**CSC 6430**

**Final Project**

**Total 200 points**

**(50 points for project documents and 100 for code submission)**

**(50 points for presentation)**

**Due in class April 28, 2014 (4:30 p.m.)**

***Both in-class and on-line students are required to attend***

***No late submissions will be considered. Turn in what you have by that date.***

***The entire project document must be contained in ONE word document.***

***No exceptions***

**Shidi Wang**

**fc7571**

You are to produce a 3D game using the **Monogames and Windows 8 frameworks** based upon your interpretation of the **FPS Land Bridge** scenario that we have been working on all term. The player will look for objects, collect resources, and avoid attacks from other game characters. In your game, you will need to have at least the following features implemented.

Give the basic design document for the game. Incorporate your answers from the associated lecture assignments relating to the game requirements into the document. Make sure that you make your points with concepts from lecture. What is the:

**Project Design: (50 points total) 5 points each.**

Describe the following:

1. Goal of the game for **current level**. Describe the overall nature of the game and how this level progresses to the next level. Precisely how are scores computed and winning assessed?

**To finish current level, you need to reach score 200. And to win the whole game, you need to collect all the collecting objects: the purple cross.**

**The player plays as a ghost – Boo, who tries to get all his lost crosses back in a dead land. However, there are corpse flower in every corner kill all the moving things close by, the player need to survive in this game and collect the lost crosses.**

**For every level the difficulties of the game has increased, and the score you need to reach doubled, for example, level 1 to level 2 is 200, level to level 3 is 400, etc.**

**The score are generated in 2 ways:**

1. **By killing the corpse flower, each kill raise the score by 10.**
2. **By collecting the cross (which are only 10 in entire game), each collection raise the score by 100.**

**Kill the corpse flower won’t get you win this game, the only way to win is find all the crosses and collecting them.**

2. What are the renewable resources in your **FPS** game economy (e.g. ammunition, and health among others)? What are their maximum values? How are they incremented? How are they depleted? Give precise relationships.

**In my game health and ammo are two renewable resources in my game. The maximum value is:**

**Health of Player: 100**

**Health of corpse flower: 10**

**Amount of Corpse Flower: 100**

**Ammo of player: 1000**

**Ammo of corpse flower: infinity**

**Health Objects: 100**

**Health and ammo of player reset after each level.**

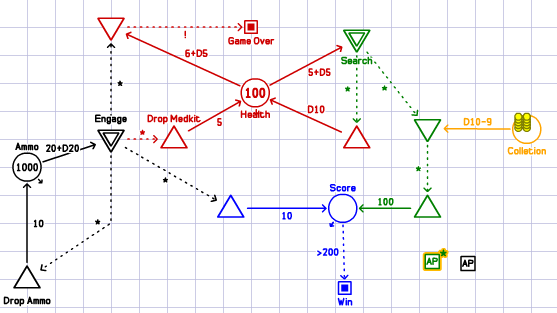
**There are health object you collect increase the health, 1 per mushroom.**

**And every time kills a corpse flower you earn 5 health and 10 ammos.**

**Health of the corpse flower can’t change by themselves in the entire game.**

**However every time you get closer to a corpse flower, it will start to attract you, every seed it shoots to you took your health by 5. The bullet you shoot to corpse flower took its health by 1.**

3. Give the machination diagram for your game economy for level 1. Describe. How many feedback loops does it have and what are their characteristics?



**This game diagram indicates 2 major renewable resources in my game, which are health and ammo, and the score system of this game.**

**It has 3 feedback loops, 2 of which are from killing corpse flower, one is collecting health object.**

**Each time you kill a corpse flower, you might lose ammo and health, but when the kill success, you win 10 ammo and 5 health.**

**When you do search, you might randomly lose health from the corpse flower among the road, nut when you collect a health object, you earn 1health.**

4. Identify two basic user strategies for level one, e.g. always attack (e.g. maximize ammo -always attack; or maximize health). Give the modified machinations diagram for each one of the artificial strategies. Describe.

**Here are two basic user strategies of this game:**

**1: Engage**

**2: Search**

**Because there are two ways to earn score in my game, kill the corpse flower or collecting the cross. From the diagram, you can clearly see:**

**In Engage, you choose to always attack the corpse flower to earn score, which you might lose 5 to 10 health and 20 to 40 ammos, but you will earn 10 scores and 10 ammo 5 health if you killed the enemy.**

**In Search, you choose to not fight back but searching through the map to collect crosses, which worth 1000 score each. But during the search you might be randomly attacked by corpse flower or find random health objects, so you will be able to lose 1 to 6 health and earn 1 to 10 health.**

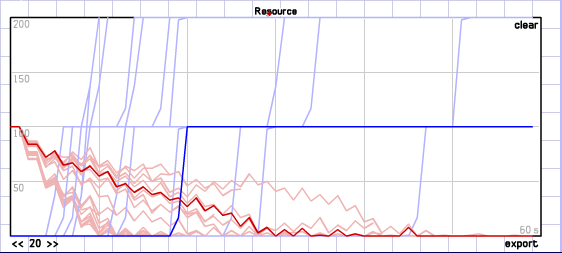
5. Give a graph with 10 runs of your game economy for each of the renewable resources for each of the two artificial strategies in level one above. What information do they provide as to the overall **balance** of your game? Describe.

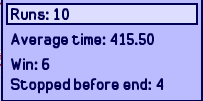
**Blue is Score;**

**Red is Health;**

**Black is Ammo;**

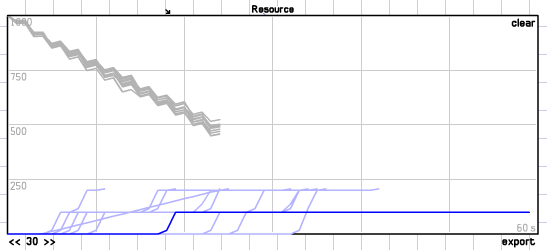
**Search & Health:**

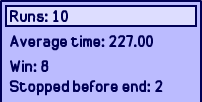




**Because it’s possible to lose health if you search without fight back, so even player could find health object and earn 1 health back, it’s possible it dies before find 2 crosses to earn 200 score.**

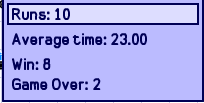
**Search & Ammo:**





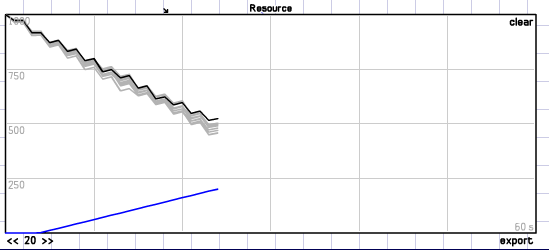
**Because of Ammo won’t lose much in search mod, more possible to win in his mod.**

**Engage & Health:**



**The score increase with kills, but health drops during kill, with the health kit feedback, it’s still easy to win.**

**Engage & Ammo**





**The ammo drop during engage mode is faster than search, which has the almost same possibility of win with search mode.**

6. through 10. Repeat each of the above questions for your level two. 5 points each.

6. Goal of the game for **current level**. Describe the overall nature of the game and how this level progresses to the next level. Precisely how are scores computed and winning assessed?

**The goal of current level is continue collecting crosses, but in order to get t next level, you need to reach score 400.**

**The way of getting score and find crosses stays the same.**

7. What are the renewable resources in your **FPS** game economy (e.g. ammunition, and health among others)? What are their maximum values? How are they incremented? How are they depleted? Give precise relationships.

**For every level the difficulties of the game has increased, and the score you need to reach doubled, for example, level 1 to level 2 is 200, level to level 3 is 400, etc.**

**The score are generated in 2 ways:**

**So in Level 2, there are:**

**Health Objects: 80**

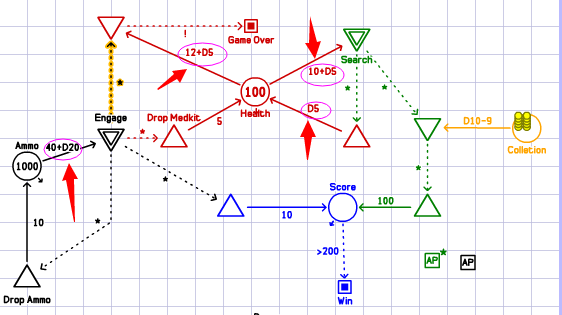
**Corpse Flower Health: 20**

**Amount of Corpse Flower: 200**

**Other stays the same.**

**This increases the speed of player losing ammo and health, which increase the game difficulty.**

8. Give the machination diagram for your game economy for level 1. Describe. How many feedback loops does it have and what are their characteristics?



**The feedback loops stay the same, but the difficulties of get ammo and med kit increased.**

9. Identify two basic user strategies for level one, e.g. always attack (e.g. maximize ammo -always attack; or maximize health). Give the modified machinations diagram for each one of the artificial strategies. Describe.

