# 

**Final Year Project Report**

**Controllable Piecewise Component Animation of a 3D model in an Augmented Reality Environment**

**Supervisor by**

**Pn. Tan Guat Yew**

**Prepared by**

**Leim Peng Peng (105906)**

**School** **of** **Mathematical** **Sciences**

**Academic Session 2010/2011**

# Abstract

This project describes the process of importing a wavefront file, identifying each component in the wavefront file so that each component can be controlled or moved individually. The output is display in augmented reality environment to enable the viewer to see the individual part in 3D environment from different angle.

The Standard Template Libraries (STL's) are useful template classes and our project uses STL to store the large array data imported from wavefront OBJfile.

Index Term: Piecewise components; Animation model; STL library; Augmented Reality

# Acknowledgement

First and foremost, I would like to express my special thanks of gratitude to my supervisor Madam. Tan Guat Yew who gave me the golden opportunity to do this wonderful project on the topic Controllable Piecewise Component Animation of a 3D model in an Augmented Reality Environment. The supervision and guidance and advice that she gave from the initial to the final level enabled me to develop an understanding of this project. She inspired me greatly to work in this project. Her willingness to motivate me contributed tremendously to this project. I also would like to thank her for showing some example and answer my question that related to the topic of my project. Besides, I would like to express to thanks her willingness to take times to help me by explain and pointing out our error of program which enable me complete my project in time. She also helped a lot by looking over my report and fixing my mistakes. The co-operation is much indeed appreciated.

Besides, I would like to thank the authority of University Science Malaysia (USM) for providing me with a good discussion class room and facilities to complete this project. Also, I appreciate the University for offering the subject Computer Modelling in Bachelor of Applied Science Degree program. It gave me an opportunity to learn about Computer Graphic.

Finally, an honourable mention goes to my families, seniors and friends for their understandings and supports on me in completing this project. Without help of the abovementioned particular, especially my parents, I would have faced many difficulties while doing this project.

Table of Contents

[Abstract 1](#_Toc324400222)

[Acknowledgement 2](#_Toc324400223)

[List of Table: 5](#_Toc324400224)

[1.0 Introduction 6](#_Toc324400225)

[2.0 Objective 8](#_Toc324400226)

[3.0 Hypothesis 8](#_Toc324400227)

[4.0 Literature Review 9](#_Toc324400228)

[5.0 Methodology 20](#_Toc324400233)

[5.1. Our File Format 20](#_Toc324400234)

[5.2. Create Wavefront Format File 21](#_Toc324400235)

[5.3. Analysis of Wavefront Format File 22](#_Toc324400236)

[5.4. Loading the OBJ File Format. 24](#_Toc324400237)

[5.5. Standard Template Library (STL) on vector 25](#_Toc324400238)

[5.6. Rendering OBJ file: 26](#_Toc324400239)

[5.7. Control each Part of the object 27](#_Toc324400240)

[5.8. Understanding how ARtoolkit works. 27](#_Toc324400241)

[6.0 Our Project 29](#_Toc324400242)

[6.1. Created simple OBJ file and MTL FILE 29](#_Toc324400243)

[6.2. Parsed simple OBJ file and MTL file into openGL 30](#_Toc324400244)

[6.3. Replace complicated model in openGL 33](#_Toc324400245)

[6.4. Render model in Augmented Reality Environment 33](#_Toc324400246)

[6.5. Navigate the model in 3D space 34](#_Toc324400247)

[6.6. Flow of project a) Flow of whole project 35](#_Toc324400248)

[7.0 Result 38](#_Toc324400250)

[8.0 Problems encountered 42](#_Toc324400251)

[8.1. The size of the array 42](#_Toc324400252)

[8.2. Parsing data 42](#_Toc324400253)

[9.0 Solution 43](#_Toc324400254)

[9.1. Using sscanf () function to extract data. 43](#_Toc324400255)

[9.2. Using vectors in Standard Temple Library to stored data. 44](#_Toc324400256)

[10.0 Achievement for the 1st semester 45](#_Toc324400257)

[11.0 Achievement in 2nd semester 45](#_Toc324400258)

[12.0 Conclusion and future work 46](#_Toc324400259)

[13.0 References 47](#_Toc324400260)

# List of Table:

Table 4.1 show the type of 3D data file. 11

Table 4.2 show summarize of 3D data file specifications. 12

Table 5.1 Properties of OBJ file. 23

Table 5.2 Properties of MTL file. 23

Table 6.1 sample of .obj file ’test.obj’ 29

Table 6.2 sample of .mtl file ’test.mtl’ 29

Table 6.3 the data structures in program. 30

Table 6.4 The key to control each part of the model 34

**List of Figure:**

[Figure 4.1 Relationship between earth-sun in AR environment 19](#_Toc324396922)

[Figure 4.2 The AR Volcano Kiosk 21](#_Toc324396923)

[Figure 4.3 The slider moves the tectonic plates. 21](#_Toc324396924)

[Figure 4.4 AR in 3D molecular Structures. 23](#_Toc324396925)

[Figure 5.1 Finalize and summarizes the steps above . 32](#_Toc324396926)

[Figure 6.1 Flow of project 38](#_Toc324396927)

[Figure 6.2 Flow of parse .obj file 39](#_Toc324396928)

[Figure 6.3 Flow of parsing .mtl file 40](#_Toc324396929)

[Figure 7.1 A simple model is rendered on openGL window. 41](#_Toc324396930)

[Figure 7.2 Model created in 3D studio Max 42](#_Toc324396931)

[Figure 7.3 Model rendered in openGL window. 42](#_Toc324396932)

[Figure 7.4 Model display in Augmented Reality environment. 43](#_Toc324396933)

[Figure 7.5 Model display in Augmented Reality . 43](#_Toc324396934)

[Figure 7.6 Model and indication panel is displays on the screen. 44](#_Toc324396935)

[Figure 7.7 control all the part of the model. 44](#_Toc324396936)

## Introduction

­In 3D computer graphics, 3D modelling also known as meshing, is the process of developing a mathematical representation of 3D object with a collection of vertices, edges that connected by various geometric entities such as triangles, line, curve, surface and etc. [1]. The product is called a 3D model. In order to meet needs related to 3D modelling, some 3D graphic system or software have been released into global market such as AutoCad, 3D Studio Max, Maya, Blender, etc. These tools are powerful where it enables users to create 3D model visually, however their 3D display techniques are currently not sufficient due to 2D interaction techniques. They often decompose 3D design operation into sequences of 2D interface operation. In addition, models that created by these tools lack of reality because they are totally different from 3D physical world. Therefore, researchers are focusing on pulling graphics out of conventional displays and integrating them into real world environment. Augmented Reality (AR) is the technology to superimpose graphics, audio and other sensory enhancements over a real world environment in real time [2].

When creating a complex object in AR environment that are challenging to design them by specifying each vertices and it’s corresponding surfaces. For this reason 3D editing programs are designed to let users visually create 3D models in 3D graphic system. Once 3D models have been created in a given editor, they can be exported or saved to Alias Wavefront Format for storing the geometry data of 3D models. Two types of files are created when the geometric data is exported to Wavefront Format in 3D Studio Max graphic system. First file is in OBJ format, it is a geometric definition file format first developed by [Wavefront Technologies](http://en.wikipedia.org/wiki/Wavefront_Technologies" \o "Wavefront Technologies) for its [Advanced Visualizer](http://en.wikipedia.org/wiki/The_Advanced_Visualizer) animation package which storing simple data format that represent the geometry data of the object ,such as the position of each vertex ,the UV position of each texture coordinate vertex, normals, and the faces that make each polygon defined as a list of vertices, and texture vertices. Vertices are stored in a counter-clockwise order by default, making explicit declaration of normals unnecessary. Second file is in Material Template Library format (MTL), it is a standard defined by [Wavefront Technologies](http://en.wikipedia.org/wiki/Wavefront_Technologies" \o "Wavefront Technologies)  for [ASCII](http://en.wikipedia.org/wiki/ASCII) files that define the light reflecting properties of a surface for the purposes of [computer rendering](http://en.wikipedia.org/wiki/Computer_rendering). This format file contain one or more material definitions, each material description in an .mtl file consists of the newmtl statement, which assigns a name to the material and designates the start of a material description. Each of material includes the color, texture, and reflection map of individual materials. Thus, all these are applied to the surfaces and vertices of objects [3].

