



Jaypee University of Information Technology
Solan (H.P.)
LEARNING RESOURCE CENTER

Acc. Num. *SP05049* Call Num:

General Guidelines:

- ◆ Library books should be used with great care.
- ◆ Tearing, folding, cutting of library books or making any marks on them is not permitted and shall lead to disciplinary action.
- ◆ Any defect noticed at the time of borrowing books must be brought to the library staff immediately. Otherwise the borrower may be required to replace the book by a new copy.
- ◆ The loss of LRC book(s) must be immediately brought to the notice of the Librarian in writing.

Learning Resource Centre-JUIT



SP05049

GAME BASED ON BLEACH ANIMATED SERIES

**Submitted in partial fulfilment of the
Degree of Bachelor of Technology**

By:

ROHIT SHARMA 051070

PRANIT SUNEJA 051071

This is to certify that the project titled "Game Based on Bleach Animated Series" submitted by Rohit Sharma and Pranit Suneja in partial fulfillment for the award of degree of Bachelor of Technology in Communications Engineering of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

(Prof. Dr. Sunil Khoscharan)



JULY 2008-MAY 2009

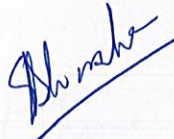
**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY-WAKNAGHAT

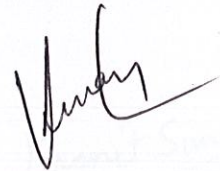
ACKNOWLEDGMENT

CERTIFICATE

This is to certify that the work entitled, "Game Based on Bleach Animated Series" submitted by Rohit Sharma and Pranit Suneja in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communications Engineering of Jaypee University of Information Technology has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.



[Prof. Dr. Sunil Bhooshan]



[Dr. Vinay Kumar]

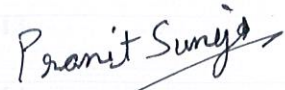
ACKNOWLEDGMENT

We would like to express our gratitude to **Prof. Dr. Sunil Bhooshan**, Head of the Department, Electronics and Communication JUIT, Wagnaghat, for encouraging and inspiring us to carry out the project .

We would also like to thank our guide, **Dr. Vinay Kumar**, Lecturer, Dept. of Electronics and Communication for his expert guidance, encouragement and valuable suggestions at every step.

We also would like to thank all the faculty members of E&C dept. for providing us with lab facilities and support towards the completion of the project.


[ROHIT SHARMA]


[PRANIT SUNEJA]

CONTENTS

List of Figures	i
1 Introduction	1
2 Roadmap	2
3 Game Design	3
4 Windows Platform	4
5 Visual C++	5
6 Creating Arena	6
7 Modelling with 3DS Max	10
8 Texturing with Adobe Photoshop	13
9 Rendering Animations to give effects	14
10 Loading final Models and Animations	15
11 Giving Sprites and other Resources	16
12 Compiling all Resources	17
13 Bibliography	18

LIST OF FIGURES

- Figure 1 DirectX format of the model
- Figure 2 Various components during compilation
- Figure 3 Height map of Arena
- Figure 4 Texture image of Arena
- Figure 5 Detail image of Arena
- Figure 6 DirectX of Skybox model
- Figure 7 Final visualisation of Arena
- Figure 8 Modelling in 3DS Max
- Figure 9 Representation of working model
- Figure 10 Final model visualisation
- Figure 11 Texturing of uv template in Adobe
- Figure 12 Texturing of model
- Figure 13 Walking animation of the model
- Figure 14 Fighting animation of the model
- Figure 15 Sprites animation in the arena
- Figure 16 Screenshot of the final game

INTRODUCTION

The course final project is to design and implement a complete game from the ground up based on BLEACH animated series. The game will follow the storyline including the main aspect as fighting between our hero Ichigo and hollows.

