Modeling Turbines and Vortices with VPython

CHARLES TIERNEY

MATH 198

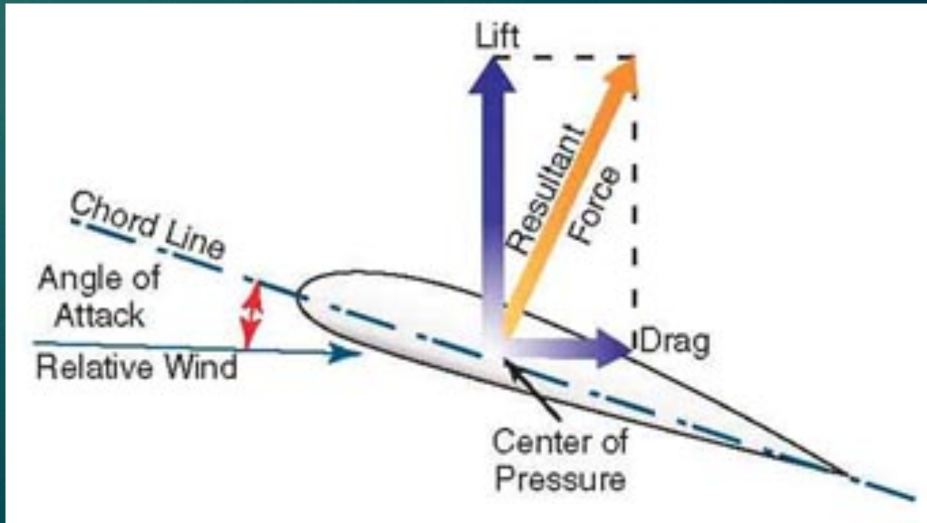
DECEMBER 15, 2015

Project Overview

- Goals
  - Model Wind Turbine
  - Model wake of vortices as turbine rotates
  - Create model wing to demonstrate forces acting on wing
  - Create streams to demonstrate vertical paths

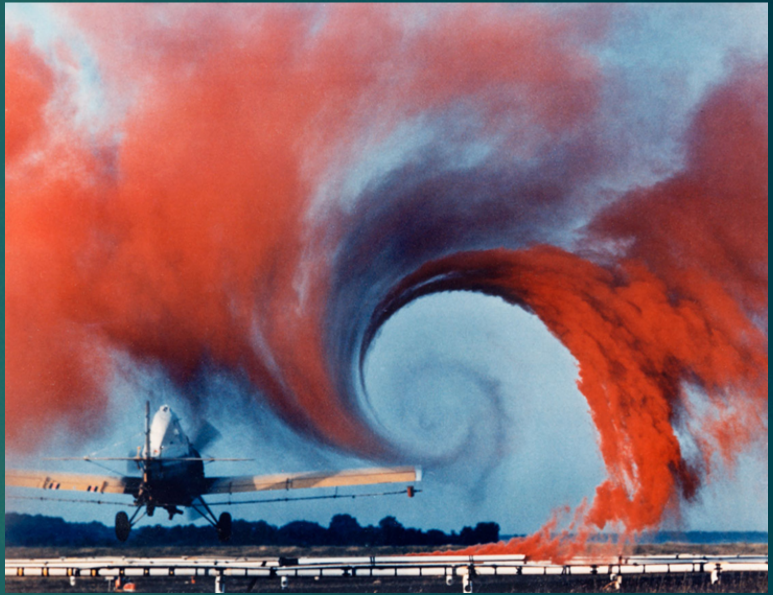
Review of Aerodynamic Forces

- Lift:
  - Reaction force to downward displacement of air, and result of pressure gradient from bottom to top of wing.
  - Acts Perpendicular to relative motion
- Drag:
  - Result of viscous effects acting on wing, also referred to as "air resistance."
  - Acts parallel to relative motion and opposes motion.
- Induced Drag
  - Result of vortices generated at the tips of wings and the "downwash" that occurs.
  - Reduces the effective angle of attack



Review Of Vortices

- Vortices result from the combination of *spanwise* (from the root of the wing to the wing tip) flow over the wing and *chordwise* (from the leading edge to the trailing edge) flow over the wing, due to the pressure gradient from the bottom to top of the wing.
- This same gradient creates lift, and the size of the vortices is generally directly proportional to the lift generated.
- These vortices produce a *downwash* on the wing, which reduces the effect angle of attack and thus the lift acting on the object



Review of Winglets

- The addition of winglets increases the effective *aspect ratio* (ratio of span to chord length) of the wing, which decreases the induced drag.
- Winglets are necessary on turbines because they cannot modify their angle of attack like planes can.
- This reduction in induced drag increases the power output of the wind turbines.

Turbines

Turbine RTICA Overview

- The turbine RTICA “vpython_turbine.py” models a Horizontal Axis Wing Turbine with three blades and winglets.
- As the turbine rotates, it traces a wake behind it to show where vortices would appear.
- Allows user to control the wind speed, and view how this affects both the speed of rotation and the turbine wake

Turbine Video

- http://www.screencast.com/users/ctierne2/folders/Jing/media/f45068fe-0be6-4c8b-80a9-89aa44da1929

Turbine RTICA

- Creation of three levels of frames:
  - World Frame
  - Blade Frames
  - Winglet Frames
- During the while loop, the entire turbine moves forward, appending the new position of each of the winglets to the three wake curves.