## **Objective**

* To allow users to create a complex model by assembling piecewise component and grouping them to form a unified object by store the vertices of the object in a Standard Template Library (STL).
* To be able to view the animation of 3D interactions or control piecewise components of an object by usability of the system by combining the tangible interaction with the traditional input devices such as mouse and keyboard in Augmented Reality environment.
* To produce a flexible interface that able to assists the users in creating 3D models quickly and easily.

## Hypothesis

Expected outcome of this project is to be able to control piecewise components of the model in an Augmented Reality Environment.

## Literature Review

### What is 3D model

In 3D computer graphics, 3D modelling is the process of developing a mathematical, wireframe representation of any three-dimensional surface of object (either inanimate or living) via specialized software. The product is called a **3D model**. 3D models represent a 3D object using a collection of points in 3D space, connected by various geometric entities such as triangles, lines, curved surfaces, etc. Being a collection of data ([points](http://en.wikipedia.org/wiki/Point_%28geometry%29) and other information), 3D models can be created by hand, [algorithmically](http://en.wikipedia.org/wiki/Algorithm) ([procedural modelling](http://en.wikipedia.org/wiki/Procedural_modeling)), or [scanned](http://en.wikipedia.org/wiki/3D_scanner). 3D models are most often created with special software applications called 3D modellers [4]. 3D models are widely used anywhere 3D graphics are used. Apart from the rendered graphic, the model is contained within the graphical data file called 3D data format file [5].

### 3D Data File

3D visualizations for data provide an intuitive means to identify underlying structures and patterns in a data set which would otherwise often remain undetected. Software tools to create interactive graphical data representations in 3D can be used effectively both to explore inherently 3-dimensional data and subspaces and projections of higher dimensional data. The latter task is becoming increasingly important, as companies and research institutions are striving to fully exploit their vast data resources.

For these reasons, a great variety of software tools has been made available in the past to generate 3D data representations with interactive means to explore different perspectives, analyse class membership and data density using color schemes and contour surfaces, or compare and interlink different data sources using overlay-plots and hyperlinks [6]. The widely used Microsoft Visual C++ programming languages for Microsoft Window API contains several software packages for 3D data analysis including a package to interconnect Microsoft Visual C++ with OpenGL and various packages for multivariate data visualization. Besides that, those wary of the capabilities of a 3D data analysis, consider that file formats listed on this page are in continuous use around the world for every conceivable type of professional [3D activity](http://www.okino.com/conv/users.htm), ranging from conversions for online 3D model databases, to animation of the model, to movie production, to multi-media content development, to 3D game development and so forth [11].

There are many different file formats in use concerning 3D data analysis. Table 4.1 the properties of 3D data file specifications.

|  |  |
| --- | --- |
| Format | Properties |
| 3D2 | stereo CAD 3D 2.0  40 named objects, 3 lights and ambient. Colour palette, positions are in fixed point format as integer. Object from vertices. |
| 3DMF | Meta format  Hierarchical description of a scene: named objects, textures, surfaces, lights, camera, position, rotation, fog, NURB, matrix ,quaternion transforms |
| 3DS | 3DStudio  Binary. Hierarchical description of a scene: named objects, textures, surfaces, lights, camera, position, rotation, distances, fog, ray tracing, morphing. |
| 3DX | Appears game oriented. 3 lights, 3 distance grades allow 3 different representations. |
| Geo TIFF | A geographic format, basically 2D with height optionally. Focuses on system transformations and projections |
| Imagine | Hierarchical description of a scene: textures, lights, fog, surfaces, tracks. |
| Infini-D | Hierarchical description of scene or animation :textures, surfaces, objects, terrains, lights, fog, text, camera, path, sequencer |
| Inventor | Hierarchical scene description. Triangle lists, quad lists, and primitives build a shape. The complexity can be controlled. The environment allows ambient light, spotlight, fog. Material, camera, picking, translation, rotation, texture. |
| ISFF | Intergraph Standard File Format  Hierarchical description of objects only. |
| Lightwave 3D | Objects stored as mesh. The surface has a detailed list of attributes, such as color, shade, glossiness, reflection mode, refractive index, transparency, texture |
| MC4D | Cinema 4D  Has a detailed list of named objects, lights, camera, textures, positions, rotations. |
| NFF | Neutral file format  Simple scene description. Background, lights, viewpoint, frustum, cylinder, cone, sphere, polygon |
| NMGF | Noise model grid format.  Specialized to hold acoustic noise data together with geographic data. |
| OBJ | Wavefront object files  Grouped vertices allow any form to be described. Simple rendering choices: colors, shadows, reflections, textures, no lights though. |
| OpenFlight | Scene description, Level of detail, sound, bounding boxes, lights, textures. |
| POL | Polygonal model.  Vertices define polygons. Optional color information. |
| POV Ray | Scene description with bodies, lights and shadows, textures, pigments, reflection, refraction, attenuation, halos, fog, dust, rainbows, animation. |
| RPI | Enhanced STL.  Instead of unordered triangles, this format uses facets, edges, faces and primitives such as cube, cylinder, cone, sphere and torus to describe a shape. The file is more compact than STL. |
| SDML | Spatial data modelling language.  Vertices and polygons define shapes. Colors, viewpoints, animation, triggers to URL's. epically suited for terrain viewing. |
| STL | Stereo lithography.  Binary or text. Triangles with or without normal. No neighbour or color information. Triangles can be grouped as objects. |
| VRML | Virtual reality modelling language  Hierarchical description of a scene. Bodies, materials, camera, transformations, lights. Newer versions have fog, animation, multimedia, sensors (triggers), grouped objects. |

Table 4.1 show the type of 3D data file.

The summary is admittedly biased with a side glance to OpenGL and GLScene. There are some formats that only hold shapes. Depending on the application they may differ in one or another aspect. Some file hold only shapes and surface information ignore color or material. Their viewer may or may not use light. There are files that hold a complete scene including a camera with a certain setting. It all depends on what user required[7].

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Format | Shapes | Surfaces | Textures | Lights | Camera | Animation |
| 3D2 | √ | √ |  | √ |  |  |
| 3DMF | √ | √ | √ | √ | √ |  |
| 3DS | √ | √ | √ | √ | √ |  |
| 3DX | √ |  |  | √ | √ |  |
| GeoTIFF | √ |  |  |  |  |  |
| Imagine | √ | √ | √ | √ |  |  |
| Infini-D | √ | √ | √ | √ | √ | √ |
| Inventor | √ | √ | √ | √ | √ |  |
| ISFF | √ |  |  |  |  |  |
| Lightwave\_3D | √ | √ |  |  |  |  |
| MC4D | √ | √ | √ | √ | √ |  |
| NFF | √ |  |  | √ | √ |  |
| OBJ | √ | √ | √ |  |  |  |
| OpenFlight | √ | √ | √ | √ | √ | √ |
| POL | √ | √ |  |  |  |  |
| POV | √ | √ | √ | √ | √ | √ |
| RPI | √ |  |  |  |  |  |
| SDML | √ | √ |  |  | √ | √ |
| SHP | √ |  |  |  |  |  |
| SLC | √ |  |  |  |  |  |
| STL | √ |  |  |  |  |  |
| VRML | √ | √ | √ | √ | √ | √ |

Table 4.2 show summarize of 3D data file specifications.