**Change on the 2 strategies.**

**1: Engage**

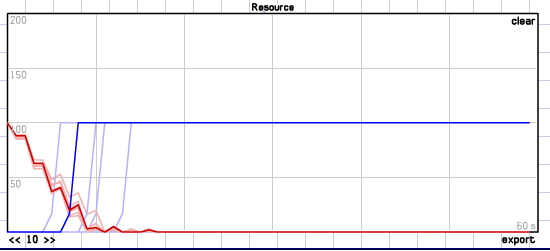
**In Engage, you choose to always attack the corpse flower to earn score, which you might lose 12 to 22 health and 40 to 60 ammos, but you will earn 10 scores and 10 ammo 5 health if you killed the enemy.**

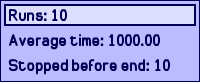
**2: Search**

**In Search, you choose to not fight back but searching through the map to collect crosses, which worth 1000 score each. But during the search you might be randomly attacked by corpse flower or find random health objects, so you will be able to lose 10 to 15 health and earn 1 to 5 health.**

10. Give a graph with 10 runs of your game economy for each of the renewable resources for each of the two artificial strategies in level one above. What information do they provide as to the overall **balance** of your game? Describe.

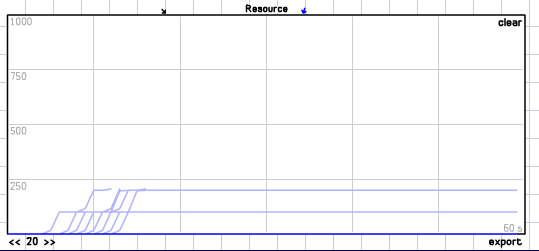
**Search & Health:**

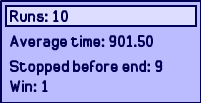




**So only search will get you lose this game because health lose too fast.**

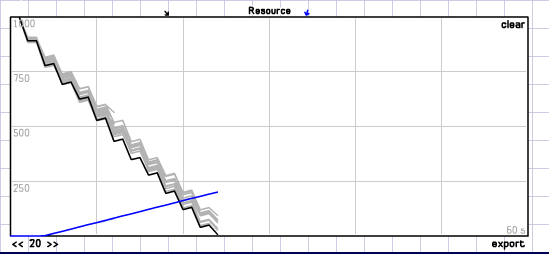
**Search & Ammo**





**Still because the difficulty increased, it’s less easy to win the game.**

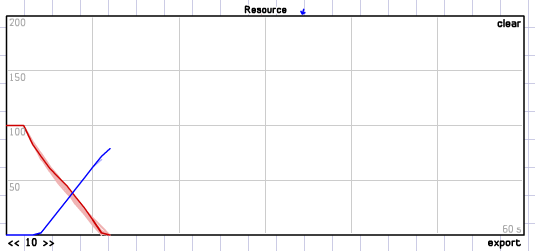
**Engage & Ammo:**





**Game sill wins, but it’s pretty clear that the ammo almost used out in the end of the game, which is really different from Level 1.**

**Engage & Health:**





**At this level, purely attack will for sure get you killed.**

**Conclusion:**

**In my game, no matter which level you are in, you need to use the 2 game strategies in the same time, or the wining will not be reached that easy. These 3 feedback loops helps keep the game balanced and last longer.**

**Implementation: Checklist: 50 points. 5 points each.**

For each of the items below briefly describe, give a screen shot, and highlight and label its location in your code.

1. Highlight the location of the mechanics for your game economy levels described above.

**My game economy goes with different object’s reaction to each other, they change the global value of health, score, bullet left, etc:**

**BulletObject2.cs**

if (distance < 10)

{

game.GameObjects.Add(new SmokeObject(game, Game.Textures["Smoke"], game.mgame.\_ghost.Position, new Vector3(20f)));

game.SoundEffects["hit"].Play(0.3f, 0, 0);

game.GameObjects.Remove(this);

if (game.mgame.health <= 100 && game.mgame.health > 0)

game.mgame.health -= 5;

return;

}

**CollectObject.cs**

if (distance <= 30)

{

game.SoundEffects["collect"].Play(1f, 0, 0);

game.mgame.Ingamescore += 100;

game.GameObjects.Remove(this);

return;

}

**Ghost.cs**

if (Mouse.GetState().LeftButton == ButtonState.Pressed)

{

if (game.mgame.bulletleft >= 0 && UpdateCount%3 == 0)

{

game.GameObjects.Add(new BulletObject(game, Position, game.Models["Bullet"], VangleY));

game.SoundEffects["shot2"].Play(0.3f, 0, 0);

game.mgame.bulletleft--;

}

}

**Corpse Flower.cs**

if (distance <= 200)

{

game.SoundEffects["gfire"].Play(1 / distance, 0, 0);

}

if (distance <= 300)

{

if (upcount % 100 == 0)

{

game.GameObjects.Add(new BulletObject2(game, position, game.Models["Orange"], towards));

game.SoundEffects["shot"].Play(0.3f, 0, 0);

}

}

position.Y = 7 - distance / 60f;

if (fhealth <= 0)

{

game.mgame.kill++;

game.mgame.Ingamescore += 10;

if(game.mgame.bulletleft<=990)

game.mgame.bulletleft += 10;

if(game.mgame.health<=95)

game.mgame.health += 5;

game.GameObjects.Remove(this);

return;

}

**HealthObject.cs**

if (distance <= 30)

{

game.SoundEffects["geto"].Play(0.3f, 0, 0);

if (game.mgame.health < 100 && game.mgame.health > 0)

game.mgame.health++;

game.GameObjects.Remove(this);

return;

}

**Mode\_Game.cs**

public override void Update(GameTime gameTime)

{

TouchCollection touches = TouchPanel.GetState();

if (health <= 0)

{

MediaPlayer.Stop();

state = State.gameover;

Score = Ingamescore;

over\_index = 0;

}

if (state == State.gameover)

{

if(TouchPanel.GetState().Count==1)

Game.SetGameMode<Mode\_HighScores>();

\_game.IsMouseVisible = true;

GameIsActive = false;

this.Deactivate();

\_game.GameObjects.Clear();

}

if (Ingamescore > 200 \* level)

{

level++;

Reset();

}

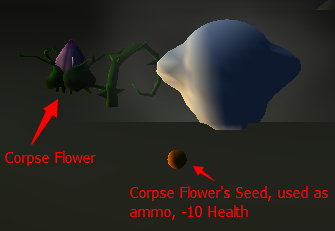
Game.UpdateAll(gameTime);

base.Update(gameTime);

}

2. What are the objects that you are hunting and collecting? Give screen shots and describe.

**Corpse Flower:**



**Everywhere in this dead land, Corpse flower hind under the ground, whenever you getting close, it suddenly appears and shot you to death.**

**Amount : 100 in level 1, 200 in level 2, 300 in level 3…**

**Health: 10 in level 1, 20 in level 2….**

**Ammo: Seed bullet, -10 health**

**Reward: kill it adds 10 bullet 5 health.**

**Special: you can’t see them when you are far, but when you getting close, you will hear scary crying noise from them, which means they are awake from the underworld….**

**Code:**

**CorpseFlower.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Graphics;

using GameFramework;

using Microsoft.Xna.Framework.Audio;

namespace Shidi\_Wang\_Project

{

public class CorpseFlower : GameFramework.MatrixModelObject

{

Game1 game;

public Vector3 position;

public float distance;

public Vector3 towards;

public int fhealth = 10;

int upcount = 0;

public CorpseFlower(Game1 game, Vector3 position, Model model) : base(game,position,model)

{

this.game = game;

this.position = position;

}

public override void Update(GameTime gameTime)

{

distance = (position - game.mgame.\_ghost.Position).Length();

towards = game.mgame.\_ghost.Position - position;

towards.Normalize();

if (distance <= 200)

{

game.SoundEffects["gfire"].Play(1 / distance, 0, 0);

}

if (distance <= 300)

{

if (upcount % 100 == 0)

{

game.GameObjects.Add(new BulletObject2(game, position, game.Models["Orange"], towards));

game.SoundEffects["shot"].Play(0.3f, 0, 0);

}

}

position.Y = 7 - distance / 60f;

if (fhealth <= 0)

{

game.mgame.kill++;

game.mgame.Ingamescore += 10;

if(game.mgame.bulletleft<=990)

game.mgame.bulletleft += 10;

if(game.mgame.health<=95)

game.mgame.health += 5;

game.GameObjects.Remove(this);

return;

}

Transformation = Matrix.CreateWorld(position,new Vector3(0,0,1), Vector3.Up);

upcount++;

}

public override void Draw(GameTime gameTime, Effect effect)

{

base.Draw(gameTime, effect);

}

}

}

**Cross:**



**The object to collect in this game, each worth 100 scores.**

**Amount: 10**

**Renewable: No**

**After collect all the cross in the game, you will win!**

**Code:**

**CollectObject.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Graphics;

using GameFramework;

using Microsoft.Xna.Framework.Audio;

namespace Shidi\_Wang\_Project

{

class CollectObject : GameFramework.MatrixModelObject

{

Game1 game;

public Vector3 position;

public float distance;

public Vector3 towards;

public int amount = 0;

public CollectObject(Game1 game, Vector3 position, Model model)

