Further Games & Graphics Assignment 3

# Introduction

In this assignment I am going to cover the particle system used in the flight simulation that I have produced in junction with this report. I will cover the physics system used in the system as well as the AI system that I used in my Simulation. Finally, I will end the assignment with the going through what I have simplified to make the basic simulation happen and what could be improved for the future.

# Particle System

The particle system in my game in based around my game object class. The game object class for each object stores the particle model; transform model and appearance model. The appearance is the model that draws the objects on my game. The transform model is the application of the rotation, scale and positioning of the object. Finally, you have the particle model that calculates all the forces on the particle.

So all the objects in my scene like my aircraft, mountains, towers etc. are a single particle in the collection they are in. They are just a large scaled particle. In the future I could implement a jet stream out the back of the aircraft by creating lots of objects that are small compared to the aircraft and then I will have a jet stream particle effect on my aircraft.

Below is a diagram that shows the relationship between the classes in my game. As you can see the game object class has a many to one relationship with the other classes. This is because the vector of game objects will all have their own transform, particle and appearance classes.



# Physics System

## Force Accumulation

The base model contains the force accumulation model that outputs the final velocity of the aircraft. The force accumulation model for my simulation contains four forces acting on it. These are thrust, lift, gravity and drag. All of these added together in different quantities to make the final force that is exerted on the aircraft to make it move in various directions. This is where I have incorporated the force accumulation equation of “FT = F1 + F2 + F3 + F4” to get the resultant force.

As the aircraft has thrust added to it will add forward moment to the aircraft. There is also drag when the aircraft is on the ground and when the aircraft is in the air. There needs to be enough thrust built up to make the aircraft actually move forward. When the thrust stops being added then the drag will be too high and bring the forward velocity back down to zero.

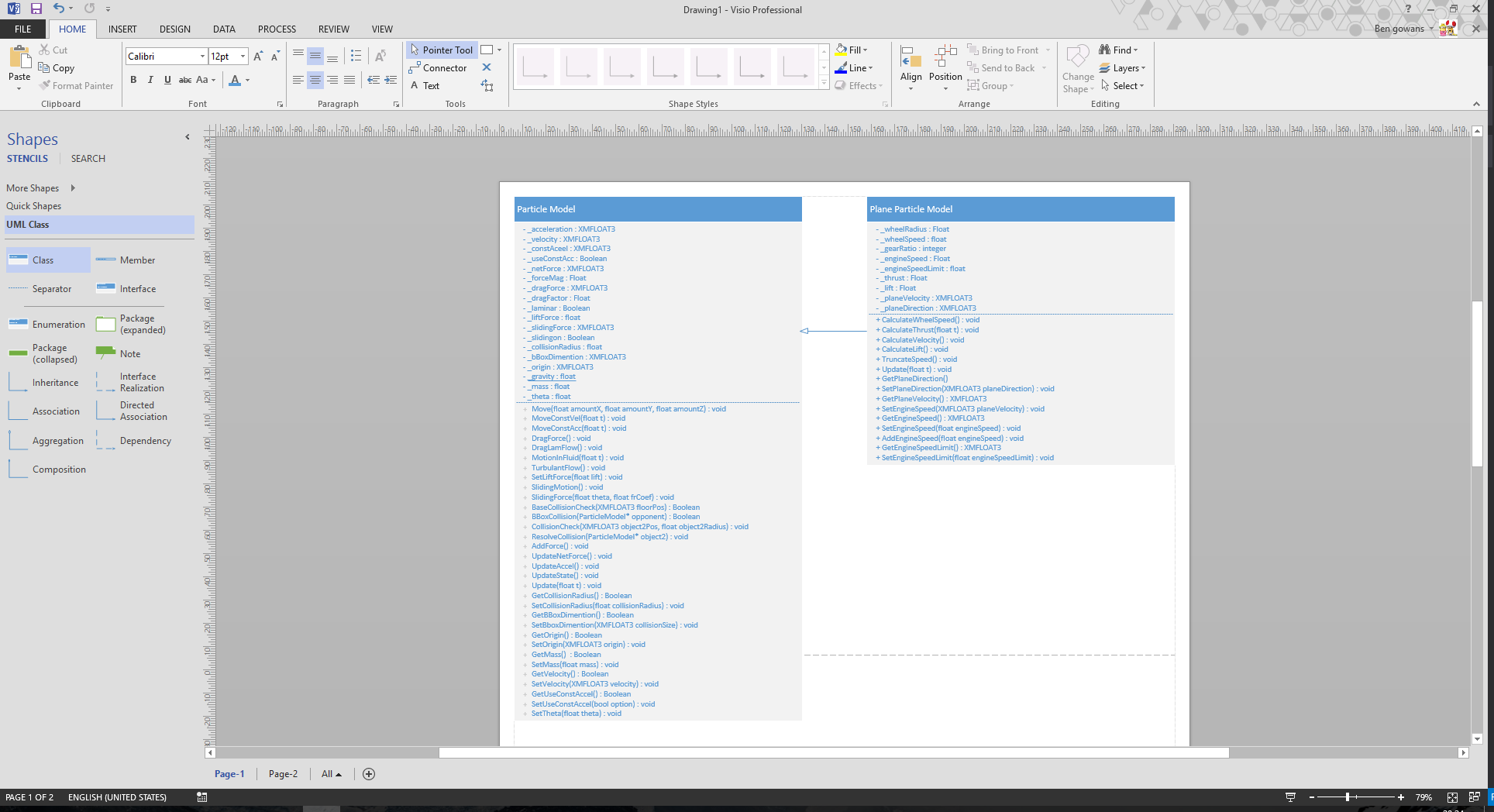
The same can be said for gravity and lift. There is always gravity on the aircraft and that gets multiplied by the mass of the aircraft to try and give it a realistic gravitational value. Then there needs to be enough lift applied to overcome the gravitation force otherwise the aircraft will not take off. The lift is calculated from the thrust that has been applied to aircraft. This works by getting the magnitude of the aircrafts velocity and scaling it up so that it can have a change to other throw the gravitational pull on the aircraft. To get the aircraft to stop at an altitude and not keep climbing; I minus a percentage of the altitude that the aircraft is at until it gets to a certain height where the lift will then equal the gravitational force and thus make it look like it is staying at that altitude.