### Definition of Augmented Reality

Augmented reality (AR) **is a technology where the real environment** **is enhanced by virtual elements. The virtual elements, which co-exist with the real environment, can be seen by the aid of wearable computing devices, e.g. head mounted devices, mobile phones with camera, personal digital assistance, etc. The purpose of AR is to enhance the information we naturally receive through our five senses, by adding superimposed, constructed virtual elements to bring complementary information and meaning that may not be possible to see by natural means. One of its most important characteristics in AR is the way in which it creates an interactive environment between computer system and user. Today’s AR environments create interactive systems that are no longer simply a face-to-screen exchange, but an interaction within the whole environment [12].**

#### ARToolKit

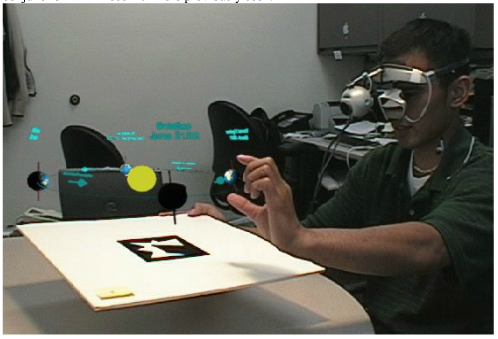
ARToolKit is a software library for building Augmented Reality (AR) applications. These are applications that involve the overlay of virtual imagery on the real world. For example, in the image to the right a three-dimensional virtual character appears standing on a real card. It can be seen by the user in the head set display they are wearing. When the user moves the card, the virtual character moves with it and appears attached to the real object [13].

### Application

Augmented Reality (AR) is a system if tools that allows a person to view one or more virtual 3D objects in the real world environment. The virtual object may be stationary or manipulated which displays on a screen. From numerous research projects have shown that visual 3Dobject are easier to understand and memorize when compared to messages transmitted by that text or sound. In almost all educational settings, whether in formal school classrooms, or on the job training, students watching a dynamic 3D animation is more useful to them for grasping the concepts of what is being conveyed rather than simply reading a textbook, or following verbal instructions. In recent year, researchers have striven to improve the visual features and flexibility of AR technology in the field of developing the teaching tools and other educational fields.

#### AR teaching relationship between sun and earth

Shelton and Headly use AR technology to display scientific exhibits. And though these scientific exhibits teacher can teach undergraduate geography student to understand about earth-sun relationships. Researcher are investigated the potential of AR to improve education by studying thirty-four student enrolled in geography during summer 2002. The student experienced three to six animated 3D Earth and Sun model using AR. The models were designed to build an understanding of the rotation and revolution, solstice and equinox, and seasonal variation of light and temperature of the northern and southern hemispheres. This project are able student understand the position and tilt of the earth the revolution of the sun, daily and seasonal conditions had experience on earth. Besides that, instructor are using this system can try to attempt to use real 3D object or props readily available in the room to demonstrate these relationship as a complete understanding of a sophisticated system operates in space and time. This might involve holding an orange or flashlight as the sun in one hand, and apple impaled on a pencil as earth with axis. Interactively, student can see how relative position along path of revolution interacts with tilt of earth’ axis and the resulting effect when illuminated by the sun. With this system, instructor can assuming all students understand the metaphor and assuming that student are visualized cosmic space [8].

Figure 4.1 Relationship between earth-sun in AR environment

#### AR Volcano Kiosk

Additional research is looking at other topics that also naturally lend themselves to3D space, such as the Volcano Kiask. The AR Volcano Kiosk was developed for Science Alive.It is an exhibit that teaches people about volcanoes, including details on subduction, rifts, the Ring of Fire, volcano formation, eruptions and tectonic plates (Figure 3). The interactive slider allows the user to control the speed of animation. As move the slider up, the animation advances, and as you move it down, it rewinds. Interactions include volcano formation, movement of tectonic plates, the sequence of events in the eruption of Mount St Helens, flows of magma in convection currents and the resultant forces applied to the Earth’s crust. The position of the slider is mapped to both the animation of the model and to various other dynamic effects such as particle system density, pixel shader brightness and ambient sound. This is highly beneficial in allowing the animated content to be presented to the viewer at their desired speed, and also allows for simple and intuitive pausing and rewinding. Modelling and animation of volcano formation, eruptions and movement of tectonic plates was outsourced to a professional 3D studio. The resulting data, in STL model format, includes high level photorealistic textures for the various scenes. While this was not ideal for all animations, it was the best format available that worked in both the design and development environments. [9]



Figure 4.2 The AR Volcano Kiosk

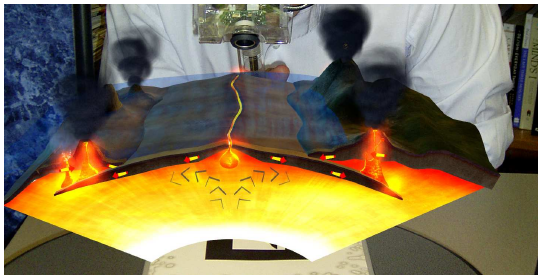


Figure 4.3 The slider moves the tectonic plates.

#### AR in 3D molecular Structures.

Besides that, a researched from The Graduate University for Advanced Studies, Japan found that learning molecular structures using AR also a good learning environment. Researcher developed VRML export function as an optional module in a visualization tool, so that the 3D data of molecular structures could be converted to VRML format, where VRML is a text [file format](http://en.wikipedia.org/wiki/File_format) for representing [3-dimensional](http://en.wikipedia.org/wiki/3-D_computer_graphics) (3D) interactive [vector graphics](http://en.wikipedia.org/wiki/Vector_graphics) and prepare the distinct VRML data for changing the visualization style such as presenting atomic relation and bond condition in the some molecular. The visualization module include a loader of the VRML file format data and rendering system of the 3D data. The rendering system were designed to contain the minimal function for viewing 3D molecular structures as follows : zoom in and out of the molecules, selection of atom(protein, nucleic acid, and ligand), change of the presentation of the model(balls, molecule-surface, and ribbon), color change (amino acid, createtini phodphokinase and acidility-alkalinelty-neutrality) and presentation of the annotations. The tangible interface of AR was used for manipulating the 3Dmodel of the molecular structures. Student able control the viewpoint of the 3D model by hand with square markers captured by a video camera in imagery scene and a mouse was used as an input device for the menu selection of the about function[10].



Figure 4.4 AR in 3D molecular Structures.

  
Figure 4.5 control the molecular structures part

## Methodology

There are few steps required in order to complete this project :

Export a 3D model from 3D studio max application into a format file, i.e. Wavefront’s OBJ format, that suitable for openGL–based visualization.

Identify each component of the 3D model from the OBJ file, and load in as individual object into an Standard Template Library (STL) in the p

Create an algorithm for manipulating the parts, and handling the user’s input. .   
Noticed that Step 3 is depending on the object that being model and no general solution can be suggested, whereas for Step1 and Step 2 do not depend on the object's nature. Thus the best way of implementing the tool is using an object-oriented library that performs visualization and manipulate with openGL entities.

### Our File Format

The 3D models created in 3D Studio Max can be exported to different file formats, for example OBJ file, VRML file, STL file, and 3DS file which are commonly used. There are many compatibility problems when exporting 3D data file. For example STL file format generated in one application cannot be converted to other file format because of its wrong file format. However, during the export/import operations from one application to another, errors are always occurred; for example, some of the import applications are unable to import the data file due to different version of the data file provided. Some applications are able to export the 3D graphical models to a data file but all the surfaces belong to individual part are saved without any unique identifier ,eg, STL file and POL file format these kind of file only contain the shape information. Besides that, when exporting an assembly model to 3D data file format, all the surface belonging to individual part are saved together without any discrete identifier and it leads to unable to control each part of the model.

In our analyses, we founds that openGL-based visualization requires some information to render a prefect model such as , coordinate of the surface mesh polygon, normal to the polygon , and optical properties of the material. However, this information can be easily obtained from OBJ file. OBJ file contains all the information that needed to construct a 3D geometry model, such as the position of each vertex, vertex normal, as well as the information about how vertices make up the faces and the smoothing groups assigned to each polygon. Besides that, an OBJ file also contains a reference MTL file which describes the visual aspects of the model. For example the information of the materials that applied to each surface of polygon in the model, and how they react in different lighting conditions or the transparency. Therefore, both OBJ file and MTL files are important and needed in order to describe the model complete.

### Create Wavefront Format File

The 3Dmodel can be created by using 3D studio Max software, 3D Studio Max is a useful 3D modelling tool that enables user to create a complex 3D model and save it, and eventually export to Wavefront Format File for storing the geometry data of 3D models. Two types of format file will be created when geometry data is exported to Wavefront Format in 3D Studio Max graphic system. First format file is OBJ format file which describes the geometry data. Second format file is MTL format file that include information about the corresponding material and light of the 3D models in the OBJ file.

