics are also applied to simulated clothes as in real-life (clothes flowing due to wind and tear due to stretching).

The avatar generation process is complex and the cost of realism is processing speed, therefore many researches are done to improve this process to be more efficient and realistic.

2.3.2. Texture mapping of the avatar

Images captured by the video input are used as textures for the avatars. The texture can be an image of the user's face, or from the head to shoulders of the user, or the whole body. This depends on the type of avatar been implemented in the application. Apart from using video images as texture for the avatar, textures can be created by an artist and texture mapped onto the avatar.

2.3.2.1 Texture extraction

Before these images can be mapped onto the avatar, they are separated from the background, so that the image contained only the user. This can be achieved by two approaches, the image taken by the video camera is captured by the system, and the user's image is segmented from the background by comparing a background image without the user. This requires more analysis and processing, and a background image

without the user given to the system, but no additional set-up (putting up the background screen) is required for the system. Therefore, the images can be captured using the standard cameras.

The other approach is implemented using a blue screen and "chroma-keying" to segment a clear image of the user. This approach is fast and does not require image analysis and comparison computations, but needs to pre-set-up at the capturing area. Therefore, this approach is more common in immersive environments, where the whole body of the user can be captured easily by the system.

In the immersive environment, the extracted image of the user is texture mapped onto a flat polygon plane or projected onto a 3D object that represented the user's head or body.

2.3.2.2 Real-time texture-fitting

G. Sannier and N. Magnenat-Thalmann [10][59] proposed an approach for a real-time texture-fitting interface that fitted a texture interactively to the 3D object. The texture mapping co-ordinates of the features on the mesh are marked on the texture, and they corresponded to the vertices on the mesh, allowing it to fall in the right position. Therefore, the user can see the effects of his/her texture manipulation directly on the mesh. The nearest 3D vertices to the marked-point are found automatically as the user added or removed a marker on the 3D model. This made texture mapping for the avatars faster and simpler, which can be implemented on any complex mesh surface (The face and body of synthetic avatars).

2.3.3. The body movement of the avatar

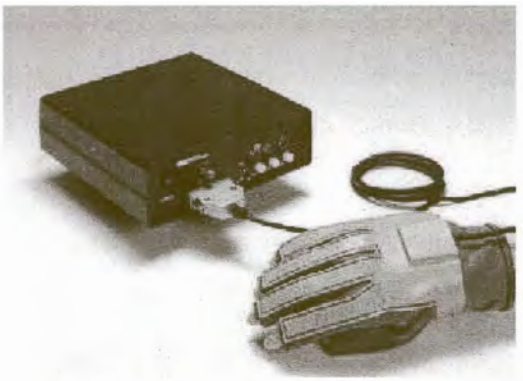
Body movement of the avatar involves animating the avatar's motion. When it interacts with other objects or avatars and when the avatar performs actions in the virtual world. The emergence of techniques like artificial intelligence and object-oriented programming and the increase in computer speed and new virtual reality interacting

devices, makes it possible to classify the movements of the avatar into: avatars with environment, avatars with other avatars and avatars with animator interactions.

2.3.3.1. Motion sensory devices

Body movement data of the avatar can be obtained by a method called, "Motion Capture". Motion capture is a process of capturing the motion of a (human) actor, which used physical devices to control animation of a virtual character [40].

Physical devices may differ in resolution/range of motion, cost, calibration, accuracy and data capturing speed. A few examples that are commonly used are the datagloves (Figure 2.3.), exoskeletons (Figure 2.4.), flock of birds, and other Virtual Reality (VR) tracking sensors. Optical sensors e.g. Ortho Trak developed by Motion Analysis Corporation (Figure 2.5), are used for full body motion capture, it is lighter and less rigid than the exoskeleton, which can captured various body motions. It is widely used in capturing motion for the avatars in computer games and films. The limitation is that it do not supported motion capture in real time, and it required a pre-set-up area to captured the sensor input via a digital camera (Figure 2.5.). Electromagnetic sensors captured body motions in real-time and it do not required a pre-set-up area for motion capture e.g. Polhemus Fastrack, the new types of electromagnetic sensors are wireless e.g. Star Trak developed by Polhemus (Figure 2.6.). These devices' sensors are placed near the joints of the body, which sensed the joints at the body limbs under motion.



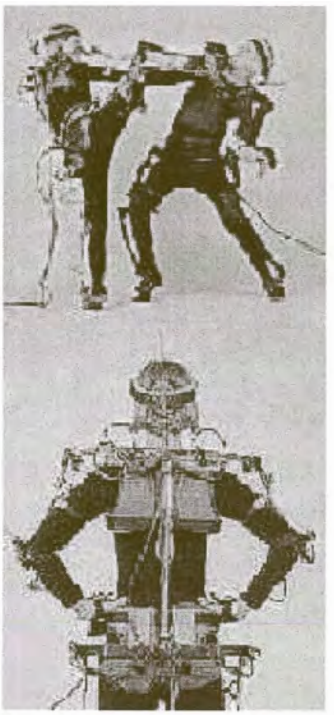


Figure 2.3. (Top) The datagloves are used in capturing motion of the actor's hands [95].

Figure 2.4. (Right) The exoskeleton is used to capture full body motion, but it is rigid and has limited body movement for actors. Data is captured in real time [96].

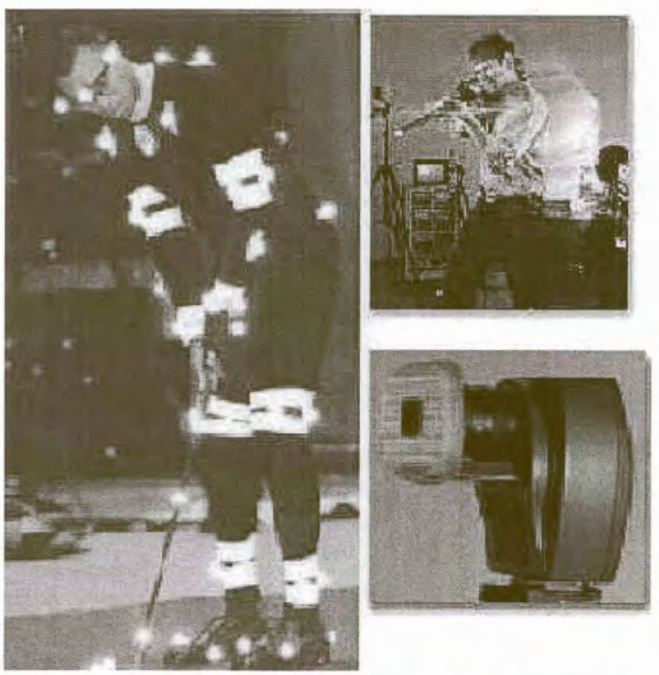


Figure 2.5. The left image showed optical sensors on Mr. Tiger Woods' body, tracking his golfing body motion for a computer game.

The right top image shows the pre-set-up area for capturing body motion [97].

The right bottom image is an example of a digital camera, Eagle, developed by

Motion Analysis Corporation [98].

Figure 2.6. Star Trak, wireless electromagnetic sensors [99].

Apart from body motion, facial movement data can also be obtained using "motion capture". The "motion capture" data played an important role in realistic avatar development. The data obtained by the motion capture process can be used off-line or on-line. During off-line, these data served as filtering for the avatar motion animation, or as Inverse Kinematics (IK). Inverse Kinematics modelled the forces that caused the motion to occur, and this is a technique for positioning end-effectors of the avatar in individual frames of the animation.

While in on-line, these data drove the avatar's motion directly based on the motion of the actor in real-time.

The body motion of the user can be captured by the video and detected via electromagnetic sensors, or virtual reality devices used by the user. Depending on the

type of input devices selected for the application and the type of VR application, the motion of the avatar can be animated differently in the system. For example, in an immersive collaborative environment, the input/output devices used are stereo cameras and projection displays. The video avatars used in the application and their motion will be based on the movements of the user in front of the camera [46]. Therefore, the kinematics of the video avatar is simply the direct video capture of the user.

When the VR application involved using synthetic avatars, the input devices used for the system are the virtual reality tracking sensor (optic or electromagnetic) [66]. Then the avatar must be constructed such that simulating the motion of the user can be done directly from the captured motion data or animated the motion directed by the animator. This required motion detection, which sensed the user's movement and obtained enough data of the user's motion to animate the avatar.

The input values for the motion of the avatar can be received from the VR devices and sensors. If the user moved his/her arm, the sensors sent the input values to the system and computed the change in body motions and then applied these same values to animated the avatar's arm.

2.3.3.2. Motion Control

Magnenat-Thalmann and Thalmann [49][19] classified the computer animation scenes involving synthetic avatars according to the types of avatar interaction and method of controlling motion. Motion control methods specified how an avatar is animated in the application. Motion control methods can be classified according to the types of information belonging to the synthetic avatar that has been animated in the system, into forward dynamics and key-frame system. In a forward-dynamics-based system, when a force acted on the parent object, any child objects are affected with the parent. E.g. If the body is pushed moving backwards, the arms will also move backwards. The information stored in the system is a set of forces and torques. In a key-frame system for an articulated body, the body is positioned at the critical frames with the current or target motion. The animation frames between the critical frames are filled with in-

between motion progressively. The information stored in the system is the angles of the joints. Motion control methods required handling geometric information like the angles of the joints.

The information for the motion control of the synthetic avatars can be separated into three categories: geometric, physical, and behavioural.

In geometric motion-control-methods, the information is of geometrical nature. Motion is defined in terms of angles, co-ordinates, and other shape characteristics. They are applied to calculated the deformations on the bodies and faces, and determined the skeletal motion.

In physical motion-control-methods, the physical characteristics and laws served as the basis for calculating motion. The information used for physical motion-control-methods included mass, moments of inertia, and stiffness. Physical laws helped controlling the skeletal motions, and the face and body deformation calculations. These deformations usually applied to the muscles in the virtual bodies and faces.

Behavioural motion-control-methods defined an avatar's motion in terms of behaviour, which is referred to as the way that animals and human act [14].
It is complex to displayed behaviour in avatars, because they vary among avatars, depending on the type and form of avatars. For example, if the avatar is feminine, then the walking motion and sitting posture will be different to an avatar representing a male user. Therefore, behaviour motion-control-methods would be less formal than geometric and physical motion-control-methods.

2.3.3.3. Avatar motion and Virtual world

Magnenat-Thalmann and Thalmann [49] described the relationship of the synthetic avatar with the virtual world as actor interfaces, which has their own sets of motion-control-methods.
Actors interface contained four basic cases:

- Single avatar situation: The synthetic avatar does not interact with other objects. The motion of the avatar is simple.
- Avatar-environment interface: The synthetic avatar is moving in the environment and it is awaken by its environment. The environment will affect the motion of the avatar. The motion of the avatar depended on the forces that exist in the environment. For example, if a steep path exists in the environment the motion speed of the avatars walking movement will changed and the physics needed to be taken into account.
- Avatar-avatar interface: The synthetic avatar responds to the action performed by the other avatar. The motion of the avatar is dependent on the motion of the other avatar. For example, if avatar A pushes avatar B, the possible motion of avatar B can be falling, moved slightly or nothing happens depending on the physics of avatar B (size, weight, etc.).
- Animator-avatar interface: The synthetic avatar responded to the action performed by the animator. Therefore, the motion of the avatar depended on the action or motion of the animator.

2.3.3.4. Retargetting motion

Although we have defined the avatar's motion control, data capture used for animation and, determined the body structure to aided the animation of avatar motion, it will be more efficient, if we can reused the motion controls and other information that we had determined from one avatar onto another avatar. This is known as "Retargetting motion", which adapted the motion animation from one character to others that might differed in geometric size and structure appearance. For example, we first developed the tall adult avatar walking motion, but we want to extend it for a small child avatar. If this motion is applied directly to the small child without retargetting, the feet of the child avatar during walking motion will not touched the floor, as the constraint of the feet touching the floor is not re-adapted in the body motion animation.

Apart from adapting the motion for one avatar onto another avatar, retargetting motion is required when motion capture is done. Without retargetting motion, the motion

capture data cannot be applied to the avatars that had different sizes or proportions than the actor from whom the motion data was obtained using the motion capturing devices. There are few techniques tried to tackle the retargetting problem. In general, users are restricted to adapted motions using the same tools that created the motion which each frame is manually tuned in the tool. Kinetix's Character Studio [79], is a commercial system that supported retargetting motion animation. In Character Studio, the keyframes are adjusted to maintain the feet-steps and the balance of the motion, when it is re-applied to a new character. (Figure 2.7.)

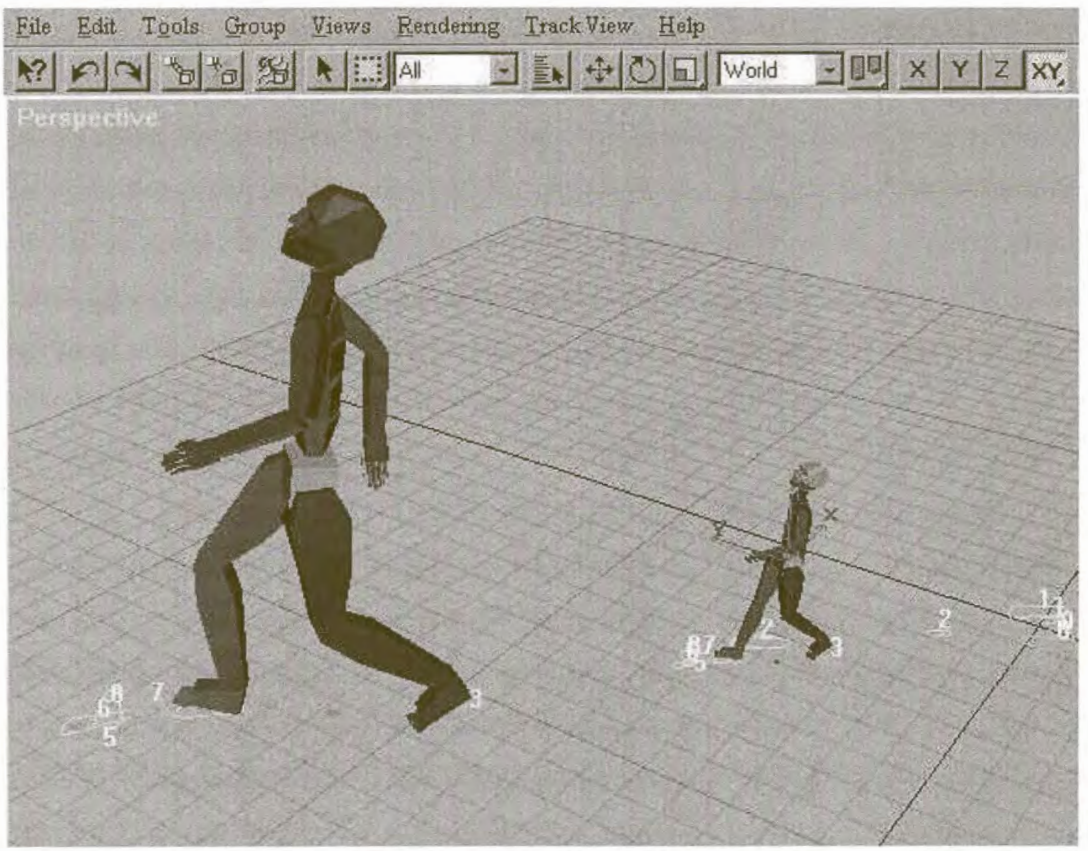


Figure 2.7. Retargetting process showed under Kinetix's character Studio R2.1, when the same motion is applied to characters of different sizes and the motion is re-adapted to each character without distorting the walking motion [79].

Hodgins and *Pollard* [81] addressed a variant of the motion re-used problem and adjusting the parameters of a physical simulation to adapted a controller used for a new character or a character that is changing it's size and shape. The procedural and

simulation based approaches for animation offered representations that are independent from the character, which generated new motions for new characters. Procedural and simulation controllers are able to adjusted different characters easily. However, these methods do not addressed the problem of retargetting, they can generated new motions for new characters, but do not reused existing motions. Re-generation of motion risked losing qualities in the original. The goal of these approaches are to created methods that adapted existing motions obtained from various sources, including the motion capture and keyframing as well as simulation and procedural generation.

In recent years, there is an interest in tools that allowed motion to be altered in ways that are independent of how it was created in the system. At their core, these tools treated animated motions as time-varying signals and applied signal-processing techniques to these signals. Litwinowicz's Inkwell system [80] demonstrated the utility of applying signal processing methods to animation data. Perlin [82] showed how existing motions could be blended together, and how the addition of noise to a motion could be used to transform it. While Gleicher [83], solved the retargetting problem by finding the adaptations required and the setting constraints for each motion that is applied to a new character. If these constraints are violated, re-adaptation process will occurred for the new character. Motions therefore can be reused on other characters independent to how they were created for the first character.

2.4. Facial expression and emotions in Avatars

Facial expressions are the best indicator of a person's mood, emotion and general "state" in the real world.

The most important facial-feature that indicates variety of emotions without other features, are the eyes. As seen in cartoons and comic strips, the simple characters have only eyes to show their facial expressions. Eyes on its own can communicate many basic emotions, but together with other facial-features more information is conveying to the respondent.

2.4.1. Designing facial expressions in avatars

Humans seek meaning through conversation and interaction. In a virtual environment, people interacted with other people through avatar interaction. Therefore, avatar interaction is not complete without facial expression and body motion. Some avatars in the virtual environment represent real people. During the interaction with other users in the virtual world, we want to know the emotion of the other user while interacting with them. Therefore, the avatar must be able to show emotion through facial expressions.

Facial expression analysis

Before facial expressions can be animated, expressions must be analysed, a head mesh must be generated, and texture mapped correctly, to allow expressions to be displayed. Face analysis can be divided into face detection, facial feature extraction, and facial recognition.

Face detection involves finding the position of the face of the actor. Facial feature extraction is concentrated around the facial features such as eyes, mouth, or eyebrows, which identified their position within the face of the actor [84].

Recognition, analysis and synthesis of expressions are often achieved using the user's expressions to drive the animations (tracking of the user's face) [53][54][55].

The expressions on the avatar's face can be determined by tracking the expression of the user's face and then the system determined the facial expression of user, this process is known as facial recognition.

There are two approaches to perform facial recognition.

Firstly, the facial feature markers are set on the video input. Capin et al. [42] defined a "Soft Mask" to manipulate the changes in the markers as the user's facial feature changed positions and orientation.

The alternative approach used markers (paint mark or beads on the face), that are on the actor/user's face and tracked them [77][22] (Figure 2.8.). The markers are usually in brighter and distinct-colours, which are easily identified by chroma keying. These markers simplified the expression recognition process and they are placed around the important facial features, eyes, eyebrows, nose, and the mouth. The limitation of this

approach is that these markers are irritating for the users, as they markers are stuck or painted around the facial features of the users.

Figure 2.8. Markers are placed on the actor's face for facial recognition [22].

Guenter et al. [52] created a system for capturing human facial expressions and replayed them as a highly realistic 3D "talking head" consisted of a deformable 3D polygonal face model with a changing texture map onto the model. In their system, fluorescent coloured paper fiducial are used as markers on the actor's face, which is used to tracked the facial motion as input data for the system.

Functional control can be used to interpolate and replay expressions that are predefined in the system. This allows expression to be created through the control panel interface manually. Expressions of the user's image from the video can also be texture mapped onto the face of the avatar directly [42].
This is unfeasible, because it required more bandwidth to stream video images and mapped them to the face mesh of the avatar continuously, than sending the data value of the position of each facial feature to the application. In case of error in transmission, the facial animation will not be smooth and the head movement will be rigid.

Waters [16] defined a facial expression of a person by muscle movements in the face. He simulated the muscle movements to generated facial expressions on the avatar's face. Therefore, facial feature values are sent to the system and the fast algorithm implemented the changes affecting the expression of the avatar. Expression is therefore broken into virtual muscle movements. Similar to Waters' approach Magnenat-

Thalmann et al. [24] described the concept of abstract muscle action procedure (AMA procedure), which simulated the specific action of a face muscle. Each AMA procedure corresponded roughly to a muscle or bone structure. AMA procedures are not independent of each other, so the order of action is relevant. Since human muscles are complex, the AMA procedures must simulated the same motion without imitating the complexity of the actual muscles. It is possible to animate the low-level expressions by manipulating the facial parameters using AMA procedures. By combining different AMA procedures, complex expressions can be animated in the application.

Mesh predefined expression

Facial expressions of the avatar generated by moving, or deforming the specific facial features created on the head mesh [42]. If jaws, teeth, hard palate and tongue is created with the head mesh, lips and speech simulation can also be implemented in the application [75][76]. The amount of movement or deformation on a specific facial feature can be obtained via capturing the user's facial movements, or predefined these expressions in the application. This enables the user to select an expression from the keyboard, which similar to the ones used in mail messages and chat-rooms, which is less complex and do not needed processing like facial recognition.

In lip movement simulation, Lavagetto [3] showed that by analysing the audio signals of speech, it is possible to extract from audio the visual parameters of lip movement. An application doing this recognition and generating motion parameters for controlling the face is connected to the VE program through the facial expression driver.
The facial representation engine developed by Lavagetto, then synthesising the face with the appropriate lip movements. A primitive version of such a system would just open and close the mouth during speech. A sophisticated system would actually synthesised realistic lip movement, which is an important aid for understanding speech.

2.4.2. Texture Mapping the face mesh

Before facial expressions are animated in the application, the face mesh must be texture mapped correctly. The face consisted of detailed facial features, if the texture is not

mapped correctly it will result in an unrealistic face and caused deviation in the facial expressions.

2.4.2.1 Face Texture extraction

Images of the user's head are captured by the video input and used as textures for the face mesh. The image of the user's head is separated from the background using "Chroma-keying", or position the camera close to the user's face to avoid the background appeared on the face texture.

2.4.2.2 Markers removal from texture

If the texture of the actor's face is captured while the markers are still on the user's face, a marker removal process is needed before the texture can be mapped onto the mesh. Guenter et al. [52] described a process for removing the markers and their associated illumination effects from the camera images of the actor. The markers are removed from the camera image sequences by substituting each pixel that is covered by the colour markers with the skin texture. Inter-reflection effects are noticeable at some parts of the face fold, as the reflective surface of some markers come into close proximity with the skin. The diffused inter-reflection effects and any remaining colour cast from stray pixels that have not been properly substituted are removed from the image.

The skin texture substitution begins by finding the pixels that correspond to coloured dots. A marker mask is generated by applying the classifier to each pixel in the image, which marked the pixels that required to be substituted by the skin texture. The skin texture is divided into low spatial frequency and high spatial frequency components. The low spatial frequency components of the skin texture are interpolated using a directional low pass filter oriented parallel to features that might introduced intensity discontinuities. This prevents smudging of colours across sharp intensity boundaries (e.g. boundary between the lips and the lighter coloured regions around the mouth. The directionality of the filter is controlled by a two dimensional mask which is the projection into the image plane of a three-dimensional polygon mask lying on the 3D face model.

Since the polygon mask is fixed on the 3D mesh, the 2D projection of the polygon mask stayed in registration with the texture as deformation is applied to the face model.

All of the important intensity gradients have their own polygon mask (the eyes, the eyebrows, the lips and naso-labial furrows). The 2D polygon masks are filled with white and the region of the image outside the masks is filled with black to create an image with a low pass filter. The intensity of the resulting image is used to control the filter's directional. The filter is circular symmetric where the image is black, far from the intensity discontinuities, and it is directional where the image is white. The directional filter is oriented so that its long axis is orthogonal to the gradient of the image.

The high frequency skin texture is created from a rectangular sample of the skin texture taken from a part of the face that has no markers. The skin sample is high-pass filtered to eliminated the low frequency components. At each marker mask's pixel location the high-pass filtered skin texture is first registered to the centre of the 2D bounding box of the connected marker region and then added to the low frequency interpolated skin texture.

The remaining diffused inter-reflection effects are removed by clamping the hue of the skin colour to a narrow range determined from the actual skin colours. First, the pixel values are converted from RGB to HSV space and then any hue outside the legal range is clamped to the extremes of the range. Pixels in the eyes and the mouth are found using the eye and lip masks. Using this marker removal process, the camera image can be used as texture for the face-model even if the markers on the actor's face are not removed from the face.

2.4.2.3. Determining texture mapping

When texture mapping the face of the avatar, mapping co-ordinates must be determined, so that the textures are mapped onto the face mesh in the correct position.

View- independent texture mapping

In order to support rapid display of the textured face model from any viewpoint, it is desirable to blend the individual photographs together into a single texture map. This texture is constructed on a virtual cylinder enclosing the face model.

Won-Sook Lee, P. Kalra and N. Magnenat-Thalmann[43] used the following approach to mapped textures on the avatar's head. The standard cylindrical texture map (Figure 2.9.), two images of the user (front and side view) can be jointed to form one texture image of the whole head, allowing the texture to be viewed at all angles. By using a cylindrical projection on these images, the front view covered from –90 degree to 90 degree, right view from 0 degree to 180 degree and left one from –180 degree to 0 degree. The front view is best for a certain range, the same for the other views of the head mesh (Figure 2.10.). By using the conventional blending method for wider range and using variation [57], it is easy to blur certain shape of features. The images on the front and side are cropped for specific points.