: base(game, position, model)

{

this.game = game;

this.position = position;

}

public override void Update(GameTime gameTime)

{

distance = (position - game.mgame.\_ghost.Position).Length();

towards = game.mgame.\_ghost.Position - position;

towards.Normalize();

if (distance <= 30)

{

game.SoundEffects["collect"].Play(1f, 0, 0);

game.mgame.Ingamescore += 100;

game.GameObjects.Remove(this);

return;

}

Transformation = Matrix.CreateWorld(position, new Vector3(0, 0, 1), Vector3.Up);

}

public override void Draw(GameTime gameTime, Effect effect)

{

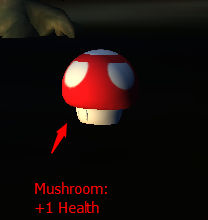
base.Draw(gameTime, effect);

}

}

}

**Health Objects:**



**Total amount: 100 in level 1, 80 in level 2, 60 in level 3…etc.**

**Renewable? Yes**

**Randomly appeared in this world.**

**Code:**

**HealthObject.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Graphics;

using GameFramework;

using Microsoft.Xna.Framework.Audio;

namespace Shidi\_Wang\_Project

{

class HealthObject : GameFramework.MatrixModelObject

{

Game1 game;

public Vector3 position;

public float distance;

public Vector3 towards;

public int amount = 0;

public HealthObject(Game1 game, Vector3 position, Model model)

: base(game, position, model)

{

this.game = game;

this.position = position;

}

public override void Update(GameTime gameTime)

{

distance = (position - game.mgame.\_ghost.Position).Length();

towards = game.mgame.\_ghost.Position - position;

towards.Normalize();

if (distance <= 30)

{

game.SoundEffects["geto"].Play(0.3f, 0, 0);

if (game.mgame.health < 100 && game.mgame.health > 0)

game.mgame.health++;

game.GameObjects.Remove(this);

return;

}

Transformation = Matrix.CreateWorld(position, new Vector3(0, 0, 1), Vector3.Up);

}

public override void Draw(GameTime gameTime, Effect effect)

{

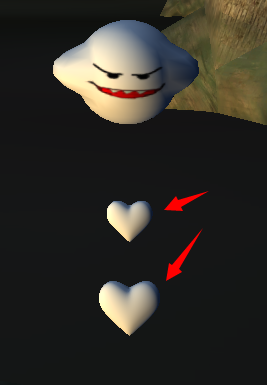
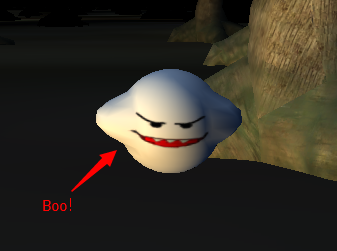
base.Draw(gameTime, effect);

}

}

}

3. Describe your avatar. Give screen shots and details.



**This tiny ghost is my avatar, which originally from Nintendo game Mario series, and his name is Boo, who can shoot heart shaped bullet to attack enemy. he has 100 health and 1000 ammos, when he lose all the health, he will die. Boo’s goal is to collect all the lost crosses of him, which are randomly appeared in this world.**

**Major code:**

**Ghost.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Audio;

using Microsoft.Xna.Framework.Content;

using Microsoft.Xna.Framework.GamerServices;

using Microsoft.Xna.Framework.Graphics;

using Microsoft.Xna.Framework.Input;

using Microsoft.Xna.Framework.Media;

using GameFramework;

namespace Shidi\_Wang\_Project

{

public class Ghost : GameFramework.MatrixModelObject

{

Game1 game;

public float VangleX = 0;

public float VangleY = 0;

float x = 0;

float y = 0;

float z = 0;

public SoundEffectInstance thurder = null;

public SoundEffectInstance thurder2 = null;

public Ghost(Game1 game, Vector3 position, Model model)

: base(game,position,model)

{

this.game = game;

this.x = position.X;

this.y = position.Y;

this.z = position.Z;

thurder = game.SoundEffects["Thurder"].CreateInstance();

thurder.Volume = 1.0f;

thurder2 = game.SoundEffects["Thurder2"].CreateInstance();

thurder2.Volume = 1.0f;

}

public override void Update(GameTime gameTime)

{

Vector3 delta;

base.Update(gameTime);

if (Keyboard.GetState().IsKeyDown(Keys.K) == true)

{

z += (float)Math.Cos(VangleY);

x -= (float)Math.Sin(VangleY);

}

if (Keyboard.GetState().IsKeyDown(Keys.I) == true)

{

z -= (float)Math.Cos(VangleY);

x += (float)Math.Sin(VangleY);

}

if (Keyboard.GetState().IsKeyDown(Keys.J) == true)

{

// x += (float)Math.Cos(VangleY);

VangleY += MathHelper.ToRadians(-1);

z -= (float)Math.Cos(VangleY) ;

x += (float)Math.Sin(VangleY) ;

thurder2.Play();

}

if (Keyboard.GetState().IsKeyDown(Keys.L) == true)

{

//x -= (float)Math.Cos(VangleY);

VangleY += MathHelper.ToRadians(1);

z -= (float)Math.Cos(VangleY) ;

x += (float)Math.Sin(VangleY) ;

thurder.Play();

}

Position = new Vector3(x, y, z);

delta = new Vector3(-(float)Math.Sin(VangleY), 0, (float)Math.Cos(VangleY));

Transformation = Matrix.CreateWorld(Position, delta, Vector3.Up);

if (Mouse.GetState().LeftButton == ButtonState.Pressed)

{

if (game.mgame.bulletleft >= 0 && UpdateCount%3 == 0)

{

game.GameObjects.Add(new BulletObject(game, Position, game.Models["Bullet"], VangleY));

game.SoundEffects["shot2"].Play(0.3f, 0, 0);

game.mgame.bulletleft--;

}

}

}

public override void Draw(GameTime gameTime, Effect effect)

{

base.Draw(gameTime, effect);

}

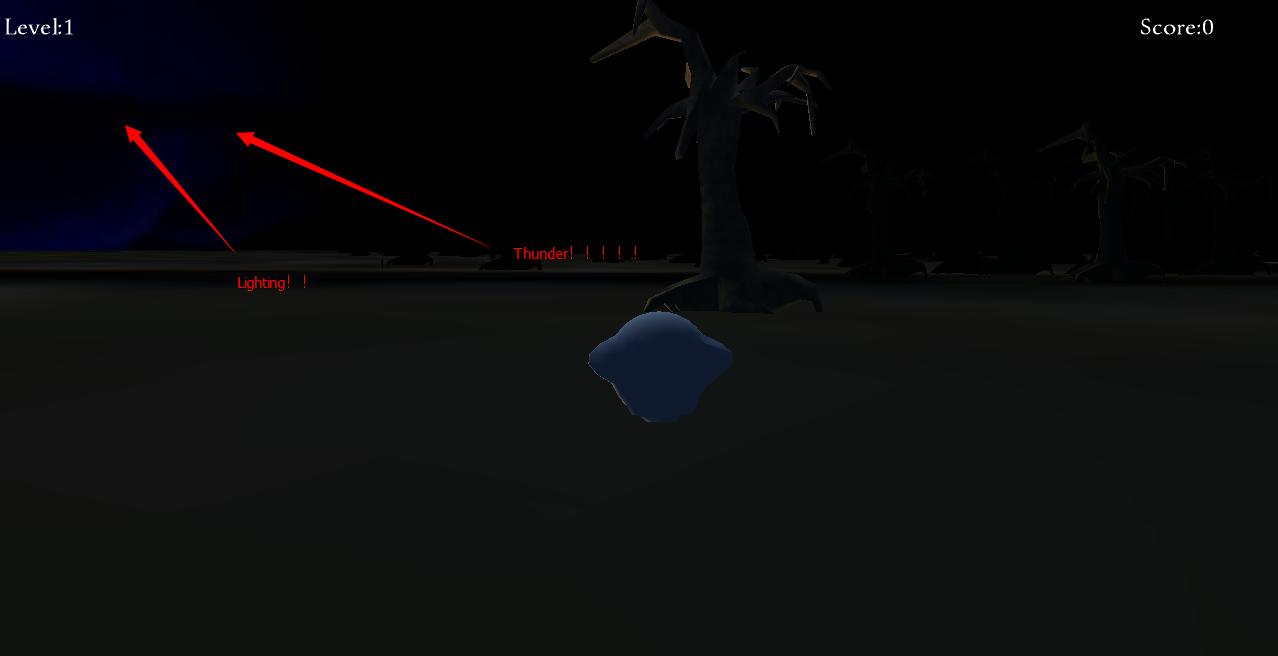
}

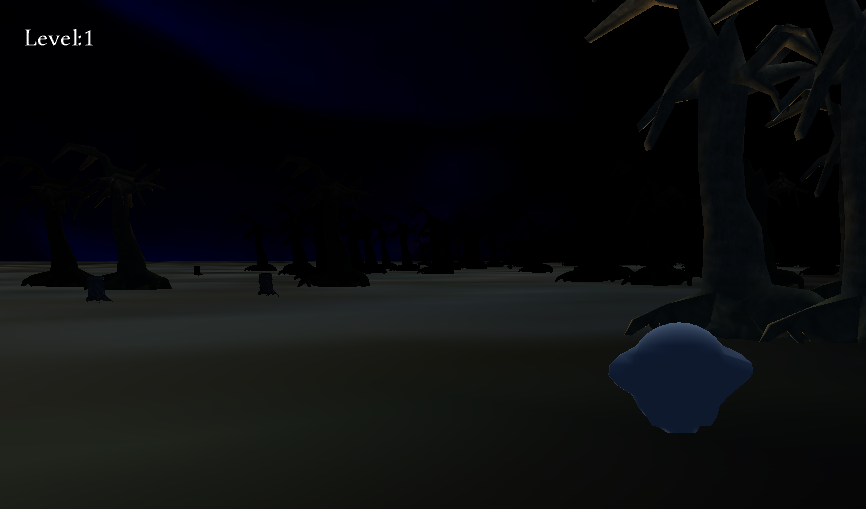
}

4. Describe the skybox used and the ground texture employed and the associated height map. Describe any textures used for the ground.