### Analysis of Wavefront Format File

Data analysis and parsing wavefront format file is a process to conduct performance reviews which we loading an OBJ file model and its material library and passing the data to openGL rendering commands. This section we are going to provide a thorough explanation of the most important part of a wavefront format file.

The OBJ file format supports lines, polygons, and freeform curves and surfaces. Lines and polygons are described in terms of their points, while curves and surfaces are defined with control points and other information depending on the type of curve. Since, in our project, lines and polygons were sufficient to represent the model appropriately.

An OBJ file is a text file consisting of lines of statements, comments, and blank lines.

Each line of the OBJfile starts with one or two letters that indicated what the line for. There are three types of basic OBJ statements: shape-related, those for grouping, and ones for using materials.

Below are listed various token of obj file :

|  |  |
| --- | --- |
| Prefix | Definition |
| ”#” | A comment line. Simply skipped by OBJ file readers. |
| ”g” | A group line.  Determine the start of a group. ”g” token is followed by the group name. |
| ”v” | A shape line, provide the information, concerning vertices.  This letter is followed a single space will be three floating-point values that indicated x, y and z coordinates of the vertex. |
| ”vn” | A shape line. Contain information, concerning the normal of a vertex.  This letter is followed by x, y and z coordinates of the normal. |
| ”f” | A face line, describing face; provide the information, concerning polygons.  This letter follows by a number of a triples which is equal to number of vertices, the polygon has.  Each triple inform v1//n1 v2//n2 v3//n3  where (v1, v2, v3) is the index of vertex position array for each vertex, (n1, n2, n3) is the index of normal vector array for each vertex.  It define a polygon consisted of 3 vertex; mean face is formed by triangular mesh. |
| ”mtllib” | A material files line. The followed by material file name. |
| ”usemtl” | Presumed that the MTL file names and defines various materials.  This token followed by material name.  All subsequence faces should be rendered with this material name, until a new material is invoked. |

Table 5.1 Properties of OBJ file.

An MTL file is a reference file containing definitions of materials that may be accessed by an OBJ file. An MTL file contains a sequence of definitions of materials. Each definition begins with a ‘newmtl’ command that defines the name of the material, followed by lines specifying particular properties.

The quantities that may be defined for a material include:

|  |  |
| --- | --- |
| Command: | Definition |
| **Ka** r g b | The ambient color of the material to be (r,g,b). The default is (0.2,0.2,0.2) |
| **Kd** r g b | The diffuse color of the material to be (r,g,b). The default is (0.8,0.8,0.8). |
| **Ks** r g b | The specular color of the material to be (r,g,b). This color shows up in highlights. The default is (1.0,1.0,1.0);. |
| **d** alpha | Defines the transparency of the material to be alpha. The default is 1.0 (not transparent at all) Some formats use **Tr** instead of **d**. |
| **Ns** *s* | The shininess of the material to be s. The default is 0.0. |
| illum *n* | Denoted the illumination model used by the material. **illum** = 1 indicates a flat material with no specular highlights, so the value of **Ks** is not used. **illum** = 2 denotes the presence of specular highlights, and so a specification for **Ks** is required. |

Table 5.2 Properties of MTL file.

### Loading the OBJ File Format.

We have developed an OpenGL program under Microsoft Visual C++ environment to load OBJ file format. The OBJ file does not contain any sort of header, so we need some sort of resizable array to hold vertices, face and other component. In the implementation shown here, everything is loaded into an STL vector, a kind of resizable array.

Each data type has its own structure. Example Vertex and Normal contain 3float and is used to hold a single vertex position. Face contain 6 unsigned integers, where 6 integers for each vertex indexes, normal indexes. Material contains 1 character for material name, six float and 2 integer to store the information of the material. Last of all is FaceMaterials which contain 3 integers and is used hold the index of first face of the group, last face of the group, and the group ID.  
After the data structures have been created, we can started read OBJ file and parsed it value in the program. The best way to do this is read in one line at time, then separate each string to multi number of the argument, followed by check the prefix of the line and extract the rest of the values using the sscanf function.

If first argument with "v", and takes the following data as 3D coordinate of the vertex (x y and z) and stores the integer values in a class called Vertex, and adds it to an array of Vertex objects.

If first argument with "vn" (Vertex Normal) and takes following data as normal vertex (nx ny nz)and store the integer values in a class called Normals and add it to an array.

If first argument "mtl" ,take the following token as mtlfile name . Then open the MTL file and parse it line by line.

When a “newmtl” argument is encountered, the current Material object is added to the materials array list .

A subtle aspect of the OBJ format is how materials are linked to faces. After a material is named in a “usemtl”, all subsequent faces will formed a group of object and use it for rendering until another “usemtl” line is encountered.

When “usemtl” statement is encounters, it captures the link by passing the current face index and stored in a class called Face.

Then continue reading OBJ file to tracking face index. The data for a single face is stored in two arrays of vertex, and normal indices. If first argument with ‘f’

For example, if the string is:

f 1//2 3//4 5//6

then the vertex indices array will hold {1, 3, 5} and the normal indices array is{2,4,6}.Stored the vertex indices and normal indices into a class called Faces. Combine all the faces index and mapped it with material to form a new class called FaceMaterial. Each class of the FaceMaterial will become one of the part of the model.

### **Standard Template Library (STL) on vector**

The purpose using Standard Template Library (STL) in our program is to make tasks easier; to support faster programming, and avoid error occurred because of largest memory **allocation.** The C++ STL has been thoroughly tested and optimized for performance. There are two types of memory in C++ where there are *memory stack and memory heap*. Allocation for the stack is much faster because it should be used for temporary variables within scopes. Allocating memory on the heap is permanent, but much more expensive and can significantly lead to internal/external memory fragmentation over time if used extensively, which in turn slows down future program execution**.** In case of OBJ files, a vector of a vector of vertices for each face and a geometry of about 40,000 faces, there are need largest of memory space and slows down the program execution. So the best way to store a lot of similar data is using arrays or STL vectors when using C++. STL on vector implements an array with fast [random access](http://en.wikipedia.org/wiki/Random_access) and an ability to automatically resize when appending elements [14]. The STL's are a set of C++ template classes to provide common programming data structures and functions such as list, paired arrays map, expandable arrays (vector), large string storage and manipulation (rope), etc.. By using *struct*  function, we are able to hold some data such as coordinate of the vertices. Besides that, the STL *class* function also available to classing all the vertices in a list of or classing the index of vertex and faces in a group[15].

### Rendering OBJ file:

After the OBJ and MTL file have been read in openGL and all the data is collected and stored, we can start to render the model. We are rendered a model group by group. A array class in FaceMaterial represented a group of information of a face. Use the FaceMaterial to index into the array of the vertices, normal and material. There is one little [obstacle](app:ds:obstacle) here. The creators of OBJ file format are makes 1 as the first index, instead of 0. To counter this we need substract 1 from each of the vertex and normal indexes that make up a face.

Create a new function in program by naming using the part of the object. In the function, after set up the colour for the group’s material, we sets up the vertex buffer for the position, normal and drawn the triangles associated with that group. Then loops over each group by creating a new function.

### **Control each Part of the object**

In this section will explain how to do an animation of the model. According to the research, an animation of the model involving in modelling transformation. The modelling transformations allow manipulating the position and setting the orientation of a model by moving, rotating, and scaling. There three routines for modelling Transformations in openGL are: glTranslatef(), glRotatef(), and glScalef().These routines transform an object by moving, rotating, stretching, shrinking, or reflecting it.

glTranslate(dx dy dz) , which translates by displacement vector (dx, dy, dz),and glRotatef(angle, vx, vy, vz) rotates about axis (vx, vy, vz) by angle specified in degrees.