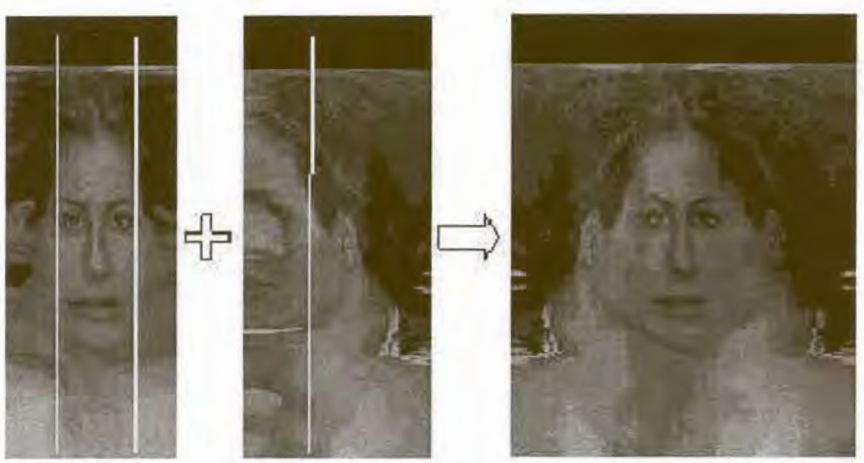


Figure 2.9. A standard cylindrical texture maps the front and side view of the user's face texture together [43].

With the information of the positioning of the eyes, the front view can be cropped automatically, the repetition of the same process for the left and right views and

assembled them to form a facial texture in 360 degrees. Won-Sook Lee, P. Kalra and N. Magnenat-Thalmann used a multi-resolution spline method to assembled two images [58]. Then texture mapped with a composed image onto the 3D-head model by projecting 3D mesh of control points on the image and calculating Voronoi and Delaunay triangulation on the 2D points. The local barycentric co-ordinates of the non-feature points with a surrounding triangle of feature points are calculated in the process, and then determined the texture co-ordinates on each vertex on the 3D surface from a 2D-texture image.

Figure 2.10. The left image indicates the head mesh before mapping. The right image shows the result from mapping the standard cylindrical texture onto the head mesh [43].

View-dependent texture mapping

The main disadvantage of the view-independent cylindrical texture mapping is that its construction involved blending together re-sampled versions of the original images of the face. Due to this re-sampling, the resulting texture is slightly blurry. This problem can be alleviated to a large degree by using a view-dependent texture mapping [93] in which the blending weights are adjusted dynamically, according to the view. For view-dependent texture mapping, the model is rendered several times, each time using a different input photograph as a texture map and blending the results.

View-dependent texture maps have many advantages over the cylindrical texture maps. Firstly, the texture map covered the lack of detail in the model. Secondly, if the model is projected onto a cylinder with overlapping, the cylindrical texture map will not contained data for some parts of the model. View-dependent texture map will contained all the data of the model as the geometry of the mesh matches the photograph.
One disadvantage of the view-dependent texture mapping is its higher memory requirements and slower speed due to the multi-pass rendering. Another limitation is that the resulting images are much more sensitive to any variations in exposure or lighting conditions in the original photographs.

The parts of the mesh that correspond to the eyes, teeth, ears and hair are textured in a separated process. The face usually occludes the eyes and teeth, and it is complex to extract a texture map for these parts in every facial expression. The ears have an intricate geometry with many folds and it cannot be projected without a cylinder that has overlapping. The hair has fine-detailed texture that is difficult to registered across facial expression. Therefore, each of these facial features and hair has an individual texture for realistic head models.
Texture mapping can be applied to a complex and detail head achieved higher level of realism. However, the process of finding the texture co-ordinates is more difficult as the number of vertices becomes numerous.

2.4.3. Facial Animation in Avatars

Over the years, there are various facial animation approaches developed and a distinct generalisation can be made from them. The one is texture manipulation based and the other is face mesh manipulation based.

Texture manipulation of facial animation is achieved by morphing the texture of a neutral face that is pasted on a model of a head or a face mesh (moving pixels in the texture) to a texture with facial expression. The alternative approach is adjusting the texture co-ordinates of the facial features in the face mesh, using texture co-ordinate displacement approach as proposed by Valente and Dugelay [37]. This approach

generated the facial expressions by displacing the texture co-ordinates of the face texture that is texture mapped onto a face or head mesh to simulated the facial expression, which is less in computation and complexity. Since facial animation is done at the texture level, unlike the other approaches where facial animation is done at the mesh level.

For sheer photo-realism, one of the most effective approaches used 2D morphing between the photographic image [91]. The only limitation of this approach is that it requires the animators to specified a few correspondences between physical features of the actor in every frame, and it does not correctly account for changes in the viewpoint or object pose. These approaches use few system resources, low computations and simpler than the approaches that involved morphing the face mesh.

Face mesh manipulation approaches for facial animation can be further divided into low-level muscle motion simulator, known as action units, abstract muscle action procedures, or minimum perceptible actions.
The following are face mesh manipulation approaches for facial animation:

- *Key-framing*, one of the earliest approaches taken, this involved linear transformations from one face mesh to another. This approach involves extensive computation and requires large data set. This approach is inflexible as the number of expressions that can be generated are restricted by the key-frames that were already digitised and it is also difficult to generalised the work on one face mesh to another.

- *Parametric Deformations*, which models the human face as parametric surface and recorded the transformations as control points' movements, in order to minimise the data storage requirements [21]. The limitation of this approach is that it is difficult to generalise over different face meshes.

One attempt is to utilise the B-spline patches that were defined manually on an actual digitised face mesh and the control points of the B-spline patches are moved to effect

the distortion of the face mesh. This method is powerful, but there is no automatic way of defining the relevant control points for the B-spline patch [28].

The other attempt used rational free-form deformation [60] to move points inside a defined volume with respect to the control points placed at the edges of the volume. The volume can be distorted by changing the position of the control points. This approach has the same limitation as the above approach.

- *Anatomically correct muscles*, this approach simulates the actual human face muscles underneath the skin (Figure 10.). This approach can simulate large number of facial expressions, but it is complex and difficult to understand [16][45].

- *Pseudo-muscles*, this hybrid approach simulated muscles that can be anatomically incorrect. Only the muscles that are related to facial animation are taken into account. It is easier than the face meshes manipulation approaches mention previously and it only required small data set [21].

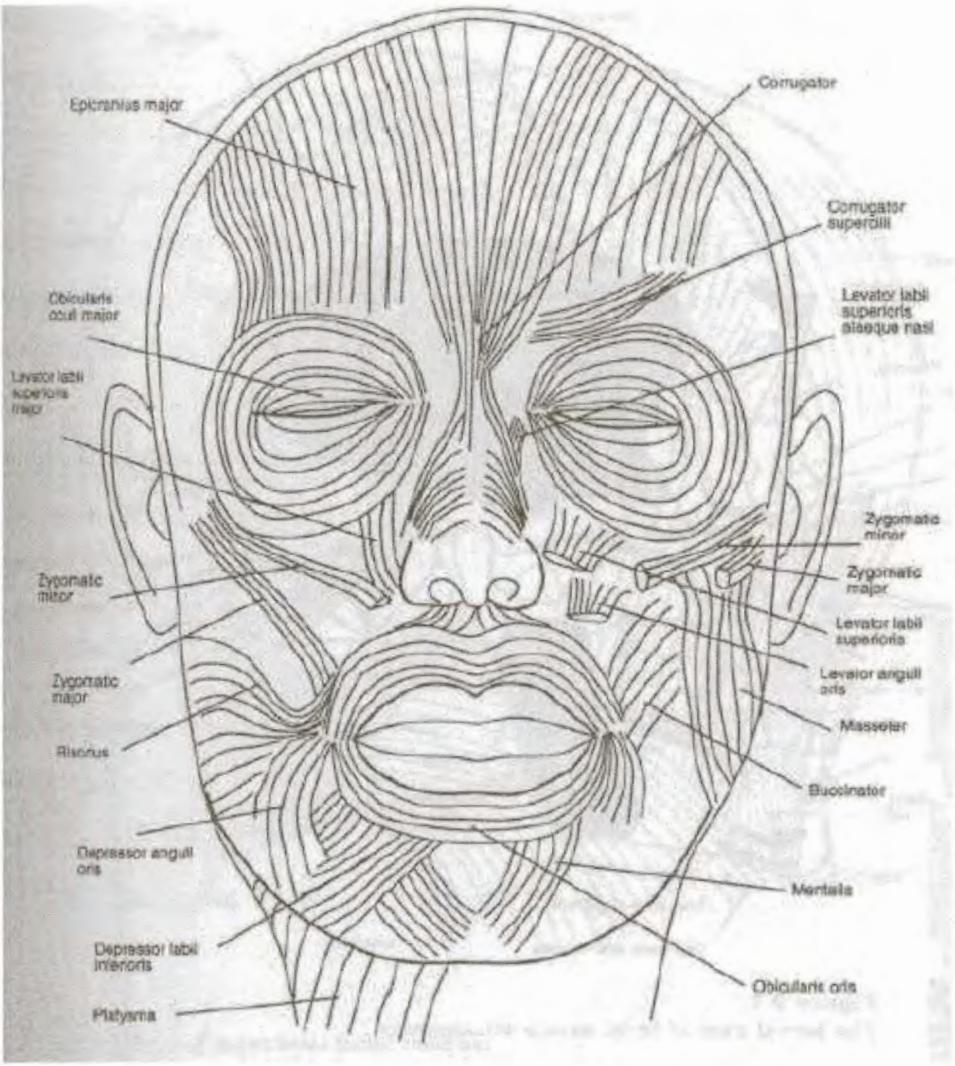


Figure 2.11. The underlying human face muscles are simulated under Water's Anatomically correct muscles approach [16].

MPEG-4 *Facial Animation Parameters*:

The multimedia MPEG-4 standard describes one of the primitive objects defined in this standard is the "face model", which can be displayed and animated according to a number of predefined rules [20]. In particular, FAPs (Facial Animation Parameters) are used to model and encode facial movements. Therefore, the face object is treated as a pseudo-muscular model [87]. Facial Animation Parameters (FAPs) are utilised in the framework of MPEG-4 for facial animation purposes. This enables efficient hybrid

coding for synthetic avatars with natural video and enables the animators to focus on local or global actions on the face, by means of "scripting" the animation sequence. For example, the animator can instruct the synthetic model of a human face to "open mouth" or "close eyes". This instruction is passed to the MPEG-4 decoder, which deforms the model by translating the vertices that correspond to the area of the facial feature to be animated in the animation. The standard does provide for the abstract definition of expressions and emotions as a collection of FAPs, and their subsequent interpolation into intermediate expression. This does not necessarily mean that all possible expressions and emotions can be modelled using this approach [88]. The definition parameters defined by the MPEG Group allow a detailed definition of body/face shape, size and texture, while the animation parameters facilitating the definition of facial expressions and body postures [89]. These parameters are designed to accommodate all natural possible expressions and motions, thus covering not only representation purposes, but also entertainment. This approach provides realistic results, however the limitation of this approach is that not all expression can be modelled using FAPs and it requires expensive hardware to computed complex manipulations of mesh points.

In 3D deformation, the facial expressions are captured from the video and calculated to obtained deformation data for the face mesh to animated facial expression. The result achieved from this approach is very realistic with life-liked face meshes, however the analysis video stream to determined the deformation is a time-consuming process, which required specialise and expensive hardware [52].

Pighin et al. [92] showed how 2D morphing techniques could be combined with 3D transformation of a geometric model to automatically produced 3D facial expression with a high degree of realism. The cameras capture multiple views of the actor with a facial expression; these photographs are digitised to generate the head mesh of the actor from a generic mesh. The texture of the mesh is extracted from the photos. While facial animation is produced by interpolating between two or more different 3D head models, and blended the textures together simultaneously. Since all the 3D models are

constructed from the same generic mesh, there is a natural correspondence between all geometric points for performing the morph.

After the different facial expression models have been generated, transition between expression can be produced automatically without specifying the correspondences between any expressions. This approach is attractive, because interpolating different basic facial expressions can animate complex facial expression. However, creating a set of different face meshes with different facial expression is high computation and time-consuming, while the eyes' gaze is fixed in all face models, unless the user gazed in the other direction in front of the camera.

Self-Evolving Personalities

As Capin et al. [42] suggested, it is possible for the avatar to change their emotions as the avatar's personality changes through time in the system using the object-oriented personality components.

The object-oriented personality components, with their characteristic behaviours, can be re-arranged and interconnected to form a personality matrix specified by the user. Personality component libraries would be available to a user who wishes to construct a "root personality" for their avatar, over which they would have initial total control. The facial expression of the avatar is predefined and they are linked to the avatar personality. However, just as with evolutionary personality growth in humans, the emotional components of the avatar personality would tend to generate their own nuance predilections over time.

2.5. Discussion

Apart from the implementation mentioned above, it is possible to combine the implementation of video avatar and synthetic avatar to produce faster and realistic avatars for collaborative virtual environment. For example, the video camera can capture the facial image of the user, then project the image onto a 2.5D avatar's head, and use a synthetic body for the avatar. In this way, the user can be identified through

his/her face by other users in the virtual environment. Alternatively, the user can use a Pure Avatar's synthetic body and tracking devices for better interaction and communication to other users in the collaborative virtual environment.

The selection of an appropriate approach for creating avatars depends on the type of interaction required for the application, the speed of system processing and the realism required. The methods for implementation can be selected based on their trade-off and advantages, or combination of the approaches. Currently, one of the fastest growing fields for avatars is in the entertainment industries. In the gaming video sequence, guided avatars act the introduction or the in-game cut-scene video through the guidance of the animator. While in the game, the autonomous characters/avatar represented the background characters or opponents, and the users are represented by guided avatars. The users can also interacted with other users using guide avatars in the gaming environment in multi-player mode.

Besides the academic organisations and gaming industries, research regarding avatars is conducted by many institutions from around the world.

The *IMPROV project* by NYU Media Research lab. This research project builds the technologies to produce distributed 3D virtual environments in which human-controlled avatars interacted with computer-directed avatars in real-time, through procedural animation and behaviour scripting techniques.

VET (Virtual Environment for Training), and the Steve avatar. At the Educational Technology Group, USC Information Sciences Institute is another example of research in this area

OZ project (CMU) (Avatar for Entertainment) Carnegie Mellon University (CMU). The Oz Project at CMU is developing technology that combines art to help an artist create high quality interactive drama, based partly on AI and Avatar Technologies. Creating avatars in an interesting dramatic virtual world.

MIT Synthetic Characters Group.

The goal of the Synthetic Characters Group at the MIT Media Laboratory is to understand how to build interactive characters that come alive in the eyes of the users who interact with them. They combined the ideas and disciplines of animal/human behaviour, traditional character animation, AI, robotics, avatar modelling and computer graphics.

Avatars aided in many applications, simulating events in the real world or the virtual environment, enabled distant users to participated in meetings and communicated in the same virtual environment. However, without the existence of avatars, interactions in the virtual environment will be impossible or non-stimulating.

In the Coven experiment [65], the users communicated in a virtual meeting environment. The results indicated that the avatars cannot fully represent people, if facial expressions and gestures are not available, then interaction and communicate becomes unnatural, because facial expressions and gestures allowed people to expressed themselves in a meeting or social environment.

Therefore, the avatar development process discussed in this thesis will concentrate on creating fairly realistic facial expressions using the expressive texture approach and generating simple, upper body movement for avatars, that is suitable for interacting in a synthetic social environment. The next chapter describes the expressive texture approach, in creating facial expressions by manipulation of the face texture.

Chapter 3

Expressive Textures

Expressive Textures approach uses textures of images of both the synthetic faces and the real faces captured from video and a simple low-polygon face/head model. It provides an interactive way of fine-tuning and adjusting the underlined model, which allows a more realistic mapping for a special facial image. Further more, it concentrates in creating facial expressions by texture manipulation. The first section of this chapter discusses the process of face model generation in Expressive Textures approach. Section 3.2 discusses how texture images are texture mapped onto the face model and the process of texture mapping adjustment. Later, Section 3.3 discusses how facial expression is created using Expressive Textures. Section 3.4 looks in-depth at the eye animation and discusses how this can be achieved using Expressive Textures. Finally Section 3.5 gives a short summary of the results of the expressive textures approach.

3.1. Face model

There are many approaches developed to generate the face model as discussed in the previous chapter. Different face mesh generation method can be applied depending on the application and the realism required in the application. In expressive textures, the amount of system resources (System Memory, CPU time, etc) used should be low, while providing a fair amount of realism with low computations. This allowed the possibility of extending the approach to be used over low bandwidth networks for distributed collaborative virtual environments, which is improved by a sense of mutual awareness

using facial expressions. Therefore, keeping the resources used to a minimum allows the application to support more face models in a single virtual environment. If the whole avatar body is implemented in the virtual environment, extra system resources can be used to focus on computing body animation of the avatar.

First the face model will be designed with low polygons and simple geometry, so that texture mapping face images on the face model can be done more easily, and allow animated facial expressions, while keeping the system resources usage and computations lowest.

In this thesis a generic, low complexity face mesh is manually created, which different images of real or synthetic human faces can be mapped onto it. The reasons for creating a generic face mesh than using scanning equipment generated a face model from a human subject is:

- Scanning equipment is expensive.
- The face model result from a scan have too many polygons/vertices, which increased face model complexity, computations (e.g. texture mapping computation) and usage of system resources.
- The face model is difficult to map texture onto it, because there are numerous texture-mapping co-ordinates and the amount of texture-mapping computations is large.
- The face models generated from a scan are usually difficult to animated facial expressions, due to large number of control points.
- By creating a generic face mesh, we can avoid the need for creating more specialised face mesh. This also avoids the face mesh morphing computations required re-adaptation from the default face mesh before applied to another face mesh that do not resembled the face of the actor/user.

However, there is a disadvantage by manually creating a generic simple face mesh, the face mesh is less realistic than the face mesh generated from scanning equipment.

To simplify the face mesh, eye sockets and eyeballs are not created in the face mesh. This differs to other face models implemented by other researchers. Since a face model with eye sockets and eyeballs simulates real eye movement by rotation of the eyeballs, but this increases the number of polygons in the face model (Figure 3.1). Therefore in this thesis real eye rotation is simulated at the texture level instead of the face mesh level reducing the number of polygons; this process is further discussed under eye animation at section 3.4.

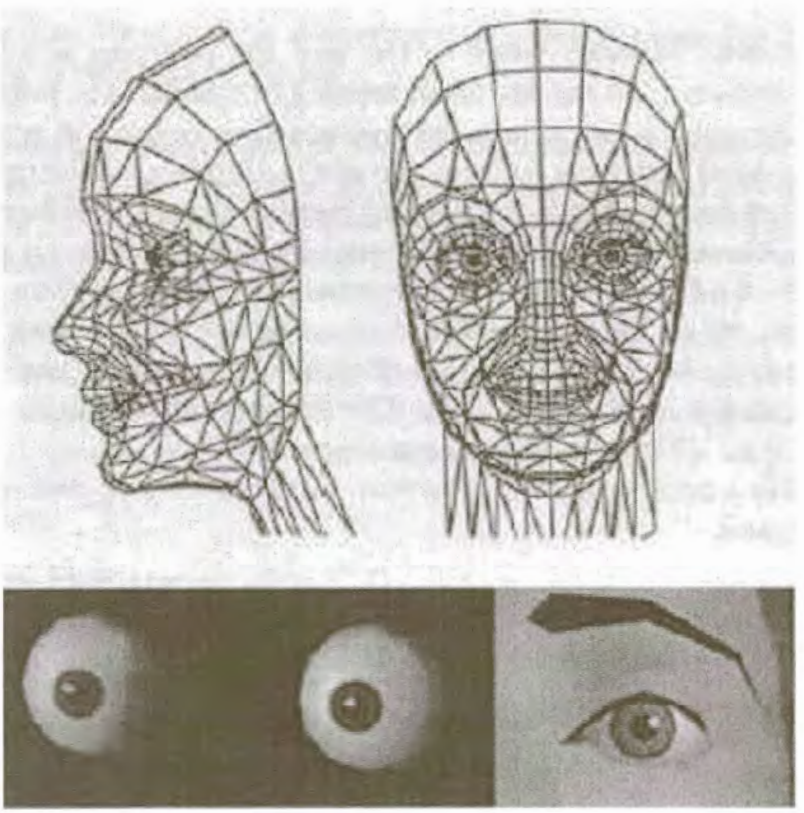


Figure 3.1 A synthetic face mesh with eye sockets and eyeballs, which can simulate the eye animation by rotating the eyeballs [87].

Some facial parts that are not affected under the orientation of the face and can be viewed from different angles – such as the mouth and cheeks are modelled as slightly round surfaces, while the eyes and eyebrows are modelled as slant planes. The nose is modelled so that it protruded from the face mesh, giving the face mesh a non-flat appearance.

Once the face mesh is designed in the application environment, a few vertices in the face mesh are defined as control points (Figure 3.2). Control points are the points in the face mesh, which controlled the facial animation. The control points are set only around the important facial features (Figure 3.3), because the facial expressions are created by the changes at those facial features. In other face models, the control points are usually the vertices that are translated to simulate the movement in the face. In our face model, the control points remain at the same position, only the texture mapping co-ordinates at the control points are translated during facial animation.

Figure 3.2 (Top)
Indicating the positions of the control points around the facial features, this marks the facial features to be mapped in the same corresponding region in the face mesh.

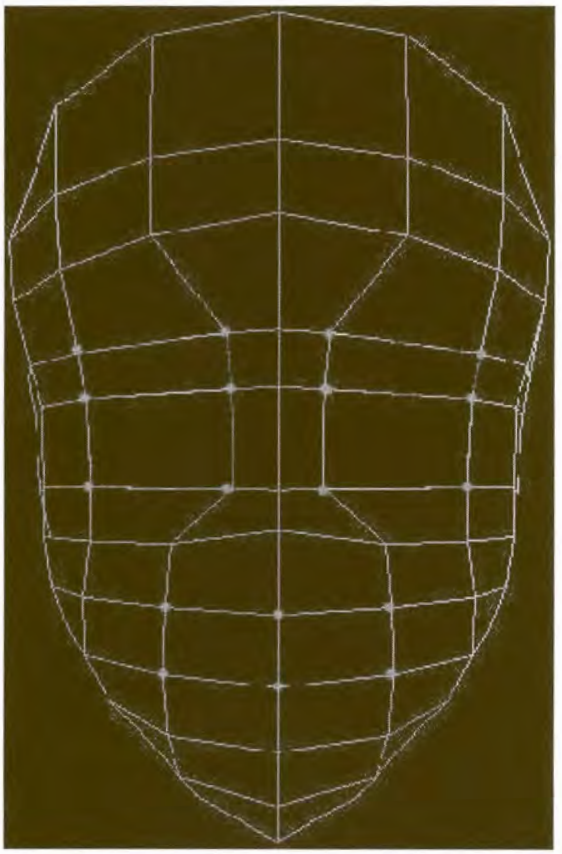


Figure 3.3 (Right)
The face mesh with the dots indicating the control points.

Although we have defined the control points for the face mesh, animating the face expression is only possible if the texture of the facial features is texture mapped around the control points.

It is difficult to texture map the whole face image onto the face mesh when the number of texture mapping co-ordinates are numerous, so instead of mapping the whole face image at once, the texture mapping process is broken down according to each part in the face mesh. This should facilitate texture mapping of the face image and we can pay more attention to areas around the facial features during texture mapping. Therefore, the face mesh is divided into strips of polygons, which separate the face mesh into regions, so texture mapping the facial features can be done more easily. Since faces vary in size and length, by dividing the face into regions, texture mapping different face images onto the same generic face mesh can be fine-tuned in the application environment.
After generation, the face mesh consists of 138 vertices and 238 polygons with 18 control points. Now we can determine texture mapping for the face mesh.

3.2. Texture Mapping

Texture mapping can be seen as a process where an image is pasted onto objects, but if the position of the image is not determined, the image pasted on the object can be distorted on the object.
Since we have divided the face mesh into strips of polygons regions, so we can determined the texture mapping co-ordinates for each polygon strip regions instead of the whole face mesh. In an image (Figure 3.4), a pixel lies in the pixel co-ordinate system in a square between the value (0,0) and the respective size of the image (width of the image, height of the image). In a texture (Figure 3.5), in OpenGL for example, a Texel lies in the texture co-ordinate system between the values (0,0) and (1,1). Any texture mapping value larger than value one will result in tiling the texture repetitively or clamping the texture. Therefore it is important to establish a relation between the pixel co-ordinate system and the actual texture co-ordinate system.