**My skybox is really special, it has the thunder effect and dark depth.**

**It’s a dark deep purple sky covered by black cloud, lighting shows up frequently.**





**The Ground is a desert look land, I used this non reflect texture to look like the sand, and it has very small height change because my avatar and he’s enemy can only shoot straight forward bullets, too big up or downs will bollock the sight of each other.**

**Code:**

**SkyBox.cs**

using System;

using System.Collections.Generic;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Graphics;

using System.Text;

namespace GameFramework

{

public class MatrixSkyboxObject : MatrixObjectBase

{

//-------------------------------------------------------------------------------------

// Class variables

// Declare a static array of vertices. As they will be the same

// for every cube, this saves us having to use further vertex

// memory for each SkyboxObject instance.

private static VertexPositionColorTexture[] \_vertices;

private static VertexBuffer \_vertexBuffer;

internal bool \_renderedThisFrame = false;

//-------------------------------------------------------------------------------------

// Class constructors

public MatrixSkyboxObject(GameHost game, Texture2D texture, Vector3 position, Vector3 scale)

: base(game)

{

// Have we already built the cube vertex array in a previous instance?

if (\_vertices == null)

{

// No, so build them now

BuildVertices();

// Create a vertex buffer

\_vertexBuffer = new VertexBuffer(game.GraphicsDevice, typeof(VertexPositionColorTexture), \_vertices.Length, BufferUsage.WriteOnly);

\_vertexBuffer.SetData(\_vertices);

}

// Set other object properties

ObjectColor = Color.White;

ObjectTexture = texture;

Position = position;

Scale = scale\*50;

}

//-------------------------------------------------------------------------------------

// Object Functions

/// <summary>

/// Update the ground position and calculate its transformation matrix

/// </summary>

public override void Update(GameTime gameTime)

{

base.Update(gameTime);

// Calculate the transformation matrix

SetIdentity();

// Observe the camera's position if one is active

if (Game.Camera != null)

{

// Read the camera's calculated position

//ApplyTransformation(Matrix.CreateTranslation(Game.Camera.Transformation.Translation));

}

// Now apply the standard transformations

ApplyStandardTransformations();

// Indicate that the skybox has not been rendered this frame

\_renderedThisFrame = false;

}

/// <summary>

/// Draw the cube

/// </summary>

public override void Draw(Microsoft.Xna.Framework.GameTime gameTime, Effect effect)

{

// Prepare the effect for drawing

PrepareEffect(effect);

// Disable lighting but remember whether it was switched on...

bool lightingEnabled = ((BasicEffect)effect).LightingEnabled;

((BasicEffect)effect).LightingEnabled = false;

// Disable the depth buffer

DepthStencilState depthState = effect.GraphicsDevice.DepthStencilState;

effect.GraphicsDevice.DepthStencilState = DepthStencilState.None;

// Set the active vertex buffer

effect.GraphicsDevice.SetVertexBuffer(\_vertexBuffer);

// Draw the object

foreach (EffectPass pass in effect.CurrentTechnique.Passes)

{

// Apply the pass

pass.Apply();

// Draw the sky box

effect.GraphicsDevice.DrawPrimitives(PrimitiveType.TriangleList, 0, \_vertices.Length / 3);

}

// Re-enable lighting and the depth buffer if required

if (lightingEnabled) ((BasicEffect)effect).LightingEnabled = true;

effect.GraphicsDevice.DepthStencilState = depthState;

// Indicate that the skybox has been rendered this frame

\_renderedThisFrame = true;

}

/// <summary>

/// Build the vertex array that stores the positions and colors of the cube vertices

/// </summary>

private void BuildVertices()

{

int i;

Color thisColor = Color.Black;

// Create and initialize the vertices

\_vertices = new VertexPositionColorTexture[24];

// Set the vertex positions for a unit size cube.

i = 0;

// Front face...

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, -0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(0.5f, 0.5f, 0.5f);

// Right face...

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(0.5f, 0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(0.5f, 0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(0.5f, 0.5f, 0.5f);

// Back face...

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, -0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(0.5f, 0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(0.5f, -0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, -0.5f);

// Left face...

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, -0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, -0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, 0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, 0.5f, -0.5f);

\_vertices[i++].Position = new Vector3(-0.5f, -0.5f, 0.5f);

// Set the texture coordinates

i = 0;

// Front face...

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.25f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.25f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.25f, 0.001f);

// Right face...

\_vertices[i++].TextureCoordinate = new Vector2(0.25f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.5f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.5f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.25f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.5f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.25f, 0.001f);

// Back face...

\_vertices[i++].TextureCoordinate = new Vector2(0.5f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.75f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.75f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.5f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.5f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(0.75f, 0.001f);

// Left face...

\_vertices[i++].TextureCoordinate = new Vector2(0.75f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.75f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.999f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(0.75f, 0.001f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.999f);

// Set the vertex colors -- all white

for (i = 0; i < \_vertices.Length; i++)

{

\_vertices[i].Color = Color.White;

}

}

}

}

**GroundObject.cs**

using System;

using System.Collections.Generic;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Graphics;

using System.Text;

using GameFramework;

namespace Shidi\_Wang\_Project

{

class GroundObject : GameFramework.MatrixObjectBase

{

//-------------------------------------------------------------------------------------

// Class variables

// Declare a static array of vertices.

private static VertexPositionNormalTexture[] \_vertices;

//-------------------------------------------------------------------------------------

// Class constructors

public GroundObject(Game1 game, Texture2D texture)

: base(game)

{

Scale = Vector3.One\*1000;

PositionY = 0;

// Have we already built the ground vertex array in a previous instance?

if (\_vertices == null)

{

// No, so build it now

BuildVertices();

}

// Set other object properties

ObjectTexture = texture;

}

//-------------------------------------------------------------------------------------

// Object Functions

/// <summary>

/// Update the ground position and calculate its transformation matrix

/// </summary>

public override void Update(GameTime gameTime)

{

base.Update(gameTime);

// Calculate the transformation matrix

SetIdentity();

// Now apply the standard transformations

ApplyStandardTransformations();

}

/// <summary>

/// Draw the ground

/// </summary>

public override void Draw(Microsoft.Xna.Framework.GameTime gameTime, Effect effect)

{

// Prepare the effect for drawing

PrepareEffect(effect);

// Draw the object

foreach (EffectPass pass in effect.CurrentTechnique.Passes)

{

// Apply the pass

pass.Apply();

// Draw the square

effect.GraphicsDevice.DrawUserPrimitives(PrimitiveType.TriangleList, \_vertices, 0, \_vertices.Length / 3);

}

}

/// <summary>

/// Build the vertex array that stores the positions and colors of the ground vertices

/// </summary>

private void BuildVertices()

{

int i;

Color thisColor = Color.Black;

// Create and initialize the vertices

\_vertices = new VertexPositionNormalTexture[6];

// Set the vertex positions for the ground

i = 0;

\_vertices[i++].Position = new Vector3(-15.0f, 0.0f, -15.0f);

\_vertices[i++].Position = new Vector3(15.0f, 0.0f, -15.0f);

\_vertices[i++].Position = new Vector3(-15.0f, 0.0f, 15.0f);

\_vertices[i++].Position = new Vector3(15.0f, 0.0f, -15.0f);

\_vertices[i++].Position = new Vector3(15.0f, 0.0f, 15.0f);

\_vertices[i++].Position = new Vector3(-15.0f, 0.0f, 15.0f);

// Set the texture coordinates for the ground

i = 0;

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 0.0f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.0f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 1.0f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.0f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 1.0f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 1.0f);

// Set the normals

for (i = 0; i < \_vertices.Length; i++)

{

\_vertices[i].Normal = new Vector3(0, 1, 0);

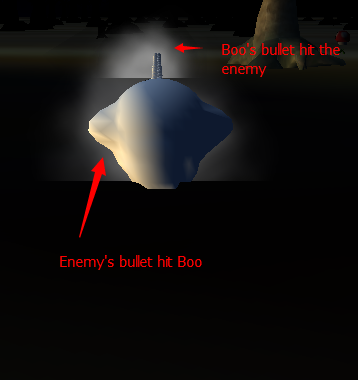
}

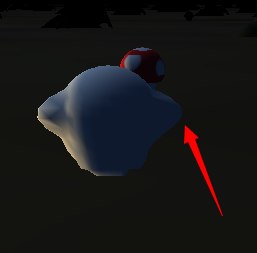
}

}

}

5. Collision Detection: Display the bounding spheres used to identify collisions between game objects. Give a screen shot showing the result of a collision.