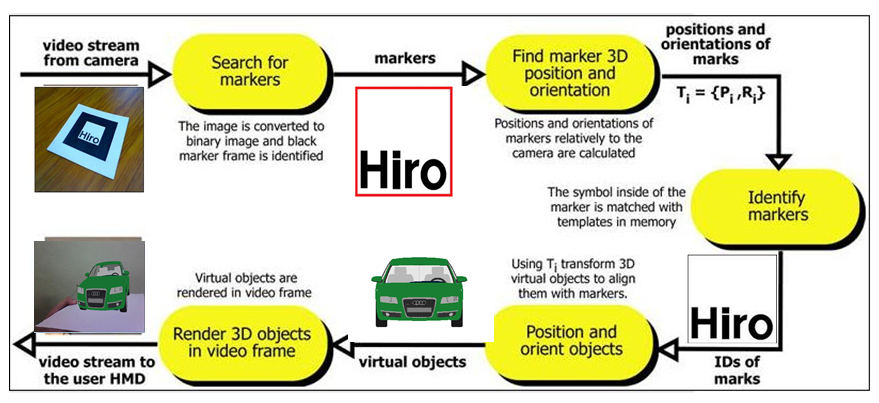
To able control each part of the model; this operation is performed by using keybroad Callback Function. The keybroad callback function is called whenever press a key. The following step modify the parameter has been assigned in the function, by increase or decrease the value of the parameter.

### Understanding how ARtoolkit works.

According to the research, ARToolKit applications allow virtual imagery to be superimposed over live video of the real world. Basic of computer version of ARToolkit method used as tracking marker. The ARToolKit tracking works as follows:

1. The camera captures video of the real world and sends it to the computer.
2. Software on the computer searches through each video frame for any square shapes.
3. If a square is found, the software uses some mathematics to calculate the position of the camera relative to the black square.
4. Once the position of the camera is known a computer graphics model is drawn from that same position.
5. This model is drawn on top of the video of the real world and so appears stuck on the square marker.
6. The final output is shown back in the handheld display, so when the user looks through the display they see graphics overlaid on the real world.

ARToolKit is able to perform this camera tracking in real time, ensuring that the virtual objects always appear overlaid on the tracking markers.

Figure 5.1 Finalize and summarizes the steps above .

## Our Project

### Created simple OBJ file and MTL FILE

In the beginning of our project, we created a very simple model(2 cube) as our first in 3D studio Max software and exported it to wavefront format file by naming as “test.obj” and “test.mtl”.

# Max2Obj Version 4.0 Mar 10th, 2001

mtllib test.mtl

# object Box01 to come ...

v -19.326 0.000 22.472

v -2.717 0.000 22.472

:

vn 0.000 -1.571 -0.000

vn 0.000 -1.571 -0.000

:

g Box01

usemtl material\_for\_box1

f 1//1 3//3 4//4

f 4//4 2//2 1//1

:

g Box01

usemtl material\_for\_box2

f 2//2 4//2 5//3

Table 6.1 sample of .obj file ’test.obj’

# Max2Mtl Version 4.0 Mar 10th, 2001

#

newmtl material\_for\_Box1

Ka 0.24 0.02 0.80

Kd 0.24 0.02 0.80

Ks 0.9 0.9 0.9

d 1.00

Ns 0.00

illum 2

#

newmtl material\_for\_Box2

Ka 1.00 0.05 0.67

Kd 1.00 0.05 0.67

Ks 0.90 0.90 0.90

d 1.00

Ns 0.00

illum 2

#

# EOF

Table 6.2 sample of .mtl file ’test.mtl’

### Parsed simple OBJ file and MTL file into openGL

After analysis both of the wavefront formats file. We try to load the “test.obj” file and it material file “test.mtl” into Microsoft visual C++ program.

First of all, we created a classic data structures implemented using Standard temple library vector to store data. In a class,

|  |  |
| --- | --- |
| Array Class | Structure |
| Vertex | coordinate that contain 3 float variable (x,y,z) |
| Normal | coordinate that contain 3 float variable (x,y,z) |
| Face | 6 unsigned integers (v1, v2, v3, n1, n2, n3) used to hold each vertex indexes and normal indexes. |
| Material | A character for material name ,  4float variable to holds the ambient color, diffuse color , specular color , and alpha value. |
| FaceMaterial | 3 integers to hold the first face index, last face index and the group ID. |

Table 6.3 the data structures in program.

The name of .objfile will called in int() function.   
Then we are created a new function to parse data of .obj file and .mtlfile, by naming as load\_model(). We passed the .objfile name(test.obj) form int() function. Then we passed the data file to load\_model() function.

In the function we open .obj file and start to parse .obj file line by line.   
When read every line of .obj file, separate each line from string to multi number of argument, then check first argument as prefix of the line to determine the content of the line. In this function we are divided to several cases, assigned the condition of the each case by using prefix of each line.

Case 1:   
If first argument is ‘mltlib’, which tells us the name of the material library file. We parsed the second argument to mtlfile, and open MTL file then parse the content of the mtlfile.

In a material library file, we known that the ‘newmtl ’ prefix is the begins a new material definition. We are parsed the information after ‘newmtl’ line where consist of ambient color, diffuse color, specular color, and alpha value the float number and stored in material arraylist.

Case2:  
If the prefix consist of geometry symbol(v, vn), it is parsed to a tuple of float values and stored in the appropriate array. For instance,

The line

v -19.326 0.000 22.472

it would be parsed to the tuple (-19.326, 0.000, 22.472) , stores the float values in a coordinate struct and appended to an array of vertex class.

The line,

vn 0.000 -1.571 -0.000

it would be parsed to the tuple (0.000,-1.571,-0.000), stores the float values in a coordinate struct and appended to an array of vertex normal class.

Case3:   
If the first argument is equal ‘usemtl’, which tells us which material subsequent face will use. In this case, we are review the material name in material class and mark it to objectID, then created a new FaceMaterial group to group the objId and the face information that will follow.

Face are defined with the ’f’ prefix, we parse the information of the face line which consist of a word for each vertex in the face 3 for triangle. Each word contains indices into the vertex, and normal lists, separated by a forward slash character (/). For instance,

f 1//2 3//4 5//6

These triplets of indices is parsed to integer value are store in the current Face array list and will be used to reconstruct the model shape when we come to render it. All the faces in usemtl is counted, so that we can realized the number of face in a FaceMaterial group.

Case 4:

If the prefix is “g”, it indicates us that the following argument is the name of the part model. We are print out the name of the part model. While printed out the name of part model, we also numeration the name of the part model. The number of name of the part model will match with the index of the FaceMaterial array list.   
This loadmodel() function are enough to read all the information we need from OBJ file and mtl file, and we able write the code to send the geometry to openGL.   
We use the FaceMaterial array list and Face array list to draw the model.  
A FaceMaterial group will become one of the part of the model. This mean that a index number of the FaceMaterial array is a part of the model. However, Index of the Face array is the vertex of each face. By using this two index and openGL draw function our first model are success render.

### Replace complicated model in openGL

The purpose of using 2cube as our first model is ensured that our program is work to parsed objfile, this is because of 2cube contain seldom data. Then we try to loader complex model where we using car as our model in this project. Similar with simple model, complex model is created in 3Dmax studio and export to objfile with the name ‘car.obj’.

This car model is formed by 230face. After we make sure car model is able render in openGL library, then we create many new function to separate every part of the model by identified using FaceMaterial array. The several functions have been created to draw the part of the model is antenna(), bumper(),type(), hood(),door(), window(), mirror(), headlight(), foglight(), taillight(), door1(), door2(), Interior(), Exhaust(),Trunk(), Grilles(), Roof(), statical().All of the function, will called in car\_model() function.

### Render model in Augmented Reality Environment

After find section part 6.2, 6.3, we import our source code to ARToolkit program. A perfect 3D car model will show in the screen by using camera and marker as input device.

### Navigate the model in 3D space

In this section we will discuss about how to control each part of model by using keybroad.

By using the glTranslatef() and glRotatef() function which provided in openGL library in the every part of the model function where we are discuss in part 6.3.

To control the motion of each part model, we are assigned variable as parameter in glTranslatef(), and glRotatef() function, and the figure of the variable will change by using keybroad function.

For instance, we use pressing button ‘a’ , increased the figure of the variable *to Z* in the function glTranslatef(0,0,toZ); then the position of antenna shift to upside.