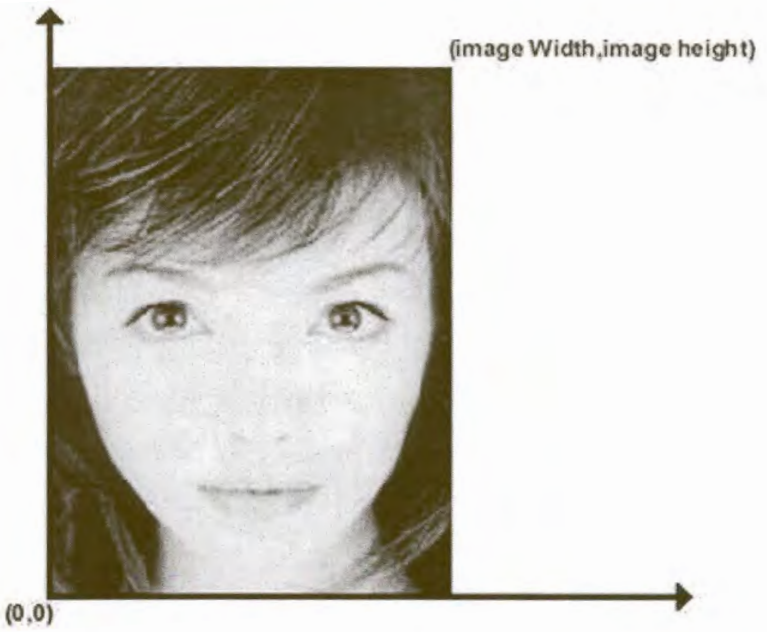


Figure 3.4 The Face Image in pixel co-ordinate system

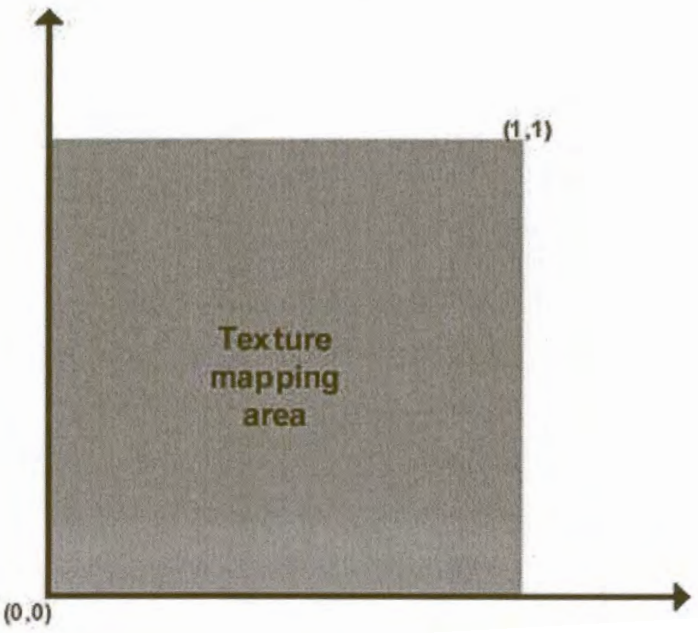


Figure 3.5 Texture mapping co-ordinate system, the texture co-ordinates lie in the texture mapping area will not cause the texture to be tiled or clamped.

Let $\lambda(x, y)$ be a pixel in the image and x and y are the co-ordinates of the pixel in terms of image size. Let $\mu(x, y)$ be the size of the image with x and y as the width and height of the image.
Then the texture mapping position m(x, y) is expressed by:

$$mx = \lambda x/\mu x \quad , \qquad my = \lambda y/\mu y$$

This can also be expressed as:

$$m(x, y) = \lambda(x, y)/ \mu(x, y)$$

Many image applications can help us resize an image, according to a ratio, so that the face images fitted into the generic face mesh. Unfortunately, we cannot simply resize the image by comparing the image size of the image that fitted the mesh to the one we want to texture mapped onto the face model, so that the image is not from a shoulder height and that it includes only the face.

In Expressive Textures, texture scaling is achieved by first finding the interpupillary distance (distance between the centre of the iris) in the facial image that correctly maps on to the mesh. Then dividing that by the interpupillary distance in the facial image that is texture mapped onto the face mesh. It can also be done, by drawing a perpendicular line between the eyes and the tip of the nose, then use this line as distance for comparing the two images. The distance of the texture image mapped onto the face mesh divides this distance of the new texture image.

These results give the ratio required for scaling the image. Then any portions that are not required in the image are cut away from the final face texture, e.g. shoulders.

Once the still image of the face is scaled to the correct size, it can be mapped to the face mesh using the default texture mapping co-ordinates or adjusted texture mapping co-ordinates (Figure 3.6). The adjusted texture mapping process is discussed later in this chapter.

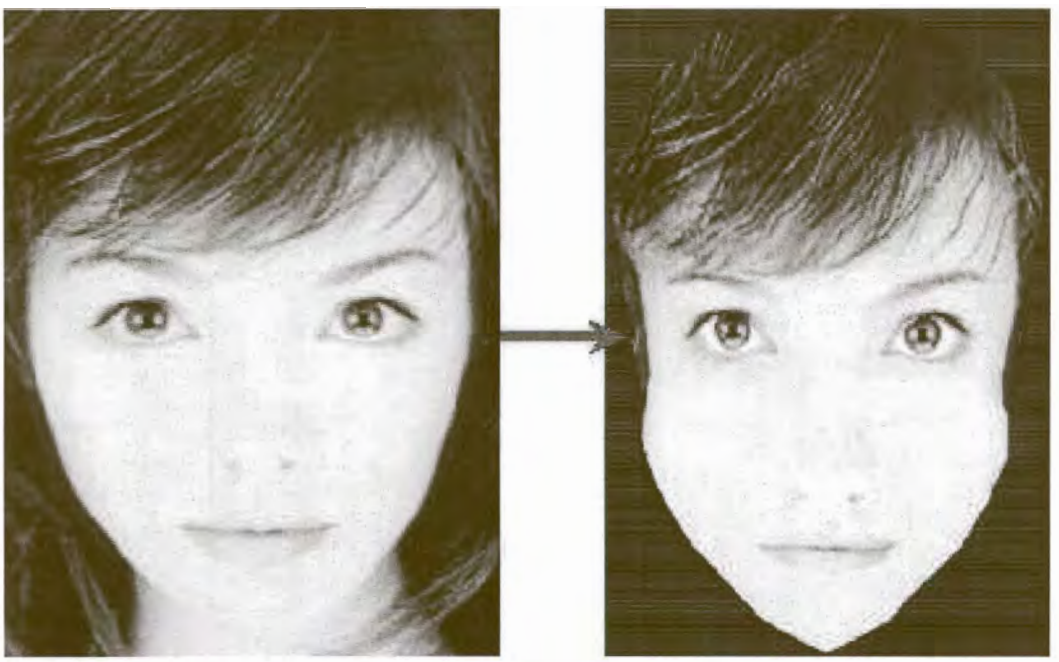


Figure 3.6 A still Photo image (left) is first scaled to the correct size, and then texture mapped onto the face mesh using the adjusted texture co-ordinates (right).

Alternatively, the user can create synthetic images and texture mapped them onto the face mesh using the same texture mapping process.

3.2.1 Video Images

Apart from using still photo images as textures for the face mesh, video images can be captured and texture mapped correctly to the face mesh using the same method as with the still photo images. The video camera must capture the image of the user's face; to avoid unwanted portions map onto the face mesh. After the video image is captured, the video image is scaled to the correct size before it is mapped to the generic face mesh with the default mapping co-ordinates. The default mapping co-ordinates distorted the video face image on the generic face mesh, because the faces are different in length and width. Therefore, tuning is required to adjust the texture co-ordinates for mapping the video image (Figure 3.7).

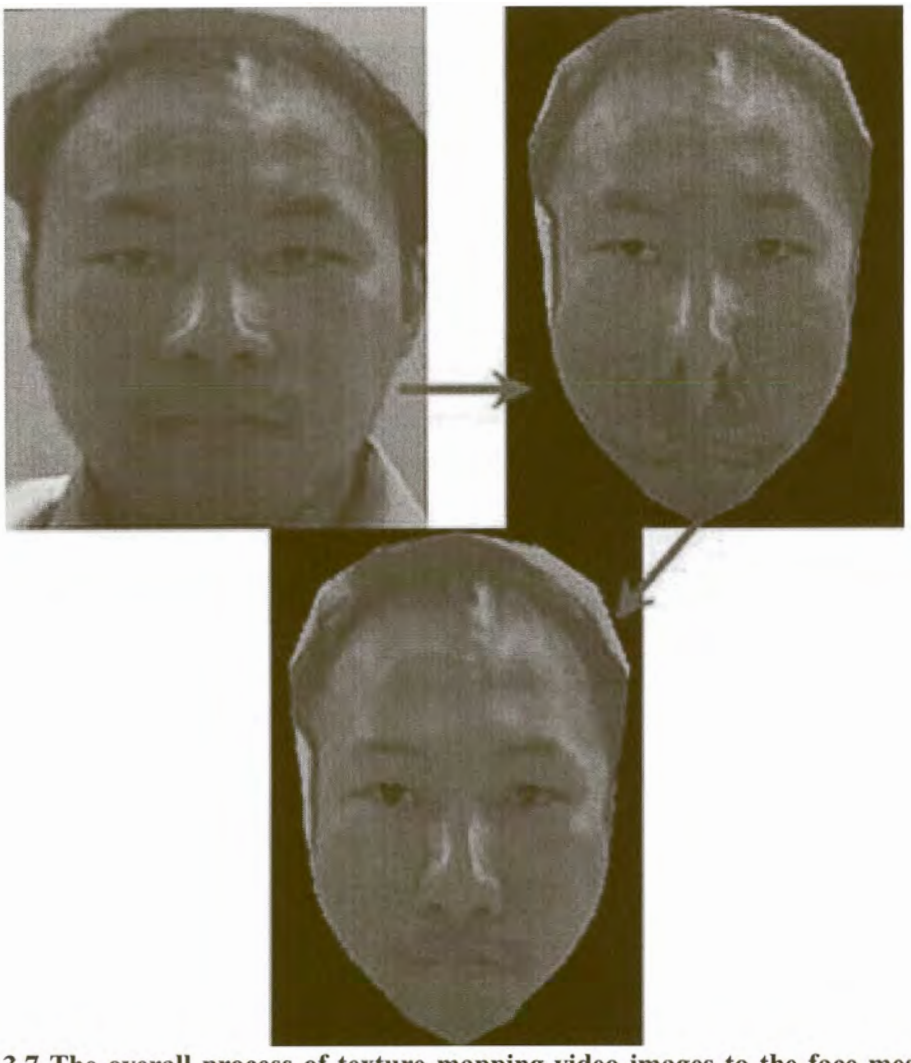


Figure 3.7 The overall process of texture mapping video images to the face mesh, first the captured video image is mapped onto the face mesh using default texture co-ordinates values, then the texture co-ordinates are adjusted to eliminate the distortion.

3.2.2 Texture Mapping Adjustment

Texture mapping co-ordinates can be adjusted at each vertex, but this is very time consuming even if the face mesh is simplified, because there are numerous vertices in the face mesh. Therefore, the Texture mapping adjustment process is broken down into easier steps of texture mapping adjustment. Initially, the default texture mapping co-ordinates map the face texture image's edge onto the face mesh's edge and approximate the texture mapping co-ordinates for the facial features. In some cases, the face image texture might not be in the centre of the image. To accommodated this problem, the

texture adjustment allows the user to position the face image's nose at the centre of the face mesh's nose position by shifting the texture mapping co-ordinates moving left, right, top and bottom directions. As the nose is a good indication for the centre of the face, this helps the user to position the face texture before tuning the texture mapping at the other facial features. The face is not flat and varies in length and width, thus it does not correspond to the positions in the generic face mesh. Therefore, some facial features (eyes, eyebrows and mouth) will be distorted or be at the wrong position. The next step in texture mapping adjustment is to position the facial features that are not mapped correctly, because the face mesh is divided into polygon strips texture mapping co-ordinates for the facial features can shifted horizontally. This will speeds up the texture mapping for the facial features. The reason is the eyebrows; eyes and mouth are usually level. Therefore when shifting the entire strip of texture mapping co-ordinates horizontally, we will found the texture mapping co-ordinates for both eyes or eyebrows at once, instead of mapping the one eye first and the other later. Once the facial features are mapped in correct position horizontally, the user can fine-tuned at texture mapping co-ordinates vertically (Figure 3.8).

This texture mapping process allows the user to texture map the overall face image and the main facial features quickly and then apply the fine adjustments at each texture co-ordinate when needed, at a later stage. Since the face image and the mask (an image usually in two colours, e.g. black and white, which marks a portion in the texture image to be transparent or act as filters) have the same size, texture-mapping adjustment is applied to the texture image and automatically to the mask too. This avoids the texture mapping co-ordinates to be re-determine for the mask, and prevents the masking area to differ in positioning at the face image from the mask and mask the wrong area of the face image. Texture mapping adjustment provides a simple texture mapping method with low computation.

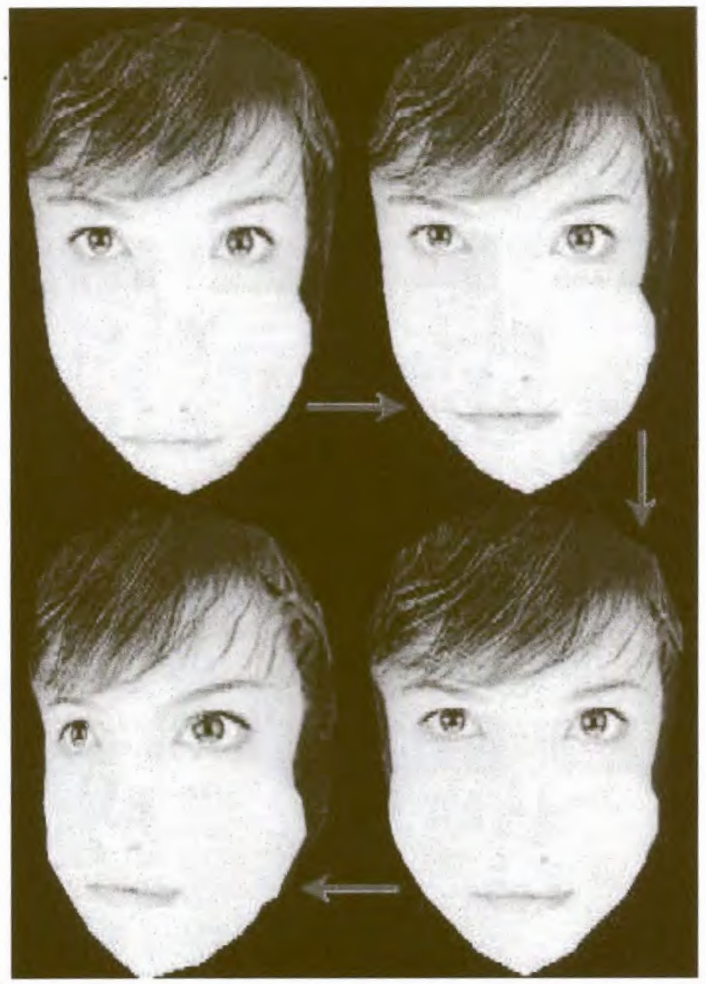


Figure 3.8 The texture mapping adjustment process:
The face texture is mapped using the default texture mapping co-ordinates initially.

Then, the nose is positioned at correctly first and the other facial features are later adjusted based on the nose position. The fine tune adjustment can be applied if necessary. The result will give us the adjusted texture mapping.

3.3. Facial Expressions

Facial expressions result from facial feature movement caused by the underlying muscles beneath the skin contracting and relaxing. To simplify the facial expressions, facial features are only taken into account and the underlying muscle movements are ignored, this makes facial expression easier to animate based on the movement of the facial features. Simple Facial expressions are build up from simple movements of the facial parts namely: eyes, eyebrows, and the corners of the mouth. Psychological research had classified six facial expressions that relate to distinct universal emotions: disgust,

melancholy, fear, happy, anger and surprise [16]. Notice that four of them are negative emotions: disgust, despair, fear, and anger. Only happy and surprise are classified as positive emotions. It is possible to group a few facial expressions with the common facial features, so during animation merging different facial feature movements may expand the number of facial expressions to be animated in the application.

In order to create a facial expression, each facial expression must be generalised into a set of facial feature movements (motion cues). From these motion cues, movement of each facial feature is known for a specific facial expression during animation and they can be applied to different faces. In Table 3.1, is a classification of each facial expression with their corresponding motion cues, it is also possible to add other facial expressions and classify their motion cues.

Expression	**Motion cues**
Melancholy	Lowering the mouth corners, raise the inner portions of the eyebrows.
Surprise	Eyebrows slide up, eyes wide open.
Fear	Inner eyebrows raise, eyes open, and mouth lowers slightly.
Happy	Raising mouth corners, lower outer eyebrows, eyes close slightly and upper cheeks expand slightly.
Disgust	Upper-lip raised, eyes close slightly and lower-eyebrows.
Anger	Inner-eyebrows lowered, outer-eyebrows raise and lips press against each other or lower slightly.
Cunning	Eye lids nearly close the eyes, and mouth corners raise slightly

Table 3.1 Motion cues of facial expressions.

Once the face mesh is correctly textured, we can start animating the facial expressions that are classified in the table above (Table 1.). In the face mesh manipulation approach,

the control points around the facial feature are moving to simulated a facial feature movement, so the change in control points positions must be determined for the face mesh. Alternatively in Expressive textures, the texture co-ordinates around the facial feature are moved to simulate a facial feature movement. Therefore, the texture co-ordinates for each facial feature under a specific facial expression must be determined and group according to each facial feature. For example, all the texture co-ordinates for the mouth are grouped together. After the texture co-ordinates data of all facial features are grouped, they are stored in the system as different arrays. Consequently, the user can select from the facial cue's movements defined previously and created new, different facial expressions. Additionally, the implementation is simplified and changes in any facial feature's movement will not affect other facial features.

A particular facial expression is animated by moving the texture co-ordinates of all facial features with the current facial expression towards the target facial expression's facial feature texture co-ordinates progressively. For instance, to simulated the closing eyes, the two upper texture co-ordinates can be moved closer to the two lower texture co-ordinates of the eyelid texture (Figure 3.9). To simulate eyes completely closed; this cannot be achieved by moving the two upper texture co-ordinates lapping over the two lower texture co-ordinates of the eyelid texture of an open eye. This is because the eyes texture will be extremely distorted and make the eye look unrealistic. Therefore, another still image of the same face with the eyes closed is required, since both face images can be merged to simulate eyes blinking or eyes closed (Figure 3.10).

Although displacing the texture co-ordinates simulated fairly realistic facial expressions, but in some cases, this is not sufficient. When a person smiles, the face muscles around the cheek contracted causing the cheek to expand slightly. Morphing of the face mesh is required to achieve this effect. When the happy expression is animated, the vertices around the cheek region in the face mesh will expanded slightly, to simulated the face muscle contraction during a smile. The application can now simulate the six universal expressions and other expressions by combining the motion cues defined in the system (Figure 3.11).

Figure 3.10 (Top image)
The face image's right eye is merged with the image of the face with closed eyes.

Figure 3.9 (Left image)
The eyes on top are at neutral position, while the eyes at the bottom shows the effect of moving the upper texture co-ordinates closer to the lower texture co-ordinates that map the eyelid texture to simulate eye close.

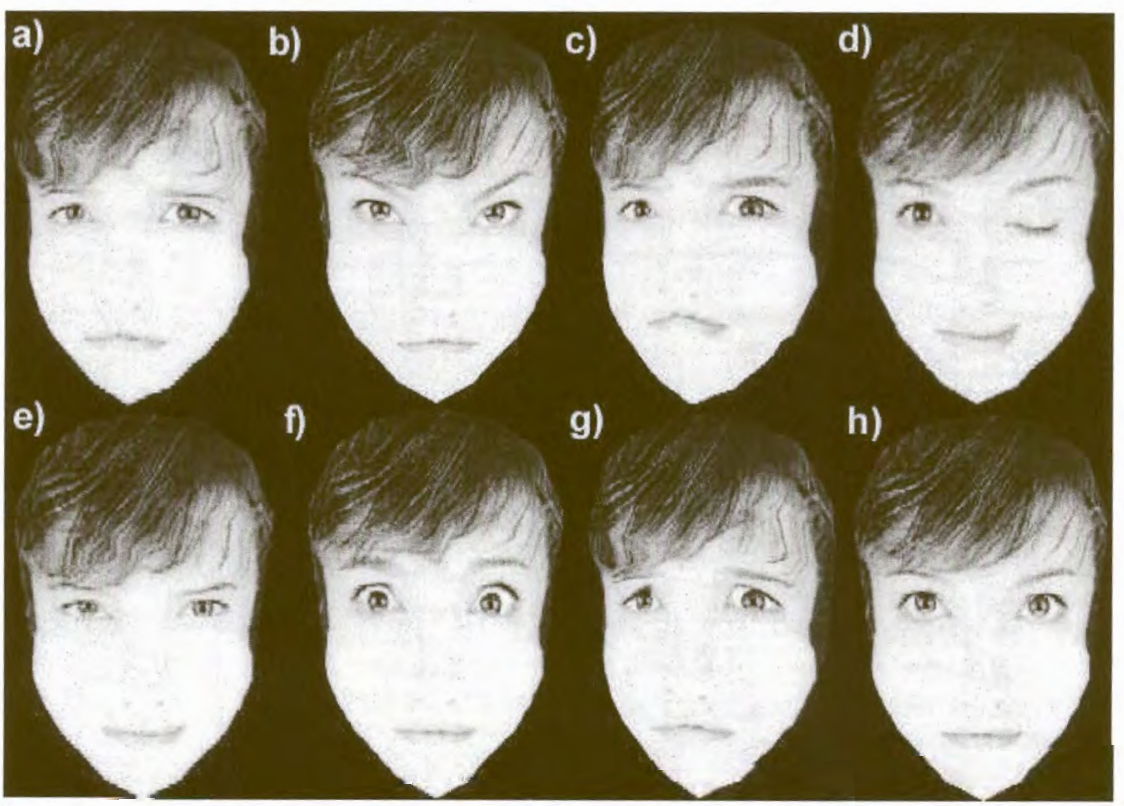


Figure 3.11 The facial expressions animated using expressive textures: a)Melancholy b)angry c)disgust d)happy with left eye closed e)cunning f)surprise g)fear h)neutral

3.4. Eye Animation

The human eyes are considered one of the most important features in the face for facial animation, because at times of extreme anger, showing a sign of boredom or a shy expression, we will usually avoid eye contact. Especially in virtual conferencing [30] the eyes can convey the attention of the user to the collaborating partner. If the user's eyes are directed away from the partner, this shows a lack of gaze awareness and general disinterest. If the user's eyes are directed at the partner, it creates eye contact and an attentive facial expression [94].

In order to show eye gaze, expressive texture approach must use a technique to simulated eye rotations even when still images are used in the application. This is more complex compared to the approaches that uses a face mesh with eyeballs and eye sockets, since simulating the real eye movements can be achieved by simply rotating the two eyeballs.

In expressive textures, although the eyeballs are flat in the face mesh, the eye socket must receded into the face mesh. This is important for creating a realistic look when the face mesh rotates up, down, left and right. First, we will need to extract the pupil texture from the face image and create a hole on the face image. This is achieved by using two masks; one mask is applied onto the face image to generate a pupil texture with only the pupils visible in the original face texture. The other texture-mask creates a hole at the eyeball position in the original face texture using the transparency channel. After both textures are generated, the face texture is mapped on top of the pupil texture with the pupils appeared under this opening (Figure 3.12). To simulate rotation of the eyes, the texture co-ordinates of the pupil texture are displaced on the face mesh.

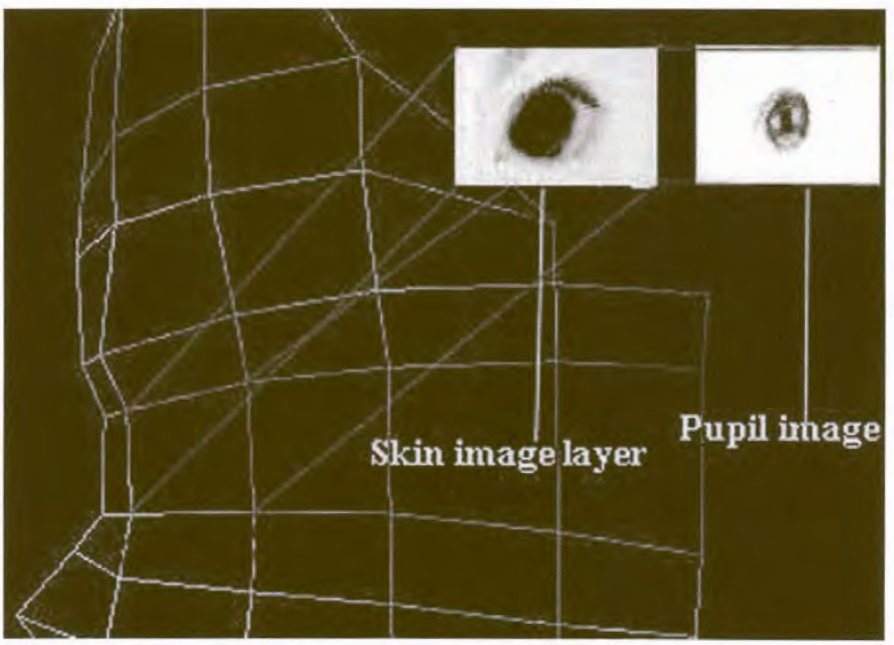


Figure 3.12 Image of the skin layer with a hole created by the masking that appears over the pupil image

The problem arising from this approach is that the speed of the eye movement cannot be adjusted in the application, so this approach will required to calculated the eye movement before the pupil texture co-ordinate is moved towards the target co-ordinate. The eye movement is calculated by:

Let "a" be the pupil image X or Y texture co-ordinate.

