

3D Visualization of Cooperative Trajectories

John A. Schaefer*

Missouri University of Science and Technology, Rolla, MO, 65409

Aerodynamicists and biologists have long recognized the benefits of formation flight. When birds or aircraft fly in the upwash region of the vortex generated by leaders in a formation, induced drag is reduced for the trail bird or aircraft, and efficiency improves. The major consequence of this is that fuel consumption can be greatly reduced. When two aircraft are separated by a large enough longitudinal distance, the aircraft are said to be flying in a cooperative trajectory. A simulation has been developed to model autonomous cooperative trajectories of aircraft; however it does not provide any 3D representation of the multi-body system dynamics. The topic of this research is the development of an accurate visualization of the multi-body system observable in a 3D environment. This visualization includes two aircraft (lead and trail), a landscape for a static reference, and simplified models of the vortex dynamics and trajectories at several locations between the aircraft.

Nomenclature

AC1	=	lead aircraft in formation
AC2	=	trail aircraft in formation
VRML	=	Simulink Virtual Reality Modeling Language
v	=	free stream velocity of AC2
w	=	induced upwash from the vortex of AC1
α, L, D	=	angle of attack, lift, and drag of AC2
$\Delta\alpha, \Delta L, \Delta D$	=	change in angle of attack, lift, and drag of AC2 due to w
ϕ, θ, Ψ	=	roll, pitch, and yaw angles [rad]
SORA	=	Simultaneous Orthogonal Rotation Angle
$\delta_x, \delta_y, \delta_z$	=	orthogonal rotations about the x, y, z axes [rad]
\mathbf{k}	=	unit vector used in axis-angle representation
Θ	=	angle used in axis-angle representation [rad]
b	=	wingspan [ft]
b_0	=	spanwise separation of vortex pair after rollup [ft]
d	=	decent velocity of vortex system [ft/s]
W	=	weight [lb]
ρ	=	density of air [slugs/ft ³]
\mathbf{x}_w	=	wake object position
\mathbf{x}_{w0}	=	initial wake object position
\mathbf{v}	=	total wake object velocity
Δt	=	time since wake object generation

I. Introduction

DRAG reduction is possible during formation flight when two more aircraft fly in such a way that a trail aircraft positions itself in the upwash region generated by the wing tip vortex of a lead aircraft. This positioning rotates the lift vector of the trail aircraft forward, reducing induced drag and improving fuel efficiency. The geometry of formation flight is depicted in Fig. 1, and the rotation of the lift vector is depicted in Fig. 2. The distance from AC2 to the vortex of AC1 is of particular importance. The three components of this distance are designated long track, cross track, and vertical track. When the long track between aircraft is deemed to be significantly large—that is, on the order of 25 to 50 times the wingspan of the lead aircraft—the aircraft are said to

*Summer Intern at NASA Dryden Flight Research Center, Dynamics and Controls Branch, AIAA Student Member.

be flying in a cooperative trajectory. This scenario allows the trail aircraft to retain most of the benefits of formation flight while maintaining a safe following distance. As the price of fuel continues to increase, cooperative trajectories could aid in the effort to lower both military and civilian operating costs.

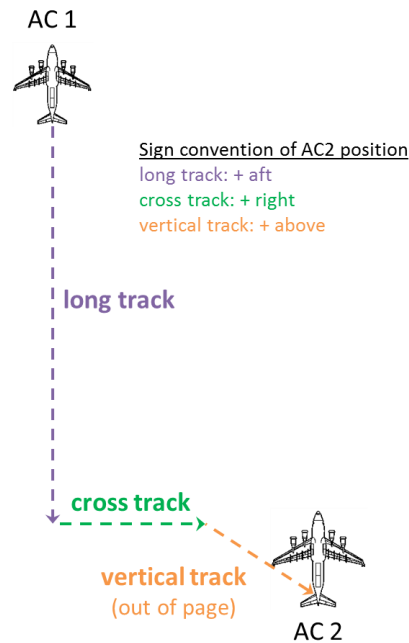


Figure 1. Formation flight geometry. Long track, cross track, and vertical track are the displacements of AC2 measured from the wake of AC1. Long track is positive in the aft direction, cross track to the right, and vertical track up. Each is shown in the positive sense here.