The following table show the key to control which part of the car model.

|  |  |
| --- | --- |
| Button | Part of the car |
| A | Antenna |
| B | Bumper Molding |
| C | Component of type |
| D | Hood and Fender |
| F | Door and Window |
| G | Side mirror |
| Q | Headlight |
| R | Foglight |
| S | Taillight |
| T | Roof |
| U | Grilles |
| V | Interior |
| W | Exhaust |
| X | Trunk |
| 1 | Restore the all figure of the transformation function to intiatial value |
| ESC | exit |

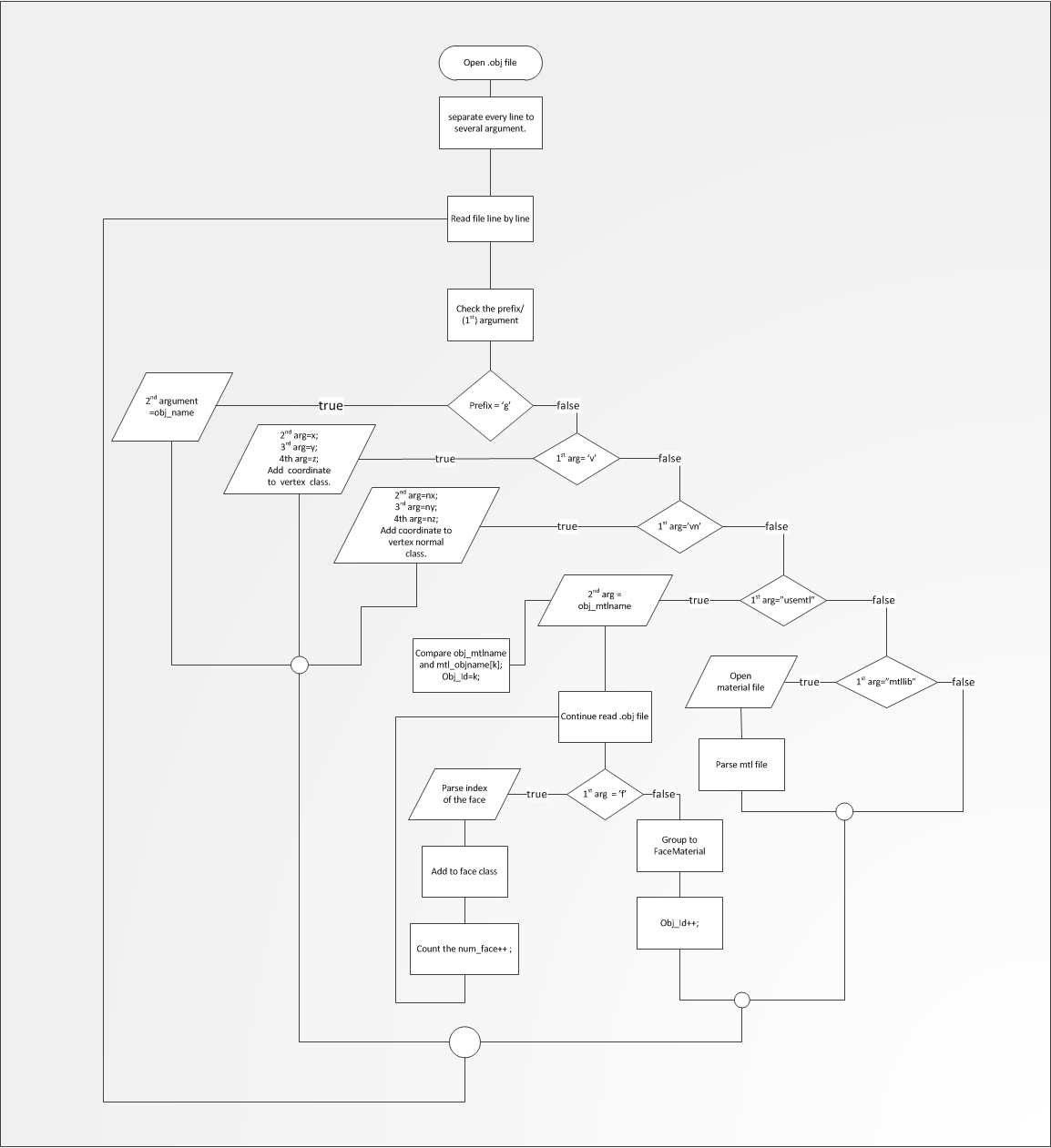
Table 6.4 The key to control each part of the model

### Flow of project a) Flow of whole project

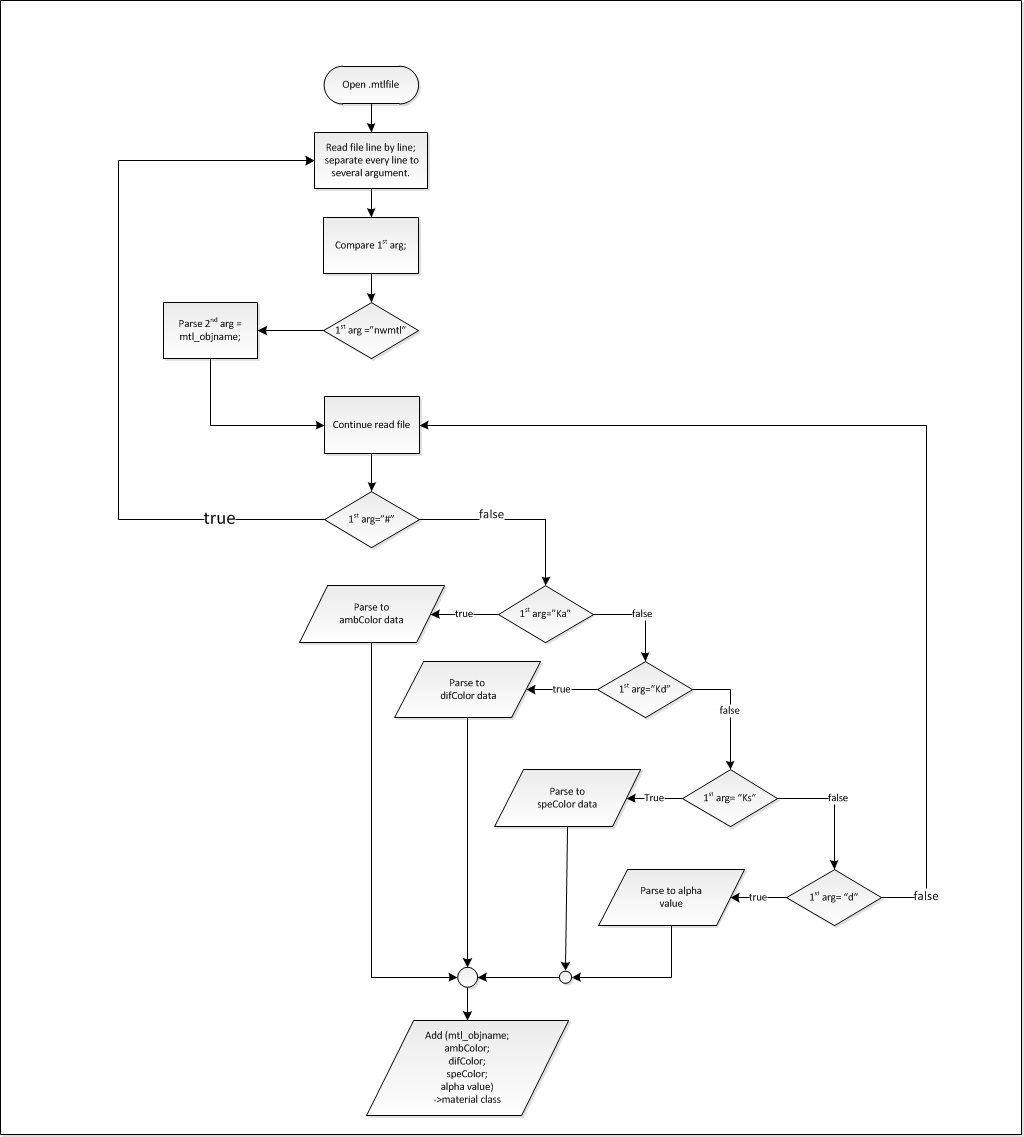
### 

Figure 6.1 Flow of project

b) Flow of parsing .obj file.