Let "b" be the target position of the pupil's X or Y texture co-ordinate, "ob" the current position of the pupil's X or Y texture co-ordinate, and "c" be any small value to control the amount of eye movement.

Then the equation is expressed as: a = a – c*(b - ob);

When animating a facial expression the eyeballs should be able to rotate freely as real eyes, because the facial expressions are independent from the eyeball rotation. With masking, the underlying pupil texture is independent from the image of the skin layer. When a facial expression is animated, only the texture skin layer is morphed according

to the motion cues and this morphing process (Figure 3.13) does not affect the pupil texture.

Both masking mode and non-masking mode are provided. In the non-masking mode, only the face texture is enabled and we can simulate facial expressions without providing the masks, but the pupils will be slightly distorted compared to the pupils at neutral expression.

Figure 3.13 The shifting of the pupil texture simulates the rotation of the real eye.

3.5. Summary

The Expressive texture approach for facial animation achieves fairly realistic results with low computations and simple implementation. If an image with skin fold (wrinkles) is available, blending can be applied to simulate movement of the skin on the face. The advantage of using this approach to animate the faces of avatars is that the facial expressions defined in the system can be applied to all the avatars once the face texture is correctly texture mapped onto the avatar's face. The other advantage is that synthetic face images can be mapped onto the avatar to create synthetic avatars and mapped video or photo face images to generate an avatar that represent the users.

The trade-off in this approach is that the hair on the still image will be distorted during animation, if the hair is lying close to the eyes or eyebrows. Another limitation is that mapping an open mouth onto a closed mouth is not realistic.

The next chapter will further discuss how full body avatars are designed in this thesis by combining the Expressive Textures approach, and design upper body animation for these avatars. The next chapter also discusses in detail, the theoretical approach and design of these avatars in a synthetic social-interaction environment.

Chapter 4

Social Interaction

People gained new knowledge by communication and social interaction. Virtual collaborative environments allow people communicate with others to exchange ideas, and interact with virtual objects and distant users in the same virtual environment. Social interaction, is when people communicate and interact with others in the real world, facial expressions and gestures stimulate this communication process. If the avatars only express its emotions through facial expressions, this creates an unnatural feeling for the users when they communicate and interact with other avatars in the virtual world. This is because gesture is another form of expressing emotions during communication and social interaction in the real environment, so attention must be given when modelling and animating the avatar body.

The first section in this chapter looks into the process of creating a simple full-body avatar suitable for a synthetic social environment and in how to animate the upper body movements of the avatar associated with facial emotions and other simple actions. Section 4.2 discusses the interaction process between different avatars and their virtual environment. Section 4.3 discusses briefly the effects of combining sounds in the virtual environment. Section 4.4 discusses the design of a simple state machine that will co-ordinate the interaction process between all the avatars in the synthetic social environment. The last section summarises this chapter.

4.1. Full body Avatar

When designing the body of the avatar, the body must be designed such that it is low polygon, able to perform animation, and the user must recognise that the model is a simplified human model. The reasons for these criteria are, when the number of avatars in the virtual environment increase, the computation and system resource usage will increase. By lowering the polygon structure of the body, the amount of system resource use will decrease, and more avatars can participate in the virtual environment. The body model must be able to animate body movements or simple gesture, to simulate real people interactions and communications in the virtual world. Although the body model is simplified, during interaction the users should feel or see that he/she is interacting with a human-like avatar and not a robot or a figure that cannot be identified easily.

4.1.1. Body Creation

The avatar's body is referred as a body model, because it is a representation of the human body. The body model is of geometric nature, because the model is a collection of components with well-defined geometry (the body consist of arms, hands, legs, feet, .etc) and interconnection between components exist (the hand is joined to the arm).
The advantage of representing the body as a geometric model is, because the inheritance and hierarchical geometric properties exist in geometric models, which similar to the human body e.g. when the arm moves, the hand move along with it.

The body model is decomposed into a collection of different parts, so that each part is modelled as an individual object e.g. the arm is separate into upper arm and lower arm. The reasons for modelling each limb as a separate object is, because the desired joint angle for each body part are determined more easily and the body with all the limbs can be structured hierarchically.

When drawing the human body artist usually first draw the head or the body and then the rest of body [100]. This is because the body must be position correctly in the picture first and then draw the rest of the body parts according to the position and size of the body. This is known as the top-down construction process.

In hierarchical modelling, the body model is created by a bottom-up construction process. This means that the lower-level body (e.g. Hand) parts are modelled first and served as building blocks from the higher-level body parts (e.g. Arm). The higher-level body parts are modelled after the lower-level body parts, if the body parts are out of proportion, they can be scaled to the correct size. Although the body model is constructed in a bottom-up manner, it does have one top-down characteristic in terms of colour, e.g. the hands has the same colour as the arms.

The body model can be simplified by modelling the clothing of the avatar as part of the body. The skeleton of the avatar is not modelled and complex limbs are further simplified by polygon reduction. E.g., the human hand is the most complicated limb on the human body, because it consists of many joints and folds. Instead of modelling each finger with joints, the fingers are modelled as a single polygon box (Figure 4.1).

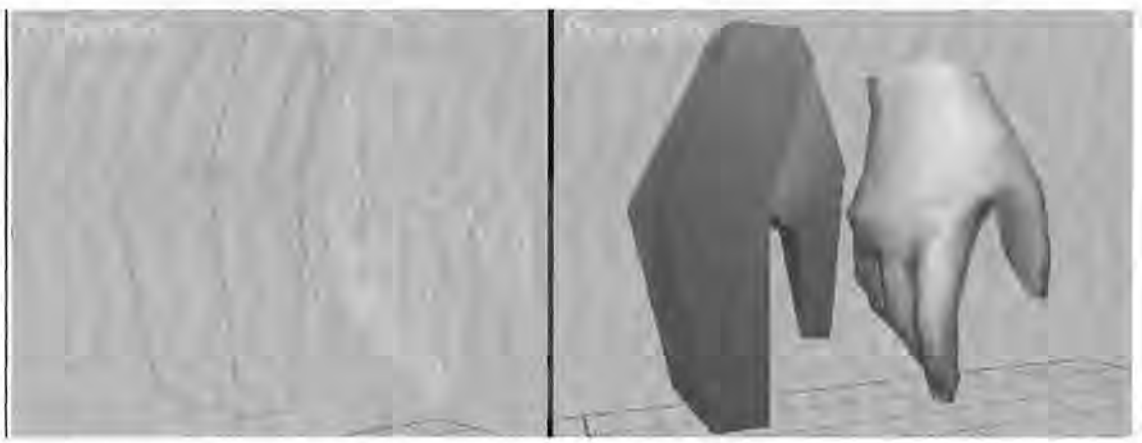


Figure 4.1 The complex hand model (Right) is re-modelled with fewer polygons, which result in a simplified hand model (Left).

In a real social environment, both genders exist and they interact with each other, therefore two different body models must be created, because the male figure is different in representation to the female figure. Instead of create a new set of body parts for the female model, the limbs that are similar in both models are reused (Hands, arms, neck, face and head). We created a dress model to represent the lower body of the female model, while the thighs and legs models are created for the male model. The reason for creating the dress in the female model is that the dress simplified the female model hierarchy and contains fewer polygons than the legs in the male model. In Figure

4.2 and Figure 4.3, the hierarchies of both models are represented in tree graphs; this will illustrate the connectivity between different body parts in both models.

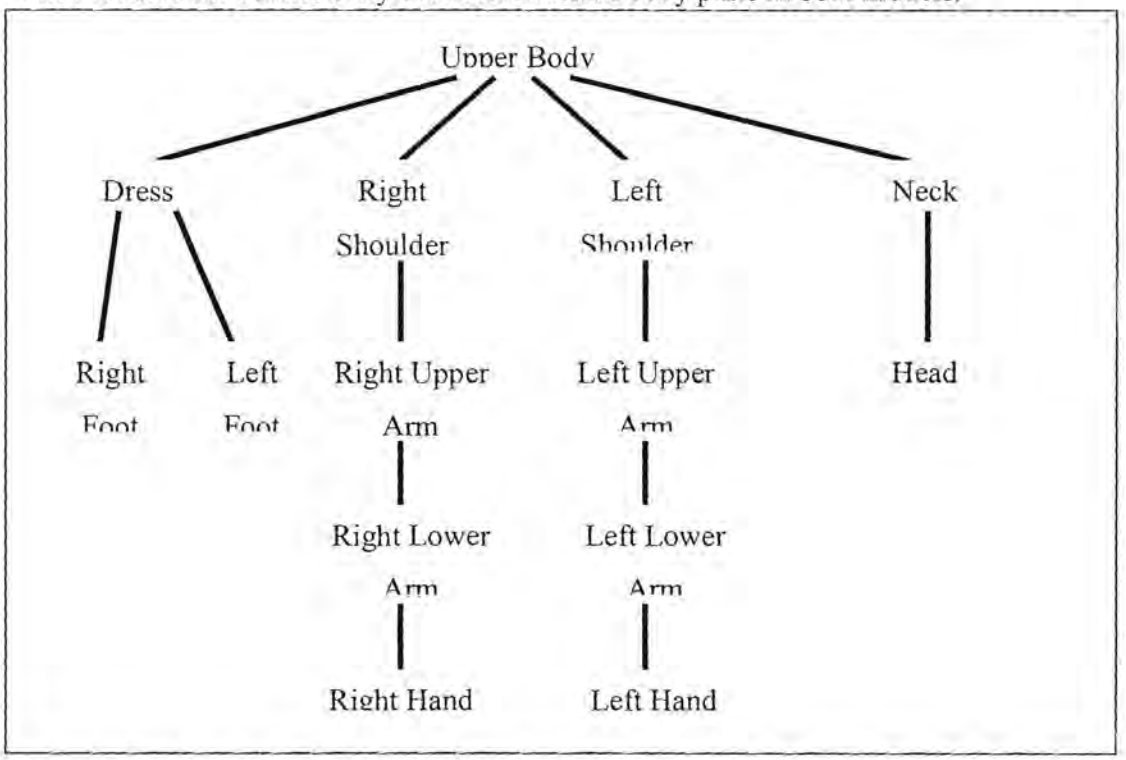


Figure 4.2 Tree hierarchy of the female model

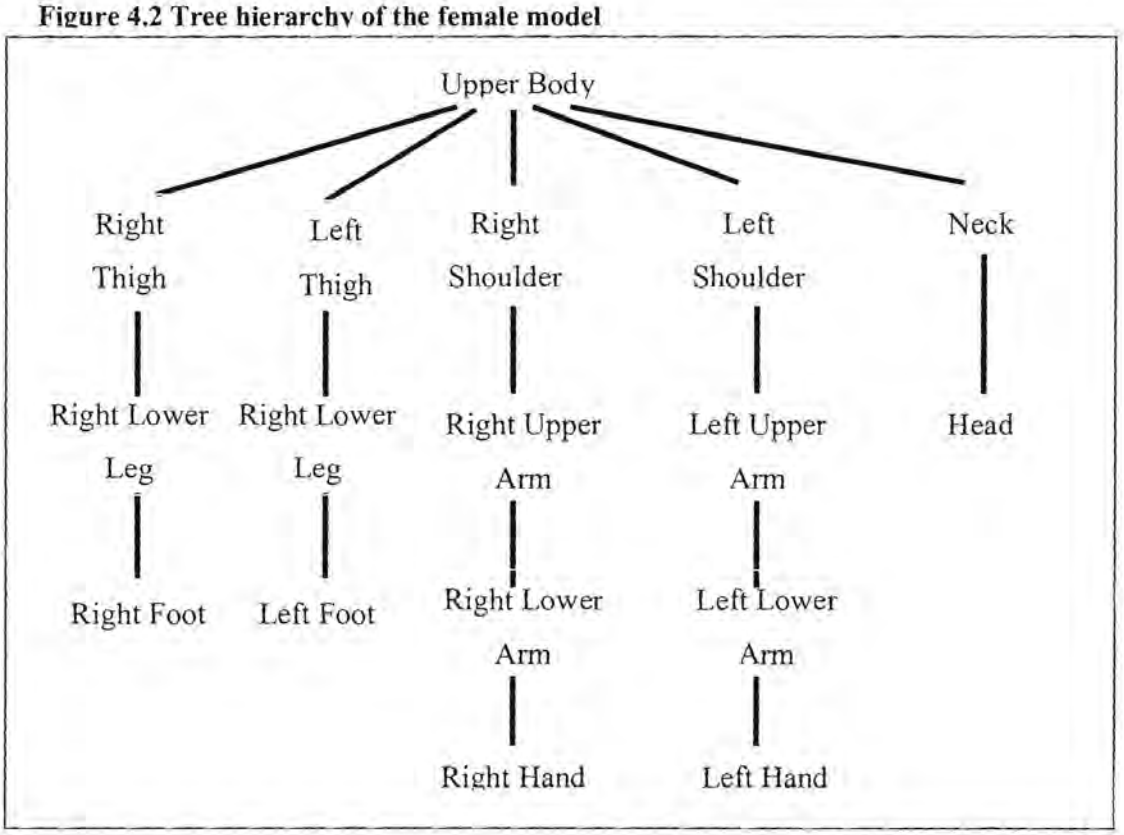


Figure 4.3 Tree hierarchy of the male model

The head model is divided into two parts the face model and the back of the head. The face mesh is reused from expressive texture approach, so those texture mapping, implementation and facial animation do not needed to be re-implemented. The back of the head is modelled such that it fitted behind the existing face mesh. This allows separate texture mapping for the face mesh and the back of the head model.

After the body structure is defined in details, the both body models are model in a 3D-modelling tool before it is implemented in the system. The 3D modelling tool created the prototypes of the body models, which aided the design of body animation for the body models and problems in the models structure are adjusted before implementing the models into the system. E.g. The sizes of different body parts are scaled into correct proportions relative to the body. The prototypes of both body models are shown in Figure 4.4 and Figure 4.5, different colours are allocated to each type of body part, which allows the body part to be identified more easily during body animation.

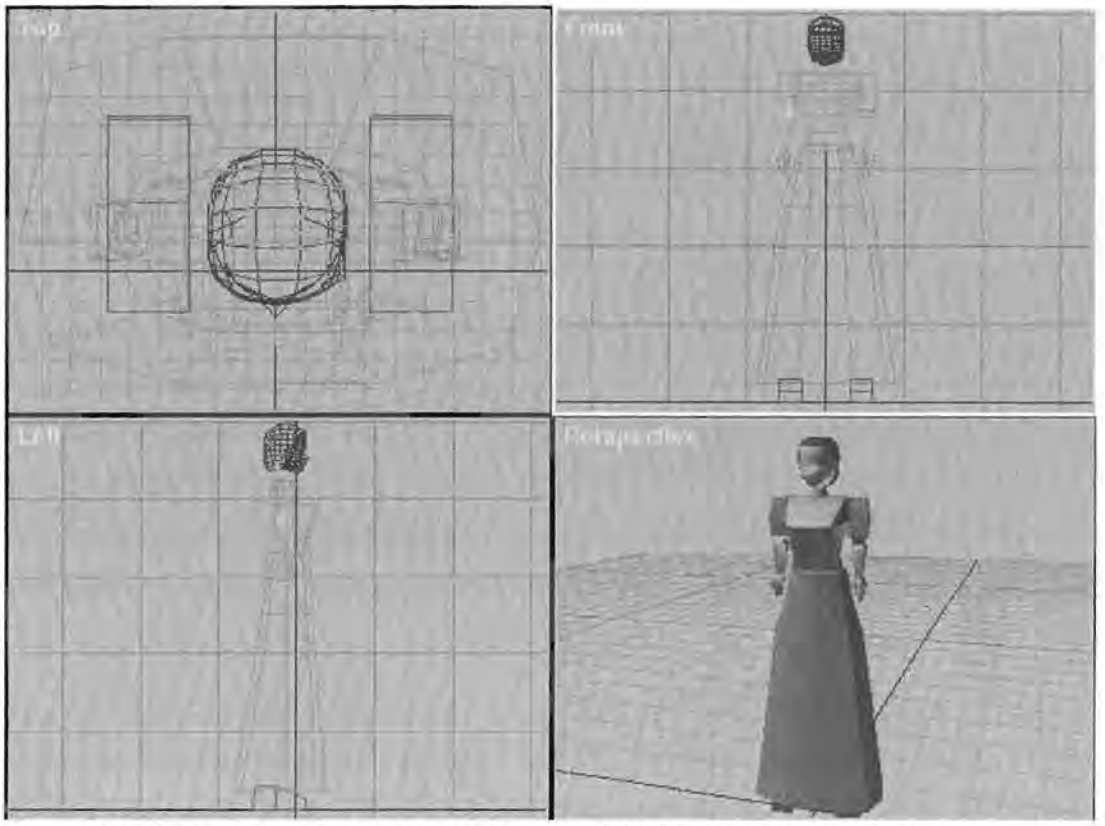


Figure 4.4 The female body model prototype modelled in 3D studio Max R2.5.

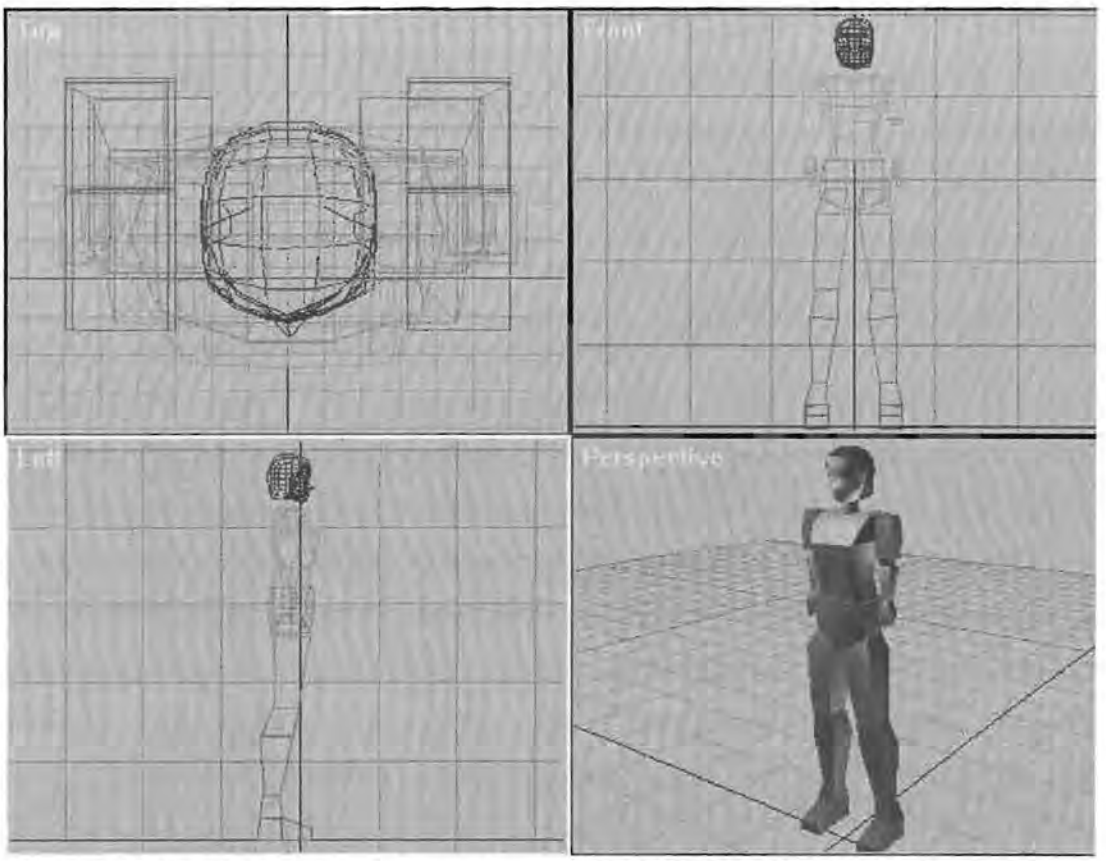


Figure 4.5 The male body model prototype modelled in 3D studio Max R2.5.

The body model prototype allow us to identified body parts that has the same shape and sizes, so during implementation these body parts can be implemented in a modular fashion.

4.1.2. Body Animation

In body animation, the position of all body joint angles must first be determined, and then the motion cues of the body motions must be classified.

Positioning Upper Body Joint Pivots

When animating the body motions, the movement of the avatar body must looked natural to the user, because each joint between the limbs in the human body has angle constraints. E.g. the elbow joint between the lower arm and upper arm of the human body cannot rotate an angle larger than 170 degrees, unless it is broken.

Therefore, the angle constraints and angle positioning in the joints are very important when animating body motions. The angle positioning is determined by the local co-ordinate system of each object model, in many commercial 3D modelling tools, the models created have an initial local co-ordinate system and the object pivot at centre of the object model. This means when the model animates object rotation, it rotates along its pivot axis in the local co-ordinate system centred at the object. When animating the avatar joint rotation the pivot of the body part that requires animating rotation is positioned between two body parts. E.g. When rotating the hand model, the pivot of the hand model is positioned between the hand and lower-arm models (Figure 4.6).

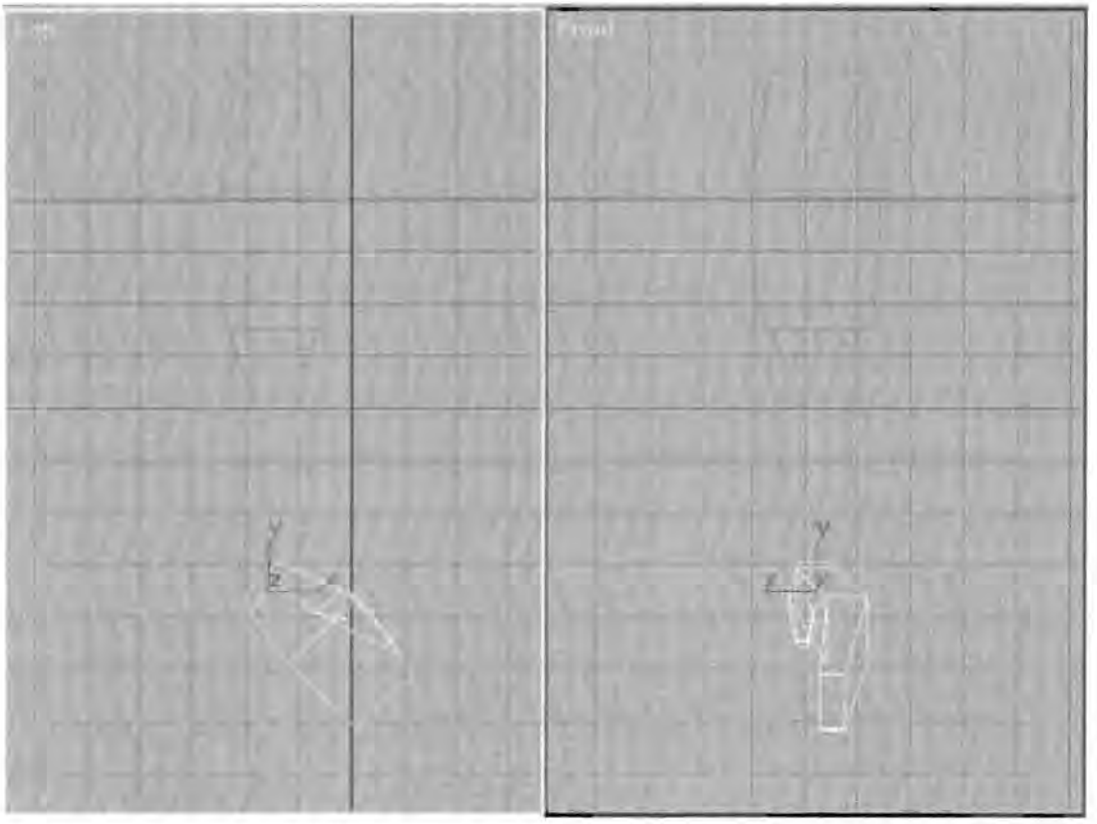


Figure 4.6 The hand model rotation simulating wrist movement.

The body model is structured in hierarchy, this means a consistent behaviour exists between the different parts of the body. The body parts remains connected after rotation and the rotation of the body part at the higher level of the hierarchy will result in the rotation of the lower level object. E.g. If the arm moved, the hand moved with the arm (Figure 4.7), but it the hand moved, the arm remains stilled.