The series BLEACH opens telling how a teenager called Kurosaki Ichigo is given Soul Reapers power by Rukia. As he assumes the duties of a Soul Reaper, Ichigo has to fight against giant creatures called hollows which are the souls of people who are not able to rest in peace after death. As he faces several challenges, some of his friends awaken their own powers due to being close to him and the fact that his spirit pressure is constantly leaking. However Soul Society, send Soul Reapers to arrest Rukia for giving her powers to Ichigo and prepare to execute her. Unwilling to let her die, Ichigo and his friends go to rescue her from Soul Society, causing them to infiltrate Soul Society's inner walls and face many Soul Reapers. This causes the unrevealing of many mysteries and leads to the rebellion of Aizen, a former Soul Reaper captain, and the formation of his own army of hollows with which he will attack Soul Society.

The Game will be sent in the outer districts of Heuco Mondo. Here Ichigo is trapped and surrounded by Menos which are his enemies. He has to pass through four blocks of varying difficulty in order to find the path back to the real world.

ROADMAP

Our basic idea was to use Dark GDK to create our arena and then importing 3d models with synchronisation and user control. Thus our roadmap was as follow:

1. Our first step was to decide upon the type and storyline for the game.
2. Decided to build our game on windows platform.
3. Understood windows programming and concepts of VC++.
4. Built Arena with DARK GDK.
5. Creating models with Autodesk 3 DS Max and loading them on our map.
6. Texturing of models with Adobe Photoshop.
7. Rendering animations to give effects to the characters and map.
8. Loading our final models and animations on our map.
9. Giving sprites and other resources to our game.
10. Compiling all resources to create the game.

1 GAME DESIGN

Deciding upon the storyline we took Bleach Animated series as our basic idea and type behind our game. Our game is first person type providing the user control of our central character Ichigo. Our game will focus on fighting between Ichigo and many Hollows in outer districts in Heuco Mondo where Ichigo is trapped and he has to find the key to unlock the door to come back to the earth.

Ichigo has his sword as his weapon and can fire big attack and small attacks. Big attack will be fired from the sword to give more damage at range while small attack consist of series of sword movement giving direct hit from his sword. Big attack can only be fired when a minimum amount of soul power is there and this attack will consume ample amount of soul power.

Hollows will be there protecting the key and will only fire long range attack. Ichigo has to defeat these hollows to find the key in the map. He will find many items on the way which will allow Ichigo to heal himself and to regain his soul power.

The player will be able to move in up, left, right, back directions and will be able to jump and run. A combination of keys would allow the user to fire big attacks. The player would be able to see his life and soul power on the top left corner of the screen.

2 WINDOWS PLATFORM

We decided to use windows platform due to the following reasons:

1. Ease of access: We have been using this platform for a long time and are comfortable with it. It is user friendly and it has support for graphics through the COM object model.
2. Compatibility: Most of the software's available are built on the windows platform. We decided to use Visual C++ Express Edition 2008, Dark GDK, Autodesk 3DS Max 2008, Adobe Photoshop CS3 and Adobe Audacity.
3. Most widely used: The windows platform is used by a lot of programmers to build games and support is easily available as compared to other operating systems.
4. DirectX: It's basically a system of software that abstracts video, audio, input, networking, installation, and more, so no matter what a particular PC's hardware configuration is, the same code can be used. We will also use DirectX to import our 3D models from 3DS Max in .X format. The DirectX format for our model is shown below:

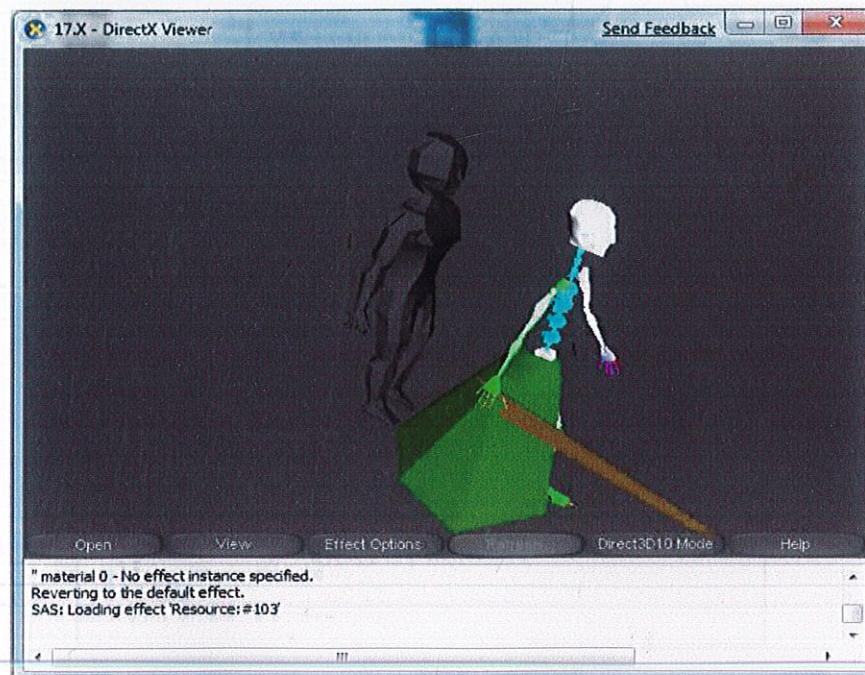


Figure 1

3 VISUAL C++

We are using Visual C++ 2008 express edition as our basic tool to create our game. Visual C++ is a very powerful package but, as is usually the case with powerful software, it is also very complex. We will be using VC++ to compile our code. Dark GDK is compatible with it and we can also use DirectX components through it.

The following figure tells us about the various components that are connected or required while compiling:

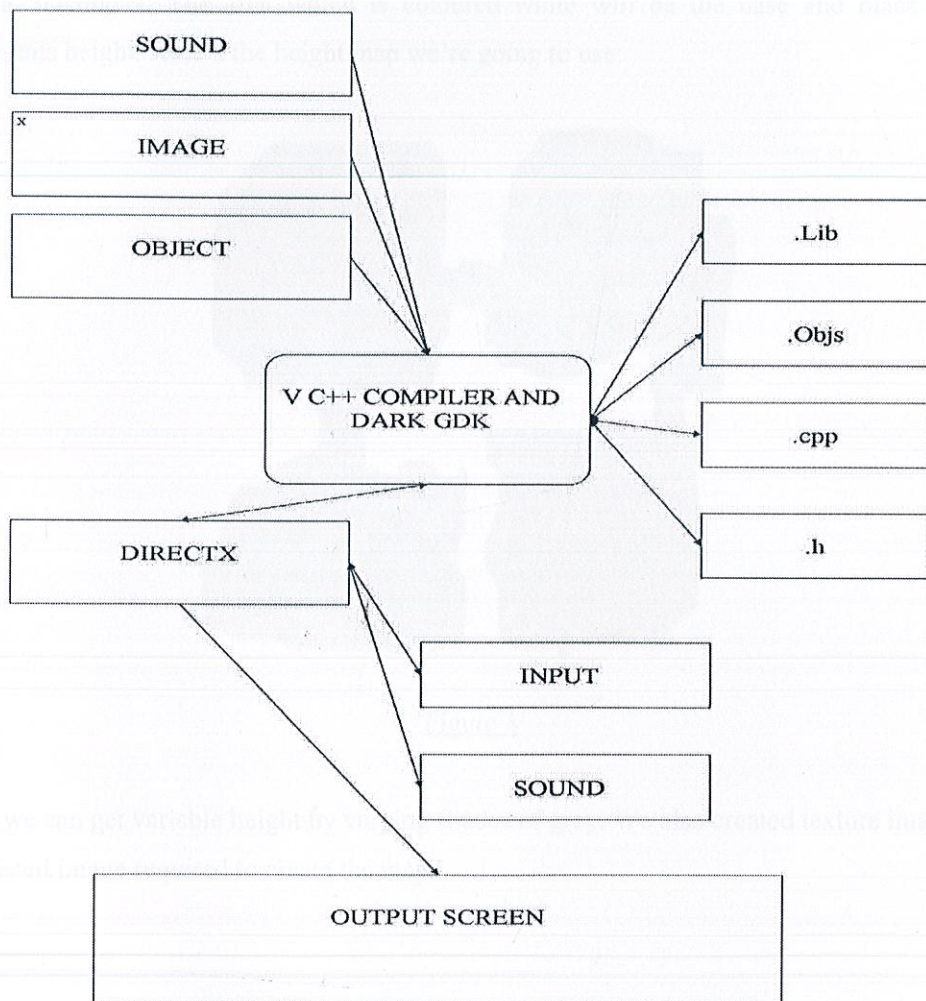


Figure 2

4 CREATING THE ARENA

Dark GDK is a game development kit which consists of various functions which give us more flexibility to create the game. The functions built in the Dark GDK basically encapsulate various Visual C++ graphics and DirectX functions.