Figure 6.2 Flow of parse .obj file

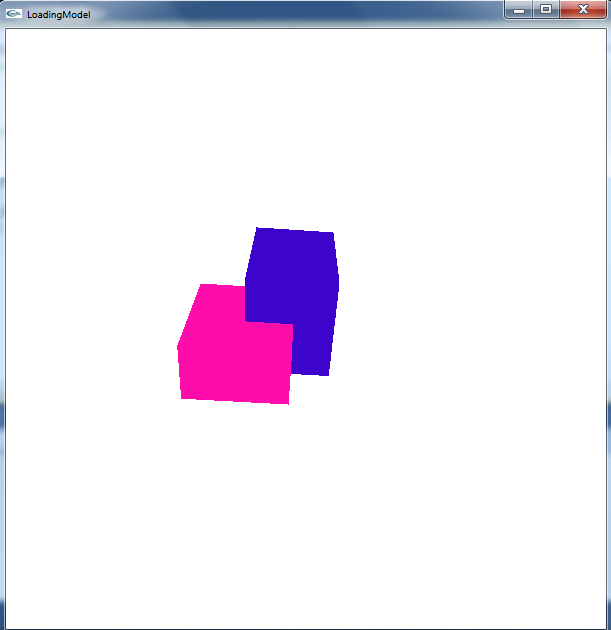
c) Flow of parsing .mtl file

Figure 6.3 Flow of parsing .mtl file

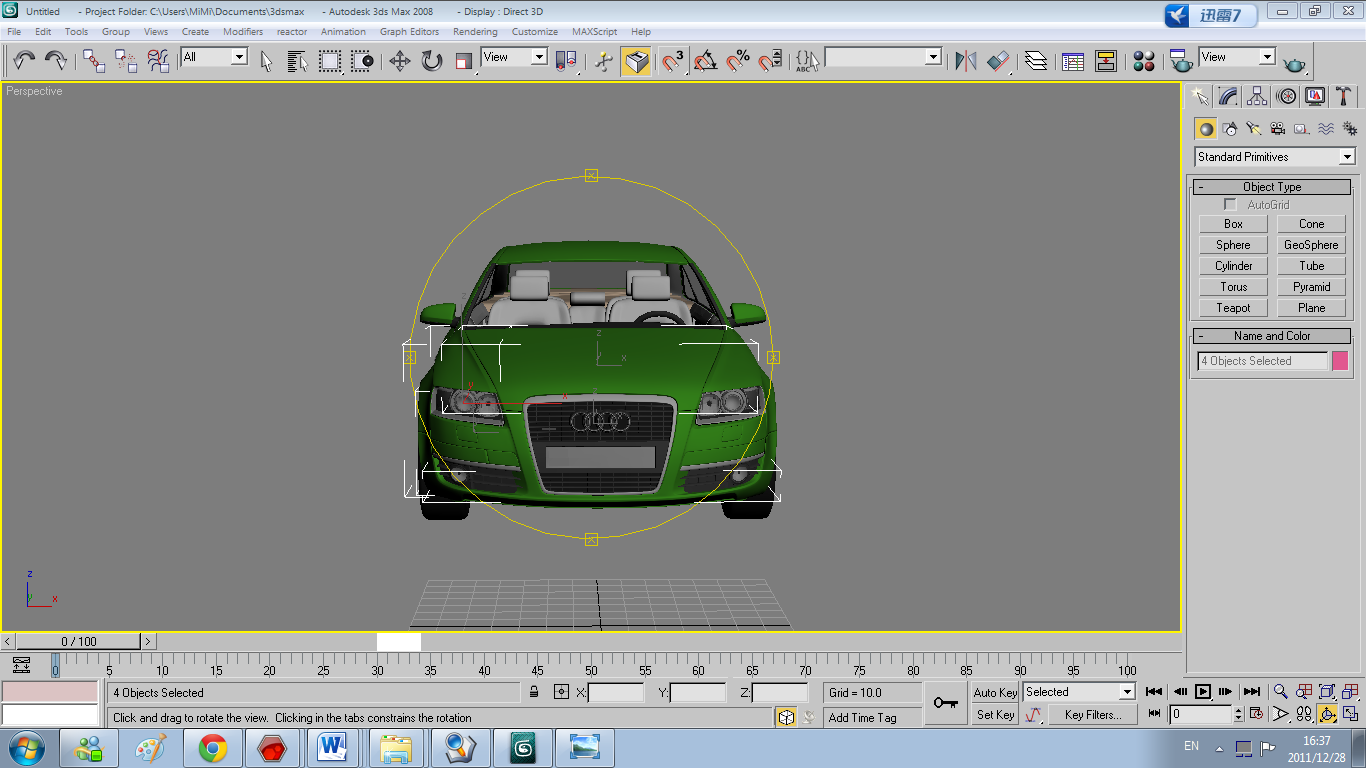
## Result

Finally, we are able to display our model in augmented Reality environment and control each part of the model. Below are some screen shots of our result:

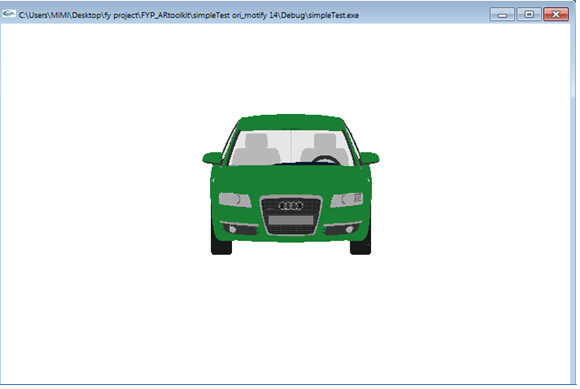
1. first result where simple model was parsed is OBJ file and render in openGL.

  
Figure 7.1 A simple model is rendered on openGL window.

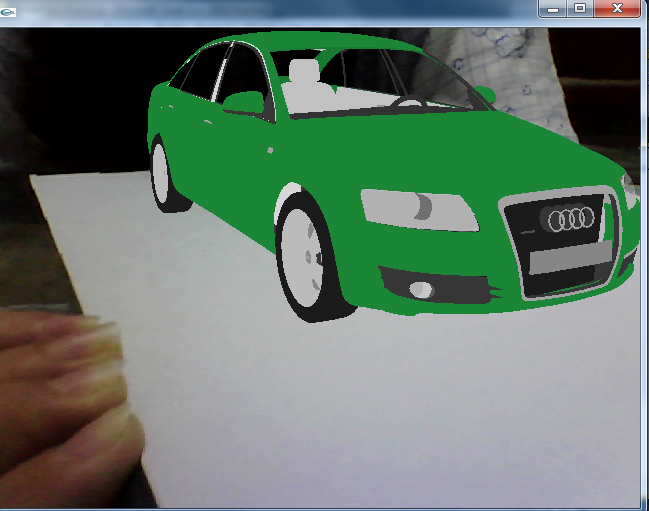
1. Our project car model is created in 3Dmax Studio.

  
Figure 7.2 Model created in 3D studio Max

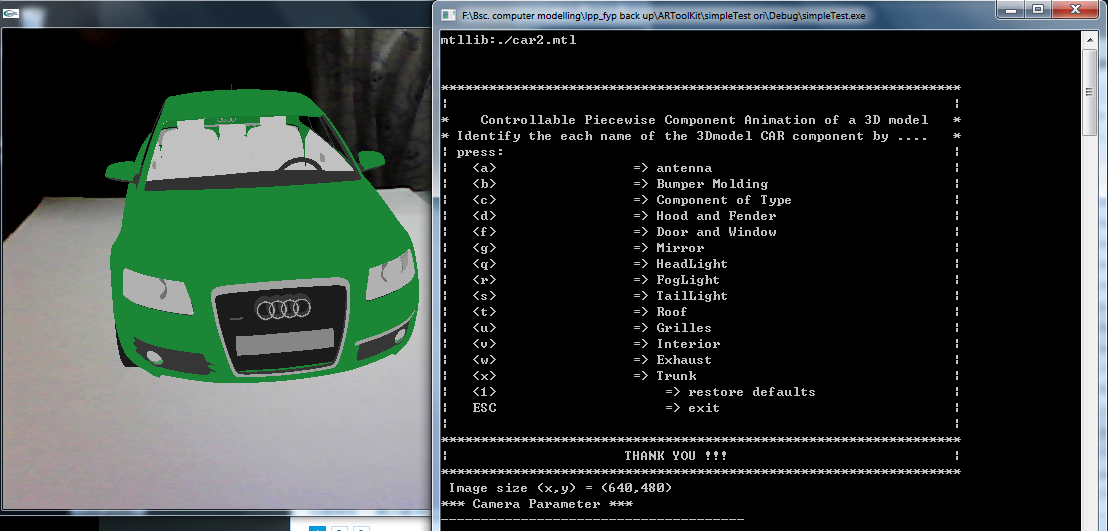
1. After created 3Dmodel in 3Dstudio max, we were rendering our model in openGL window.