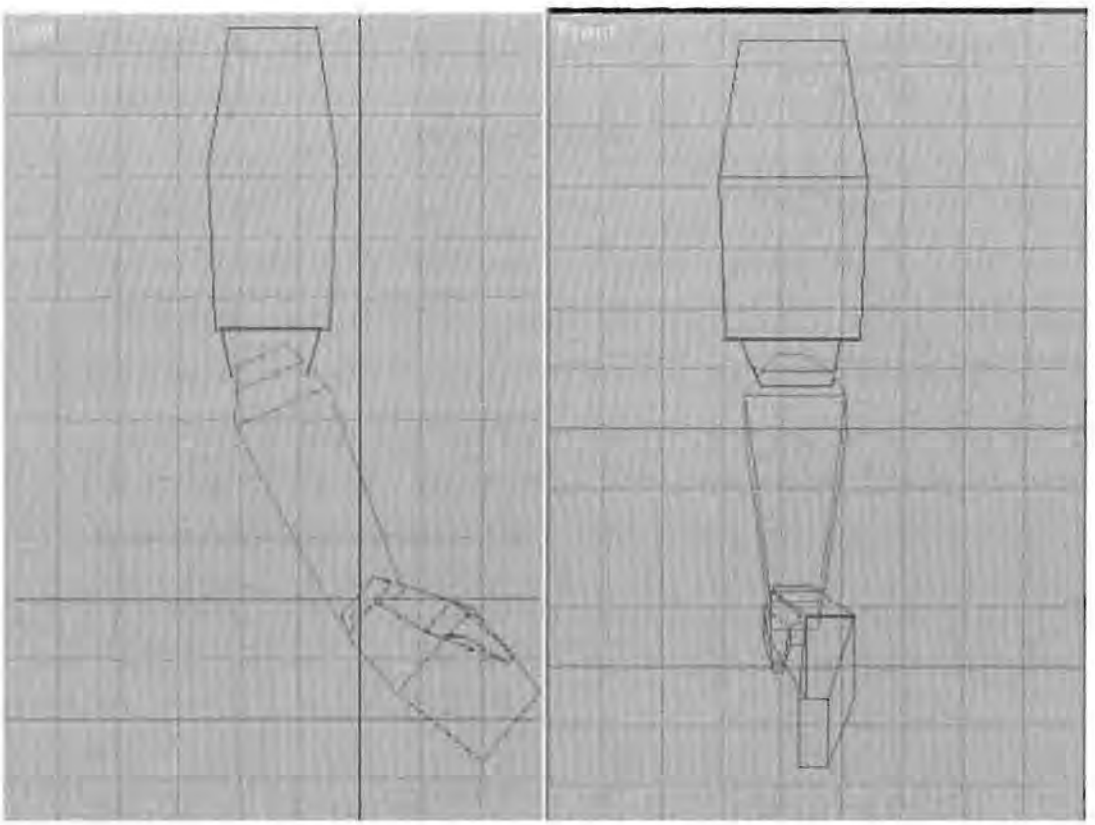


Figure 4.7 The rotation of the arm around its pivot caused the hand moved with the arm, while remain connected with the arm.

Therefore, the pivots of all upper body parts (head, upper arms, lower arms and hand) are adjusted for upper body animation (Figure 4.8).

Body motion cues

After the pivots are defined in the male and female body models, the body motion cues can be classified for each body motion associated with an emotion and action in the environment. During communication and interaction, each individual has unique ways of expressing their emotions using body motions and using gestures. Therefore, the body motion cues classified for each emotion in Table 4.1 is only a basic generalisation. E.g. When Japanese people communicate and interact with each other, they tend to use less or no gestures and body motions, because some body movement are seen as hostile from the Japanese people's point of view. American people on the other hand use gesture and body motions more frequently during communication and interaction.

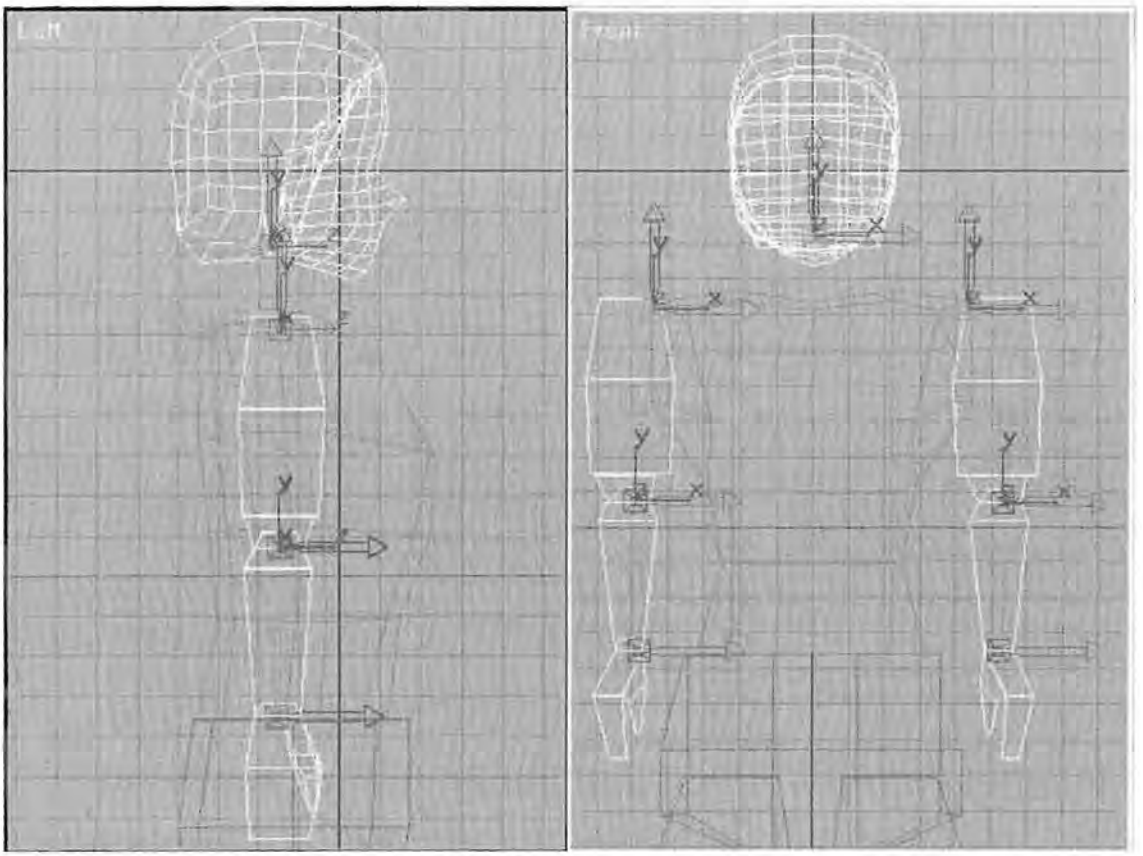


Figure 4.8 The adjusted pivots of all upper body parts for upper body animation.

Emotion	Body Motion Cue
Angry	Head face in the other direction, eyes staring at the respondent, arms crossed.
Melancholy	Head looked down, the hand is placed in front of the face.
Surprise	Head tilted slightly, hand placed over the chest.
Disgust	Head looked slightly away from the respondent, the hand raised in front of the avatar in a "rejection position".
Fear	Head tilted slightly, hand placed against the chest.
Teasing	Head face slightly away from the respondent with eyes focus on the respondent, one arm arched at the waist
Happy	No body movement
Cunning	Head looked in the other direction, arms crossed

Table 4.1 The Body Motion Cues classification for each emotion

The body motions are applied in the same manner for all avatars under a specific emotion and retargetting body motions is not required, because all the avatars are equal in size, this avoids the feeling of superiority from the avatars. From the coven experiment, some users think an avatar larger than a small avatar is more superior. This causes the interaction between avatars that differ in size inflexible [65], because users feel that this is similar to a situation where a subordinate communicates with ones superior.

4.2. Interaction

Before interaction can be determined, we have to create a virtual environment that demonstrates a simple application for the avatars and their interaction in the environment. This allows the users to understand the behaviour of people under a simple, closed social environment without considering the external factors. When avatars interacted in the virtual environment, their actions are dependent on the possible interactions in the environment. Therefore, we decided to create a virtual pub environment for the avatars, because the people communicate and interact with each other in the pub, so the avatars can represent people interacting and communicating in the pub.

In the real environment, interactions are separated into physical interaction and social interactions. Physical interaction, is when people interacting with objects in the environment and Social interaction, is when people interact with other people. Similarly, the avatar interactions in the virtual environment can also be classified into physical and social interactions.

The possible interactions for avatars in the virtual pub environment:

- Drinking (Physical)
- Watch the television in the pub (Physical)
- Change television channel (Physical)
- Talk to other avatars (Social)
- Play music from the juke box (Physical)
- Change music (Physical)

The classification of interactions made it possible to identified which interaction of the avatar is dependent on the objects or other avatars in the virtual environment. E.g. if the avatar talk to other avatars, the interaction is only possible if there are other avatars close by in the virtual environment.

Interaction Restrictions

In the real environment, people cannot interact with all objects and other people, if the object or the other person is not reachable by the individual. Therefore, constraints must be applied to the avatar interactions, so that interaction in the virtual environment is similar to those in the real world environment for the users. Constraints can be applied to interactions defined in the virtual pub environment, so that specific interaction is only possible by the identity of the avatar.

The physical interactions are restricted by object availability, if the object in the environment has limited access then when the object is fully accessed, other avatars cannot interact with the object. E.g. the jukebox can only play one music tune at a time. Therefore, only one avatar at a time can play music from the jukebox. Apart from limited access, the object can also be accessible only by specific avatars. E.g. In the pub, the television is own by the Barman, so only the barman can change the television channel.

The physical interaction restrictions in the virtual pub environment are:

- The television channel is changeable only by the Barman avatar
- The jukebox can be used by any avatar except the Barman avatar, because the barman has to stand at the counter to serve customers.
- Any avatar can change the music on the jukebox
- Any avatar can perform drinking except the Barman
- All avatars can watch television except the Barman

Social interaction is restricted by avatar availability, if there is no avatars surround the avatar wishing to talk then the avatar cannot perform talking with other avatars.

Once all possible interactions for the avatars and constraints are determined, each interaction can be classified into motion cues, which allow us to animate the avatar's body motion during interaction (Figure 4.2).

Interaction	Motion Cue
Drinking	The hand that is holding a drink moved close to the mouth, the hand must return to neutral position after drinking.
Watch the television in the pub	The avatar turn to face the television and look at television with arm cross
Change TV channel	Press the Remote control
Talk to other avatars	Turn to face other surrounding avatar
Play music from the juke box	Move in front of the juke box and switch on juke box
Change music	Move in front of the juke box and switch on juke box

Table 4.2 Interaction Motion Cues

4.3. State Machine

Although the interactions and interaction constraints are defined, but the effects of avatar interactions affecting other avatars and the virtual environment are still not established. In the real world, each action performed by an individual always influenced the emotions of others surround this individual and people interacted with objects for satisfying their goals. If people are prevented or blocked from achieving their goals then people will become frustrated which lead to anger or despair. E.g. If an individual want to used the public phone, but someone is stand blocking the way and not willing to give way, then the individual can become angry.

Therefore, the consequence results of interaction by each avatar will change the emotional state of other avatars and state of the objects in the virtual environment. These emotional and state changes in avatars and objects must be determined and defined in the system as output of the avatar interaction.

The emotional state of an avatar can change because:

- The avatar wants to talk to other avatars, but no other avatars are nearby, so the avatar's emotion changes to melancholy.
- The avatar wants to talk to other avatars, there is other avatar nearby, so the avatar's emotion changed to happy.
- The avatar is watching television, but the Barman changes the channel, therefore the avatar becomes angry with the Barman.
- The avatar is hearing music, and no one is disturbing it, therefore the avatar becomes happy.
- The avatar is hearing music, but another avatar try to change the music tune to another, therefore the avatar is disgust at the other avatar.
- The avatar is shouted by another avatar, therefore the avatar's emotion changes to melancholy.
- The Barman avatar is shouted by another avatar, therefore the Barman avatar's emotion changes to melancholy.
- Result from drinking too much the avatar becomes drunk, which made the avatar looks cunning.

In order to simplify the state changes in objects, only the television and the jukebox in the pub has changeable states with two channels or two music tunes. The state of the television in the virtual pub environment can be showing channel1 or channel2.
The state of the jukebox in the virtual pub environment can be off or on, when it is on it can be playing music tune1 or music tune2.

After all the state changes and interactions are defined, the transitions between interactions and state changes, and interaction processes are implemented into the system using a state machine. A state machine is a finite automation process, which implement the states that an object may be in, as well as the transitions between the states. The state machine is deterministic and the resulting state is unique. The avatars' interactions are synchronised by the state machine, because when an avatar perform an interaction, the interaction is determine in the state machine to identified which virtual

environment object state is changed. These object state changes are stored in the system process and the result of the avatar's interaction is a change in the emotional state of the avatar as the avatar achieved or failed to reach its goal. An avatar can affect other avatar's emotional state by modifying the object's state which cause the other avatar fail to achieve its goals or direct influence (shouting at the other avatar) (Figure 4.9).

The state machine is represented by a state diagram, so that the interactions and transitions in the state machine is scenario tested and corrected before implementation. A state diagram is a model, which describe the states that an object may be in and the transitions between states.

The initial state in the state diagram is the initial Interaction State of the avatar before interacting with the environment and other avatars. The final in the state diagram is the results of the avatar interaction after the avatar interacting with the environment and other avatars. The emotion of the avatar changes or remain the same before interaction.

The rounded rectangle in the state diagram represents the interaction of the avatar, the boxed rectangle represents the objects in the environment and the arrows are representing the events/transitions between the interactions.

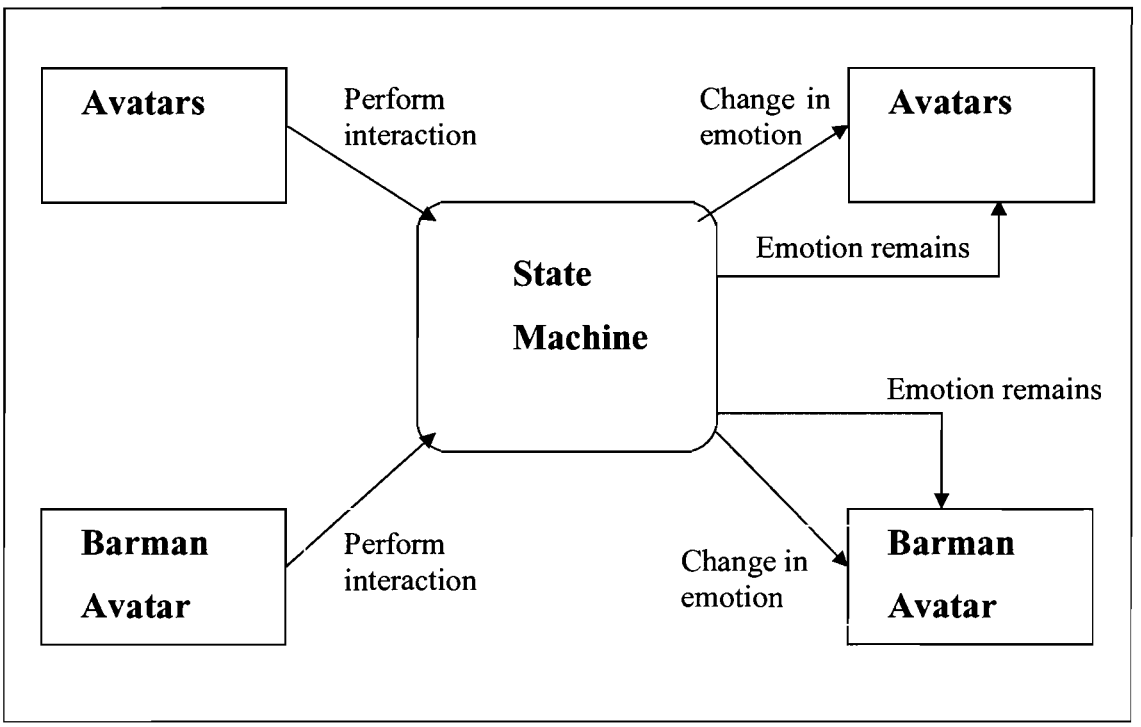


Figure 4.9 The use-case diagram of the state machine.

In Figure 4.10 and Figure 4.11, the state machine is subdivided into different interaction processes, which is easier to understand the different interactions and their connection.

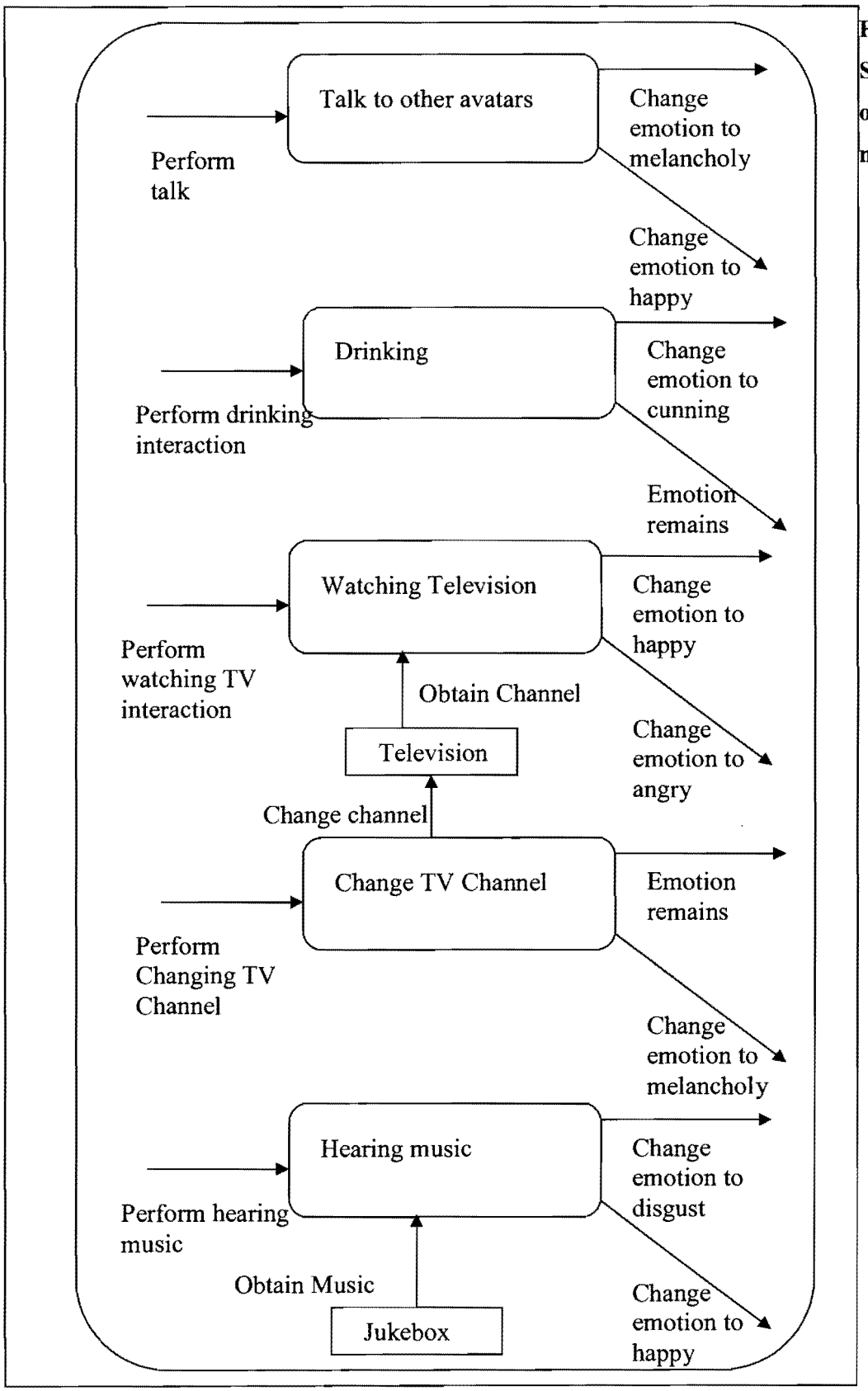


Figure 4.10 State diagram of the State machine.

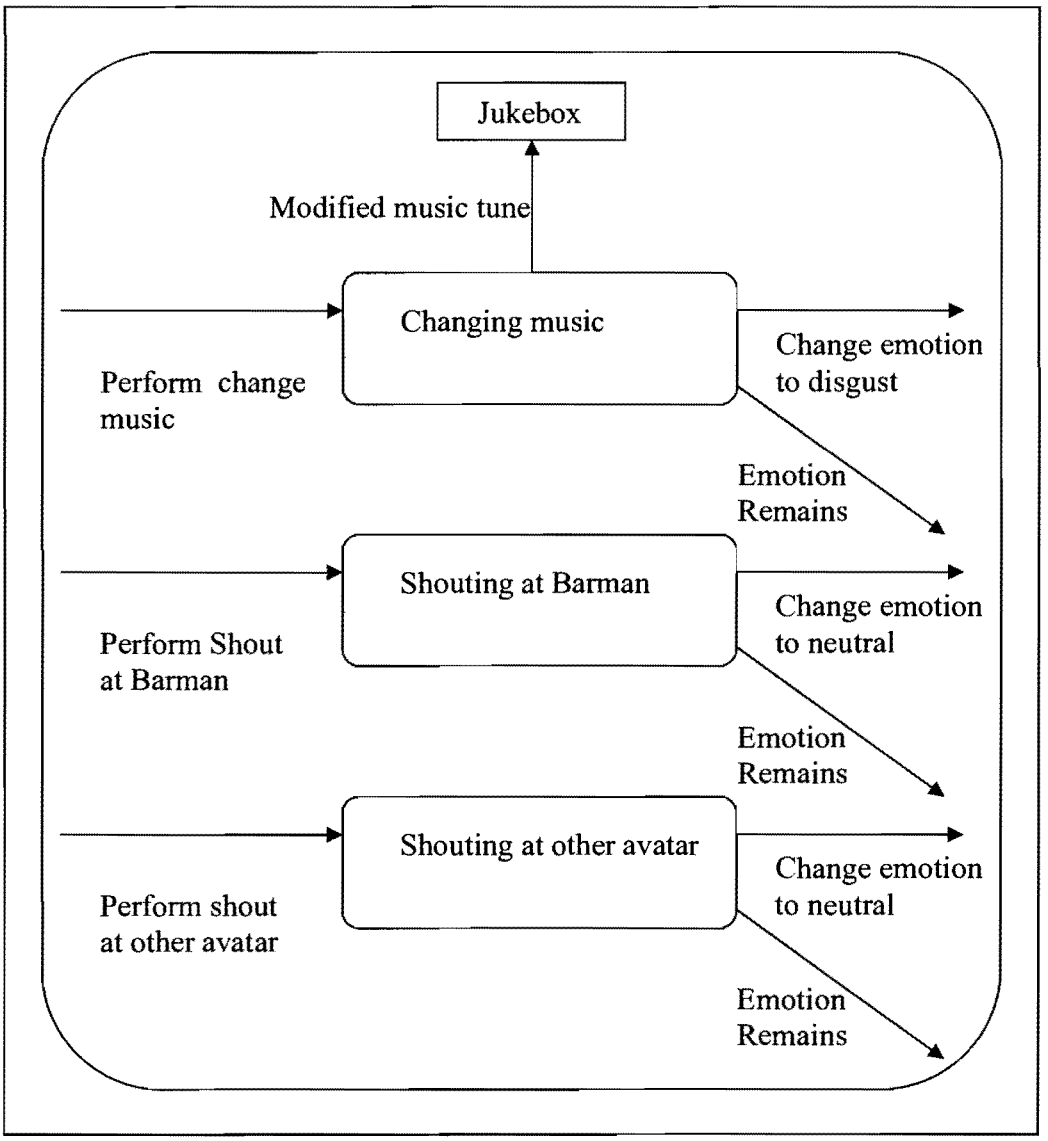


Figure 4.11 (Continue) The state diagram of the state machine.

In figure 4.11, two new interaction processes is added, "shouting at Barman" and "shouting at other avatar". The avatar performed these interaction processes automatically, if the "Hearing music" and "Watching television" interaction process is interrupt by the barman avatar or other avatars. In the real world, people will become angry if another person changed the television while they are watching, and people usually scream at the person who changed the channel to switch back to the previous channel. Therefore, the avatar will continually perform the "shouting at Barman" or

"shouting at other avatar" interaction processes until the avatar that changed the television or jukebox object's state back to previous state. Once the environment objects is returned to previous state, the avatar will continue the interaction process before the interruption by other avatars.

The talking and drinking interactions contained complex internal states that differ to other interaction process. These interaction processes are represented by the internal state diagrams in Figure 4.12 and Figure 4.13, which gives the in-depth view of these interaction processes. In the drinking interaction process, a drinking counter is allocated to each avatar, this counter calculated the number of drinks that each avatar had, if the avatar drink exceed the limit then it is indicated as been drunk.

