User Manual

Versatile Aerodynamics Package

Support: mrk.benecke@yahoo.com

Introduction

General Description:

The Versatile Aerodynamics Package is a behavior solution for implementing flight into your Unity project. Most flight simulation packages fall into one of two categories, those which offer realism and those which offer simplicity. Often, to enter each category, a package will have to make trade-offs. A highly realistic flight simulator invites complexity and technical barriers which could be time-consuming to set up for practical applications. On the other hand, a flight solution that is very simple may be easier to set up and integrate into your project, but will often be plagued by unrealistic physics behavior that could detract from the experience you are creating. This is where the Versatile Aerodynamics Package comes in to offer a balance between realism and ease-of-use, striking a claim somewhere in the happy medium. It accomplishes this mainly because its primary component, the Aerodynamics class, simulates a simple idealized wing using thin airfoil theory. You can arrange multiple wing objects in any orientation and shape you want and the simulation will be able to follow. The package also includes some additional behaviors, such as input controllers and propulsion scripts, to help configure an aircraft, which I imagine is the primary use case for this package, though by no means the only one.

Be warned though, that balancing realism and ease-of-use does come at the cost of relying on the user to make some manual inputs and judgements when configuring your wings, and not to mention some understanding of the basic principles of aviation in order to create a flyable aircraft. I will try to explain the necessary principles as they arise but you are encouraged to learn more on your own. In fact, that you have already read this far into the documentation without giving up is already a sign you have what it takes to make full use of this package.

Hopefully you will have as much fun with this package as much as I have had making it.

Setup Guide:

There is a demo scene included which contains some prefabs for you to inspect, play around with, and learn how everything works. In the prefab folder there is an airplane which is the result of following the instructions below.

1. Creating The Airplane Body
    a. Create an empty gameobject and name it something like “biplane”.
    b. Drag the BiplaneModel asset from the models folder and drop it onto your new gameobject so the model is its child.
    c. Create a new child object for your plane and call it “Colliders”
    d. Now create three or so colliders under the Colliders object in order to represent your plane’s body. I recommend imitating the colliders I made for the sample scene (box collider for the fuselage, box collider for the wings, and a capsule for the landing gear). And for the collider you use to represent the wheels, change the physic material to the Wheels material included with the scene.
    e. Finally, add a rigidbody to your main plane object and set its mass to 300. Your hierarchy should look like this.

2. Adding Aerodynamics To The Plane
    a. First, from the VersatileAerodynamics folder find the Aircraft script and drag it onto your main plane object (same place where the rigidbody is). Anything you want to have aerodynamics simulated on must have one and only one of these components. It has several

instance specific fields such as fluid density, center of mass, and the roll torque multiplier to adjust roll rates.

b. Under Airframe Properties set the New Center of Mass field to be 0,0,-1. This overrides the center of mass of the rigidbody, and is represented by the blue orb gizmo. This is important for making your airplane stable (more info on the center of mass can be found in the Principles of Aviation section).
c. Next, Create an empty child object on your plane and call it something like “Aerodynamics”.
d. Make an empty child object of this new child object and call it “Upper Wing”. Then add the Aerodynamics component to it from the VersatileAerodynamics folder.
e. Set the position of the Upper Wing object to 0, 1.5, -1.5 and the rotation to 4,0,0.
f. Now set the following fields in the Aerodynamics inspector.
    i. Planform_Area = 13.
    ii. Aspect Ratio = 7
g. Duplicate the Upper Wing object and call it “Lower Wing” and set its position to 0,-0.3,-0.6. You have now done everything needed for the main wings to work. But now we want to control the plane.
h. Create a new child called Fuselage, add aerodynamics to it, and input these values.
    i. position = (0,0,-1)
    ii. Planform_area = 5
    iii. Side_area = 7
    iv. Aspect ratio = 0.14

3. Setting Up Control Surfaces
    a. Create four empty gameobjects as children of Aerodynamics and rename them as “Left Aileron”, “Right Aileron”, “Elevator”, and “Rudder”.
    b. Give them these positions
        i. Left aileron: 3.5, -0.4 ,0.
        ii. Right aileron: -3.5, -0.4, 0.
        iii. Elevator: 0, 0.5, 5.
        iv. Rudder: 0, 0.5, 5.

