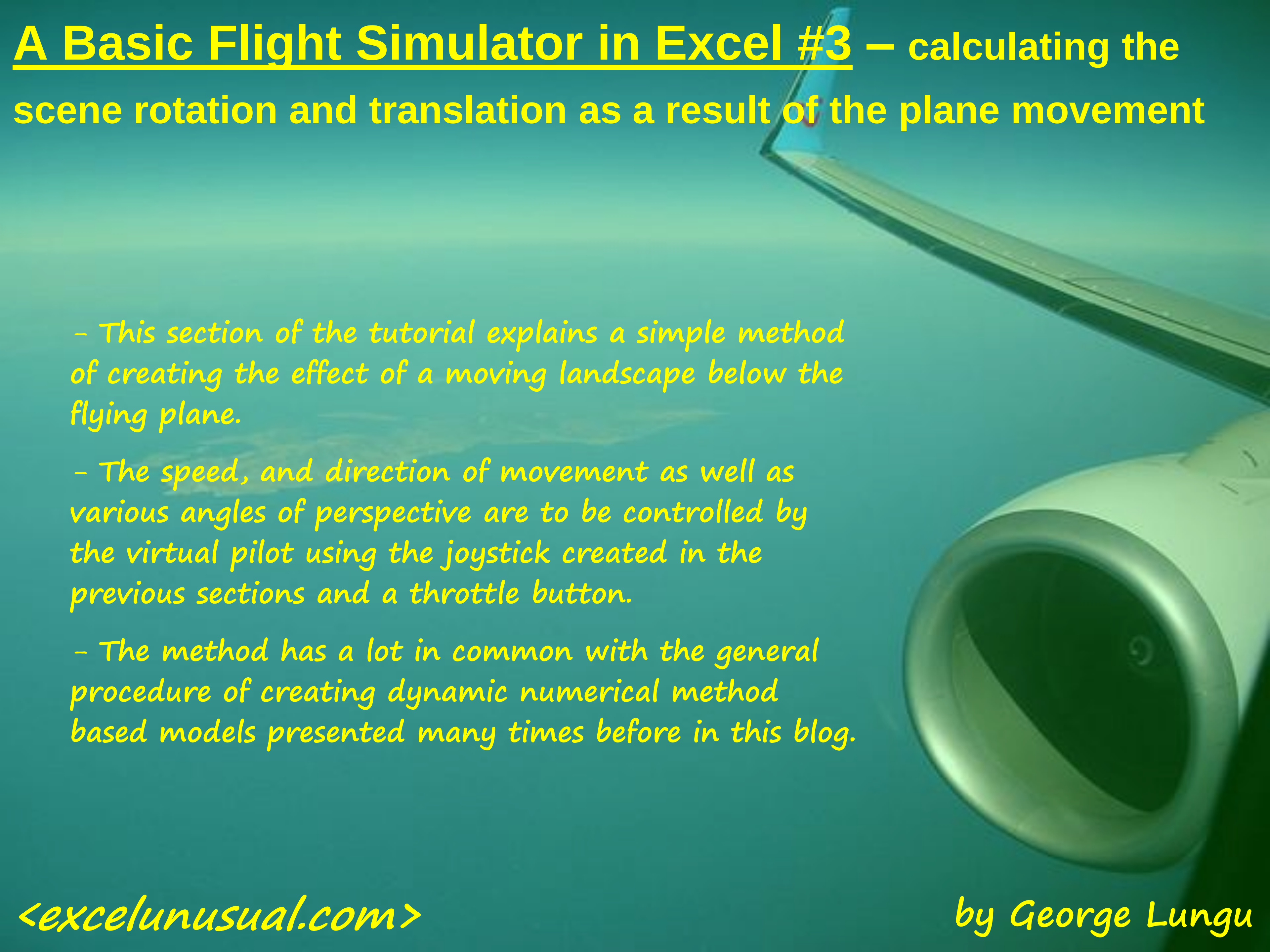


A Basic Flight Simulator in Excel #3 – calculating the scene rotation and translation as a result of the plane movement

- This section of the tutorial explains a simple method of creating the effect of a moving landscape below the flying plane.

- The speed, and direction of movement as well as various angles of perspective are to be controlled by the virtual pilot using the joystick created in the previous sections and a throttle button.

- The method has a lot in common with the general procedure of creating dynamic numerical method based models presented many times before in this blog.

<excelunusual.com>

by George Lungu

Overview of the method:

- The basic method used in this model has the following steps:

=> *Build a 3D wireframe landscape* *(done once at the beginning).*

=> Translate and rotate the 3D image based on the input parameters

=> Convert the 3D image into a 2D perspective image

=> Display the perspective image on a 2D scatter chart.

(loop through this sequence while changing input parameters)

- The last three steps are done while constantly changing perspective parameters in a continuous loop (one linear coordinate and two angular coordinates) based on two input parameters coming from the joystick and one input parameter coming from the throttle.