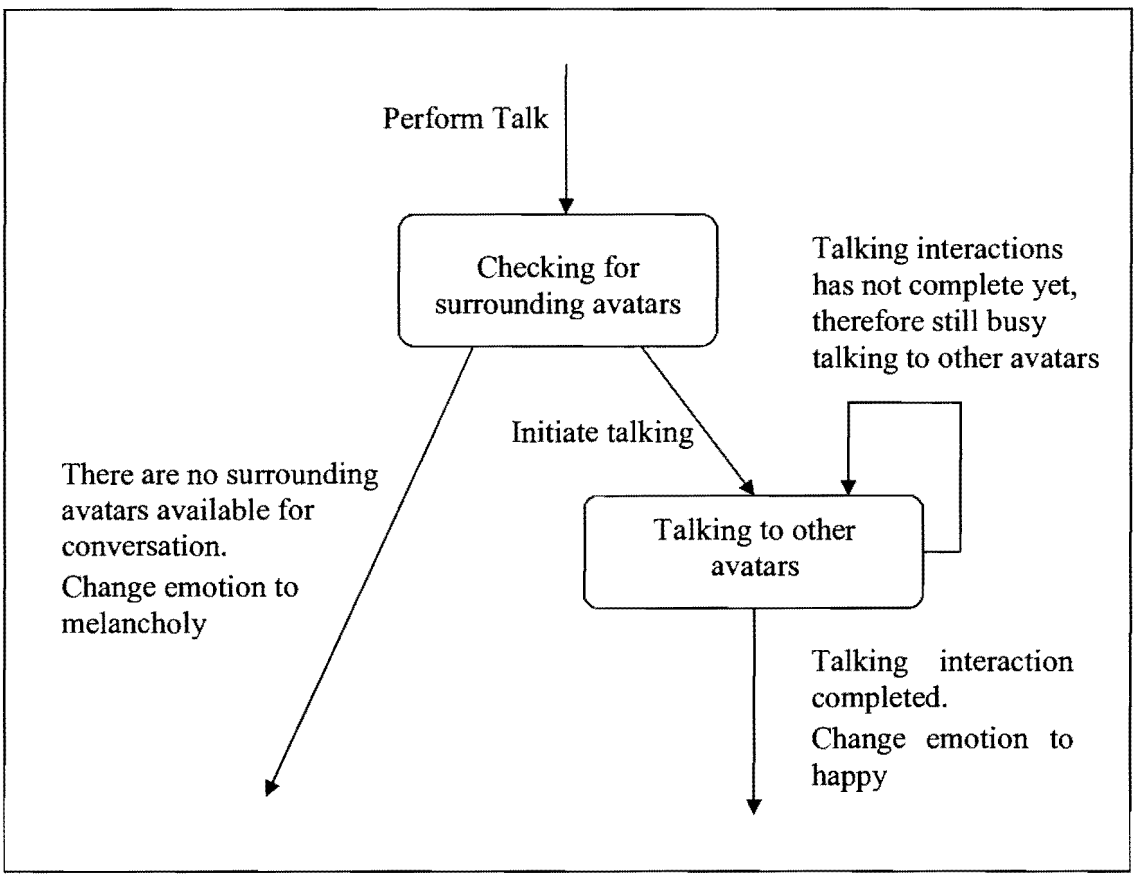


Figure 4.12 Internal state diagram of the talking interaction process.

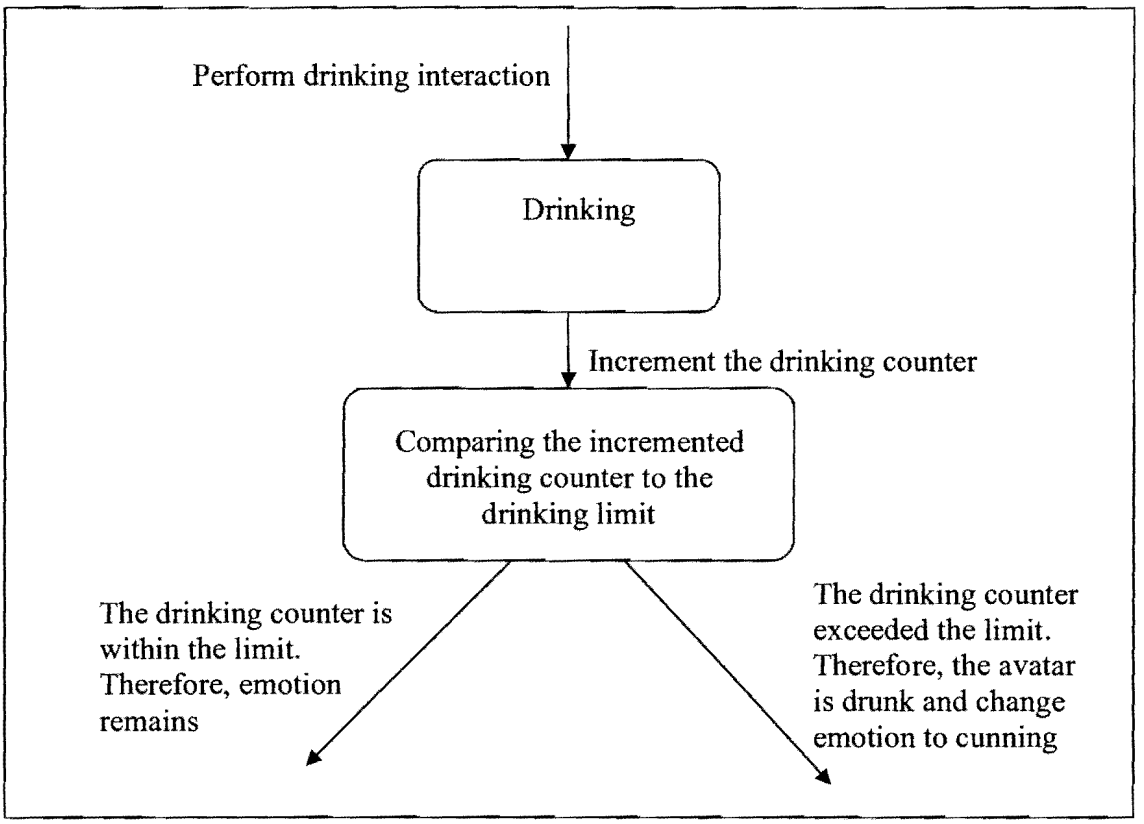


Figure 4.13 Internal state diagram of the drinking interaction process.

The state machine can be implemented according to these state diagrams, but identity checking is required in the actual implementation of the system, before the avatars entering the state machine and perform interactions. This identity checking process determine if an avatar is a customer or the barman and restricted the avatar from performing interactions that it is not supposed to perform.

4.4. Sounds

When avatars interact and communicate in the virtual environment, if sounds are not available, the interaction in the virtual environment will become dull and communication without words is not amusing. In the virtual pub environment, short speech sounds are linked to avatar interactions, so when the avatar perform an interaction, the user can hear the comment made by the avatar. E.g. When the avatar has no one to talk to, the avatar's expression becomes melancholy and it makes a remark

"Boring", then user understand that the avatar is bored because there is no avatars to communicate with it.

Two sets of speech sounds must be provided, because we have to accommodate a set of speech sounds for the male avatar and the other set for the female avatar. The user will found it weird, if the female avatar has a male voice or vice versa. The limitation of this is that all the male or female avatars will have the same voice, because the speech sounds and the environment sounds together required large amount of system resources to loaded. Therefore, the system can only provided one set of speech sounds for each gender. Music tune sounds are also provided in the virtual pub environment when the avatars interacted with the jukebox. This is because the user will not know if the avatar is just standing in front of the jukebox or interacting with it, if music tunes is not played during the interaction processes. The speech and music sounds added more information to the visual interaction made by the avatar in the virtual environment and made interaction more amusing.

4.5. Summary

In order to simulate the physical and social interaction in the virtual environment similar to the real world and study the interactions of the avatars, the avatar body structure must be modelled such that animating interactions is possible. When modelling the avatar, gender must also be taken into account, because both genders exist in the real environment and they differ in appearance. Since the male and female models has equal size, and similar upper body structure, the body motion of the male avatar can be applied to the female avatar or vice versa without retargetting. By presenting interactions of the avatar in state diagram, the transitions between interactions are easier to understand and error appeared in the transition between interactions are detected and corrected before implementation.

The next chapter will discuss the actual design and implementation of the virtual pub application and the texture mapping avatar faces tool, which takes the theoretical approaches and prototypes into implementation.

Chapter 5

Design and Implementation

This Chapter discusses the implementation of the Expressive Texture theoretical approach described in chapter 3. An avatar creation tool and an interactive virtual pub environment for these avatars are implemented. The interactive virtual environment simulates a real-world social environment, which allows the user to observe the social interaction between avatars, and the emotional changes of avatars caused by the social interaction. Section 5.1 discusses the implementation and section 5.2 discusses the creation process of the virtual pub environment. Section 5.3 discusses the modification of the expressive texture tool into the avatar creation tool. Later in Section 5.4, the implementation of the social interaction application will be discussed and determined how the avatar creation tool was linked to it. The chapter is summarised in section 5.5.

5.1. Avatar Generation

An avatar can be generated by importing the prototyped-avatar models from the modelling tools into the application or obtained the geometric data of the prototypes and implement the avatar based on these data. The application was developed using C++ and OpenGL. Because the existing importing plug-ins for does not support complex texture mapping on the models, the avatar models were not importing.

5.1.1. Body Creation

The avatar data model (the position of all vertices) was created in a 3D modelling application. Based on the data of the prototyped avatar, the avatar model was implemented in OpenGL by constructing each body part as a polygon object. Each body part of the avatar model was created using the vertex data of the corresponding body part of the prototyped model. Therefore, the avatar model in OpenGL had the same scale as the prototyped model and the positioning of each body part remained in the correct position as the body parts in the prototyped model. Each body part can be coloured individually. However, the hierarchy structure of the body and the adjusted pivot positions at the joints of the upper limbs were not preserved in the vertex data. Therefore, the body parts are regenerated in a bottom-up manner in OpenGL. The pivot points at the joints between the upper body limbs are also adjusted.

In OpenGL, the hierarchical body structure was created using the Matrix Transformation Stack, the objects were pushed into the matrix stack using a method "glPushMatrix()" and they were popped by calling the method "glPopMatrix()". The Higher level objects (e.g. body) were pushed into the stack before the lower level objects (e.g. arms, legs, etc.) were pushed into the Matrix stack. Because the Matrix stack has LIFO (Last in, first out) properties, the lower level objects were drawn first. Any transformation and rotation applied on the lower level objects will not affected the higher level objects, because the transformation and rotation occurred before the higher level objects were drawn. However, the rotation of the higher-level object causes the lower level objects to rotate as the higher level rotation is applied after the lower level object was drawn.

Apart from ensuring that the correct rotation and transformations are applied to the specific levels of objects, the matrix transformation stack also preserved the root co-ordinate position.

Figure 5.1, shows a simple example of the code of the method drawing the female avatar body using the matrix transformation stack. The structure was based on the tree representation of the female avatar model in the previous chapter.

The male avatar's drawing method is coded similar to the female avatar, but the difference is that the male avatar has a hierarchy coding with "draw_RightThigh()", "draw_RightLowerLeg()", "draw_LeftThigh()" and "draw_LeftLowerLeg()" instead of the "draw_Dress(lower)" method.

```
Public void draw_femaleAvatar(){

    glPushMatrix();
        draw_neck();
        draw_Body(shtGender, top);
        glPushMatrix();
            draw_face(avaNum);
            draw_backHead(avaNum);
        glPopMatrix();
        glPushMatrix();
            draw_RightShoulder(top);
            draw_RightUpperArm();
            glPushMatrix();
                draw_RightLowerArm();
                glPushMatrix();
                    draw_RightHand();
                glPopMatrix();
            glPopMatrix();
        glPopMatrix();
        glPushMatrix();
            draw_LeftShoulder(top);
            draw_LeftUpperArm();
            glPushMatrix();
                    draw_LeftLowerArm();
                    glPushMatrix();
                        draw_LeftHand();
                    glPopMatrix();
                glPopMatrix();
            glPopMatrix();
```

```
                    glPushMartix();
                        draw_Dress(lower);
                        glPushMartix();
                            draw_RightFoot(shoe);
                            draw_LeftFoot(shoe);
                        glPopMatrix();
                    glPopMatrix();


                glPopMatrix();
            glPopMatrix();
}
```

Figure 5.1 A simple method using matrix transformation stack to perform the drawing of the female avatar.

5.1.2. Animating Body motion

When each body part was created in OpenGL, the origin of the local co-ordinate system is initially positioned at the centre of the body part object. When adjusting the pivot of the body limb for animating body motion correctly, appropriate modelling transformations are applied to the body limb that is going to bend at a specific angle. E.g. When animating the lower arm bending, 3 matrix is used to accumulated the translation, rotation and scale of the avatar body parts (body, upper arm and lower arm). The lower arm is animated based on the origin of the avatar, it is translated to the origin of the avatar. Then rotation was applied to the lower arm, which it rotated around the origin of the avatar and rotation is applied to the lower arm, before translated it back to the position connected to the upper arm and draws the lower arm object (Figure 5.2). These translations applied during the animation of lower arm model are apply quickly between frames at run-time before drawing the lower arm, such that the user will not noticed any flickering, when animating the lower arm rotation. This modelling translation simulated the bending of the lower arm and similarly these translations can be applied to the other body limbs to simulated limb movement.

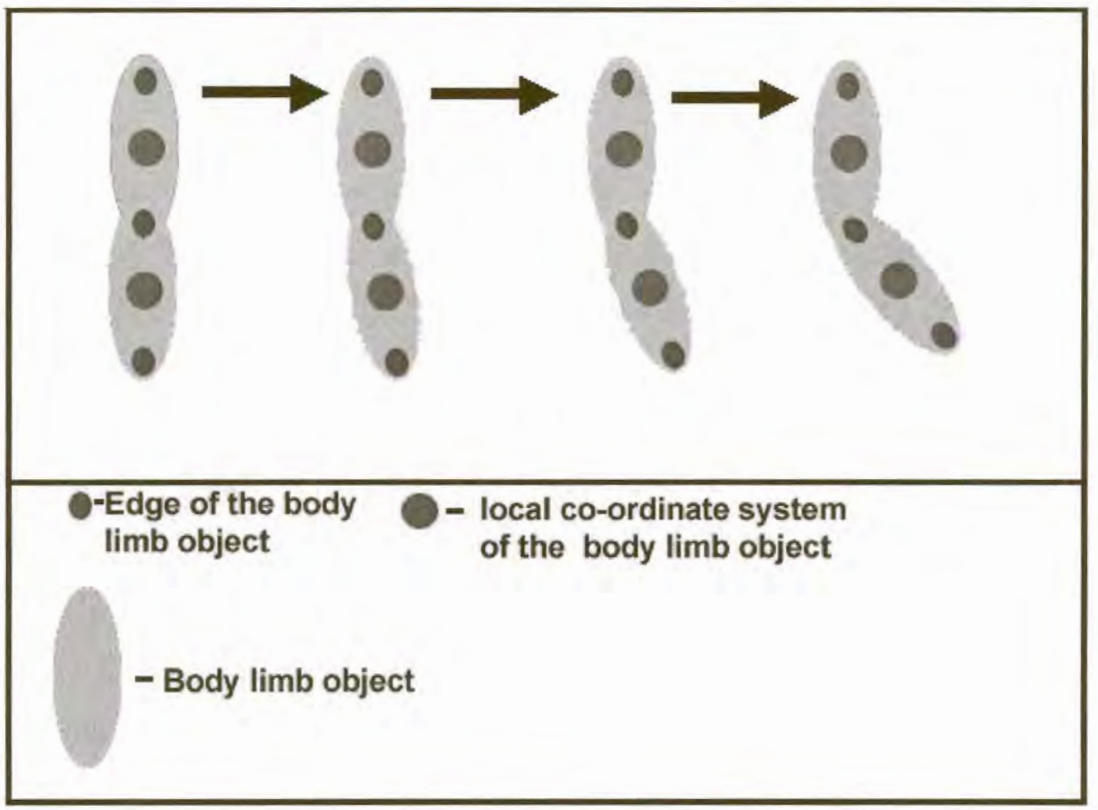


Figure 5.2 The side view of the modelling transformation sequences for a bending body limb.

Once all the modelling transformations are determined for every movable body limb, upper body motion can be animated on the avatar based on the motion cues of the possible body motions of the avatar defined in the previous chapter.

The data containing the amount of upper body movement for each emotion and action are determined and stored in the system. These data are used as the targeted limb motion when calculating body animation, which the limb object incremented its motion values (Pivot angle of the specific limb) from the current limb position to reach the target position during animation. E.g. When animating the lower arm bending, the pivot angle between the lower arm and the upper arm must be increased gradually, starting from zero degrees angle when the lower arm is in a resting position.

The motion value is continuously increased or decreased gradually during animation. It is implemented in a method or function that is updated in every frame. The function call is, glutIdleFunc() in the OpenGL Utility Toolkit. After the calculations and motion

values are coded in the idle function, the body motion of each avatar is animated when an action or emotional event triggers the body animation (Figure5.3, Figure 5.4, and Figure 5.5).

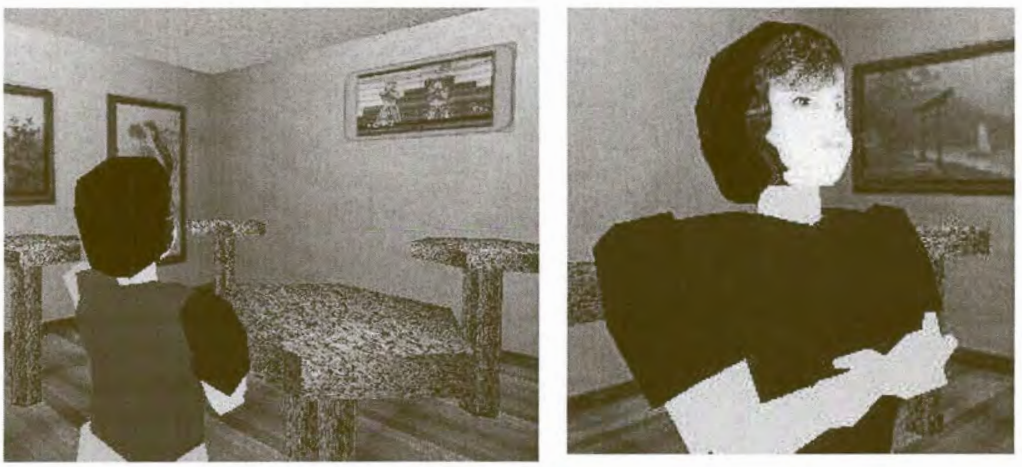


Figure 5.3 The avatar is watching television.

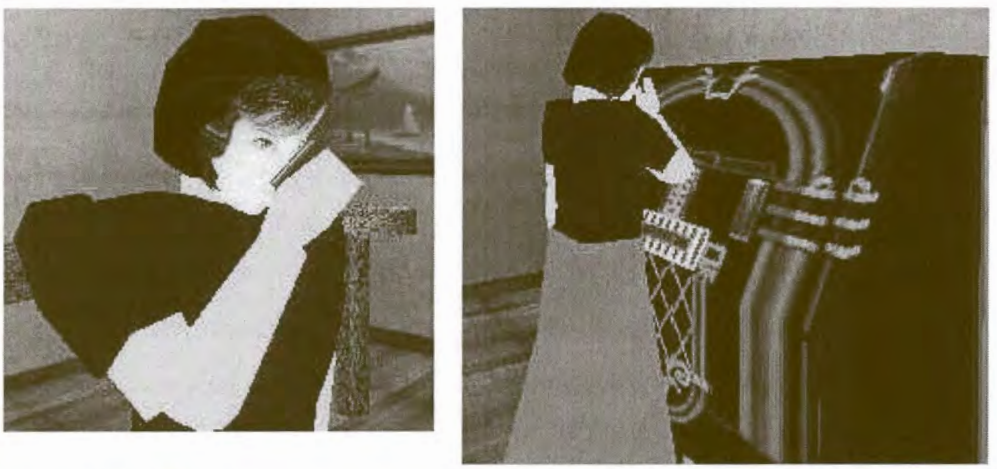


Figure 5.4 The left image shows the avatar drinking a can of beer and the right image the avatar is listening to the music from the jukebox.

Figure 5.5 The avatars talking to each other.

5.2. Virtual Environment Generation

The virtual environment of the avatar must be designed and generated to support the interaction of the avatars. The objects in this environment are designed and drawn in the correct proportions corresponded to the real world objects, so interactions between the avatar and objects in the virtual environment appear natural and fairly realistic from the user's point of view. Similar to the avatar, the objects in the virtual environment are modelled with low polygon number and textured with surface images of real objects.

5.2.1. Virtual Pub Environment and objects creation

The virtual environment and objects can be implemented into the system without prototyping, because the geometry of the virtual environment and objects are simple and do not have the complex hierarchy structure. In order to simplify the system, the virtual environment and objects cannot perform animation and there are no external forces (e.g. virtual wind) affecting the objects in the virtual pub environment, because it

is a closed environment. Therefore, the model of the virtual pub and the objects in the virtual environment are modelled as simple geometric shapes.

5.2.1.1. Virtual Pub Environment

The virtual pub environment is generated as a room-like environment with six rectangular faces that represented the walls, ceiling and floor of the pub. A large rectangular opening is created in one of the wall to represent a door opening. The door was created with the same size as this opening in the wall, which can translated horizontally simulating a sliding door.

In the real environment, people, objects and wall structures were solid and had properties occupied space, which prevented people walking through it. However, these properties are not represented in the objects and wall structures in the virtual environment. Therefore, some form of collusion detection method was required to prevent this problem form appearing during the avatar movement.

Instead of computing the object surface for collusion detection, a simple grid approach was implemented for collusion detection. The floor of the virtual pub environment was sub-divided into a grid structure with square cells that had the width of the avatar (Figure 5.6). The floor of the virtual pub environment is similar to a room plan that contains the position of furniture, the position of the virtual objects and avatars are marked on the cells in the grid. When the avatar walks in the virtual pub environment, it can detected if other avatars or a virtual object occupied the adjacent cells and try to move past that cell without walking through the virtual object or other avatar in that cell.

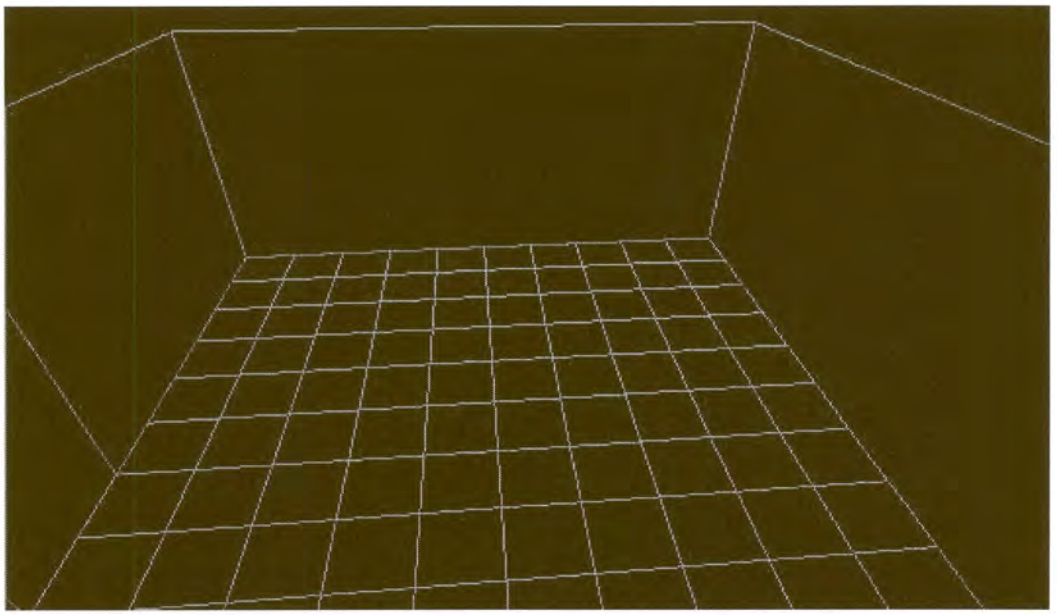


Figure 5.6 The wire-frame view of the floor in the virtual pub environment, which is divided into a grid to detected space availability.

5.2.1.2. Virtual objects

The possible objects in the virtual pub environment are tables, television, and jukebox, bar table, lights and cupboard (Figure 5.7). Instead of drawing all the objects in the virtual environment using a single method and each object were drawn by calling the corresponding method that drawn the object. The advantage of drawing each object using a single method is that the object can be repeated by calling the method again with another transformation that would position or scale the object in a new place.