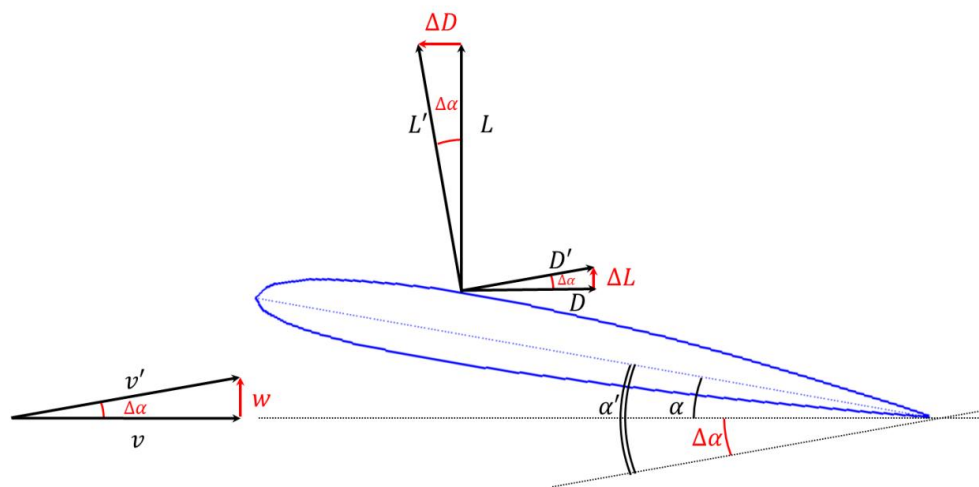


Figure 2. Rotation of the lift vector. The induced upwash (w) from the vortex of AC1 rotates the attack of AC2, α , by $\Delta\alpha$ to α' . This rotates the lift and drag vectors, L and D , to L' and D' by ΔL and ΔD . The free stream velocity, v , is rotated to v' . Note that ΔD points upstream; this indicates that drag is reduced for AC2.

The concepts of formation flight and cooperative trajectories have been tested in flight. In 1996, two Do-28 aircraft achieved a 15% power reduction while flying in close formation.¹ In 2000, two F/A-18 aircraft achieved a 20% drag reduction and an 18% fuel flow reduction with similar aircraft spacing.² In 2010, two C-17 aircraft achieved a fuel flow reduction of 7-8% with a long track of approximately 18 wingspans.³

Since the completion of these experiments, a Simulink simulation has been developed to model an automatically piloted cooperative trajectory. The simulation was developed by the NASA Langley Research Center and the NASA Dryden Flight Research Center. It includes three major subsystems: Two aircraft models and a vortex and relative navigation subsystem. A simplified diagram of the simulation is included in Fig. 3.

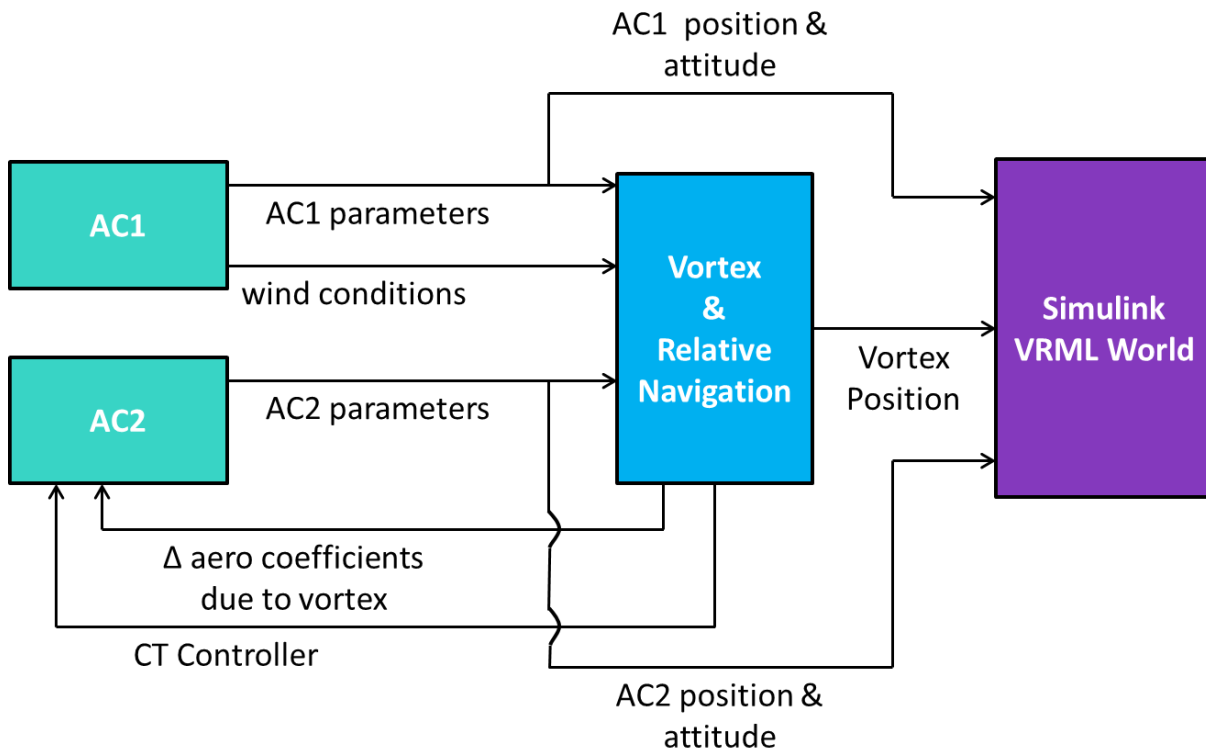


Figure 3. Simplified cooperative trajectory Simulink model. Parameters from both aircraft are constantly fed into the vortex and relative navigation subsystem, where formation flight control laws are computed. The cooperative trajectory controller sends commands to AC2, and the cycle repeats. Simultaneously, geometric data for both aircraft and the vortices from AC1 are sent to the Simulink VRML world subsystem.