c. Now, from the BiplaneModel find the child named Cube.001 and drag this from its current place in the hierarchy onto Left Aileron so that it is now a child of Left Aileron. If you get a warning preventing you from doing this right click BiplaneMode l->Prefab -> UnpackCompletely and then try again. (moving these will allow us to animate the control surfaces automatically).
d. Repeat the last step as follows
    i. Cube.002 -> Right Aileron
    ii. Cube.004 -> Elevator
    iii. Cube.005 -> Rudder
e. The hierarchy should look like this.

f. Now add an Aerodynamics component to each of the Ailerons, the Elevator, and the Rudder, and set these fields
    i. Ailerons: Planform Area = 2, Aspect Ratio = 7
    ii. Elevator: Planform Area = 2, Aspect Ratio = 2

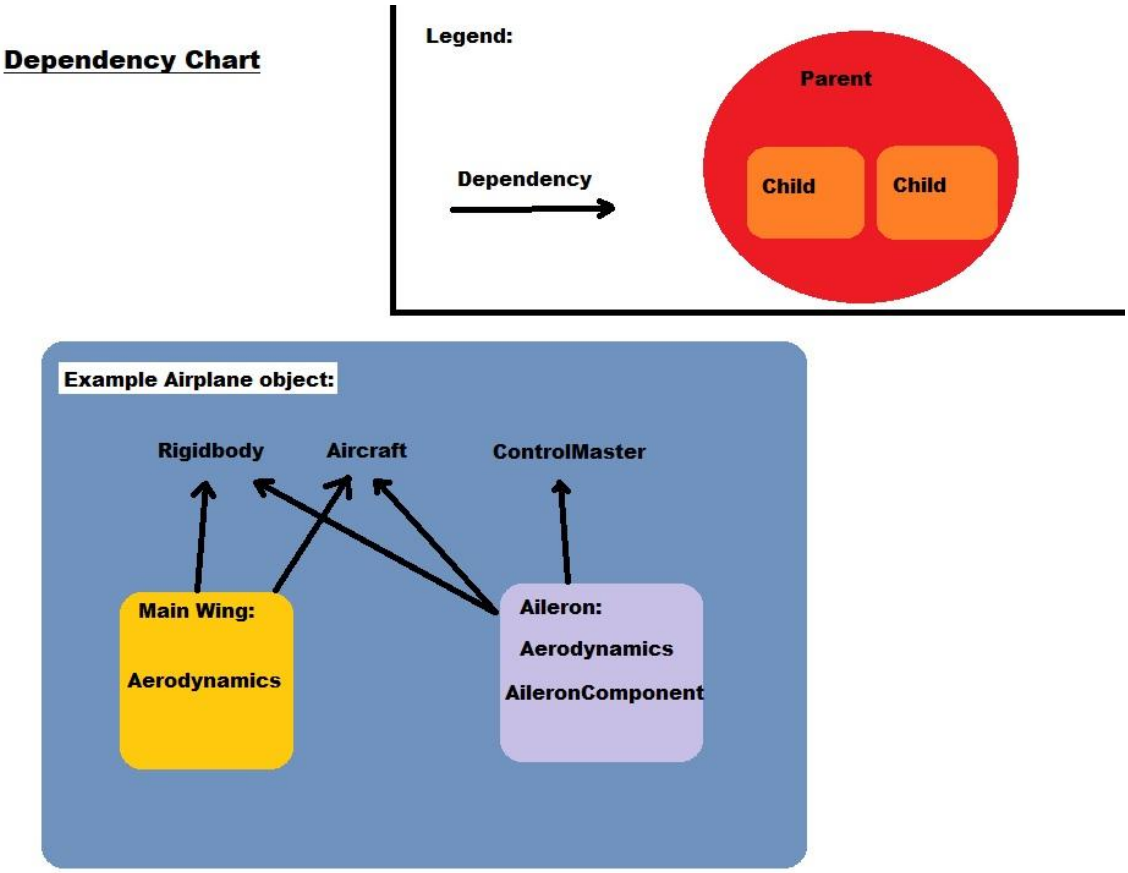
        iii. Rudder: Side Area = 2, Aspect Ratio = 1
    g. Before we can hook up these controls to a control scheme we need an identifier on them. Go to the Control Surface Components folder in Versatile Aerodynamics and drag one of each respective component onto each of its corresponding objects in the hierarchy (this is just an empty behavior so that the Control Master can find these objects).
    h. Finally, place a ControlMaster from the Versatile Aerodynamics folder onto the main plane object (where you placed the Aircraft component and rigidbody). You should see some fields for pitch, yaw, and roll in the inspector under Control Inputs.
    i. Now test that everything works! Hit the play button and try entering some values for roll, pitch, and yaw. You should see the surfaces moving on their own if you set everything up correctly.
4. Setting Up An Engine
    a. Create a new child object of the main plane object and name it “Engine”.
    b. Set the position to 0, 0.15, -2.5 and set the rotation to 0,180,0. (This is because my model was imported backwards and I couldn’t be bothered to fix it.)
    c. Now find the Propulsion script in the Versatile Aerodynamics folder and drag one onto the engine object.
    d. Set the max Thrust field to 1700 and ensure the engine State is in 2.
5. Setting Up Controls
    a. Control inputs should all be handled by the Control Master. It serves as a layer of separation between your player controller and the physical control surfaces of your airplane. Think of this analogy to help you understand its purpose. When a pilot flies a plane and wants to initiate a roll he does not run out along the wing and physically move the aileron one way or another then run to the rudder and push that around; no, that’s absurd. A pilot controls his plane by moving his stick around in the cockpit. The stick is connected mechanically or electrically to the control surfaces so