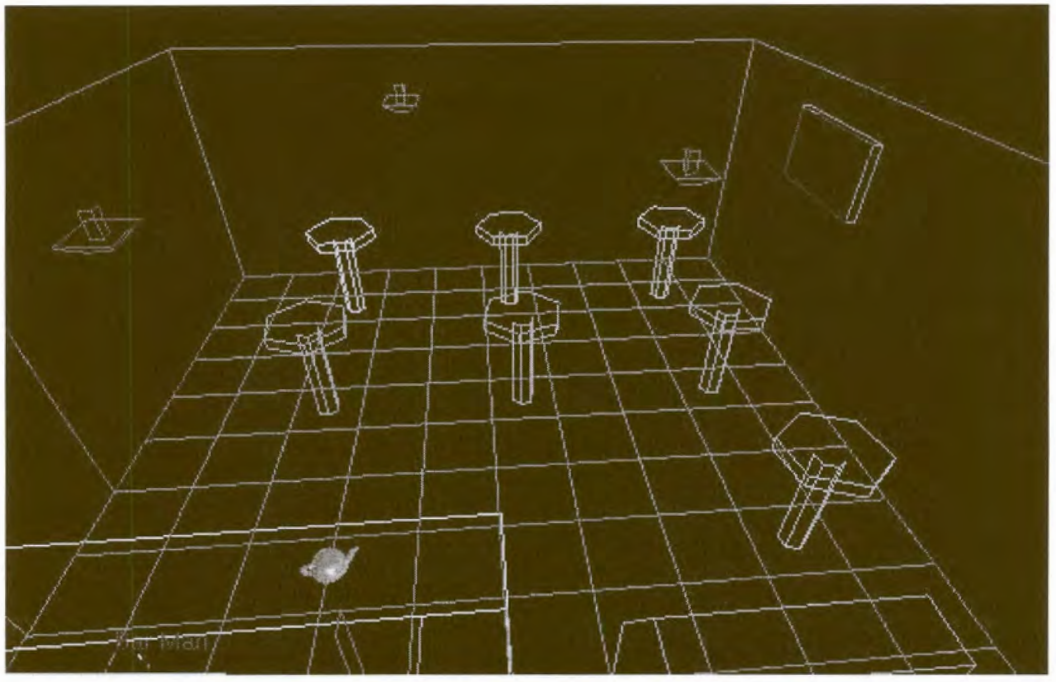


Figure 5.7 The wire-frame view of the virtual pub environment with all the virtual objects.

5.2.2. Texture mapping virtual environment and objects

To make virtual objects more realistic details are added by texture mapping images of real objects or the texture of real object onto the virtual object.

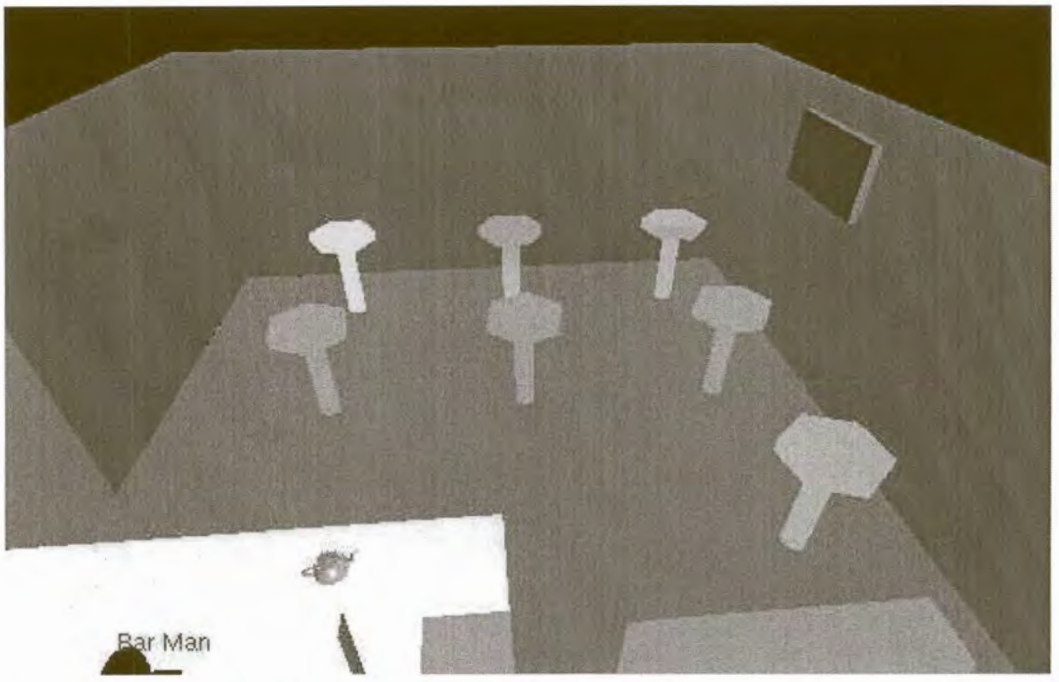


Figure 5.8 The virtual pub environment without textures.

To create a fairly realistic environment a high number of textures needs to be loaded, requiring large amounts of system resources. Therefore, the textures of all the virtual objects and the pub environment are combined into a single high resolution texture and complex texture mapping is applied to all the virtual objects and the virtual environment using this single texture (Figure 5.9). The combined texture is divided into ratio portion, and the texture occupied a specific portion depending on the size of the corresponding virtual object. The reason for this type of texture allocation is that the user will noticed the blur effects and stretching pixels on the object caused by mapping a large virtual object with a small texture. Alternatively, a small texture can be applied to a large surface by tilling the texture. This usually generates a surface with a continuous pattern and the texture is generated seamlessly (without a distinct edge), because the textures can be joined from any direction without any distortion between the repeated textures.

Figure 5.9 A single texture containing all the textures of the virtual pub environment and the virtual objects.

Figure 5.10 shows the result of texture mapping all the virtual objects and the pub environment. After the virtual pub environment is completed, functionality, control and logic are added to the system enabling the user to manipulate the application of the avatars in this virtual pub environment.

Figure 5.10 The result of the virtual pub environment and objects texture mapped with the single combined texture.

5.3. Avatar creation tool

The avatar creation tool is a modification of the expression tool, that allows the user to texture map a face image onto the avatar, customise the body colour and initial

expression of the avatar, instead of texture mapping only the face mesh. The avatar creation tool allows the user to create and design the avatar before it is introduced into the virtual pub environment. The texture mapping process is interactive. The user sees the adjustment made on the face texture and fine tunes the texture mapping of the avatar's face. The user may preview the face expression animations of the avatar in the avatar creation tool. When the user completes the customised design features of the avatar, the avatar together with all its details is sent to the virtual pub environment, without the need to close the avatar creation tool.

The avatar creation tool is interactive (Figure 5.11), and the interaction sequences are simple with enough options and choices for the user to generate an avatar for the virtual pub environment. The interaction devices for the avatar creation tool are the keyboard and the mouse. The basic interaction and selecting options in the avatar creation tool is performed using the mouse, while detail texture mapping adjustment is performed using the keyboard.

Figure 5.11 The avatar creation tool.

The graphical user interface (GUI) of the avatar creation tool and the detail functions of all controls in the GUI are summarised in the Appendix.

5.4. Social Interaction application

The theoretical design of the virtual pub environment and the geometric models of the virtual environment are implemented into a social interaction application. The user interacted with the avatar, after the avatar was sent into the virtual pub environment. The social interaction tool is initialised to include the virtual pub environment and a single avatar playing the role of the barman. Other avatars are first created by the creation tool and are then inserted in the social interaction application.

The interaction between the user and the application is based on selection. First the user selects the avatar and then the task or action that the avatar should perform (Figure 5.12). The avatars' action choices are the actions determined in the previous chapter; namely, drinking, talking to other avatars, watching television and listen to music, while the tasks are 3D interaction tasks which involves rotating the avatar and position it in the virtual pub environment. The interactions between the avatars in the virtual pub are automatic. Once the user selects an action for one of the avatar in the virtual pub environment, the avatar will engage into interaction with another avatar. Although the interaction between the avatars is performed automatically, it requires the user to initiate the action in one of the avatar.

Figure 5.12 The social interaction application with one of the avatar selected for interaction, this was indicated by the bounding box.

The actions of avatars are presented as buttons situated at the top panel of the Social interaction GUI and the 3D interaction tasks are situated at the bottom panel in the Social interaction GUI (Figure 5.13). The mouse is used as the input device for the social interaction application.

Figure 5.13 The actions for the avatars were situated at the top panel in the GUI and the interaction tasks are situated at the bottom panel in the GUI.

The feedback from the avatar is based on visual output onto the screen; the user observes the avatar's emotional expression and body animation when it performs an action or interacts with other avatars in the virtual pub environment. The maximum number of avatars in the virtual pub environment is set to ten, because each avatar had a set of face textures to be loaded and each avatar's action increased the computation of the application. If the maximum number of avatars is not defined in the application, the user might overloaded the virtual pub environment with avatars, which caused the application to performed poorly or even failed to performed execution. The

performance of the application also depends on the number of avatars in the virtual environment and the system resources available for the social interaction application.

The detail functions of all controls in the social interaction GUI will be further discussed in the Appendix.

5.5. Summary

In this chapter, the avatars were generated according to the prototyped models and animated based on the body motion cues defined during the theoretical design of the avatar. Therefore, modelling transformations are applied to the avatar models in OpenGL to achieve correct modelling and motion of body parts.

The theoretical designs of the virtual environment are further implemented into the actual application. The floor of the virtual environment is divided into a grid, for collusion detection and position determination.

The expressive texture tool is modified into the avatar creation tool, instead of only texture mapping a face image onto the face mesh, the dress colour of the avatar can be selected by the user and previewed. The user can create the avatar in the avatar creation tool and then send the data of the avatar into the virtual pub environment.
The social interaction application described in this chapter is one possible application of the avatar and provides basic interaction between different avatars in the virtual pub environment, as well as the user-avatar interaction. The interactions in the social interaction application have a simple to use graphical interface. The selected avatar performs actions selected by the user and interacts automatically with other avatars if the action performed affects other avatars in the virtual environment.

The next chapter concludes the findings and results of this thesis, discusses future work and possible improvements to the development of avatars and social interactive application.

Chapter 6

Conclusions and Future work

This Chapter discusses the findings and results of this thesis, from designing the theoretical prototyped avatar to the actual implementation of the avatar creation tool and of the interactive virtual pub environment. The results of this interactive virtual environment simulate a basic real-world social environment, which allows the user to observe the social interaction between avatars, and the emotional changes of avatars caused by the social interaction. Section 6.1 concludes this thesis, by discussing the results of generating avatars that can displayed facial expressions using the expression texture approach, combined with body motions to created expressive avatars and generated this avatar into the virtual pub environment, allowing the avatar interacted with other avatars and the user. Section 6.2 discusses the possibility of improving the avatar creation process and the modification of the virtual pub environment and the social interaction application tool into a fully functional avatar application and future work.

6.1. Expressive Textures

The goal of this thesis is to create the expressive textures approach and implement the tools to create expressive avatars and allow them to appear and interact in a virtual pub environment. The expressive Texture approach used synthetic or video images of faces, which use few system resources and provides a fair amount of realism. The expression of the avatars is animated, by displacing the texture mapping co-ordinates of the face

texture mapped onto the face mesh. These avatars are implemented in a simple virtual environment, which demonstrate the interaction process between different avatars in this virtual environment and the effectiveness of the expressive texture approach combined with body animation when implementing these kinds of avatars in the virtual environment.

The expressive texture approach is implemented by determining the motion cues of all the basic expressions. Based on these motion cues, the texture co-ordinates of the face texture that is mapped onto the face mesh of the avatar are displaced to animated facial expressions. Complex masking is also applied to the face mesh to simulate eye animation using the face texture and pupil mask. Simple body animations of the expressive avatar are created by analysing the possible basic motions of the avatar in a specific virtual environment (e.g. The virtual pub) and classified them into body motion cues. Based on these motion cues, the joint rotation at specific body part is applied by the accumulated matrixes that contain the translation and rotation of the body part relative to the origin of the avatar. (E.g. translate the body part to the origin, applied the rotation of the body part, translate back to the correct position and then draw the body part object).

The expressive avatars are implemented in the virtual pub environment that tries to simulate a real environment and allows different avatars to interact in this environment. The virtual pub environment and the virtual objects in this environment are modelled with a low polygon count, and texture mapped with a single complex texture. This avoids use of too many system resources for drawing the environment, than for computing the animations in the environment. The virtual pub environment is a closed environment, therefore no virtual forces are simulated in the environment. However, the floor of the pub environment is structured such that simple space and position detection by the expressive avatars is possible in the virtual pub environment.

Expressive Texture approach to avatars is a useful approach for numerous applications that requires use over a network, such as virtual conferencing or virtual conference

room, as well as in collaborative distributed virtual environments and low bandwidth teleconferencing. This allows the distant users to interacted will each other in these virtual environments, in which they are represented by expressive texture avatars. This enables the users to recognise each other based on the face texture image or remained anonymous by using a synthetic texture image. When the avatar is combined with more complex body motion, it can be also used in entertainment, creating expressive animated characters that show facial expressions or emotions together with body motions. This can improve the computer games system and lower the required system resources, than the virtual characters that only had body expressions or when a series of face images with facial expressions are used to animate the facial expression of the character.

The implementation of the expressive avatars into the virtual pub environment was successful, and the interaction between different avatars is fairly realistic. The GUI of the virtual pub environment is simple to use with a few selection options that the user required to interact with the selected avatar. However, the actions of the avatar rely on the actions from the user or other avatars. Therefore, the virtual pub environment will be more realistic and stimulating, if an autonomous interaction process is implement on the avatars in the virtual pub environment. The other limitation is that for slower computers, the initialisation process is long due to the loading of large audio files for the avatars' speech and environment sound. The alternative will be lowering the quality of the audio files or used a better audio compression format for the application.

6.2. Conclusions and Future work

The result of using texture co-ordinates displacement in simulating facial expression is fairly realistic. If an image with skin folds (wrinkles) are available the Expressive Textures approach can use blending to animated skin folds to achieved more realistic facial expression, while keeping the computations at the lowest and using fewer system resources. When still images are used the trade off is that this approach can distort the hair on the image, if the person's hair is lying close to the eyes or eyebrows. Another

limitation is that mapping an open mouth of the user or actor onto the avatar's closed mouth is not realistic, when human speech is modelled with this approach.

When the body animation of the avatar is animated together with the facial expressions of the avatar, the avatar looks more interesting and realistic than animating the facial expressions alone. However, the body animation requires longer computation than animating facial expressions using the expressive texture approach, which makes the body animation not synchronised with the facial animation. Therefore, the facial animation completes faster than the complex body animation in slower computers. The user will find this difficult to determine the emotional expression of the avatar, when the avatar performed a series of emotional expressions one after the other. Therefore, a synchronisation between facial and body animation is required for different CPU speed and dependent to the system resources available for animation.

The virtual environment can become more realistic by including more virtual objects that resembled the real objects, but this require more system resources for loading textures for these virtual objects and drawing them in the virtual environment. When complex texture mapping is applied to the virtual environment, the number of textures loaded for the virtual environment is reduced to one. If the virtual objects required detailed texture image, the resolution of the combined texture can be increased to accommodate more detailed textures for the virtual objects in the virtual environment, but more system memory is then used as the size of image file increased according to the resolution of the texture. Obviously, the realism of the virtual environment is dependent on the hardware of the system. Apart from the virtual pub environment, the avatars can be inserted into other virtual environments e.g. office room, depending on the application targeted. The number of interaction options can changed based on the application. However, the number of options should be kept to a minimum with all the important interactions available to the user, to avoid confusion and simplify the interaction process.

For future work, the expressive avatars can be implemented in a networked virtual environment under a low bandwidth network, where users can interact with each other and participate in grouped activities (e.g. virtual conferencing system or a virtual workshop). This can determines the participants' aptitude and if the participants will benefit, when working in a grouped situation under a virtual environment represent by expressive avatars and compare the results with a real working environment.

Appendix

Avatar creation tool and Social interaction Application

This Chapter explains the functions of the avatar creation tool and the social interaction application, and summarises the execution sequences of these applications. This chapter also looks into the function of each control (e.g. button and menu option) in the graphical user interface of the avatar creation tool and the social interaction application. Section A.1 explains the initialisation of the social interaction application and the avatar creation tool. Section A.2 discusses the function of each control in the graphical user interface of the avatar creation tool and the manipulation of the avatar creation tool. Section A.3 explains the function of each control in the graphical user interface of the social interaction application and the manipulation of the social interaction application. Section A.4 gave a summary of this appendix.

A.1. Initialisation of the avatar creation tool and social interaction application

Both applications begin with initialisation, which loads the textures, data files and music files of the avatar creation tool and the social interaction application onto the system memory.

Figure A.2 shows that the texture mapping values in the input file are read into the application and then the application loads the textures of the avatar creation tool. Output "Init 1" indicates that the initialisation of the avatar development tool was successful. If the initialisation of the avatar creation tool fails, it will terminate the application immediately. The texture images of the avatar is loaded and after the initialisation process, the avatar is texture mapped with the texture images using the data stored in the input file.

After the avatar creation tool is initialised, the initialisation of the social interaction application commences automatically. The initialisation of the social interaction application load the textures for the graphical user interface (e.g. button textures), the virtual objects, the virtual environment, and the face texture for the barman avatar. Apart from loading textures of the application into system memory during initialisation, the music files of the jukebox and the voices of the avatars are loaded too (indicate by "Sound Init" in Figure A.3). The number of audio files loaded into system memory is large, therefore the application will take longer time to initialise and perform with lower frame-rate on slower computers with lower system memory. Once the audio files is loaded, the texture of the virtual environment, the virtual objects and the application is loaded. A successful initialisation is indicated by the output "InitBar 1" and the application commence with the loading of texture of the barman avatar.

The state of the virtual environment is set to neutral at the initialisation of the social interaction application. (E.g. the position of the virtual objects in the virtual environment is marked as occupied at the floor grid, while the empty grid space is marked as open). The viewpoint (the direction in which the user view into the virtual environment) is positioned in the centre of the virtual environment at the initialisation.

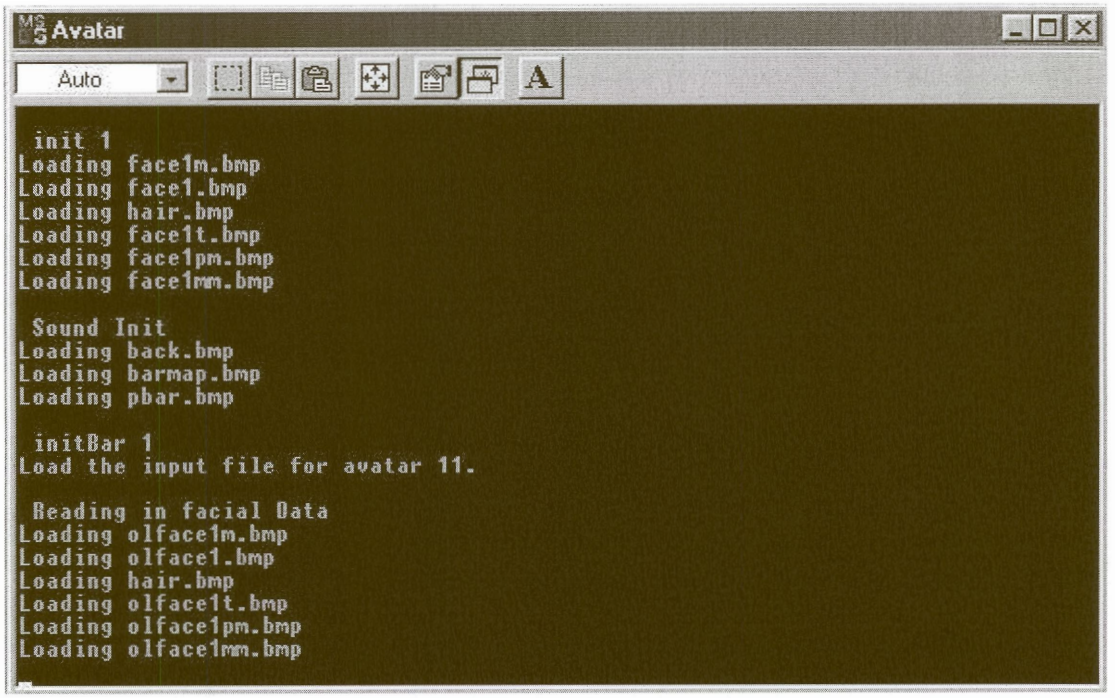


Figure A.3 Once the avatar creation tool completed the initialisation, the social interaction application commences with its initialisation.

After initialisation is completed successfully, the avatar creation tool and the social interaction tool wait for the user's interactions (the user selecting the controls on the graphical user interface).

A.2. Avatar creation tool

The avatar creation tool window is titled as "avatar builder". The basic controls of the avatar creation tool are on the display screen, while the other controls are in the pop-up menu. The user can select these controls by using the mouse-input device.

The face texture of the avatar is mapped onto the face mesh of the avatar using the texture mapping values in the input file. If a new input file is created the default texture mapping values will be used to map the face texture onto the face mesh of the avatar. The face texture used for mapping the face mesh is shown on the tool at control number "1" in Figure A.4.

Figure A.4 The GUI of the avatar creation tool with the controls numbered for easier identification.

The functions of all the controls on the GUI are explained in the following table (Table A.1) and they are shown in the table according to the number in Figure A.4. When the body rotation option is on, the avatar can be viewed from any direction by dragging the mouse pointer on the screen. When the body rotation option is off, the eye of the avatar looks at the direction of the mouse movement.

Control Number	Button Control Function
2	This button control enables and disables the masking mode. When the masking mode is disabled, the pupil mask is not applied to the face mesh, so the eye rotation cannot be animated in the application. The masking mode is enabled at the initialisation of the tool.
3	This button control changes the appearance of the models in both the avatar creation tool and social interaction application to a mesh, or wire-frame, vertex appearance. Initially the button is set such that the model appears in a mesh form.
4	This button can change the speed of the application. The button is set to high at the initialisation of the tool.
5	This button decreases the texture mapping adjustment value.
6	This button increases the texture mapping adjustment value.
7	This button causes the view to zoom out of the avatar.
8	This button causes the view to zoom in of the avatar.
9	This button changes the face expression of the avatar to the previous expression in the tool.
10	This button enables the avatar to perform the facial animation in the avatar creation tool, and enables the avatars to perform the facial and body animation in the social interaction application. During initialisation, the animation option is disable in both applications.
11	This button changes the face expression of the avatar to the next expression in the tool.

Table A.1 The button control function of the avatar creation tool.

Figure A.5 The pop-up menu of the avatar creation tool.

In the pop-up menu, there are options that allow the user to perform more complex manipulation than the selection options in the GUI. The function of each selection option in the pop-up menu is explained in Table A.2.

Selection Option name	Function of the selection
"adjustment x"	Adjusts the texture mapping co-ordinates shifting it along the x-axis of the face mesh.
"adjustment y"	Adjusts the texture mapping co-ordinates shifting it along the y-axis of the face mesh.
"adjustment eye brow level"	Adjusts the texture mapping co-ordinates at the eyebrow region. The user can simply move the eyebrows in the texture to the correct position in the face mesh by selecting the adjustment value button or enter the "-" and "+" key.
"adjustment eye level"	Adjusts the texture mapping co-ordinates at the eye region. The usage of this selection is similar to those of "eyebrow adjustment level" selection.
"adjustment mouth level"	Adjusts the texture mapping co-ordinates at the mouth region. The usage of this selection is similar to those of "eyebrow adjustment level"
"Load Other Face map"	Changes the face texture at any time by selecting this option.
"Switch head/eye rotation"	Switches the mouse drag properties to perform rotation on the whole avatar according to the head direction or rotation the eyes of the avatar.
"Reset head/eye rotation"	Resets the rotation values back to the neutral position.
"Expression"	This option has a sub menu that contains all the basic facial expression of the avatar. When the user selects the expression, the avatar will change its facial expression to the one selected by the user.
"Disable/Enable Grid Background"	Disables or enables the background image.

"Change screen size (500-800)"	Changes the window size between 500x500 and 800x800 size.
"Quit Program"	Closes the avatar tool and the social interaction application together and save the texture mapping data into the input file.
"Gender"	The female or male model is selected
"Add Avatar"	Adds the avatar in the avatar creation tool into the virtual pub environment.
"Print Avatar"	Prints the information of the avatars.
"Color"	Colour selection option that contains predefined colours for colouring the avatar's body.
"Avatar part color"	Enables the user to select the upper body, lower body or the shoes of the avatar and provide a colour for this avatar part.

Table A.2 The functions of the selections in the pop-up menu.

Complex texture mapping adjustments on the face mesh are performed using the keyboard, this allows the user to adjust the face texture by fine tuning the texture co-ordinates at each vertices in the face mesh. The user can increase the mapping value at a vertex by pressing the "z" key, while decreasing the mapping value by pressing the "x" key. The user can move between vertices by pressing "c" to move to the next vertex or pressing the "v" key to move back to the previous vertex. While the "b" and "n" key allows the user to move to the first vertex at the next or previous row of vertices along the face mesh.

A.3. Social interaction application

The controls of the social interaction application are on the display screen and the user can select the controls by using only the mouse-input device.