The aircraft models are fully non-linear with 6 degrees of freedom and contain detailed models of actuator and engine dynamics, aerodynamics, sensors, gravity, atmosphere, and landing gear. They represent twin engine, civilian transport sized aircraft. The lead and trail aircraft in formation are designated AC1 and AC2 respectively. In addition to the aforementioned parameters, both aircraft subsystems contain a wind model. The output of the wind model from the AC1 subsystem is routed to the vortex and relative navigation subsystem. See Ref. 4 for more information on the aircraft models.

The vortex and relative navigation subsystem takes inputs from the two aircraft subsystems. From this information, a simplified vortex dynamics model predicts the position and movement of the wake generated by AC1.⁵ Formation flight control laws then dictate commands for AC2.

As the simulation progresses, the positions and attitudes of AC1, AC2, and the wake region of AC1 are continuously transferred into a Simulink Virtual Reality Modeling Language⁶ (VRML) subsystem. Within the subsystem, these inputs are manipulated appropriately so that they can be used in a VRML world, which is the topic of this research.

II. Simulink VRML World

A VRML world is a fully immersive 3D environment which allows inputs from a Simulink model to control virtual objects in real time, or near real time. The VRML editor is a tree based application in which objects are created and manipulated, and it is used to set up the baseline parameters of a VRML world. Objects can be visible (such as an aircraft) or invisible (such as a camera viewpoint). They contain variables such as position, rotation, color, texture, etc. Often times a single object will consist of many smaller objects sharing the same local coordinate system. For instance, a house modeled in VRML would contain individual walls, windows, doors, and a roof. When this occurs, the large encompassing object is referred to as a “parent node” and all of the smaller objects are referred to as “children” of the parent node. An example of the VRML tree structure is included in Fig. 4.

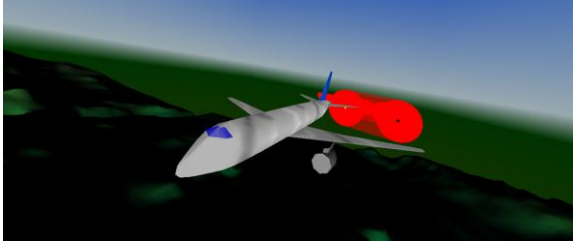


Figure 4. VRML tree structure. This screenshot shows an example of the VRML tree structure. Notice that the nodes “Lead_757” and “AC1_Transforms” contain properties such as scale, rotation, and translation. All physical components of the aircraft are children of the “AC1_transforms” node. In the figure, the “Bottom_Left_Horz” node (representing the bottom left horizontal tail surface of AC1) is expanded. It contains properties such as geometry, texture, and material.

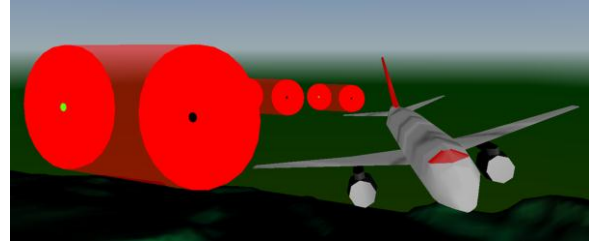
VRML has its own specific Cartesian coordinate system which is inconveniently different than any of the standard coordinate systems used in aerospace engineering or in the Simulink Aerospace Blockset.⁷ The problems caused by this inconvenience will be discussed individually as they relate to each of the objects in the following sections. For the cooperative trajectory simulation, a VRML world was created which includes two B-757 aircraft, a ground and horizon, mountainous terrain, visualized wake objects, and several camera viewpoints.

A. Aircraft Objects and Dynamics

The VRML editor includes a library containing a large number of pre-built objects. From this library, two B-757 objects were selected for their use in the VRML world. The B-757 objects are parent nodes containing many children, such as the wings, fuselage, and engines. The children were individually assigned appearance parameters. In order to visually discern the difference between the two aircraft while viewing the VRML world, AC1 and AC2 were assigned blue and red vertical tails and windshields respectively. Images of the B-757 objects are included in Fig. 5.



(a)



(b)

Figure 5. B-757 objects. (a) AC1, with a blue windshield and vertical tail, and (b) AC2, with a red windshield and vertical tail. The visualized wake objects are generated aft of AC1, and AC2 is observed flying near them.