that the pilot can stay in his seat. This is in essence what Control Master does as well. You put control inputs in Control Master and it will move the surfaces for you.

b. I already have a rudimentary player controller in the demo scene however, you will probably have your own solutions as to how you want to handle player inputs. Use this for now to understand how to interface between a player controller and the Control Master. Find the playerController component in the demo scene folders and place it on the main plane object (where Control Master and the rigidbody are).

6. Take Off!
    a. Use the scroll wheel to power up the throttle and awsd to for the control axes.
    b. If you got your plane to take off and fly then congratulations! You are well on your way to making a plethora of airplanes.

Core Classes



Aerodynamics:

This class is the workhorse of the aerodynamics package. Here is where the physics are simulated. It can only function if there is both a Rigidbody and an Aircraft component in a parent of this object. The usage of this component goes as follows.

One of these components simulates one aerodynamic/lifting surface. The parameters you input into the fields need to be known beforehand based on the shape of the wing you are simulating. You can use this website to help you determine these values (https://www.ecalc.ch/wingdesigner.htm). You can use one

Aerodynamic component for your main wing or break it down to two halves and use two of these to simulate each half separately or break it down even further. Because of this composite nature, you can use many simple wing sections to compose larger and more complex wing shapes. As you can see in the example scene even the control surfaces are all simulated as little individual wings, and it is by deflecting these that the torque can be applied to the plane thus turning it.

The physics works by approximating wings as thin airfoils which leads to simplifications in how the forces are applied. One drawback of this though is that there is no zero angle of attack lift. This means you will have to have an angle of attack to produce any lift at all. Depending on your feedback I may end up implementing cambering airfoil simulation later on.

Fields:

- Aerodynamic Center: (Vector3) This is the position in local space where all aerodynamic forces are applied.
- Planform Area: (float) This is the total area of the wing as seen from a top-down view, or in other words this is the area of the orthographic projection of the wing onto an x-z plane.
- Side Area: (float) like the planform area but the area seen from the side.
- Aspect Ratio: (float) $wingspan^2 / wing\ area$. This is important for drag characteristics and rolling properties. In general, high aspect ratios lead to less drag. A glider has a very high aspect ratio.
- Lift Multiplier: (float) simply multiplies the lift by said amount. Use this if you are having a hard time getting your plane to fly the way you want it to, otherwise, leaving it at its default value of 1.0 is recommended.
- Stall Threshold: (float) The angle in degrees, of the angle of attack, at which airflow separation will begin to take effect and the coefficient of lift will fall leading to the aircraft ‘stalling’.
- Lift Drop Off: (LiftProfile) This option determines what profile the lift coefficient follows after angle of attack surpasses the Stall Threshold. Logarithmic makes the lift coefficient decrease suddenly creating a more dramatic stall and the linear option creates a more gentle lift drop off.