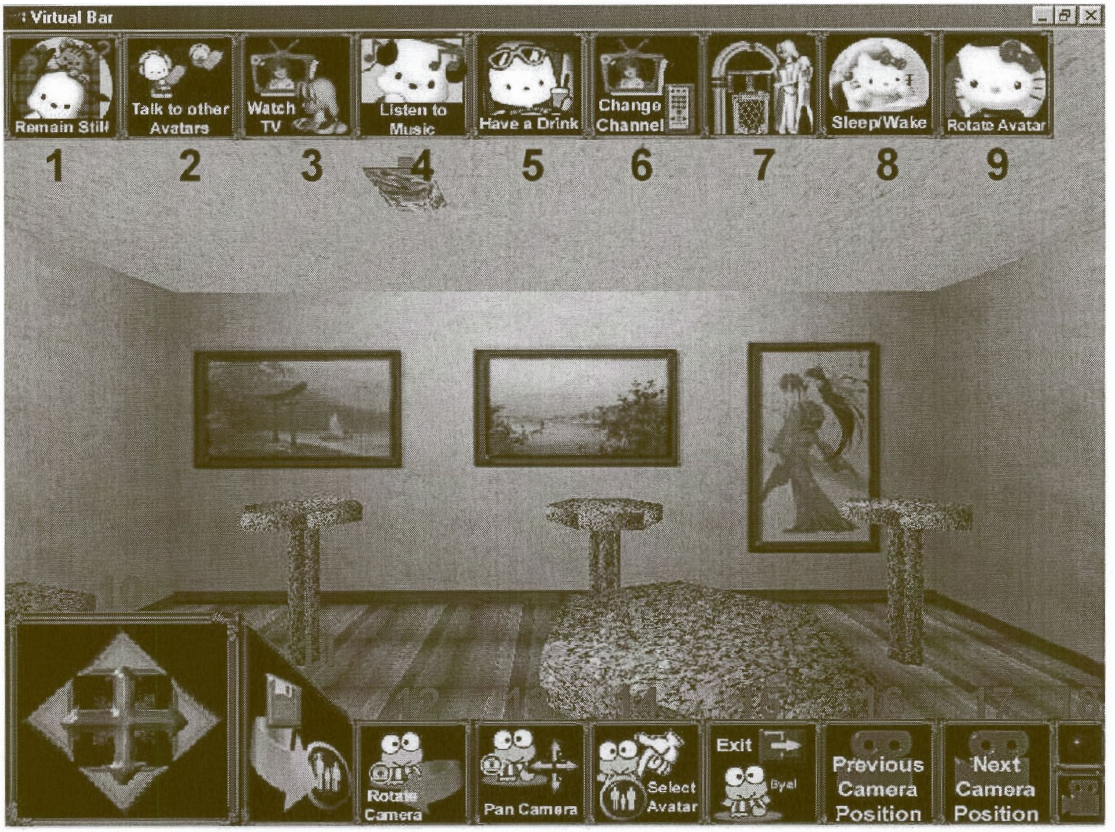


Figure A.6 The selection controls of the social interaction application GUI.

The functions of the selection controls on the Graphical User Interface (GUI) of social interaction application are explained in the following table (Table A.3) according to the numbers in Figure A.6. When the user drags the mouse point on the screen, the camera views will change according to the camera mode selected by the user.

Selection Control Number	Selection Control Function
1	This selection control causes the selected avatar to stop the action that it is busy performing and to remain still.
2	When this control is selected, the selected avatar will initiate a conversation with one of the surrounding avatars.
3	This control causes the avatar to watch the television at

	the wall of the pub.
4	When the user selects this control, the selected avatar will walk towards the jukebox and if other objects or avatars do not block its path, it will play music from the jukebox.
5	This selection control instructs the selected avatar to have a drink.
6	If the selected avatar is a barman, the selected avatar will change the television channel.
7	This control instructs the selected avatar to change the music of the jukebox and listen to this music.
8	When this control is selected, the selected avatar will switch between sleeping mode and active mode. Initially all avatars are in sleeping mode, and avatars can only perform the actions when they are active (Awake).
9	This selection control rotates the avatar at 90 degrees angle.
10	The user can position the avatars in the pub by selecting the direction arrows, and it will move the avatar in that direction.
11	When this control is selected, the initial positions and the face textures of all the avatars in the pub will be loaded into the application.
12	This control switches the camera mode to the camera rotation mode.
13	This control switches the camera mode to the camera pan mode.
14	This control enables the user to select an avatar in the pub sequentially.
15	This control quits the social interaction application and

	the avatar creation tool.
16	The user can move to a previous camera position (centre view, view from the barman, view At the selected avatar and top view) using this control.
17	The user can move to a next camera position using this control.
18	This control re-targets the camera on the selected avatar, after the position of the avatar has changed in the pub.

Table A.3 The selection control functions of the social interaction tool GUI.

The selection button controls in the GUI of the social-interaction application can automatically be resized based on resolution set by the operating system at full screen mode and maximised window mode. It is best if the user select the auto-hide option for the taskbar when executing this application at maximised window mode, because the height of the taskbar affects the calculation on the available window space for this application on the desktop. By entering the " f " key on the keyboard with the selected application, this will cause the application to run in full screen mode. In full screen mode, the application is not affected by the size of the desktop area, and if the " w " key is enter, it will set the application to return to window mode.

A.4. Summary

This appendix should give the user an overview of the functionality of the controls in the avatar creation tool and the social interaction application, as well as the functions and interactions in both of these application. The avatar creation tool has many selection controls than the social interaction application, because it has more complex manipulation that involve texture mapping the face and setting up the avatar for the social interaction application.

Bibliography

[1] A. Azarbayejani et al., "Visually Controlled Graphics", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 15, No. 6, 1993, pp.902-905.

[2] Brooke, N.M. & Scott, S.D. "Computer graphics animations of talking faces based on stochastic models", IEEE (Proceedings of the International Symposium on Speech, Image-processing and Neural Networks), Hong Kong, 1994, pp. 73-76.

[3] F. Lavagetto, "Converting Speech into Lip Movements: A multimedia Telephone for Hard of Hearing People." IEEE Trans. On Rehabilitation Engineering, Vol. 3, No. 1, 1995, pp. 90-102.

[4] D. Thalmann, "A New Generation of Synethetic Actors: the Interactive Perceptive Actors", Proc. Pacific Graphics 96, National Chiao Tung University Press, Hsinchu, Taiwan, 1996, pp.200-219.

[5] D. Terzopoulos and K. Waters, "Techniques for Realistic Facial Modeling and Animation", Proc. Computer Animation 91, Springer-Verlag, New York, 1991, pp.59-74.

[6] D. Thalmann, "Using Virtual Reality Techniques in the Animation Process", Virtual Reality Systems, Academic Press, San Diego, 1993, pp.143-159.

[7] F. I. Parke, "Computer Generated Animation of Faces." In Proceedings ACM annual conference, August 1972.

[8] E.M. Caldognetto et al., "Automatic Analysis of Lips and Jaw Kinematics in VCV Sequences", Proc. Eurospeech '89 Conf. 2, European Speech Communication Assoc. (EXCA), Grenoble, France, 1989, pp.453-456.

[9] E.C. Patterson, P.C. Litwinowich, and N. Greene, "Facial Animation by Spacial Mapping", Proc. Computer Animation 91, Springer-Verlag, New York, pp. 31-44, 1991.

[10] G.Sannier, N.Magnenat-Thalmann, "A User-Friendly Texture-Fitting Methodology for Virtual Humans", Computer Graphics International'97, 1997.

[11] H. Li, P. Roivainen, and R. Forchheimer, "3D Motion Estimation in Model-Based Facial Image Coding", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 15, No. 6, 1993, pp. 545-555.

[12] I.S. Pandzic et al., "Real-Time Facial Interaction", Displays, Vol. 15, No. 3, 1994, pp.157-163.

[13] I. Essan and A. Pentland, "Facial Expression Recognition using Visually Extracted Facial Action Parameters", Proc. Int'l Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland, 1995.

[14] J. P. Granieri et al., "Behavioral Control for Real-Time Simulated Human Agents", *Proc. ACM Symposium on Interactive 3D Graphics*, ACM Press, 1995, pp 173-180.

[15] K. Mase and A. Pentland, "Automatic Lipreading by Computer", Trans. Inst. Elect. Information and Communication Eng., Vol. J73-D-II, No. 6, 1990, pp. 796-803.

[16] K. Waters, "A Muscle Model for Animating Three-dimension Facial Expression", In SIGGRAPH'87, volume 21 of Computer Graphics Annual Conference series, July 1987, pp.17-24.

[17] K. Waters and D. Terzopoulos, "Modeling and Animating Faces using Scanned Data", J.Visualization and Computer Animation, Vol. 2,No 4, 1991, pp.123-128.

[18] L. Emering, R. Boulic, and D. Thalmann, "Interacting with Virtual Humans through Body Actions", IEEE CG&A, Vol.18, No. 1, Jan./Feb. 1998, pp.8-11.

[19] M. Cavazza, R. Earnshaw, N. Magnenat-Thalmann, D. Magnenat-Thalmann, "Motion Control of Virtual Humans", IEEE Computer Society, September/October, pp. 24-31, 1998.

[20] M. Escher, N. Magnenat-Thalmann, "Facial Deformation from MPEG-4", Proc. Computer Animation'98, 1998.

[21] M. Escher, N. Magnenat-Thalmann, "Automatic 3D Cloning and Real-Time Animation of a Human Face", Proc. Computer Animation, IEEE Computer Society, pp. 58-66, 1997.

[22] M.J. Black, Y. Yacoob, "Tracking and Recognizing Rigid and Non-Rigid Facial Motions using Local Parametric Models of Image Motions", ICCV 1995.

[23] N. Jennings, and M. Wooldridge (1996) "Software Agents." IEEE Review, January 1996, P. 17-20.

[24] N. Magnenat-Thalmann, E. Primeau, and D. Thalmann, "Abstract Muscle Action Procedures for Human Face Animation", Visual Computer, 1988, 3 , P.290-297.

[25] N.Magnenat-Thalmann, D.Thalmann, "Animating Virtual Actors in Real Environments", ACMMS'97, Switzerland, Springer, Vol.5, No2, 1997, pp.113-125.

[26] N.Magnenat-Thalmann, D.Thalmann, "Complex Models for Visualizing Synthetic Actors", IEEE CG&A, Vol. 11, No. 5, September 1991, pp. 32-44.

[27] N.Magnenat-Thalmann, I.S.Pandzic, J-C.Moussaly, "The Making of the Terra-Cotta Xian Soldiers", Digital97, Geneva, Switzerland, Digital Creativity, 8(2), July 1997, pp.66-73.

[28] Nahas M., Huitric H., Rioux M., and Domey J., "Facial Image Synthesis Using Skin Texture Recording", Visual Computer, December 1990.

[29] O. Renault, N. Magnenat-Thalmann, and D. Thalmann, "A Vision-Based Approach to Behavioral Animation," J. Visualization and Computer Animation, Vol. 1, No. 1, 1990, pp.18-21.

[30] P. Eisert, B. Girod, "Analyzing Facial Expressions for Virtual Conferencing", IEEE Computer Graphics & Application, September/October, 1998, pp.70-78.

[31] Parke, F.I "Techniques for facial animation". In N. Magnenat-Thalmann and D. Thalmann (Eds.) New Trends in Animation and Visualization. John Wiley, Chichester, 1991, pp.229-241.

[32] P. Maes et al. "The ALIVE system: Full-body interaction with Autonomous Agents", Proceedings of the Computer Animation '95 Conference, Geneva, Switzerland, IEEE-Press, April 1995.

[33] P. Karp and Steven Feiner. "Issues in the automated generation of animated presentations". In Proceedings of Graphics Interface '90, May 1990, pp. 39-48.

[34] S. Balcisoy et al. "An Interactive Interface for Directing Virtual Humans". In Proc. ISCIS 98, IOS Press, 1998.

[35] S. M. Drucker et al. "CINEMA: A system for procedural camera movements". In David Zeltzer, editor, Computer Graphics (1992 Symposium on Interactive 3D Graphics), volume 25, pages 67-70, March 1992.

[36] S. M. Drucker and D. Zeltzer. "CamDroid: A system for implementing intelligent camera control". In Mychael Zyda, editor, Computer Graphics (1995 Symposium on Interactive 3D Graphics), volume 28, pages 139-144, April 1995.

[37] S. Valente, J. Dugelay, "Face Tracking and Realistic Animations for Telecommunicant Clones", Eurecom Institute, IEEE MultiMedia, January - March 2000), pp.34 - 43.

[38] S. K. Semwal, R. Hightower, and S. Stansfield, "Closed Form and Geometric Algorithms for Real-Time Control of an Avatar", Proc. IEEE VRAIS 96, IEEE Computer Society Press, Los Alamitos, Calif., 1996, pp. 177-184.

[39] Takaaki Akimoto, Yasuhito Suenaga and R. Wallace, "Automatic Creation of 3D Facial Models", IEEE Computer Graphics & Application, September, 1993, pp.16-22.

[40] T. Molet, R. Boulic, and D. Thalmann, "A Real-Time Anatomical Converter for Human Motion Capture", Proc. Eurographics Workshop on Computer Animation and Simulation, R. Boulic, ed., Springer, Wien, Austria, 1996, pp.79-94.

[41] T.K.Capin, I.S.Pandzic, D.Thalmann, N.Magnenat-Thalmann, "A Dead-Reckoning Algorithm for Virtual Human Figures", Proc.VRAIS'97 (IEEE Press), Albuquerque, USA, 1997, pp.161-168.

[42] T.K.Capin, I.S.Pandzic, H.Noser, N.Magnenat-Thalmann, D.Thalmann, "Virtual Human Representation and Communication in VLNET Networked Virtual Environments", IEEE Computer Graphics and Applications, Special Issue on Multimedia Highways ,Vol.17, No.2, March/April, 1997, pp.42-53.

[43] W.S Lee, P.Kalra, N.Magnenat-Thalmann, "Model Based Face Reconstruction for Animation", Proc. MMM'97 (World Scientific Press), Singapore, 1997, pp.323-338.

[44] Yanghee Nam et Kwangyun Wohn, "Recognition of hand gestures with 3D, nonlinear arm movements", Pattern Recognition Letters, vol. 18/01, 1997, pp105-113.

[45] Y. Lee, D. Terzopoulos, K. Waters, "Realistic modeling for facial animation", In Computer Graphics Annual Conferences series (Proc. SIGGRAPH'96), 1996, pp. 55-62.

[46] M. Hirose, T. Ogi, T. Yamada, K. Tamagawa, "Development of Stereo Video Avatar in Networked Immersive Projection Environment", IEEE International Conference on Image Processing (ICIP'99), 1999.

[47] T. Kanade, P.W. Rander, "Virtualized Reality: Being Mobile in a Visual Scene", Proceeding of ICAT/VRST'95, 1995, pp.133-142.

[48] S. Moezzi, A. Katkere, D. Kuramura, R. Jain, "Reality Modeling and Visualization from Multiple Video Sequences", IEEE Computer Graphics & Application, November, 1996, pp.58-63.

[49] N. Magnenat-Thalmann, D. Thalmann, "Complex Models for Animating Synthetic Actors", IEEE Computer Graphics & Application, September, 1991, pp.32-44.

[50] D. Terzopoulos and K. Waters, "Physically-Based Facial Modeling, Analysis and Animation" , Journal of Visualization and Computer Animation, March, 1990, pp.73-80.

[51] D. Terzopoulos and K. Waters, "Analysis and synthesis of Facial Image Sequences using Physical and Anatomical Models", IEEE Trans, Pattern Analysis and Machine Intelligence, June, 1993, pp.569-579.

[52] B. Guenter, C. Grimm, D. Wood, H. Malvar, and F. Pighin. "Making Faces." In SIGGRAPH 98, Computer Graphics Annual Conference series, Addison Wesley, July, 1998, pp.55-66.

[53] P. Bergeron and P. Lachapelle. "Controlling Facial Expressions and Body Movements in the Computer-Generated Animated Short 'Tony De Peltrie' ". In SIGGRAPH 85, Computer Graphics Annual Conference series. Addison Wesley, July, 1985.

[54] I. Essa, S. Basu, T. Darrell, and A. Pentland. "Modeling, Tracking, and Interactive Animation of Faces and Heads Using Input from Video." In Computer Animation Conference, Computer Graphics Annual Conference series, June, 1996, pp.68-79.

[55] L. Williams. "Performance-Driven Facial Animation." In SIGGRAPH 90, volume 24 of Computer Graphics Annual Conference series, Addison Wesley, August, 1990, pp.235-242.

[56] N. Magnenat-Thalmann, H. Minh, M. de Angelis, and D. Thalmann. "Design, Transformation and Animation of Human faces. The Visual Computer, March, 1989, pp.32-39.

[57] Tsuneya Kurihara and Kiyoshi Arai, "A Transformation Method for Modeling and Animation of the Human Face from Photographs", Computer Animation, Springer-Verlag Tokyo, 1991, pp.45-58.

[58] P. J. Burt and E. H. Andelson, "A Multoresolution Spline With Application to Image Mosaics", ACM Transactions on Graphics, October 1983, pp.217-236.

[59] G. Sannier, N. Magnenat-Thalmann, "A flexible texture fitting model for virtual clones", Proceedings of Computer Graphics International, IEEE Computer Society, 1997, pp.167-176.

[60] P. Kalra, A. Mangili, N. Magnenat Thalmann, D. Thalmann, "Simulation of Facial Muscle Actions Based on Rational Free Form Deformations", Computer Graphics Forum (Proceedings of Eurographics'92), Blackwell Publishers, Oxford, 1992, pp.59-69.

[61] R. Boulic et al. "The HUMANOID environment for Interactive Animation of Multiple Deformable Human Characters", Computer Graphics Forum (Proceedings of Eurographics'95), Blackwell Publishers, Oxford, 1995, Vol. 14, No.3.

[62] M. Carignan, Y. Yang, N. Magnenat-Thalmann, D. Thalmann, "Dressing Animated Synthetic Actors with Complex Deformable Clothes", Computer Graphics (SIGGRAPH proceedings 1992), 26(2), 1992, pp.99-104.

[63] P. Volino, N. Magnenat-Thalmann, J. Shen, D. Thalmann, "An Evolving System for Simulating Clothes on Virtual Actors", Computer Graphics in Textiles and Apparel (IEEE Computer Graphics and Applications), September 1996,pp 42-51.

[64] M. Roussos, A. E. Johnson, J. Leigh, C. A. Vasilakis, and T. G. Moher, "The NICE project: Narrtive, Immersive, Constructionist/Collaborative Environment for Learning in Virtual Reality" ,ED-MEDIA/ED-TELECOM Proceedings 1997,pp.917-922.

[65] J. Tromp, A. Bullock, Anthony Steed, A. Sadagic, M. Slater, E. Frécon, "Small Group Behavior Experiments in the Coven Project", IEEE Computer Graphics & Application, November-December 1998, pp.53-63.

[66] D. J. Sturman, "Computer Puppetry", IEEE Computer Graphics & Application, January/February 1998, pp.38-45.

[67] T. Noma, L.Zhao and N. I. Badler, "Design of a Virtual Human Presenter", IEEE Computer Graphics & Application, July/August 2000, pp.79-85.

[68] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active Contour Models", International Journal of Computer Vision, 1988, pp.321-331.

[69] N. Magnenat-Thalmann, D. Thalmann, "The direction of Synthetic Actors in the film Rendez-vous à Montrèal, IEEE Computer Graphics and Applications, 7 (12), 1997, pp.9-19.

[70] M. Proesmans, L. Van Gool, "Reading between the lines – a method for extracting dynamic 3D texture. In Proceedings of VRST, 1997, pp.95-102.

[71] E. N. Coleman, R. Jain , "Obtaining 3-Dimensional Shape of Textured and Specular Surfaces Using Four-Source Photometry", IEEE Computer Graphics and Image Processing, no. 18, 1982, pp.309-328.

[72] H. Saji, H. Hioki, Y. Shinagawa, K. Yoshida and T. L. Kunii, "Extracting of 3D Shapes from the moving human face using lighting Switch Photometry", Creating and Animating the virtual world, Springer-Verlag, Tokyo, 1992, pp.69-86.

[73] P. Fua and Y. G. Leclerc, "Taking Advantage of Image-Based Constraints to Recover 3D surfaces", ComputerVision and Image Understanding, 64(1), July, 1996, pp.111-127.

[74] H. H. S. Ip, L. Yin, "Constructing a 3D individual head model from two orthogonal views. The Visual Computer, Springer-Verlag, no. 12, 1996, pp.254-266.

[75] M. M. Cohen and D. W. Massaro, "Modeling coarticulation in synthetic visual speech. In N. Magenenat-Thalmann and D. Thalmann (Eds.) Models and Techniques in Computer Animation. Tokyo: Springer-Verlag, 1993, pp.139-156.

[76] M. M. Cohen and D. W. Massaro, "Auditory/visual speech synthesis for lifelike computer characters." Lifelike Computer Characters '95, Snowbird Utah, September 26-29, 1995.

[77] D. Burford and E. Blake, "Real-Time Facial Animation for avatars in Collaborative Virtual Environments" , CS00-16-00.

[78] M. Hirose, T. Ogi, S. Ishiwata, T. Yamada, "Development and Evalution of the CABIN Immersive Multi-screen Display, Systems and Computers in Japan", vol. 30, no.1, 1999, pp.13-22.

[79] Kinetix Division of Autodesk Inc. Character studio. Computer Program, 1997.

[80] P. C. Litwinowicz. Inkwell, "A 2 ½ D animation system." In Thomas W. Sederberg, editor Computer Graphics (SIGGRAPH 91Proceedings), volume 25, July 1991, pp. 113-122.

[81] J. Hodgins and N. Pollard. "Adapting simulated behaviours for new characters" In Turner Whitted, editor, SIGGRAPH 97 Conference Proceedings, August 1997, pp.153-162.

[82] K. Perlin, "Real time responsive animation with personality." IEEE Transactions on Visualisation and Computer Graphics, March 1997, 1(1) pp.5-15.

[83] M. Gleicher, "Retargetting Motion to New Characters", SIGGRAPH 98 Computer Graphics Proceedings, July 19-24, 1998, pp. 33-42.

[84] C. Kervrann, F .Davoine, P. Perez, R. Forchheimer and C. Labit. "Generalised likelihood ratio-based face detection and extraction of mouth features", Pattern Recognition Letters, 1997, (18), pp.899-912.

[85] P. Pigazzini, F. Pedersini, A. Sarti, S. Tubaro: "3D Area Matching with Arbitrary Multi-view Geometry". EURASIP Signal Processing: Image Communications, special issue on 3D Video Technology, edited by S.A. Benton, B. Choquet, R. ter Horst, K.N. Ngan and M. Tanimoto. Vol. 14, Nos. 1-2, 1998, pp.71-94.

[86] F. Pedersini, A. Sarti, S. Tubaro: "Multi-resolution 3D reconstruction through texture matching". EUSIPCO 2000. Tampere, Finland, Sept. 5-8, 2000. Vol. 4, pp.2093-2096.

[87] F. I. Parke, K. Waters, "Computer Facial Animation", A. K. Peters, Wellesley, MA, 1996.

[88] K. Karpouzis, N. Tsapatsoulis, S. Kollias, "Moving to continuous facial expression space using the MPEG-4 facial definition parameter set", SPIE Electronic Imaging 2000, January 2000, San Jose, CA, USA.

[89] W. S. Lee, M. Escher, G. Sannier, N. Magnenat-Thalmann, "MPEG-4 Compatible Faces from Orthogonal Photos", Computer Animation 1999, Geneva, Switzerland, 1999.

[90] Bright Star Technologies Inc. Beginning Reading Software. Sierra Online, Inc 1993.

[91] T. Beier and S. Neely, "Feature-based Image Metamorphosis", In SIGGRAPH 92 Conference Proceedings, ACM SIGGRAPH, July 1992, pp.35-42.

[92] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, D. H. Salesin, "Synthesizing Realistic Facial Expressions from Photographs", In SIGGRAPH 98 Conference Proceedings, ACM SIGGRAPH, 19-24 July 1998, pp.75-84.

[93] P. E. Debevec, C. J. Taylor, and J. Malik, "Modeling and Rendering Architecture from Photographs: A Hybrid Geometry and Image Based Approach." In SIGGRAPH 96 Conference Proceedings, ACM SIGGRAPH, August 1996, pp.11-20.

[94] J. Gemmell, K.Toyama, C. L. Zitnick, T. Kang, S. Seitz, "Gaze Awareness for Video conferencing: A software Approach.", IEEE Multimedia, Oct-Dec, 2000, pp.26-35.

[95] Nissho Electronic Corporation, Dataglove, Virtual Reality Device, 1995

[96] VPL Research Inc, Data suit, Motion Capture Device, 1988

[97] Motion Analysis Corporation, Ortho Trak, Virtual Reality Device, 1993

[98] Motion Analysis Corporation, Eagle, Virtual Reality Device, 2001

[99] Polhemus, Star Trak, Virtual Reality Devices, 2001

[100] J. Raynes, "A Step by Step Guide to Drawing the Figure", Art Instructions, Collins & Brown limited, North Light Books, August 1997.

[101] P. Ekman and W. V. Friesen. "Manual for the Facial Action Coding System." Consulting Psychologists Press, Inc, Palo Alto, California, 1978.

[102] B. Damer, "Avatars! - Exploring and Building Virtual Worlds on the Internet." Peachpit Press, October 1997.

[103] S. Wilcox, "Web Developer.Com - Guide to 3d Avatars." John Wiley & Sons, June 1998.

[104] T. Capin et al, "Avatars in Networked Virtual Environments." John Wiley and Sons, May 1999.