Figure 7.3 Model rendered in openGL window.

1. Then we were doing some analysis and executed our program in ARToolkit so that our model is able displays in augmented reality environment.  ****Figure 7.4 Model display in Augmented Reality environment.
2. Model can easy view different view point by rotating marker.

*~~~~*  
Figure 7.5 Model display in Augmented Reality environment different point of view of the model in AR environment by rotating marker .

1. After done animation of the car model, user able identified each part of the model by following indication panel.

Figure 7.6 Model and indication panel is displays on the screen.

1. Example of moving the parts of the model



Figure 7.7 control all the part of the model.

## Problems encountered

### The size of the array

To parsing a typical ASCII-base format of Wavefront OBJ data file we are using 3D dimensional list of array where there is a list of array vertices, list of vertex index that used for each face and list of group that included the face index which mapping with material.

In the end all will be converted into one dimensional array of floats. The problem is that this file is huge and loading it as strings and processing takes huge amount of time. Some time it leaded error when loading file. Besides that to parsing ASCII file is extremely slow.

### Parsing data

When parsing .obj file format file in to Microsoft visual C++ program, we can’t simple set an integer equal to a string containing an integer in the text format and expect it to work as we thought.

## Solution

### Using sscanf () function to extract data.

We need extract the information which needed from OBJ file into program. In C++ program is pretty strict about the data types. So we using special function called sscanf() for extracting data from string and stores them according to the parameter format into the locations given by the additional arguments. The very first parameter for sscanf() is a string that holds the unformatted data. The second parameter is holds a string called the format string. The format string tells sscanf() which types of variables to look for and in what order they are. The last variable is pointer to the variables want to store and converted values. When converted string value, we need separate pointer for each value read in, because the sscanf() function take a variable number of argument, so that it can read in as many values at one time .

For instance, the vertex line is OBJ file is

v -10.341 0.000 14.575,

We were parsed the line in OBJfile to several argument in char data type.   
*“int num\_args= sscanf(line,"%s%s%s%s",arg[0],arg[1],arg[2],arg[3]);”*

Then parsed the x, y, z coordinate of the vertices, from char data type to float number, such as ,

*“sscanf(arg[1],"%f",&x);sscanf(arg[2],"%f",&y);sscanf(arg[3],"%f",&z);”*  
where “%S” indicate a string of characters, will read subsequent characters until a whitespace is found; and %f indicate float number is going to parse.

### Using vectors in Standard Temple Library to stored data.

In our program, we replace array to STL vector. Vectors are a kind of sequence container that allows to storage pretty much element. As such, their elements are ordered following a strict linear sequence. When used correctly that can be very powerful containers.

Vector containers are implemented as dynamic arrays. Just as regular arrays, vector containers have their elements stored in contiguous storage locations, which means that their elements can be accessed not only using iterators but also using offsets on regular pointers to elements. But unlike regular arrays, storage in vectors is handled automatically, allowing it to be expanded and contracted as needed. Vectors are good at accessing individual elements by their position index, iterating over the elements in any order and add and remove elements from its end[17].  
For example, the data of OBJ file, the vertex index(position index) which contains 3 float number of coordinate(element) are iterate in face list. So we created a class called vertex. We offsets vertex *class* on regular pointer to a *struct* called coordinate which storage x,y,z coordinate of the vertex.  
 *declaration: std::vector<coordinate\*> vertex;*

Besides that, we are using *push\_back()* function which obtain from STL vector to adds a new element in vertex class.

## Achievement for the 1st semester

|  |  |
| --- | --- |
| Month | Task Completed |
| September 2011 | * Get used to programming environment of Microsoft Visual C++. * Learn openGL and decide topic for final year project . |
| October 2011 | * Understand background of Wavefront format file and parse it of C++ program. |
| November 2011 | * Write algorithm to control piecewise component of model by using keyboard. * Compiled ARtoolKit. |
| December 2011 | * Write final year preview report |
| January 2012 | * Doing presentation slideshow for Viva. * First Viva |

## Achievement in 2nd semester

|  |  |
| --- | --- |
| Month | Task Completed |
| February 2012 | * Improve animation of the model by using method interaction between two marker. |
| March 2012 | * Develop a new functionality by naming each part of the model, to allow user identified the part of model. |
| April 2012 | * Write final year report. |
| May 2012 | * Prepare presentation slideshow for final Viva |
| Jun 2012 | * Final Viva |

## Conclusion and future work

As a conclusion, in our project we able to displayed a 3Dmodel and control each part of the model in augmented reality environment. The primary advantage of our project is to provide a flexible interface that vision how the feature of 3Dmodel described in quickly and easily way. User just need created a model in 3Dstudio Max software , and save it into OBJfile, setup the filename in program, print out the marker and turn on camera ,then a visual model will simply displayed on the screen without a lot of computer skill and training. Neither further libraries nor hardware devices are required. Besides that, we created a system in our project so that it able to uses as teaching material for displays conceptual idea that are difficult to interoperate on paper such as *Cell Structure* and allows students to identify every part of structure. In addition, our project able becomes amazing new advertisement for uses augmented reality technology to create a truly interactive media piece out of a 2-dimensional magazine.

We hope that augmented reality system will be utilized more by a variety of user. After completed our project , we found that our project can extended or modify to meet other purposes. For example, it can be uses to create a magic book by adding multiple of marker as page of the book and setup few button such as previous button and next button for reader to navigate the book. Furthermore, we also can improve our project and turn it into a quickly design prototypes. This means that we need provides a new method by adding new model in the program. In this system user can realize that position where intend to put a new model and give appropriate suggestions. This system can used in some scenarios like furniture arrangement, colour and texture customization to meet users specifies needs.

## References

[1] <http://en.wikipedia.org/wiki/3D_modeling>

[2] Trien V.Do and Jong-Weon Lee “3DARModeler : a 3D Modeling System in Augmented Reality Environment”, World Academy of Science, Engineering and Technology 2010

[3] <http://en.wikipedia.org/wiki/Wavefront_.obj_file>

[4]http://www.squidoo.com/3d-models  
[5] <http://www.jstatsoft.org/v36/i08/paper>

[6] Enrico Glaa , Jonathan M. Garibald and Natalio Krasnogo “An R Package for 3D Data Visualization on the Web”, University of Nottingha, 2010.

[7] <http://www.ibrtses.com/opengl/fileformats3d.html>

[8] Brett E. Shelton and Nicholas R. Hedle “Using Augmented Reality for Teaching Earth-Sun Relationships to Undergraduate Geography Students” University of Washington,2002.

[9] Augmenting the Science Centre and Museum Experience

[10] Book, supporting learning Flow through Intergrative Technologies T.Hirashma al.(Eds.) Ios 2007.

[11]<http://ingaldojbmad.blogspot.com/2009/07/3d-conversor-de-archivos-polytrans.html>

[12] en.wikipedia.org/wiki/Augmented\_reality

[13] http://www.hitl.washington.edu/artoolkit/

[14] <http://en.wikipedia.org/wiki/Sequence_container_%28C%2B%2B%29>

[15] [http://www.yolinux.com/TUTORIALS/LinuxTutorialC++STL.html#VECTOR](http://www.yolinux.com/TUTORIALS/LinuxTutorialC++STL.html" \l "VECTOR)

[16] <http://www.cplusplus.com/reference/stl/vector/>