**The bounding spheres are majorly used between bullet and its target, when Corpse Flower’s bullet hit the avatar, it explodes on the avatar and generate smoke, when Avatar’s bullet hit the enemy, it explodes and generate smoke, too. It also used between avatar and collecting items, when avatar is enough close to a cross or a mushroom, they disappear.**

**Code：**

public override void Update(GameTime gameTime)

{

position += 4\*towards;

delta = towards;

Transformation = Matrix.CreateWorld(position, delta, Vector3.Up);

upcount ++;

distance = (position - game.mgame.\_ghost.Position).Length();

if (distance < 10)

{

game.GameObjects.Add(new SmokeObject(game, Game.Textures["Smoke"], game.mgame.\_ghost.Position, new Vector3(20f)));

game.SoundEffects["hit"].Play(0.3f, 0, 0);

game.GameObjects.Remove(this);

if (game.mgame.health <= 100 && game.mgame.health > 0)

game.mgame.health -= 5;

return;

}

if (upcount > 1000)

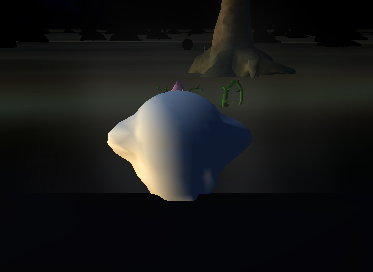
{

game.GameObjects.Remove(this);

return;

}

6. Describe each of the particle effects that you are using. Give a screen shot.



**The smoke is the particle effect I have in my game, it happens whenever the avatar or enemy were hit by bullets.**

**SmokeObject.cs**

**Code:**

using System;

using System.Collections.Generic;

using System.Linq;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Audio;

using Microsoft.Xna.Framework.Content;

using Microsoft.Xna.Framework.GamerServices;

using Microsoft.Xna.Framework.Graphics;

using Microsoft.Xna.Framework.Input;

using Microsoft.Xna.Framework.Input.Touch;

using Microsoft.Xna.Framework.Media;

using GameFramework;

namespace Shidi\_Wang\_Project

{

class SmokeObject : GameFramework.MatrixParticleObjectBase

{

private Game1 \_game;

// Declare a static array of vertices.

private static VertexPositionNormalTexture[] \_vertices;

private static VertexBuffer \_vertexBuffer;

// The initial position of the particle

private Vector3 \_startPosition;

private Vector3 \_startScale;

// The velocity of this particle

private Vector3 \_velocity;

// The particle's z-axis rotation speed

private float \_rotateSpeed;

public SmokeObject(Game1 game, Texture2D texture, Vector3 position, Vector3 scale)

: base(game, texture, position, scale)

{

\_game = game;

// Have we already built the ground vertex array in a previous instance?

if (\_vertices == null)

{

// No, so build it now

BuildVertices();

// Create a vertex buffer

\_vertexBuffer = new VertexBuffer(game.GraphicsDevice, typeof(VertexPositionNormalTexture), \_vertices.Length, BufferUsage.WriteOnly);

\_vertexBuffer.SetData(\_vertices);

}

// Set object properties

ObjectTexture = texture;

// Store the supplied start position

\_startPosition = position;

\_startScale = scale;

// Reset the particle to an initial state

ResetParticle();

}

internal void ResetParticle()

{

int grayLevel;

// Become active

IsActive = true;

// Reset to the smoke start position

Position = \_startPosition;

Scale = \_startScale;

// Offset the position to randomize around the center of the fire

PositionX += GameHelper.RandomNext(-0.03f, 0.03f);

PositionY += GameHelper.RandomNext(0.0f, 0.02f);

PositionZ += GameHelper.RandomNext(-0.03f, 0.03f);

// Start speed

\_velocity = new Vector3(0.006f, 0.012f, 0.000f);

// Random angle

AngleZ = GameHelper.RandomNext(0, MathHelper.TwoPi);

\_rotateSpeed = MathHelper.ToRadians(GameHelper.RandomNext(-5.0f, 5.0f));

// Set a random color and mid-level alpha

grayLevel = GameHelper.RandomNext(200, 255);

ObjectColor = new Color(grayLevel, grayLevel, grayLevel, 150);

}

public override void Update(GameTime gameTime)

{

base.Update(gameTime);

// Return immediately if we're not active

if (!IsActive) return;

// Update the alpha

SetAlpha(ObjectColor.A - 2);

// Have we become invisible? If so, become inactive

if (ObjectColor.A <= 0) IsActive = false;

// Update the object position

Position += \_velocity;

// Scale up

Scale \*= 1.01f;

// Rotate the object

AngleZ += \_rotateSpeed;

// Calculate the transformation matrix

SetIdentity();

// Apply the billboard transformation

ApplyTransformation(CreateBillboard(Position, Game.Camera.Transformation.Translation, Game.Camera.Transformation.Up, Game.Camera.Transformation.Forward));

// Rotate and scale

ApplyTransformation(Matrix.CreateRotationZ(AngleZ));

ApplyTransformation(Matrix.CreateScale(Scale));

}

private void SetAlpha(int alpha)

{

// Keep in the range 0 to 255

if (alpha < 0) alpha = 0;

if (alpha > 255) alpha = 255;

// Update the color

Color c = ObjectColor;

c.A = (byte)alpha;

ObjectColor = c;

}

public override void Draw(Microsoft.Xna.Framework.GameTime gameTime, Effect effect)

{

// Return immediately if we're not active

if (!IsActive) return;

// Prepare the effect for drawing

PrepareEffect(effect);

// Disable lighting but remember whether it was switched on...

bool lightingEnabled = ((BasicEffect)effect).LightingEnabled;

((BasicEffect)effect).LightingEnabled = false;

// Disable writing to the depth buffer

DepthStencilState depthState = effect.GraphicsDevice.DepthStencilState;

effect.GraphicsDevice.DepthStencilState = DepthStencilState.DepthRead;

// Enable transparency

BlendState blendState = effect.GraphicsDevice.BlendState;

effect.GraphicsDevice.BlendState = BlendState.AlphaBlend;

// Set the active vertex buffer

effect.GraphicsDevice.SetVertexBuffer(\_vertexBuffer);

// Draw the object

foreach (EffectPass pass in effect.CurrentTechnique.Passes)

{

// Apply the pass

pass.Apply();

// Draw the sky box

effect.GraphicsDevice.DrawPrimitives(PrimitiveType.TriangleList, 0, \_vertices.Length / 3);

}

// Restore the lighting, depth buffer and blandstate to their original values

if (lightingEnabled) ((BasicEffect)effect).LightingEnabled = true;

effect.GraphicsDevice.DepthStencilState = depthState;

effect.GraphicsDevice.BlendState = blendState;

}

private void BuildVertices()

{

int i;

Color thisColor = Color.Black;

// Create and initialize the vertices

\_vertices = new VertexPositionNormalTexture[6];

// Set the vertex positions for the ground

i = 0;

\_vertices[i++].Position = new Vector3(-1.0f, -1.0f, 0.0f);

\_vertices[i++].Position = new Vector3(1.0f, -1.0f, 0.0f);

\_vertices[i++].Position = new Vector3(-1.0f, 1.0f, 0.0f);

\_vertices[i++].Position = new Vector3(1.0f, -1.0f, 0.0f);

\_vertices[i++].Position = new Vector3(1.0f, 1.0f, 0.0f);

\_vertices[i++].Position = new Vector3(-1.0f, 1.0f, 0.0f);

// Set the texture coordinates for the ground

i = 0;

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 1.0f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 1.0f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.0f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 1.0f);

\_vertices[i++].TextureCoordinate = new Vector2(0.0f, 0.0f);

\_vertices[i++].TextureCoordinate = new Vector2(1.0f, 0.0f);

// Set the normals

for (i = 0; i < \_vertices.Length; i++)

{

\_vertices[i].Normal = new Vector3(0, 1, 0);