We decided to create the map using Dark GDK, in which there are four blocks. In order to create a terrain we need to supply a height map image. A height map is used as a way of representing our terrain in a 2D image. This is then imported into our world and transformed into a 3D object. The area which is coloured white will be the base and black colour represents height. Here's the height map we're going to use:

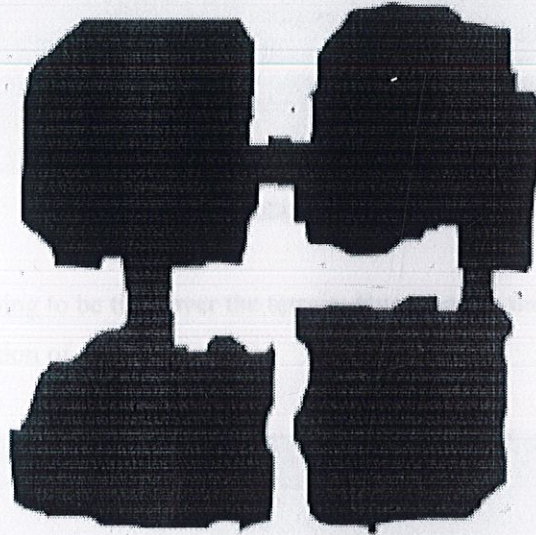


Figure 3

Thus we can get variable height by varying shades of gray. We also created texture image and detail image required to create the map.

Texture image is going to be stretched over the terrain we created and will provide the colour information for the map.



Figure 4

The detail image is going to be tiled over the terrain. By repeating this texture over the terrain we can create the illusion of detail.



Figure 5

Some Functions used to create the terrain:

dbSetupTerrain

It is necessary to call this function before making any terrains. This function is used as a method of setting up some internal settings.

dbMakeObjectTerrain

The initial creation of a terrain object starts with a call to this function. It takes an ID number and this ID number is shared with those from the Basic3D function set.

dbSetTerrainHeightMap

This function and others that set properties of a terrain can be called after the terrain has been made using dbMakeObjectTerrain. This particular function takes an ID number and a pointer to a string.

dbSetTerrainScale

When a terrain has been created its initial scale may not conform to the results you desire. This can be easily controlled by calling dbSetTerrainScale. This function takes three parameters that allow you to control how the terrain is scaled on the X, Y and Z axis.

dbSetTerrainTexture

This function is used to control which textures are applied to the terrain. It takes three parameters. The first is an ID number for the terrain while the second is an ID number for a diffuse image and the third is an ID number for a detail image.

dbBuildTerrain

Once all properties of a terrain have been set you can build it. This is the stage when everything will come together and be inserted into the world.

We also created sky box through DirectX to give the appearance of the sky and loaded it into the programme through the following functions:

dbLoadObject

dbSetObjectLight

dbScaleObject

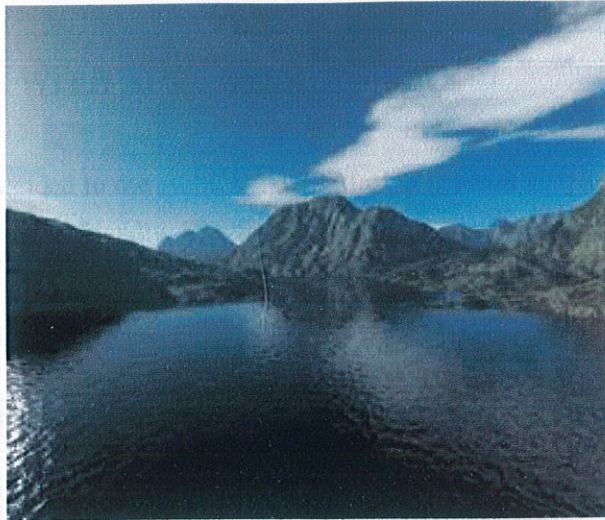
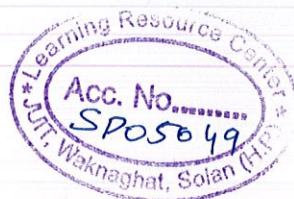


