I. 서론

II. Simulator 의 구성

이 장에서는 몇 가지 가정을 도입하여 좌표계를 설정하고, 힘, 모멘트, 자세, 위치에 관한 항공기 6 자유도 운동 방정식을 유도 한다. NASA 의 풍동 실험 자료를 이용하여 F-16 항공기를 대상으로 하는 공력 계수와 엔진을 모델링 하고, MATLAB 의 Simulink 로 Simulator 를 구성한다. Simulator 의 각 부분에 대한 설명을 한 후에 Simulation 방법을 예를 들어 설명한다. 마지막 부분에는 Simulation 의 결과로 얻어진 항공기의 운동 자료를 3 차원 그래픽으로 표현하는 프로그램을 제시한다.

II.1. 운동 방정식의 유도

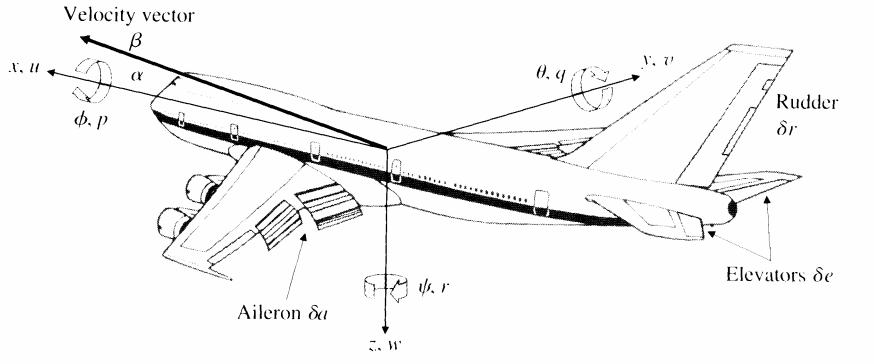
가. 기본가정

- 지구 표면에 고정된 좌표계를 Newton 의 법칙이 성립하는 관성 좌표계로 사용한다.
- 항공기는 강체(Rigid Body)이다. 따라서 항공기의 질량과 질량 분포는 일정하며 자유도는 6 이다.
- 항공기에 작용하는 힘은 일정한 크기의 중력, 공기역학적 힘, 추력 뿐이다.

나. 좌표계의 설정

운동 방정식은 항공기의 무게 중심을 원점으로 하고 항공기에 고정된 좌표계(Body Fixed Axis)에서 유도 한다.

좌표계와 상태변수 제어면의 설정은 다음 그림과 같다.



x, y, z = position coordinates
u, v, w = velocity coordinates
p = roll rate
q = pitch rate
r = yaw rate
ϕ = roll angle
θ = pitch angle
ψ = yaw angle
β = side-slip angle
α = angle of attack

다. Force Equation

Newton 의 제 2 법칙을 강체(Rigid Body)에 적용하면, 외부에서 작용하는 힘(External Force)은 무게 중심의 Linear Momentum 을 관성 좌표계에 대해 미분한 것과 일치한다는 결과를 얻는다.

$$\dot{\mathbf{L}} = \sum \mathbf{F}_{Ext}$$

Body Fixed 좌표계는 관성 좌표계에 대해 회전하고 있으므로

$$\dot{\mathbf{v}}_B + \Omega \times \mathbf{v} = \mathbf{g} + \frac{\mathbf{F}_A}{m} + \frac{\mathbf{T}}{m}$$

이다. 아래 첨자 B 는 Body Fixed 좌표계에 대한 미분을 의미한다. 이 경우에 Body Fixed 좌표계는 항공기를 따라 회전하므로, 좌표계의 각속도 Ω 는 항공기의 각속도 $\omega = [P, Q, R]$와 일치한다. 위 식을 각 성분에 대해 정리하면 다음과 같다.