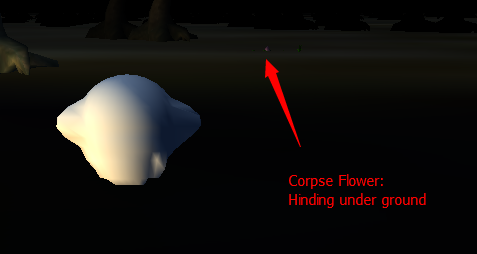
}

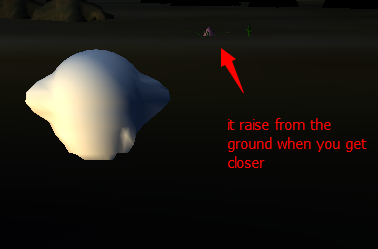
}

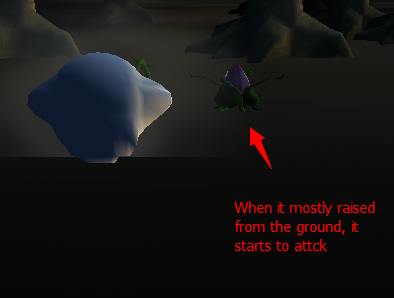
}

}

7. What game objects are animated? Give a screen shot describing each.







**And if you didn’t kill it, it goes back to ground and sleep when you run away.**

**Code:**

**CorpseFlower.cs**

public CorpseFlower(Game1 game, Vector3 position, Model model) : base(game,position,model)

{

this.game = game;

this.position = position;

}

public override void Update(GameTime gameTime)

{

distance = (position - game.mgame.\_ghost.Position).Length();

towards = game.mgame.\_ghost.Position - position;

towards.Normalize();

if (distance <= 200)

{

game.SoundEffects["gfire"].Play(1 / distance, 0, 0);

}

if (distance <= 300)

{

if (upcount % 100 == 0)

{

game.GameObjects.Add(new BulletObject2(game, position, game.Models["Orange"], towards));

game.SoundEffects["shot"].Play(0.3f, 0, 0);

}

}

position.Y = 7 - distance / 60f;

if (fhealth <= 0)

{

game.mgame.kill++;

game.mgame.Ingamescore += 10;

if(game.mgame.bulletleft<=990)

game.mgame.bulletleft += 10;

if(game.mgame.health<=95)

game.mgame.health += 5;

game.GameObjects.Remove(this);

return;

}

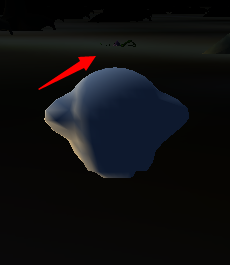
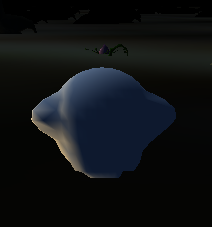
Transformation = Matrix.CreateWorld(position,new Vector3(0,0,1), Vector3.Up);

upcount++;

}

8. Describe the Artificial Intelligence kinematics used for your NPC game characters and give a screen shot.

**The enemy, Corpse Flower has the AI which it hind itself when the avatar are not in its shooting distance, so the avatar won’t be able to kill them without being attacked by them. Also, the bullet from avatar won’t be able to hit them if most of their body is under the ground, this force the avatar to get closer and start the battle.**

**Code:**

**Corpse Flower**

public CorpseFlower(Game1 game, Vector3 position, Model model) : base(game,position,model)

{

this.game = game;

this.position = position;

}

public override void Update(GameTime gameTime)

{

distance = (position - game.mgame.\_ghost.Position).Length();

towards = game.mgame.\_ghost.Position - position;

towards.Normalize();

if (distance <= 200)

{

game.SoundEffects["gfire"].Play(1 / distance, 0, 0);

}

if (distance <= 300)

{

if (upcount % 100 == 0)

{

game.GameObjects.Add(new BulletObject2(game, position, game.Models["Orange"], towards));

game.SoundEffects["shot"].Play(0.3f, 0, 0);

}

}

position.Y = 7 - distance / 60f;

if (fhealth <= 0)

{

game.mgame.kill++;

game.mgame.Ingamescore += 10;

if(game.mgame.bulletleft<=990)

game.mgame.bulletleft += 10;

if(game.mgame.health<=95)

game.mgame.health += 5;

game.GameObjects.Remove(this);

return;

}

Transformation = Matrix.CreateWorld(position,new Vector3(0,0,1), Vector3.Up);

upcount++;

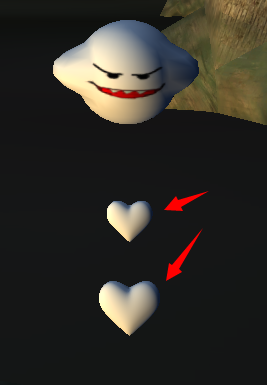
}

9. What audio is produced as the result of each collision? Describe and give a screen shot.

**I added a lot of audio in this game.**

**Shooting Sound:**

**Happens when avatar shoots a bullet:**

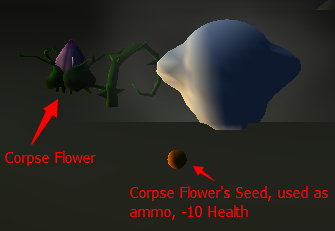


game.GameObjects.Add(new BulletObject(game, Position, game.Models["Bullet"], VangleY));

game.SoundEffects["shot2"].Play(0.3f, 0, 0);

game.mgame.bulletleft--;

**Shooting sound of Corpse flower**



if (upcount % 100 == 0)

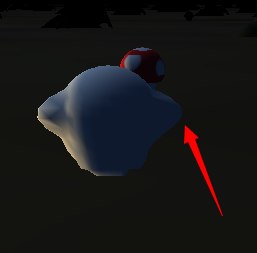
{

game.GameObjects.Add(new BulletObject2(game, position, game.Models["Orange"], towards));

game.SoundEffects["shot"].Play(0.3f, 0, 0);

}

**Getting Collecting Object:**



**A “ding” sound when you collect a health mushroom.**

if (distance <= 30)

{

game.SoundEffects["geto"].Play(0.3f, 0, 0);

if (game.mgame.health < 100 && game.mgame.health > 0)

game.mgame.health++;

game.GameObjects.Remove(this);

return;

}

**A Pleasant music When you collect the Cross**



if (distance <= 30)

{

game.SoundEffects["collect"].Play(1f, 0, 0);

game.mgame.Ingamescore += 100;

game.GameObjects.Remove(this);

return;

}

10. Describe your splash screens and give a screen shot.

**Starting Scene.**

**Tells the creator of the game, me ☺**

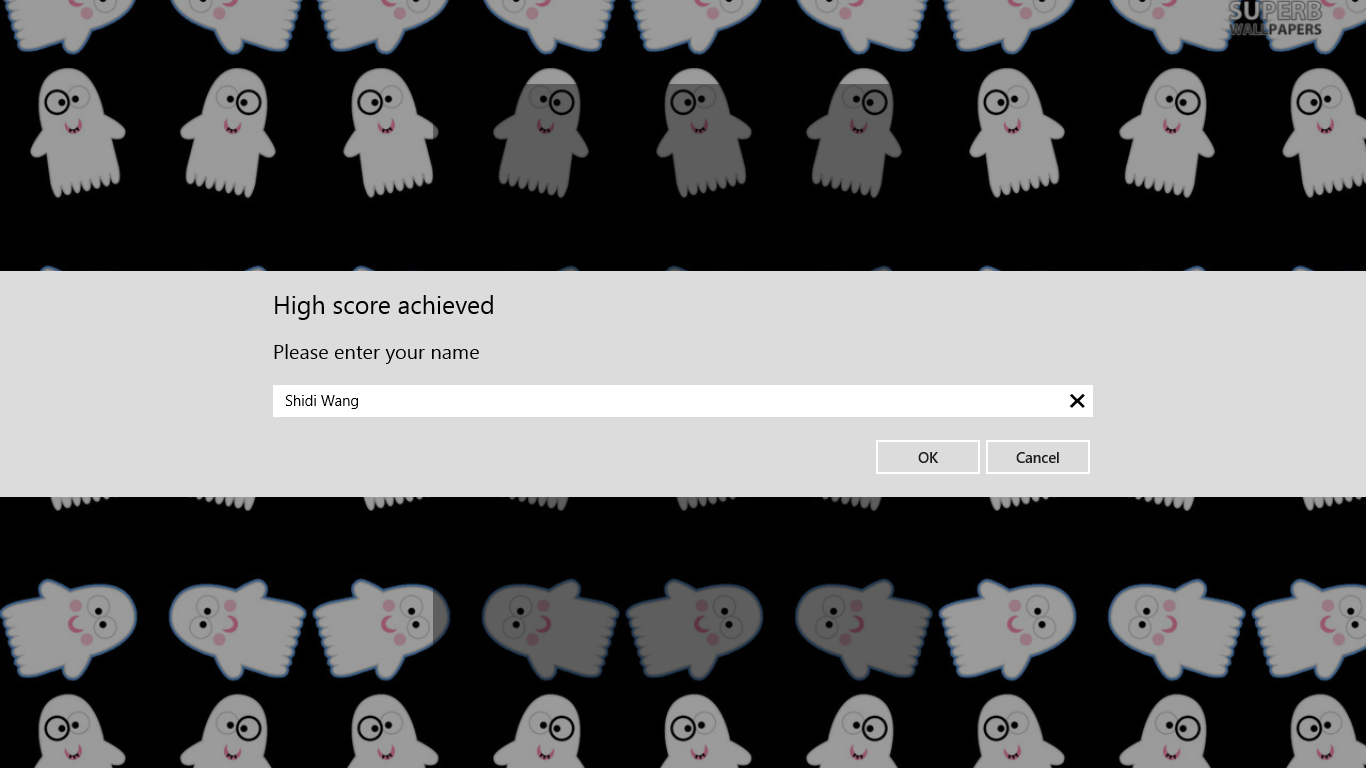
**And gives the menu you can choose from, stat a new game, view high score and exit.**





