Homework #4: Aperture Science Xbox Multicorification Center

Abhishek Venkatesh

Chosen Game for multi-corification: 2-D vertical scrolling shooter. The player controls a fighter plane(F14) and fires missiles at the enemy fighter jets. If the enemy missiles hit you, the player dies. The player gets 3 lives. The game includes effects for clouds and it scrolls the background to give a feeling that the plane is flying over a city.

URL for Game:http://aroganworld.blogspot.com/2006/09/f14-xna-game.html

1) Multi-Corification: The game had the following three sub components: Collision Detection, Creating the enemy fighter jets and updating their position and the last is to show the explosion when a collision occurs. All these were being done by the main thread. I have created separate threads for each of them and set the processor affinity to use different cores on the Xbox. Some data was being shared between these components. I have added a simple locks to ensure the threads access the data structures sequentially to ensure integrity.
Code Snippet:

Creating the Threads

```
            if (firsttime == 1)// create the thread the first time update is called
            {
                Thread t1 = new Thread(CheckCollisions);
                t1.Start();
                Thread t2 = new Thread(delegate() {
                  Enemies.ThreadedUpdate(true); });
                t2.Start();
                Thread t3 = new Thread(delegate() {
                  Explosions.ThreadedUpdate(); });
                t3.Start();

            }
```

Setting the Processor affinity inside the threads for XBox

```
        public void ThreadedUpdate(bool ignoreDestory)//float elapsed,
GameTime agt)
        {
#if XBOX360
            int[] cpu = new int[1];
            cpu[0] = 3;
            Thread.CurrentThread.SetProcessorAffinity(cpu);
#endif

            while (!glb.exiting)
            {
                Thread.Sleep(10);
                Update(elapsed,ignoreDestory, agt);
            }
```

```
        }
```

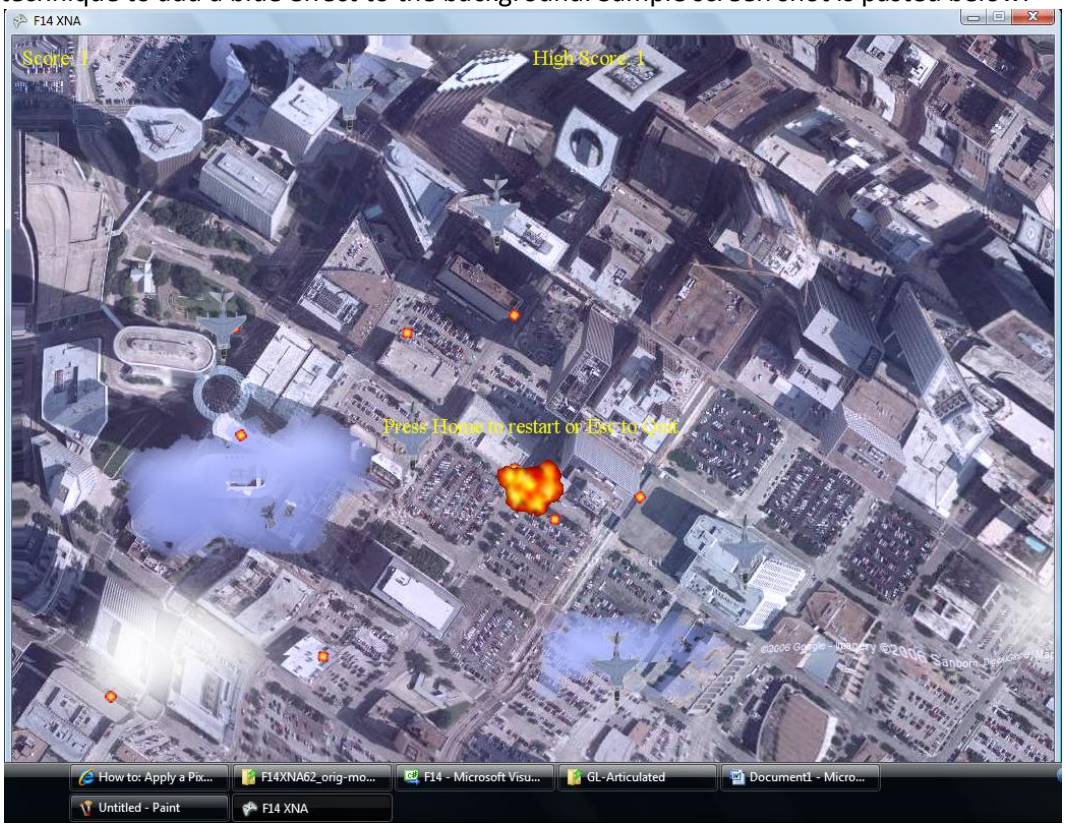
Example of lock used

```
        protected override void UpdateEx(float elapsed, GameTime agt)
        {
            //move enemy
            UpdateMovement(elapsed,agt);

            lock (glb.GroupProjectileslocker)/* here multiple threads are
accessing "GroupProjectiles", I added a lock so that threads access this data
structure sequentially */
            {

                //update the projectiles
                foreach (SceneCollection gp in GroupProjectiles)
                {
                    gp.Update(elapsed, agt);
                    //fire
                    this.Fire(gp);
                }
            }
        }
```

2) Shader: The game did not have any shader at all. I have taken the basic shader code from http://msdn2.microsoft.com/en-us/library/bb313868.aspx and added it to the game. I have applied this shader to the scrolling background and clouds to give a wavy\rippple effect. I have added another technique to add a blue effect to the background. Sample screen shot is pasted below:



How to Deploy on Xbox:

1) Open the project F14 under the directory “F14XNA62” in visual studio
2) Choose the Xbox360 configuration
3) Connect to a Xbox
4) Press F5 and ensure there are no errors
5) Load the game On Xbox from the My Games section and click play.