Frame Declarations

```
world_fr=frame()
blade_fr1=frame(frame=world_fr)
blade_fr2=frame(frame=world_fr)
blade_fr3=frame(frame=world_fr)
winglet_fr1=frame(frame=blade_fr1)
winglet_fr2=frame(frame=blade_fr2)
winglet_fr3=frame(frame=blade_fr3)
```

```
blade_fr1.axis=(1,0,0)
blade_fr2.axis=(cos(2*3.14/3), sin(2*3.14/3), 0)
blade_fr3.axis=(cos(4*3.14/3), sin(4*3.14/3), 0)
blade_fr1.pos=(.75,1,0)
blade_fr2.pos=(.75*cos(2*3.14/3), .75*sin(2*3.14/3)+1, 0)
blade_fr3.pos=(.75*cos(4*3.14/3), .75*sin(4*3.14/3)+1, 0)
winglet_fr1.pos=(.75,0,0)
winglet_fr2.pos=(.75,0,0)
winglet_fr3.pos=(.75,0,0)
```

Wake Function

```
new_pos1=blade_fr1.frame_to_world(winglet_fr1.frame_to_world(winglet1.pos))
new_pos2=blade_fr2.frame_to_world(winglet_fr2.frame_to_world(winglet2.pos))
new_pos3=blade_fr3.frame_to_world(winglet_fr3.frame_to_world(winglet3.pos))
wake1.append(pos=(new_pos1.x, new_pos1.y, new_pos1.z+increment), retain=1000)
wake2.append(pos=(new_pos2.x, new_pos2.y, new_pos2.z+increment), retain=1000)
wake3.append(pos=(new_pos3.x, new_pos3.y, new_pos3.z+increment), retain=1000)
```

Forces RTICA Overview

- The Forces RTICA demonstrates the three main aerodynamic forces acting on the wing (Lift, Drag, and Lift-Induced Drag)
- Allows user to modify velocity, the length of the blade, and the angle of attack.
- Features six “streams” of air particles traveling over the blade to demonstrate both normal paths over the blade and vortical paths.

Forces RTICA video

- http://www.screencast.com/users/ctierne2/folders/Jing/media/b89a3166-5dca-44eb-8281-5c0a39de6ab8

Forces RTICA

- Blade is created from extrusion of an airfoil, whose data points are carefully plotted and accurate.
- Key handling function handles all key events, and allows for manipulation of velocity, length and angle of attack. All values printed to the console as they change.
- While loop at the end of the script handles animation of the "streams" as well as updates the axis lengths for each of the forces.

Example “Normal” Stream Loop

```
for i in range(len(stream1.pos)):
    if stream1.pos[i][0] > 1:
        stream1.pos[i]=(stream1.pos[i][0]-.025,0,0)
    elif stream1.pos[i][0] < 1 and stream1.pos[i][0] > .5:
        stream1.pos[i]=(stream1.pos[i][0]-.025,stream1.pos[i][1]+.025,0)
    elif stream1.pos[i][0] < .5 and stream1.pos[i][0] > 0:
        stream1.pos[i]=(stream1.pos[i][0]-.025,stream1.pos[i][1]-.025,0)
    elif stream1.pos[i][0] < 0 and stream1.pos[i][0] > -5:
        stream1.pos[i]=(stream1.pos[i][0]-.025,0,0)
    elif stream1.pos[i][0] < -5:
        stream1.pos[i]=(5,0,0)
```

Example “Vortical” Stream Loop

```
for i in range(len(stream5.pos)):
    if stream5.pos[i][0] > 2:
        stream5.pos[i]=(stream5.pos[i][0]-.025,0,-3)
    elif stream5.pos[i][0] < 2 and stream5.pos[i][0] > 1.3:
        stream5.pos[i]=(stream5.pos[i][0]-.025,stream5.pos[i][1]+.025,stream5.pos[i][2]-.025)
    elif stream5.pos[i][0] < 1.3 and stream5.pos[i][0] > .6:
        stream5.pos[i]=(stream5.pos[i][0]-.025,stream5.pos[i][1]+.025,stream5.pos[i][2]+.025)
    elif stream5.pos[i][0] < .6 and stream5.pos[i][0] > -.1:
        stream5.pos[i]=(stream5.pos[i][0]-.025,stream5.pos[i][1]-.025,stream5.pos[i][2]+.025)
    elif stream5.pos[i][0] < -.1 and stream5.pos[i][0] > -.8:
        stream5.pos[i]=(stream5.pos[i][0]-.025,stream5.pos[i][1]-.025,stream5.pos[i][2]-.025)
    elif stream5.pos[i][0] < -.8 and stream5.pos[i][0] > -5:
        stream5.pos[i]=(stream5.pos[i][0]-.025,0,-3)
    elif stream5.pos[i][0] < -5:
        stream5.pos[i]=(5,0,-3)
```

Future Expansions

- More accurate Stream-line representations
- Including vorticity fields for the wings
- Improving the aesthetics of the programs
- PyOpenGL? Perhaps it is more powerful
- More user-friendly view manipulation
- Include “embellishments” for easy instruction during RTICA execution

Image Sources

- http://bandung-aeromodeling.com/tutorials/aerodynamic_forces-03.jpg (Lift and Drag)
- http://howthingsfly.si.edu/sites/default/files/image-large/il_wingtipvortexedit_lg.jpg (vortices)
- http://www.planetmattersandmore.com/wp-content/uploads/2012/04/wind-turbine-farm.jpg (turbines)