The position of each aircraft is output in a geodetic latitude-longitude-altitude (LLA) coordinate system from the cooperative trajectories simulation. VRML, however, operates in a unique Cartesian coordinate system. In order to transfer the aircraft positions to VRML, several Simulink blocks are exploited. An *LLA to Flat Earth* block has inputs of geodetic latitude, longitude, and altitude and an output in flat earth space. A custom Simulink *MATLAB Function* block converts flat earth coordinates to VRML coordinates. All of the coordinate systems discussed in this section and later sections are depicted in Fig. 6. Note the difference between the standard aircraft coordinate system and the VRML coordinate system. By default, the B-757 objects are oriented differently than the accepted standard.

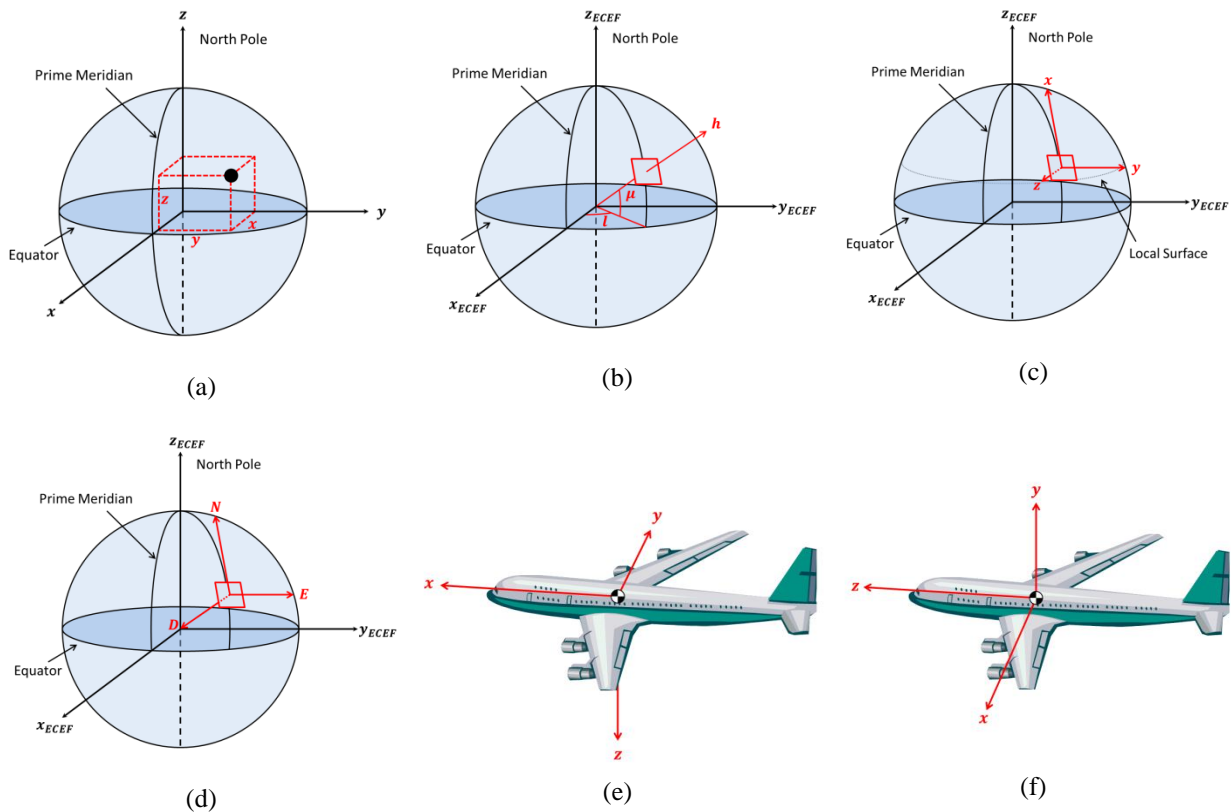


Figure 6. Coordinate Systems. (a) Earth-Centered-Earth-Fixed (ECEF), (b) latitude-longitude-altitude (LLA), (c) Flat Earth, (d) North-East-Down (NED), (e) Standard aircraft coordinate system, (f) VRML aircraft coordinate system

The flat earth coordinate system is defined within the *LLA to Flat Earth* block with the x-axis pointing north, y-axis east, and z-axis down. It uses an ellipsoidal earth model and a reference longitude and latitude equal to the initial longitude and latitude of AC1. At this initial position, a plane is defined at the surface of the earth, and as the simulation progresses all subsequent object positions are defined in reference to this plane. The flat earth approximation yields errors for long distances away from the initial longitude and latitude, however for the relatively short simulation times employed here these errors are negligible. If higher fidelity positions are desired, a North-East-Down (NED) coordinate system should be employed. Due to increased complexity and continued complications, a NED coordinate system was not used here.