## Collision

In my simulation I have a broad phase collision check in there for the controlled aircraft on central tower. This collision is box on box collision. I have created an axis aligned bounding box (AABB) around the tower and a AABB box around the aircraft. When the two boxes intersect with each other then it detects a collision has happened and the collision response currently is to negate the current velocity.

There is also collision with ground where if the aircraft gets the height the ground it is at then the height of the aircraft is set to just above it to give the illusion that it has bounced.

The diagram bellow is for the particle model used in my application for the controlled aircraft. It shows the inheritance of the base particle model to the plane particle model. The lift and thrust is in the plane particle model rather than the base particle model because they are unique naming an approaches for an aircraft. So to keep the particle model class useable in different situation like being used for the tower as well then it is kept as so.



# Artificial Intelligence System

The artificial intelligence (AI) in my game is near enough the game as my controlled aircraft the way the particle, transform and appearance models go. The main difference is rather than the user has the control over that aircraft directly; it follows a path that has been pre-determined for it.

Due to the time restrictions I had on this assignment I have implemented following of the waypoints I have created. The aircraft can be at any value on the y-axis but must be in a certain distance of the x-axis and z-axis coordinates. This is because I would have to alter how fast the aircraft turns, rises and falls so that it would hit the waypoint without fail.

On my AI aircraft I have implemented the seek steering behaviour to follow the waypoints. When the state of the aircraft is changed, it will then flee away from the controlled aircraft. For future development I would add in obstacle avoidance and a flocking behaviour so that I can have multiple aircrafts fly in formation with each other.

# Simplifications and Further Development

The ground collision is a very simplified approach to the solution of ground collision. This is because I reset the position of the aircraft when it hits the ground to give the illusion it has stopped on the ground. For future development I would implement a bit of a bounce to the plane as it descends because it would make the system look more realistic.

When the collision response happens on the tower in the middle of the map the response is to negate the current velocity to send it off in the reverse direction. For future development I would improve this simplification by doing a narrow phase check to find the side collided with and to create a more realistic collision effect rather than firing the aircraft backwards. I would also work out a rotating bounding box rather than the current one in affect to fix the collision check.

The current way of working out the direction it is going is to add the forces in force accumulation model and mode it in the direction it produces. The main issue will this is the you can really put the aircraft into a nose dive because you do not have control of how the plane rotates in all axis to make it move. So in the future I would change the system to use the accumulated forces on a forward vector of which the player has move by having control on how the aircraft would react if you were using a flight stick.