Figure 6

A model is loaded. This model is a box with textures on to represent the sky. Finally we call the dbUpdateTerrain () function. This function does not have any parameters. It is responsible for going through all terrains and updating some internal information. Finally our Arena looks like this:



Figure 7



5 MODELLING WITH AUTODESK 3DS MAX

For modelling we decided to use Autodesk 3DS Max. It is much simpler than Maya software and its GUI is more user friendly. The first step in modelling was to create the torso of the character. Next we created legs from the same object. Here are some of the working images of the very first model we created:

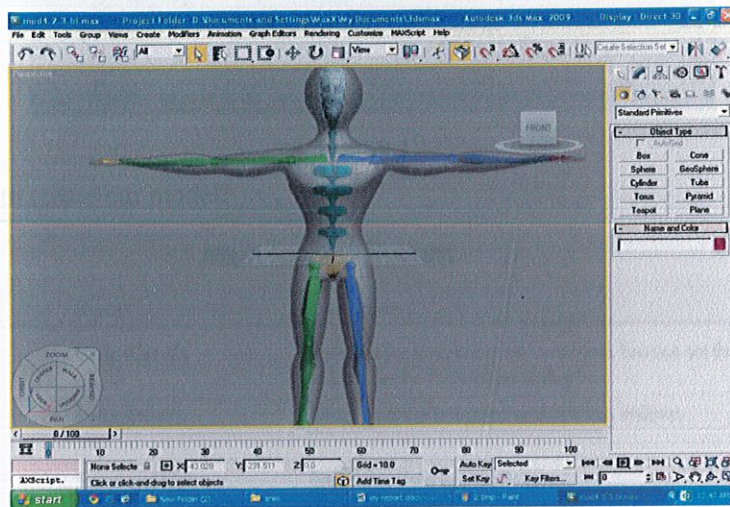


Figure 8

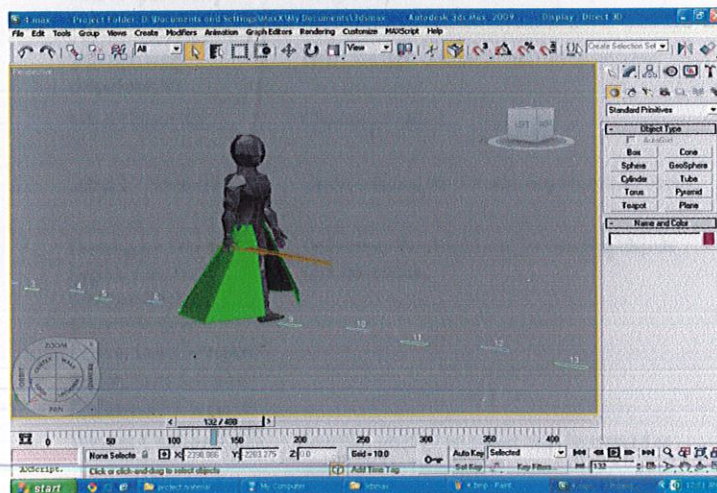


Figure 9

Here is the image of the final model we created through 3DS Max:

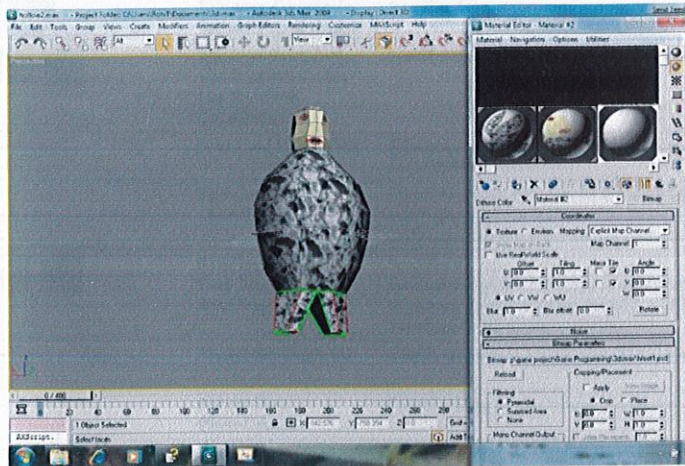








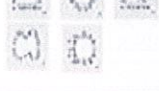






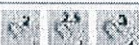













Figure 10

Toolbox used to create our model:

Main Toolbar Buttons

Toolbar Button	Name	Description
	Undo (Ctrl+Z)	Removes the last performed command. You can set the levels of Undo in the Preferences dialog box.
	Redo (Ctrl+Y)	Brings back the last command that was undone.
	Select and Link	Establishes links between objects.
	Unlink Selection	Breaks links between objects.
	Bind to Space Warp	Assigns objects to be modified by a space warp.
	Selection Filter dropdown list	Limits the type of objects that can be selected.
	Select Object (Q)	Chooses an object.
	Select by Name (H)	Opens a dialog box for selecting objects by name.
	Rectangular Selection Region, Circular Selection Region, Fence Selection Region, Lasso Selection Region, Paint Selection Region (Ctrl+F to cycle)	Determines the shape used for selecting objects with the mouse.
	Window/Crossing Toggle	Specifies whether an object must be crossed or windowed to be selected.
	Select and Move (W)	Selects an object and allows positional translations.

	Reference Coordinate System drop-down list	Specifies the coordinate system used for transforms.
	Use Pivot Point Center, Use Selection Center, Use Transform Coordinate Center	Specifies the center about which rotations are completed.
	Select and Manipulate	Selects an object and allows parameter manipulation via a manipulator.
	Keyboard Shortcut Override Toggle	Allows keyboard shortcuts for the main interface and the active dialog box or feature set to be used when enabled. Only main interface shortcuts are available when disabled.
	Snap Toggle 2D, Snap Toggle 2.5D, Snap Toggle 3D (S)	Specifies the snap mode. 2D snaps only to the active construction grid, 2.5D snaps to the construction grid or to geometry projected from the grid, and 3D snaps to anywhere in 3D space.
	Angle Snap Toggle (A)	Causes rotations to snap to specified angles.
	Percent Snap (Shift+Ctrl+P)	Causes scaling to snap to specified percentages.
	Spinner Snap Toggle	Determines the amount a spinner value changes with each click.
	Edit Named Selection Sets	Opens a dialog box for creating and managing selection sets.
	Named Selection Sets drop-down list	Lists and allows you to select a set of named objects.
	Mirror Selected Objects	Creates a mirrored copy of the selected object.
	Align (Alt+A), Quick Align, Normal Align (Alt+N), Place Highlight (Ctrl+H), Align to Camera, Align to View	Opens the alignment dialog box for positioning objects, allows objects to be aligned by their normals, determines the location of highlights, and aligns objects to a camera or view.
	Layer Manager	Opens the Layer Manager interface where you can work with layers.
	Open Curve Editor	Opens the Function Curves Editor.
	Open Schematic View	Opens the Schematic View window.
	Material Editor (M)	Opens the Material Editor window.

6 TEXTURING WITH ADOBE PHOTOSHOP

Individual polygons are selected and uvw unwrap modifier is applied upon the constructed model. The unwrap image is loaded into adobe photoshop for texturing purpose. Following are some of the screenshots while performing texturing:

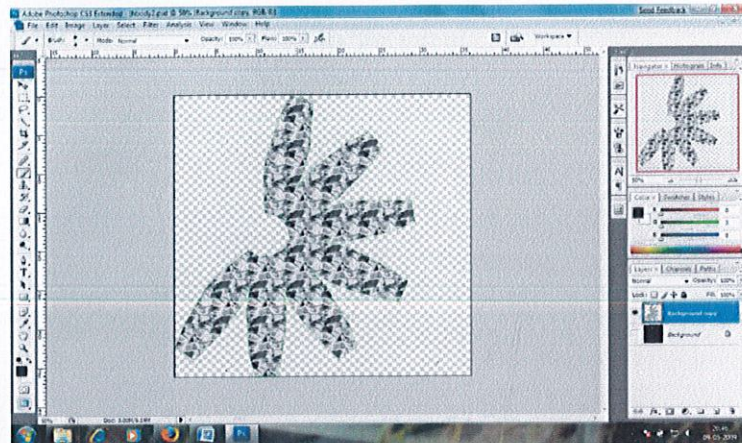


Figure 11

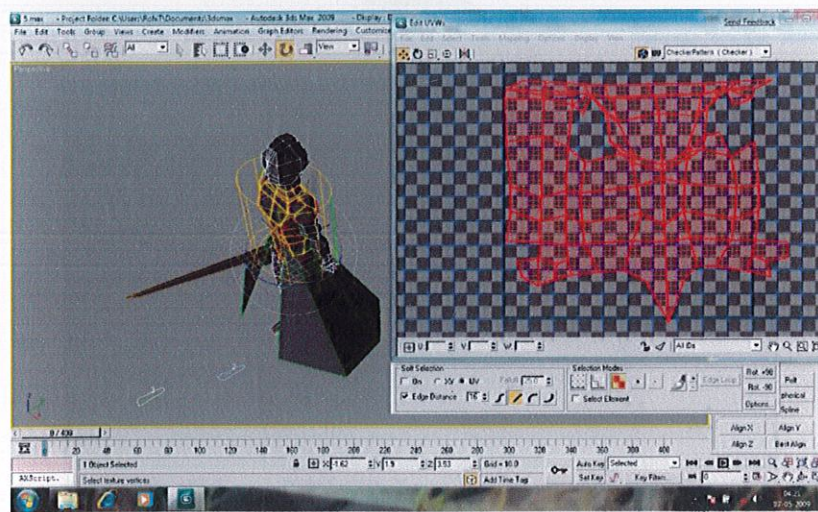


Figure 12

7 RENDERING ANIMATIONS AND MATERIALISATION

The next step was to create animations for our model. The animations were created on 3DS Max. We created many animations for the models like walk, run, big attacks and small attacks and some other special effects.

We also used the cloth modifier to give the hiori of Ichigo a cloth like effect.

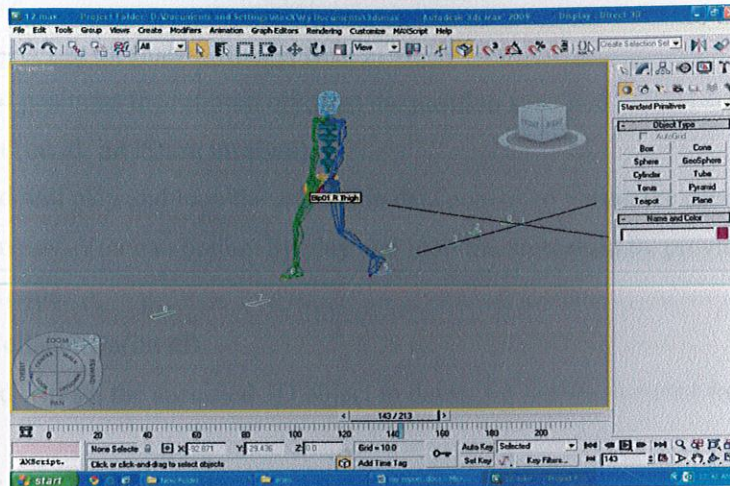


Figure 13

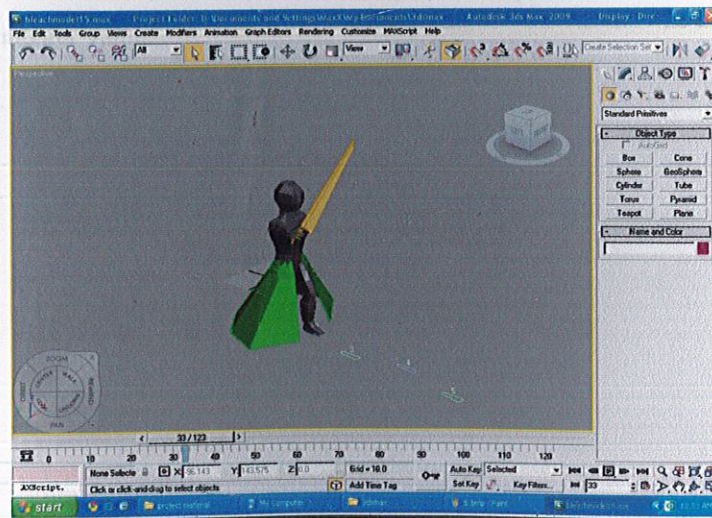


Figure 14

8 LOADING FINAL MODEL AND ANIMATION

Finally we loaded our final model and animations on our map and sorted out the positioning of objects and the camera position.