Another problem encountered with VRML was the representation of rotations. Aircraft rotations are output from the simulation in terms of pitch, roll and yaw (ϕ, θ, Ψ) angles, however VRML uses an axis-angle representation for all rotations. In an axis-angle representation, a single angle is defined to rotate about a unit vector. To convert ϕ, θ , and Ψ to the axis-angle representation, a simultaneous orthogonal rotation angle (SORA) function was written using Eq. (1) and Eq. (2). The function was implemented into the simulation using a *MATLAB Function* block. It is included in Appendix A.

$$\mathbf{k} = \frac{1}{\sqrt{\delta_x^2 + \delta_y^2 + \delta_z^2}} \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_z \end{bmatrix} \quad (1)$$

$$\Theta = \sqrt{\delta_x^2 + \delta_y^2 + \delta_z^2} \quad (2)$$

Where \mathbf{k} is the unit vector about which to rotate, Θ is the rotation angle, and $\delta_x, \delta_y, \delta_z$ are the orthogonal rotations about the x, y, z axes. A graphic depicting this conversion is included in Fig. 7.

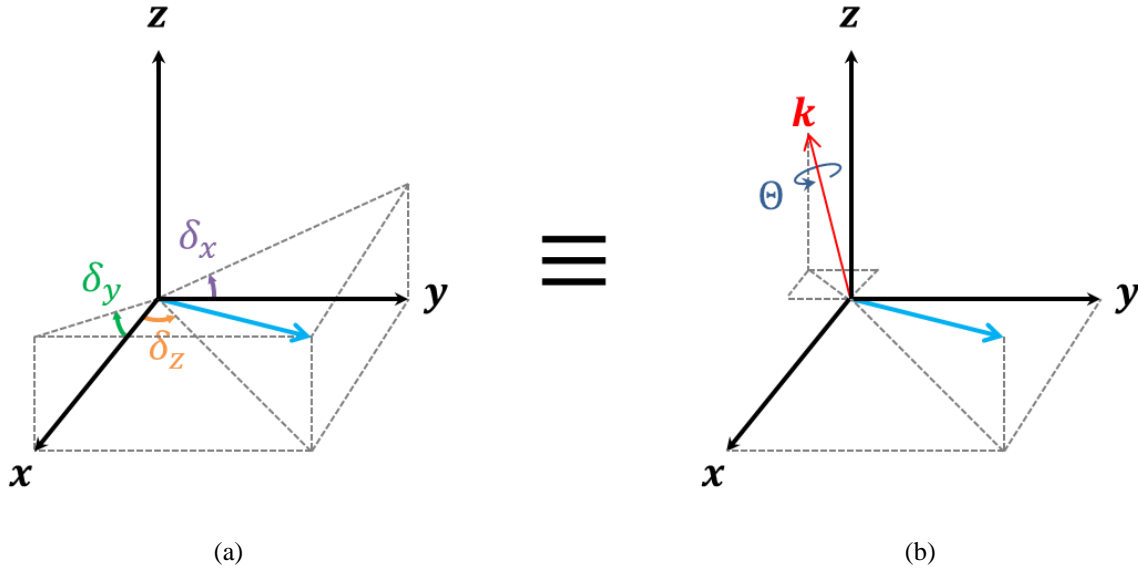


Figure 7. Simultaneous Orthogonal Rotation Angle. In this example, a **vector** originally in the direction of the x-axis is rotated by the three orthogonal rotations (a) $\delta_x, \delta_y, \delta_z$. These rotations are equivalently represented by (b) the rotation through Θ about \mathbf{k} .

B. Static Elements

The VRML world includes two static elements: a ground/sky background, and a mountainous terrain. At first these features may seem entirely aesthetic, however they have a very important purpose to the VRML world; without them, there is no visual indication of aircraft velocity or direction. While the purpose of the VRML world is

mostly to view relative aircraft dynamics, the inclusion of a static reference facilitated a much easier environment for the development of coordinate transformations. For instance, if the simulation was started and the aircraft appeared to be flying forward but pointing straight down, this would indicate that something was wrong with the SORA function.

The ground/sky background object was slightly modified from a pre-built object. It is fixed at the origin of the VRML world. The mountainous terrain was adapted from the “Terrain Visualization Example” published on the MathWorks website.⁸ The terrain object’s position is determined using *MATLAB Function*, *Step*, *Switch*, and *Memory* blocks. The *MATLAB Function* block places the terrain object at a fixed location with respect to AC1. The subsequent blocks then hold this initial position so that the terrain object does not move with AC1. A screenshot of the VRML world containing the static elements is included in Fig. 8.