**Game over scene.**

**Shows up when you lost the game.**



**Name Enter Scene for high score.**



**High Score Page.**



**The win Page after you collecting all the 10 crosses.**

**Code:**

using System;

using Microsoft.Xna.Framework;

using Microsoft.Xna.Framework.Graphics;

using Microsoft.Xna.Framework.Input.Touch;

using GameFramework;

using Microsoft.Xna.Framework.Content;

using Microsoft.Xna.Framework.Audio;

using Microsoft.Xna.Framework.Media;

using Microsoft.Xna.Framework.Input;

namespace Shidi\_Wang\_Project

{

public class Mode\_Game : GameModeBase

{

public float AverageScale = 0.1f;

public Game1 \_game;

public int bulletleft = 1000;

public int health = 100;

public int kill = 0;

public bool GameIsActive { get; set; }

Random rnd = new Random();

public int Ingamescore=0;

public int Score { get; set; }

public Ghost ghost;

public Ghost \_ghost;

public CorpseFlower cflower;

public int level = 1;

public Mode\_Game(Game1 game)

: base(game)

{

\_game = game;

\_game.IsMouseVisible = true;

// Indicate that the game is not yet active

GameIsActive = false;

}

public Song music = null;

public enum State {levelstart,playing,gameover }

public State state = State.playing;

public float level\_bannar\_Y=-20;

public float level\_bannar\_Y\_a=0;

public int over\_index;

// The player's final score

/// <summary>

/// Reset the game to its initial state

/// </summary>

public override void Reset()

{

GameObjects.Clear();

\_game.BulletsObjects.Clear();

\_game.GameObjects.Add(new GroundObject(\_game, \_game.Textures["Grass"]));

base.Reset();

//Add some Trees

for (int i = 0; i < 1000; i++)

{

int j = rnd.Next(1, 10000);

\_game.GameObjects.Add(new MatrixModelObject(\_game, new Vector3(-j + i \* 10, 5, i - j), \_game.Models["HorrorTree"]));

}

for (int i = 0; i < 100; i++)

{

int j = rnd.Next(1, 10000);

\_game.GameObjects.Add(new MatrixModelObject(\_game, new Vector3(-j + i \* 100, 0, i \* 10 - j), \_game.Models["Root"]));

}

for (int i = 0; i < 100\*level; i++)

{

int h = rnd.Next(1, 10000);

\_game.GameObjects.Add(new CorpseFlower(\_game, new Vector3(-h + i \* 100, 7, i \* 10 - h), \_game.Models["StrangeFlower"]));

}

for (int i = 0; i < 100-level\*20; i++)

{

int h = rnd.Next(1, 10000);

\_game.GameObjects.Add(new HealthObject(\_game, new Vector3(-h + i \* 100, -20, i \* 10 - h), \_game.Models["Mashroom"]));

}

for (int i = 0; i < 10; i++)

{

int h = rnd.Next(1, 10000);

\_game.GameObjects.Add(new CollectObject(\_game, new Vector3(-h + i \* 100, 2, i \* 10 - h), \_game.Models["Cross"]));

}

ghost = new Ghost(\_game, new Vector3(0, 7, 0), \_game.Models["Ghost"]);

GameObjects.Add(ghost);

\_ghost = ghost;

health = 100;

bulletleft = 1000;

// Add the camera to the game

\_game.Camera = new CameraObject(\_game);

\_game.Camera.PositionY = 7;

\_game.Camera.ChaseObject = \_ghost;

\_game.Camera.ChaseDistance = 150;

\_game.Camera.ChaseElevation = 20f;

// Add the sky box

\_game.Skybox = new MatrixSkyboxObject(\_game, \_game.Textures["NightSky"], new Vector3(0, 0.2f, 0), new Vector3(1000, 1000, 1000));

GameIsActive = true;

}

/// <summary>

/// Allows the game to run logic such as updating the world,

/// checking for collisions, gathering input, and playing audio.

/// </summary>

/// <param name="gameTime">Provides a snapshot of timing values.</param>

public override void Update(GameTime gameTime)

{

TouchCollection touches = TouchPanel.GetState();

// Has the player touched the screen?

if (health <= 0)

{

MediaPlayer.Stop();

state = State.gameover;

Score = Ingamescore;

over\_index = 0;

}

if (state == State.gameover)

{

if(TouchPanel.GetState().Count==1)

Game.SetGameMode<Mode\_HighScores>();

\_game.IsMouseVisible = true;

GameIsActive = false;

this.Deactivate();

\_game.GameObjects.Clear();

}

if (Ingamescore > 200 \* level)

{

level++;

Reset();

}

Game.UpdateAll(gameTime);

// updatecount++;

base.Update(gameTime);

}

/// <summary>

/// This is called when the game should draw itself.

/// </summary>

/// <param name="gameTime">Provides a snapshot of timing values.</param>

public override void Draw(GameTime gameTime)

{

\_game.GraphicsDevice.Clear(Color.Black);

if (state == State.playing)

{

\_game.DrawObjects(gameTime, \_game.\_effect);

\_game.DrawParticles(gameTime, \_game.\_effect, \_game.Textures["Smoke"]);

\_game.StoreStateBeforeSprites();

\_game.\_spriteBatch.Begin();

\_game.\_spriteBatch.Draw(\_game.Textures["sign"], new Vector2(\_game.GraphicsDevice.Viewport.Bounds.Width - 330, \_game.GraphicsDevice.Viewport.Height - 60), null, new Color(200 - health \* 2, 50 + health \* 2, 0), 0, new Vector2(0, 0), new Vector2(3 \* health / 100f, 0.5f), SpriteEffects.None, 0);

\_game.\_spriteBatch.DrawString(\_game.Fonts["Miramonte"], "Health:" + health.ToString() + "%", new Vector2(\_game.GraphicsDevice.Viewport.Bounds.Width - 310, \_game.GraphicsDevice.Viewport.Height - 45), Color.White);

\_game.\_spriteBatch.Draw(\_game.Textures["sign"], new Vector2(10, \_game.GraphicsDevice.Viewport.Height - 60), null, new Color(50 + bulletleft / 5, 250 - bulletleft / 5, 0), 0, new Vector2(0, 0), new Vector2(3 \* bulletleft / 1000.0f, 0.5f), SpriteEffects.None, 0);

\_game.\_spriteBatch.DrawString(\_game.Fonts["Miramonte"], "Bullet:" + bulletleft.ToString() + "%", new Vector2(30, \_game.GraphicsDevice.Viewport.Height - 45), Color.White);

\_game.\_spriteBatch.DrawString(\_game.Fonts["Miramonte"], "Score:" + Ingamescore.ToString(), new Vector2(\_game.GraphicsDevice.Viewport.Width-200, 25), Color.White);

\_game.\_spriteBatch.DrawString(\_game.Fonts["Miramonte"], "Level:" + level.ToString(), new Vector2(30, 25), Color.White);

}

else if (state == State.gameover)

{

Texture2D redbg = new Texture2D(\_game.GraphicsDevice, 1, 1);

redbg.SetData(new Color[] { Color.BlueViolet });

\_game.StoreStateBeforeSprites();

\_game.\_spriteBatch.Begin();

\_game.\_spriteBatch.Draw(redbg, new Vector2(0, 0), null, Color.BlueViolet, 0, new Vector2(0, 0), new Vector2(\_game.GraphicsDevice.Viewport.Bounds.Width, \_game.GraphicsDevice.Viewport.Bounds.Height), SpriteEffects.None, 0);

\_game.\_spriteBatch.DrawString(\_game.Fonts["Miramonte"], "Final Score: " + Score.ToString(), new Vector2(\_game.GraphicsDevice.Viewport.Bounds.Width / 2 - 90, 450), Color.White);

\_game.\_spriteBatch.DrawString(\_game.Fonts["Miramonte"], "GAME OVER !" , new Vector2(\_game.GraphicsDevice.Viewport.Bounds.Width / 2 - 220, 350), Color.White,0f,new Vector2(0,0),3.0f,SpriteEffects.None,0);

}

// End the spritebatch

\_game.\_spriteBatch.End();

\_game.RestoreStateAfterSprites();

\_game.GraphicsDevice.BlendState = BlendState.Opaque;

\_game.GraphicsDevice.DepthStencilState = DepthStencilState.Default;

base.Draw(gameTime);

}

}

}

**Project code submission (50 points)**

* The whole project on CD-ROM that is running. In addition, you are required to upload your project to Final Project folder on the blackboard. 10 points.
* Commented code listing for your program. Run instructions for accessing and running the program. You should include instructions concerning how to change the run conditions or input different variables. 20 points.