- Weight of Air Friction: (float) The scaler for drag caused by viscous air friction effects on the skin of the surface. Increase to simulate rougher surfaces.
- Weight of Inclination: (float) The scaler for the drag caused by increasing the angle of attack. As the angle of attack increases this value slowly then rapidly increases to simulate air resistance when the wing is perpendicular to the airflow.
- Weight of Induced Drag: (float) Scales the amount of drag resulting from wingtip vortices. The induced drag is highly dependent on the aspect ratio.
- Efficiency: (float) The efficiency is derived from the distribution of lift from wingtip to wingtip. The most efficient distribution is elliptical. A Spitfire has an elliptical distribution because of the shape of its wing, and how it tapers off at the wingtips. A rectangular wing will have an efficiency of 0.7. Values fall between 0 and 1, but never 0.
- Show Forces: (bool) Enable this to display gizmos of the lift and drag vectors being applied by this component in the editor.
- Lift To Drag Ratio: A graph of the lift/drag ratio where the horizontal axis is the angle of attack from 0 to 90 degrees. Do not worry too much if there are strange peaks and juts. Also this graph does not take into account any side area.
- Peak Lift/Drag ratio: This is the peak of the graph above it. Try looking up the L/D of the aircraft you are wanting to simulate. A cessna has L/D of about 12, a glider 25, and 17 for an airliner. After you have input your wing properties such as area and aspect ratio, you can use this value and the graph to fine-tune your main wing. Slowly adjust the weight of inclination and weight of induced drag until you get a profile that looks mostly smooth and peaks at your desired L/D value.

Aircraft:

This class is like what a rigidbody is to colliders in base unity. It organizes groups of aerodynamics components into discernable bodies, and its fields will affect all the aerodynamics components who are children of this object.

Fields:

- New Center of Mass: (Vector3) Sets the center of mass of the airplane during the start method. Usually you will want the center of mass to be a little bit in front of the mean center of lift for a more stable craft.
- Roll Torque Modifier: (float) The aerodynamics class at the same time as calculating the lift also calculates differential lift across a rolling wing. Picture this, as an airplane rolls the angle of attack on the left half of the plane will be different than the right half. This difference in lift creates a torque which is opposite the roll direction. The result is that there is a dampening which creates a peak rate of roll. Depending on what you desire you may not want a “sticky” rolling plane. This field allows you to scale the roll dampening how you want it.TLDR, a smaller value here allows for faster roll rate.
- Fluid Density: (float) The value for air density. Pretty self explanatory. You can access this field from scripts to control things like atmospheric changes in air pressure. Adjusting this value is also one of the most effective ways of controlling the speed of all the aircraft in your project. Higher values of air density mean more lift is generated which means you can fly at slower speeds, but it also means more drag will also be generated.
- Wind: (Vector3) This is a vector representing the wind direction and strength. Not much else to it.

ControlMaster:

This class acts as an intermediary between your own player input solution and the movement of control surfaces. Although you do not need to use this for the base aerodynamic physics to work, I included this to make implementing controls easier. The way I envisioned controls to work in my package is that the control surfaces themselves would be simulated as small wings the same as any other wing, except these are ones you can turn. This component should only be placed right next to where your Aircraft component is. Then you simply set the public fields of roll, yaw, etc…from anywhere in your project and this class will take care of the rest. For your control surfaces to be recognized by the ControlMaster however, you will need to place ControlSurfaceComponents on your different surfaces. You can have any number of rudders, elevators, etc. these empty identifier components simply allow the control master to find all of them.

Fields:

- Movement Profile: Different animation styles for the movement of the surfaces.
- Responsiveness: (float) how much time it takes the control surfaces to move to their set position. Lower values make for more sluggish controls.
- Control Surface Force: (float) The maximum amount of force that the surface can deflect. As your airspeed increases the force of the air pushing back against control surfaces will increase. Setting this to lower values will allow your plane's controls to lock up at high airspeed.
- Max […] Deflection: The maximum angle in degrees that a control surface will be able to deflect. It is recommended to keep these around the stall threshold.
- Control Inputs: The Pitch, Yaw, Roll, and Throttle are all accessible properties as long as you have a reference to the controlMaster. This is where you will set player inputs. Acceptable values range from -1 to 1.

Resources

For an introduction to unde some of the principles of aviation and terminology try reading from here. https://www.grc.nasa.gov/www/k-12/airplane/guided.htm

Airfoils and Thin Airfoil Theory. https://en.wikipedia.org/wiki/Airfoil

Wing designer tool. https://www.ecalc.ch/wingdesigner.htm

Lift to drag ratio. https://en.wikipedia.org/wiki/Lift-to-drag_ratio