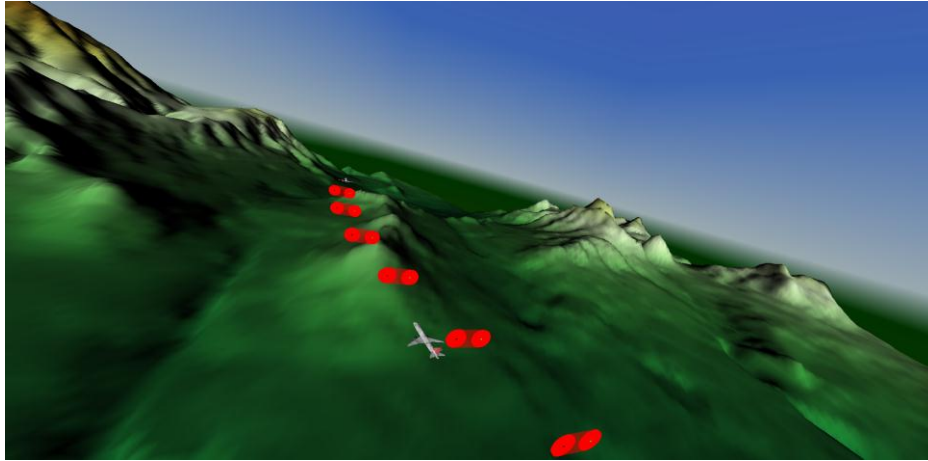


Figure 8. Static elements of VRML world. This screenshot includes both aircraft and several wake objects passing over the terrain object with the ground/sky object in the background.

C. Wake Objects and Vortex Dynamics

Perhaps the most critical displays in the VRML world are the wake objects. This series of objects provides a visual representation of the wake generated by AC1. The inclusion of these objects allows for the observation of AC2’s interaction with the wake.

One consideration was to determine how the wake should be represented. Many ideas have been proposed, but ultimately a display was chosen similar to that proposed by Holforty.⁹ This display includes both wingtips’ vortex cores, circular areas of influence, and a center “no-fly” region. The rolled up core-to-core distance (b_0) was taken from Hummel¹ and the vortex core radius was determined by Delisi et al.¹⁰ The radius of the area of influence was determined empirically. A MATLAB wake model was used to predict upwash and downwash distributions given the size, weight, and velocity of an aircraft. After running several cases, the boundary of the area of influence was chosen at a location where most upwash velocities did not exceed 10-15 ft/s. In the VRML world, the port and starboard vortex cores are represented by thin, solid black and bright green cylinders respectively; the areas of influence by thin, solid red cylinders; and the “no-fly” region by a thin, semi-transparent red rectangle. The 2D and 3D representations of the wake are included in Fig. 9.

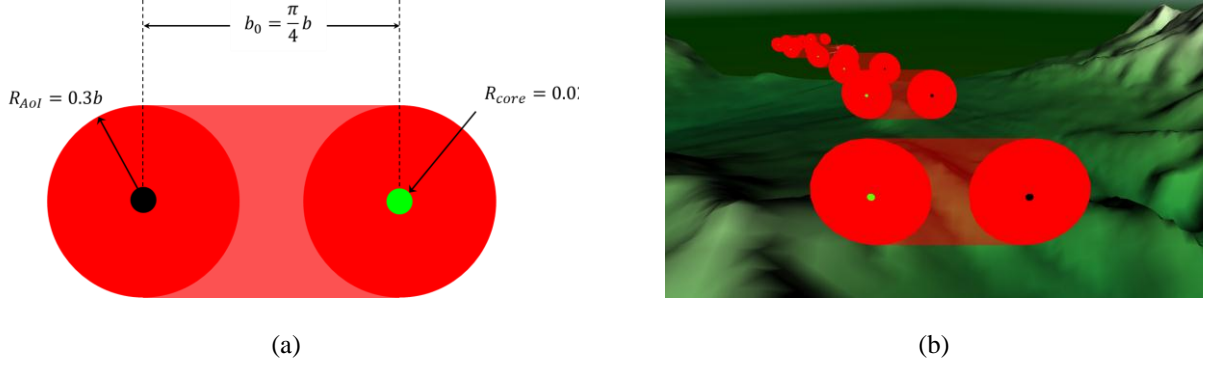


Figure 9. Wake representation. (a) Wake object dimensions and (b) 3D representation in VRML world. Note that (a) is observed from behind the wake and (b) from in front.

Simplified vortex dynamics were also considered. In straight and level flight, the unperturbed, self-induced decent velocity (d) of the vortex system is a function of aircraft weight (W), air density (ρ), aircraft velocity (V), and b_0 . This function was taken from Holforty.⁹

$$d = \frac{W}{2\pi\rho V b_0^2} \quad (3)$$

The vortex decent velocity is then added to the local wind velocity to obtain the complete 3D velocity of the vortex pair. Recall that VRML requires position, and not velocity inputs for the translation of objects. At the time that each wake object is generated, it obtains the initial conditions from w and wind velocities. As the simulation time progresses, these velocities remain locally constant for each wake object. Therefore the current position of each wake object (\mathbf{x}_w) can be represented by its initial position (\mathbf{x}_{w0}), velocity (\mathbf{v}), and time since its generation (Δt).

$$\mathbf{x}_w = \mathbf{x}_{w0} + \mathbf{v}\Delta t \quad (4)$$

In addition to position commands, the wake objects receive rotation commands for their orientation. The rotation of each wake is obtained from Ψ of AC1 after it passes through the SORA function.