**Instruction:**

1. **Open the Shidi\_Wang\_Project.sln in your VS2012(tested under VS2012)**
2. **Click Start Debuging.**
3. **Enter Main Menu:**
4. **Choose From:**



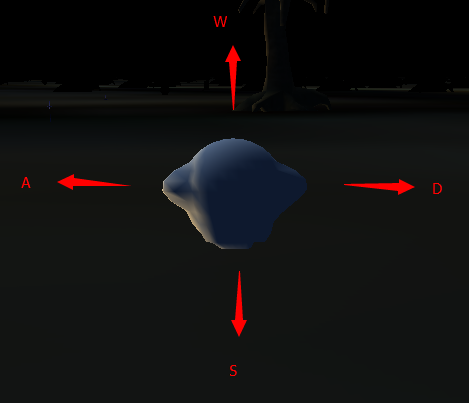
**To start a new game, view high score or exit the game.**

1. **In the game, use “W” “S” “A” “D” to control the avatar, and Left Key to shoot bullet.**
2. **After game over, enter your name in Score Board.**

* System test runs. Make sure that you include at least four different scenarios. Make sure that you describe each, using screen shots. 10 points.

1. **Moving:**

**Using “W” “S” “A” “D”, when press “A” + “D” together with “W”- Run Mode.**

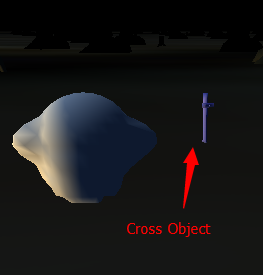


1. **Attack, using left key from your mouse, shoot bullet.**

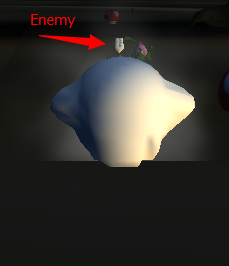


1. **Collecting Object- move closer to the object you want to collect.**

**After the collecting there will be sound effect and the object will disappear.**



1. **Fight – Using bullet to fight against the Corpse Flower.**



1. **Game Over – when your health is less than 0.**



* Overall assessment. Were you pleased with the outcome of your game? If so, why? What future extensions do you think might be useful? Why? 10 points.

**The outcome of the game is the Score Page. Right now I’m happy to see the game playing and the score I can get while playing, I believe it has a good balance and works bug free.**

**Future Extension:**

**Later on I might ass moving enemy to this game, right now all my enemies in this game are corpse flowers which can’t change their positions, but when I add moving objects the variety and difficulty will have a significant increase.**

**Project demo in class (50 points)**

* Introduce your system using a **power point** presentation. Start by describing your Game Economy as described above, and then your implementation as described above. Demonstrate the capabilities of your system during a demo run. You should be able to run the system under more than one situation. Describe each scenario that you are running your program in.
* All students must attend the class at the beginning of the presentation on April 28, 2014 (4:30 p.m.) and they are not allowed to leave until the end of all presentations. Students are not allowed to work on their projects during the presentation. We will not accept another version of your project after the beginning of first presentation.

**Extra Credit: 10 points (Credit given only upon completion of required elements).**

Describe your game mechanics for your third level by answering the five level questions above. What implementation changes did you have to make? Highlight the location of the mechanics for level three along with implementation changes made to support level three.

E1. Goal of the game for **current level**. Describe the overall nature of the game and how this level progresses to the next level. Precisely how are scores computed and winning assessed?

**The goal of current level is continue collecting crosses, but in order to get t next level, you need to reach score 600.**

**The way of getting score and find crosses stays the same.**

E2. What are the renewable resources in your **FPS** game economy (e.g. ammunition, and health among others)? What are their maximum values? How are they incremented? How are they depleted? Give precise relationships.

**For every level the difficulties of the game has increased, and the score you need to reach doubled, for example, level 1 to level 2 is 200, level to level 3 is 400, etc.**

**The score are generated in 2 ways:**

**So in Level 2, there are:**

**Health Objects: 60**

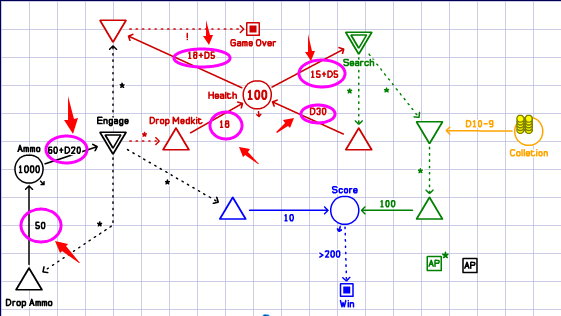
**Corpse Flower Health: 30**

**Amount of Corpse Flower: 300**

**Other stays the same.**

**This increases the speed of player losing ammo and health, which increase the game difficulty.**

E3. Give the machination diagram for your game economy for level 1. Describe. How many feedback loops does it have and what are their characteristics?



**The feedback loops stay the same, but the difficulties of get ammo and med kit increased.**

E4. Identify two basic user strategies for level one, e.g. always attack (e.g. maximize ammo -always attack; or maximize health). Give the modified machinations diagram for each one of the artificial strategies. Describe.

**Change on the 2 strategies.**

**1: Engage**

**In Engage, you choose to always attack the corpse flower to earn score, which you might lose 40 to 60 health and 60 to 80 ammos, and lose 18 to 23 health. But you will earn 10 scores and 50 ammo 18 health if you killed the enemy.**

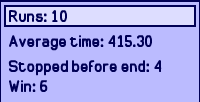
**2: Search**

**In Search, you choose to not fight back but searching through the map to collect crosses, which worth 1000 score each. But during the search you might be randomly attacked by corpse flower or find random health objects, so you will be able to lose 15 to 20 health and earn 1 to 30 health.**

E5. Give a graph with 10 runs of your game economy for each of the renewable resources for each of the two artificial strategies in level one above. What information do they provide as to the overall **balance** of your game? Describe.

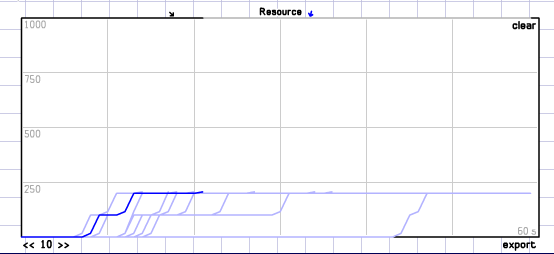
**Search & Health:**

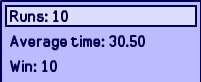




**Because the enemy is getting really strong so it’s possible you run out of health before find another cross.**

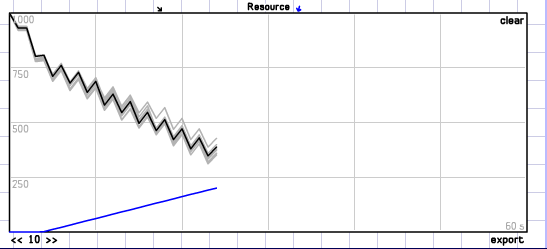
**Search & Ammo**

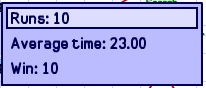




**So search will get you win this game because health not lose at all.**

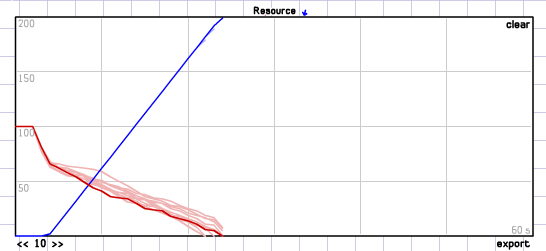
**Engage & Ammo:**





**The ammo seems to be way enough because of the rewards.**

**Engage & Health:**





**Because the rewards of killing enemy increased a lot, so pure killing isn’t a bad idea now.**

**Conclusion:**

**Because the crosses is getting less and less each level, so I increased the killing rewards, which will help the game to stay balance.**

**Submission Instructions:**

In addition to the CD-ROM used as backup only, you need to submit your project to the *Final Project* folder on the blackboard by the due date for full credit. If you are registered in both classes, AI and Game, you must provide a separate CD-ROM for each class.

Do not copy other (current or former) students’ projects. Do not seek projects from the internet. To minimize this temptation, always type up your project answers by yourself.

Make sure to include all files including Microsoft word file in one folder and compressed (zipped) your folder using (.zip) not rar.

No email or hard copy is accepted. You must follow the following format:

*Code:*

* Includes the whole project source code. Make sure that the code is well commented.
* Includes run instructions.
* Use only the code and the software associated with the required books.

*Documentation, In a Microsoft word:*

* Type each question and the answer under it. Follow the questions’ order in answering each question.
* List code, highlight changes and explain them.
* Show output/screenshots and explain them.