- The process is similar to the methods used in most of models on this blog and it is based on recurrent calculations using results from the previous time step.

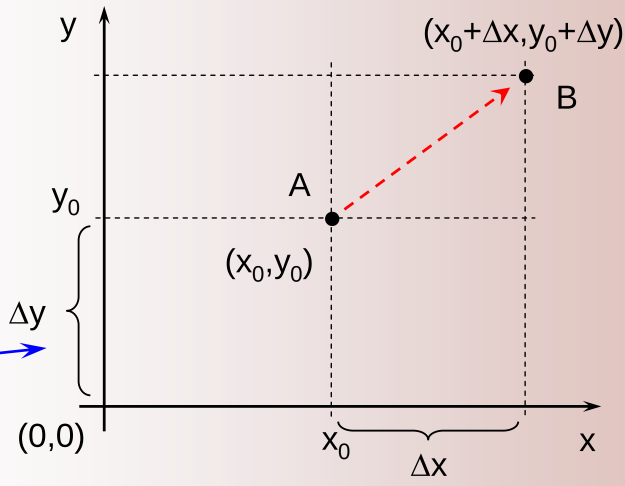
- Since the wireframe landscape has been created in the first section of this tutorial we will proceed to review the formulas for translations, rotations and the 3D to 2D perspective conversion. Translation, rotation and 3D to 2D perspective conversion were analyzed before (http://excelunusual.com/?s=3d) and only a limited review will be given in this section.

Coordinate translation:

-The following coordinate transformation of point (x_0, y_0, z_0) defines a translation by (Δx, Δy, Δz):

$$\boxed{(x_0, y_0, z_0) \rightarrow (x_0 + \Delta x, y_0 + \Delta y, z_0 + \Delta z)}$$

Illustration of a 2D translation operation (a 3D translation is harder to draw so I saved some effort by drawing it in 2D)



More details about the modeling method:

- There are two ways to model the apparent movement of the scene during the flight as seen from the cockpit.

- The easier way would be to start in an arbitrary position somewhere on the landscape. Every elementary time step we do three operations *with the scene image*:

  * move the scene backwards (along the y-axis) in the system of reference connected to the plane

  * roll the scene in a direction opposite to the joystick horizontal deviation by an angular elementary step proportional to the horizontal deviation of the joystick from neutral.

  * rotate (pitch) the scene backwards or forward in a direction opposite to the joystick vertical deviation by an angular elementary step proportional to the vertical deviation of the joystick from neutral.

- This is very easy to program and it involves the copy-paste operation of the landscape vertex matrix to be done during every time step in a loop.

- For this purpose we need three vertex matrices: an initial vertex matrix which contains the node coordinates as seen from the plane just before takeoff, a current vertex matrix and a previous (past) vertex matrix. The current vertex matrix will contain all the formulas corresponding to one translation (y) and two rotations (pitch and roll) of the data from the past matrix.

- A macro will run an infinite conditional Do loop and paste the current matrix data in the past matrix during each loop cycle (of course the copy-paste will be done after the current matrix was evaluated from the past matrix data). The data for the rotation angles and the y-translation step length is taken from the joystick and throttle.

- Through this copy-paste operation, the macro would effectively model the passage of time and the movement of the plane and its pilot through the scene as a function of the of the settings of the cockpit control devices (joystick and throttle).

General background - scene rotation:

- In our model we encounter two types of of landscape rotation seen from a system of reference based in the cockpit :

  * the ***pitch rotation*** which is a rotation around the transversal (x) axis of the airplane

  * the ***roll rotation*** which is a rotation of the scene around the longitudinal axis (y) of the airplane

Based on a previous post - http://excelunusual.com/archive/2011/01/3d-2d-perspective-mapping-in-excel-part-2/ we will be giving the results of coordinate transformation trough a 2D rotation around origin.

$$\begin{cases} x' = \cos(\alpha)\cdot x - \sin(\alpha)\cdot y \\ y' = \sin(\alpha)\cdot x + \cos(\alpha)\cdot y \end{cases}$$

We call x' and y' the new coordinates where α is the rotation angle and x and y are the original coordinates before the rotation.

Sometimes people like to put this in matrix form:

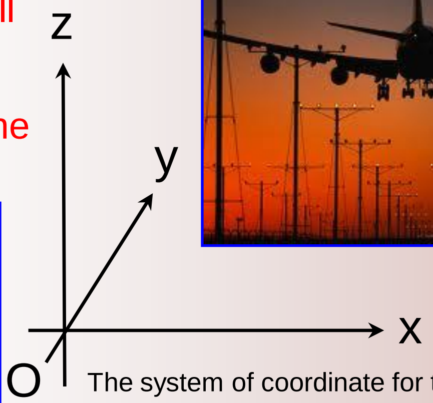
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

Let’s choose the sign conventions:

- We choose the cockpit reference so that the plane flies towards the positive y values (towards the monitor and perpendicular to it). The scene in this case will appear to approach the monitor.