Even without the use of wake objects, the simulation runs noticeably slower with the VRML viewer open. There were concerns, therefore, that using too many wake objects would slow down the simulation to unacceptable speeds. Furthermore, VRML objects cannot be easily created outside of the VRML editor, so all wake objects must exist in the VRML world before the simulation starts. Manually creating a large number of wake objects would be exceedingly tedious and impractical.

To address these problems, it was decided that a smaller number of wake objects would be created in the editor, and then recycled after a certain time period. The basic idea behind this technique is that after a certain time period, a wake object will be positioned far enough behind AC2 that it is no longer a concern to the simulation user viewing the VRML world. While the simulation is running, vortex pair positions are calculated for 21 seconds after they are generated by AC1. Each wake object tracks a vortex pair for 1 second until that vortex pair is passed on to the next wake object. The original wake object is then recycled and takes on the position of the newest calculated vortex pair. The recycling occurs at a rate of 1 Hz to match the output capabilities of a commercially available aircraft data transmission system, Automatic Dependent Surveillance-Broadcast (ADS-B).¹¹ This process is described pictorially in Fig. 10.

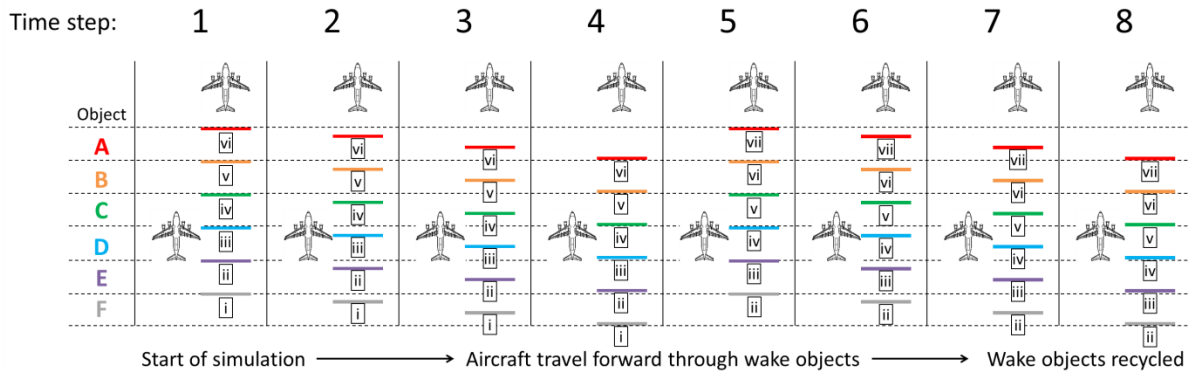


Figure 10. Time history of recycled wake objects. This figure depicts the recycling of wake objects (A-F). As the simulation time increases, the aircraft move forward through the atmosphere and the wake objects. After a certain duration (4 time steps in this case), the objects are recycled. When recycling takes place, subsequent wake objects take the positions of vortex pairs (i-vii) from previous wake objects.

In Fig. 10, **Object D** and **Object E** represent the vortex pairs **iii** and **ii** respectively for time steps 1-4. Recycling takes place between time steps 4 and 5, where **Object E** takes vortex pair **iii**, and **Object D** takes vortex pair **iv**. Vortex pair **ii** is assigned to **Object F** at this time and vortex pair **i** disappears. The recycling process occurs for all wake objects. In the VRML world all of the wake objects are identical in shape and color, so the object transition occurs smoothly in a visual sense.

In the simulation, all of the vortex pair positions are contained in a time-changing [21 x 3] array, where each row represents the ECEF position of a single vortex pair. Each row is constrained to a 1 second interval, which in turn constrains the physical space occupied by the wake object in the VRML world. The resulting physical bounds are represented by dashed lines in Fig. 10. As an example, **Object C** is always driven by the 3rd row of the array and is always the 3rd object behind AC1. Before recycling, vortex pair **iv** is the 3rd behind AC1. When a new vortex pair is created (**vii**) and recycling occurs, vortex pair **v** becomes the 3rd behind AC1, and **Object C** assumes the position of **v**.