We used following commands to load our object:

`dbLoadObject (int id, "name of the file")`

This command loads a .X file into the memory and gives it the identity number provided for future reference.

`dbPositionObject (int id, float x, float y, float z)`

This command positions the referred object at the position specified.

`dbLoopObject (int id, int iStart, int iEnd)`

This command will play and loop the animation data contained within the specified 3D object from the beginning. You can optionally play and loop the animation by providing a specified start and end frame.

`dbSetObjectCollisionOn(int id)`

This command will set the specified 3D object to detect for and be detected for any collisions that occur.

`dbObjectCollision(int id1, int id2)`

This command will return a one if the two specified 3D objects are overlapping. If the second object number is set to zero, the number of the object overlapping with the first object will be returned as an integer value.

9 SPRITES AND OTHER RESOURCES

Created the sprites required for big attack, general user communication and story element sprites. Next we built items to increase health and soul power. We also prepared the sounds to be played during different modes of game. Following are some commands for the sprites:

`dbPlaySprite (int iSprite, int iStart, int iEnd, int iDelay)`

This command will play an animated sprite. The command defines the start and end frames to be played.

`dbMakeParticles (int Particle Number, int ImageNumber, int Frequency, float Radius)`

This command will create a particles object using a specified image and radius. A particles object will emit single particles given a default set of rules than can be changed using the particle commands.

`dbPositionParticles (int ParticleNumber, float X, float Y, float Z)`

This command will position a particles object in 3D world space. By moving a particles object you will be moving at the same time every particle that belongs to the particles object.

`dbLoadSound ("name of the file", int iSound, int iFlag)`

This command will load a WAV sound file into the specified Sound Number. The Sound Number must be an integer value.

`dbPlaySound (int iSound, int iStart)`

This command will play the specified Sound Number. An optional parameter allows you to specify a start position in bytes that skips the initial part of the sample to be played.

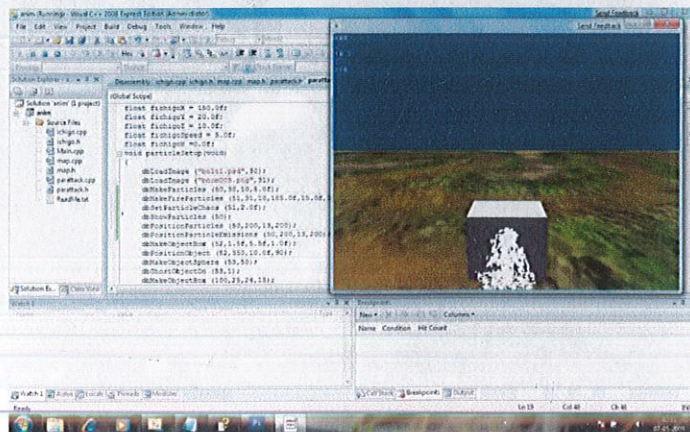


Figure 15

10 COMPILING ALL RESOURCES

All the resources are compiled together to bring out the game.

The controls for the game are as follow:

Move Forward	Up Arrow
Move Backward	Down Arrow
Move Right	Right Arrow
Move Left	Left Arrow
Jump	Space Bar
Run	Left Shift
Small Attack	W
Big Attack	Combination of S and D

The player would have to dodge the enemy fire to avoid damage.

The final project output is shown below:



Figure 16

BIBLIOGRAPHY

BOOKS REFERRED:

Andre Lamothe, *Tricks of Windows Game Programming*, Sams Publications

Kelly L Murdock, *3 DS MAX 9 BIBLE*, Autodesk

DARK GDK Documentation

WEB PAGES:

www.wikipedia.com

www.google.com

www.gamedev.com

www.dreamincode.com