- A positive roll of the plane will be a roll to the right. From the cockpit, the scene therefore will appear to roll to the left.

- A pitch is considered to increase when the plane raises its nose and in consequence the scene will appear to drop in this system of coordinate bound to the cockpit.



The system of coordinate for the landscape is bound to the cockpit

Application of the formula to the 3D pitch rotation:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_{pitch}) & \sin(\alpha_{pitch}) \\ 0 & -\sin(\alpha_{pitch}) & \cos(\alpha_{pitch}) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

- The pitch rotation takes place around x-axis, therefore the x coordinate will be unchanged. For a positive pitch angle the airplane will raise its nose, so the scene will tend to fall (z coordinate will drop). The scene therefore will rotate opposite to the airplane (the pitch angle will appear with minus in the scene rotation formula). The coordinates in the formula refer to a point of the scene.

Verification: In the formula above we can see that x'=x and which is what we need since the pitch rotation occurs around the x axis. Also checking for the sign of the pitch angle, if the pitch angle is increasing and z=0 and y=1 from the formula z' will decrease. In plain English the formula says that when the airplane raises it's nose an object in front of it will appear to be sinking which confirm the correct sign of the angle in the formula.

A positive airplane pitch is obtained by pulling the yoke or control stick. We need to make sure we take in consideration that, namely to link the y-coordinate of the joystick with the negative of the pitch rate!!!

Application of the formula to the 3D roll rotation:

$$\begin{pmatrix} x'' \\ y'' \\ z'' \end{pmatrix} = \begin{pmatrix} \cos(\alpha_{roll}) & 0 & -\sin(\alpha_{roll}) \\ 0 & 1 & 0 \\ \sin(\alpha_{roll}) & 0 & \cos(\alpha_{roll}) \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$$

- The roll rotation takes place around y-axis, therefore the y coordinate will be unchanged. For a positive roll angle (chosen to be to the right) of the airplane, the scene will rotate to the left. The scene therefore will rotate opposite to the way the airplane rolled.
The coordinates in the formula refer to a point of the scene. x', y' and z' are the coordinates after the pitch and x", y" and z" are the coordinates after the roll.

Combining the pitch, the roll and the y-translation in a single formula:

Combining the two rotation formulas and adding a negative increment to the y coordinate (due to the movement of the airplane along the y axis) we get:

$$\begin{pmatrix} x'' \\ y'' \\ z'' \end{pmatrix} = \begin{pmatrix} 0 \\ -\Delta y \\ 0 \end{pmatrix} + \begin{pmatrix} \cos(\alpha_{roll}) & \sin(\alpha_{roll})\cdot\sin(\alpha_{pitch}) & -\sin(\alpha_{roll})\cdot\cos(\alpha_{pitch}) \\ 0 & \cos(\alpha_{pitch}) & \sin(\alpha_{pitch}) \\ \sin(\alpha_{roll}) & -\cos(\alpha_{roll})\cdot\sin(\alpha_{pitch}) & \cos(\alpha_{roll})\cdot\cos(\alpha_{pitch}) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

This is the mathematical transformation that happens with a scene point as viewed from a cockpit reference during an elementary time step of the flight (as airplane advances the image will approach the observer by Δy). Knowing that this is actually part of a numerical method type of model, we can rewrite the above relationship:

$$\begin{pmatrix} x_{current} \\ y_{current} \\ z_{current} \end{pmatrix} = \begin{pmatrix} 0 \\ -\Delta y \\ 0 \end{pmatrix} + \begin{pmatrix} \cos(\Delta\alpha_{roll}) & \sin(\Delta\alpha_{roll})\cdot\sin(\Delta\alpha_{pitch}) & -\sin(\Delta\alpha_{roll})\cdot\cos(\Delta\alpha_{pitch}) \\ 0 & \cos(\Delta\alpha_{pitch}) & \sin(\Delta\alpha_{pitch}) \\ \sin(\Delta\alpha_{roll}) & -\cos(\Delta\alpha_{roll})\cdot\sin(\Delta\alpha_{pitch}) & \cos(\Delta\alpha_{roll})\cdot\cos(\Delta\alpha_{pitch}) \end{pmatrix} \cdot \begin{pmatrix} x_{previous} \\ y_{previous} \\ z_{previous} \end{pmatrix}$$

Where we define the following parameters:

$$\begin{cases} Rate_{pitch} = \dfrac{\Delta\alpha_{pitch}}{t_{step}} \\ Rate_{roll} = \dfrac{\Delta\alpha_{roll}}{t_{step}} \\ Speed_{plane} = \dfrac{\Delta y}{t_{step}} \end{cases}$$

- the pitch and roll rate will later be chosen dependent on the position of the joystick, namely they will be directly proportional to the joystick coordinates
- the speed of the plane will be controlled by the throttle
- we will discuss more about these parameters in the next tutorial section

<www.excelunusual.com>

to be continued...