Transforming simulation coordinates into VRML coordinates was once again inconvenient for the wake objects. The simulation operates at a frequency of 50 Hz and generates wake locations as a frequency of 1 Hz. At any given time, there exists data for up to 21 wake locations. This data is in an ECEF coordinate system, and several blocks from the Simulink Aerospace Blockset are required to obtain VRML coordinates. First, an *ECEF Position to LLA* block is used which has inputs of ECEF position and outputs of geodetic latitude, longitude, and altitude. These output parameters, along with a reference altitude of 0, are used as inputs for an *LLA to Flat Earth* block which operates in the same manner as that described for the aircraft. Similarly, a *MATLAB Function* block is used to convert flat earth coordinates to VRML coordinates.

III. Conclusions and Future Work

A Simulink VRML world has been created for the observation of a cooperative trajectory simulation. It includes two aircraft, a ground/sky display, a terrain, and several visual representations of the lead aircraft's wake region. Aircraft and vortex dynamics are included in the Simulink simulation and VRML world. Several coordinate system transformations were required to implement simulation outputs into the VRML world. Wake objects are recycled for increased simulation performance and efficiency.

Some evaluation of VRML as a visualization tool has been conducted. Inputting Simulink data into a VRML world is relatively simple, but correctly formatting that data has been demonstrated to be exceedingly difficult. Very little documentation concerning VRML exists, further increasing the difficulty of the task. The FlightGear Flight Simulator¹² was also considered, but due to the difficulty of implementing multiple aircraft and wake objects, it was not selected. A more robust and user-friendly tool is still desired.

Despite the limitations of VRML, this work has many applications. By FY2015 it is expected that the NASA Dryden Flight Research Center will conduct more test flights to study cooperative trajectories. Screenshots and video captured from the VRML world can be used in design reviews and other technical presentations to educate others about the mission. Soon development will begin on a wake avoidance algorithm to prevent the event of a

wake crossing during formation flight or cooperative trajectory missions. This algorithm will be implemented directly into the existing cooperative trajectory simulation, where it can be verified and demonstrated using the VRML world. As air traffic density continues to increase, it is becoming more desirable to reduce the spacing requirements between aircraft. For added safety, cockpit displays could be designed to include a visualization of other aircrafts' wakes, similar to the one developed here.

IV. References

¹Hummel, D., "The Use of Aircraft Wakes to Achieve Power Reductions in Formation Flight," *Proceedings of the 1996 AGARD FDP Symposium on The Characterization & Modification of Wakes from Lifting Vehicles in Fluid*, AGARD CP-584, Paper No. 36, Trondheim, Norway, 20-23 May 1996, pp. 1-13.

²Vachon, M. J., Ray, R. J., Walsh, K. R., and Ennix, K., "F/A-18 aircraft performance benefits measured during the autonomous formation flight project," in *the AIAA Guidance, Navigation and Control Conference*, 2002, pp. 2002-4491.

³Pahle, J., Berger, D., Venti, M., Duggan, C., Faber, J., and Cardinal, K., "A Preliminary Flight Investigation of Formation Flight for Drag Reduction on the C-17 Aircraft," AIAA 2012-4802, *AIAA Atmospheric Flight Mechanics Conference*, Minneapolis, Minnesota, August 2012.

⁴Hueschen R.M., "Development of the Transport Class Model (TCM) Aircraft Simulation From a Sub-Scale Generic Transport Model (GTM) Simulation," NASA TM-2011-217169, Hampton, VA, Aug. 2011.

⁵Burnham, D. C., and Hallock, J. N., "Chicago Monostatic Acoustic Vortex Sensing System, Volume IV: Wake Vortex Decay," DOT/FAA/RD-79-103, Vol. IV, July 1982, pp. 2-3.

⁶"Virtual Reality Modeling Language (VRML)," The MathWorks, Inc. [<http://www.mathworks.com/help/sl3d/vrml.html>]. Accessed Aug. 6 2013.].

⁷"Aerospace Blockset," The MathWorks, Inc. [<http://www.mathworks.com/products/aeroblks/>]. Accessed Aug. 6 2013.].

⁸"Terrain Visualization Example," The MathWorks, Inc. [<http://www.mathworks.com/products/3d-animation/examples.html?file=/products/demos/3d-animation/r2007b/vrterrain.html>]. Accessed Jun. 2013.].

⁹Holforty, W. L., "Flight-Deck Display of Neighboring Aircraft Wake Vortices," Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Stanford Univ., Stanford, CA, 2003.

¹⁰Delisi, D. P., Greene, G. C., Robins, R. E., Vicroy, D. C., and Wang, F. Y., "Aircraft Wake Vortex Core Size Measurements," AIAA 2003-3811, *21st AIAA Applied Aerodynamics Conference*, Orlando, FL, June 2003.

¹¹"ADS-B Academy," Garmin Ltd. [<http://www.garmin.com/us/intheair/ads-b>]. Accessed Jul. 24, 2013.].

¹²"FlightGear Flight Simulator," flightgear.org. [<http://www.flightgear.org/>]. Accessed Aug. 6 2013.].