$$\dot{U} = VR - QW - g\sin\Theta + \frac{F_X + T}{m}$$

$$\dot{V} = PW - RU + g\sin\Phi\cos\Theta + \frac{F_Y}{m}$$

$$\dot{W} = QU - PV + g\cos\Phi\cos\Theta + \frac{F_Z}{m}$$

$$F_X = C_{X_T}\bar{q}S$$

$$F_Y = C_{Y_T}\bar{q}S$$

$$F_Z = C_{Z_T}\bar{q}S$$

공기역학적 계수는 받음각과 옆미끄럼각에 따라 변화하며 공기역학적 힘의 크기는 항공기에 대한 공기의 전체 속도에 비례하므로, 상태 변수 중에서 각 축에 대한 속도성분 U, V, W 를 V_T, α, β 로 변화하는 것이 더 편리하다.

$$\dot{V}_T = \frac{U\dot{U} + V\dot{V} + W\dot{W}}{V_T}$$

$$\dot{\alpha} = \frac{U\dot{W} - W\dot{U}}{U^2 + W^2}$$

$$\dot{\beta} = \frac{\dot{V}V_T - V\dot{V}_T}{{V_T}^2\cos\beta}$$

라. Moment Equation

Newton 의 제 2 법칙을 강체(Rigid Body)에 적용하면, 외부에서 작용하는 모멘트(External Moment)는 무게 중심의 Angular Momentum 을 관성 좌표계에 대해 미분한 것과 일치한다는 결과를 얻는다.

$$\dot{\mathrm{H}}_G = \sum \mathrm{M}_G$$

$$\mathbf{I}\dot{\omega}_B + \Omega \times (\mathbf{I}\omega) = \mathrm{M}_A$$

아래 첨자 B 는 Body Fixed 좌표계에 대한 미분을 의미하며, $\mathbf{I}$ 는 inertia matrix 이다. 이 경우에 Body Fixed 좌표계는 항공기를 따라 회전하므로, 좌표계의 각속도 Ω 는 항공기의 각속도 $\omega = [P, Q, R]$와 일치한다. $I_{xy} = I_{yz} = 0$ 을 이용하여 위의 식을 정리하면 다음과 같다.

$$\left[I_x I_z - I_{xz}{}^2\right]\dot{P} = I_{xz}\left[I_x - I_y + I_z\right]PQ - \left[I_z\left(I_z - I_y\right) + I_{xz}{}^2\right]QR + I_z L + I_{xz} N$$

$$I_y \dot{Q} = \left(I_z - I_x\right)PR - I_{xz}\left(P^2 - R^2\right) + M$$

$$\left[I_x I_z - I_{xz}{}^2\right]\dot{R} = -I_{xz}\left[I_x - I_y + I_z\right]QR + \left[I_x\left(I_x - I_y\right) + I_{xz}{}^2\right]PQ + I_x N + I_{xz} L$$

$$L = C_{l_T}\bar{q}Sb$$

$$M = C_{m_T}\bar{q}S\bar{c}$$

$$N = C_{n_T}\bar{q}Sb$$

마. Kinematic Equation

$$\dot{\Phi} = P + \tan\Theta\left(Q\sin\Phi + R\cos\Phi\right)$$

$$\dot{\Theta} = Q\cos\Phi - R\sin\Phi$$

$$\dot{\Psi} = \frac{Q\sin\Phi + R\cos\Phi}{\cos\Theta}$$

바. Navigation Equation

$$\dot{p}_N = U\cos\Theta\cos\Psi + V\left(-\cos\Phi\sin\Psi + \sin\Phi\sin\Theta\cos\Psi\right)$$
$$+ W\left(\sin\Phi\sin\Psi + \cos\Phi\sin\Theta\cos\Psi\right)$$

$$\dot{p}_E = U\cos\Theta\sin\Psi + V\left(\cos\Phi\cos\Psi + \sin\Phi\sin\Theta\sin\Psi\right)$$
$$+ W\left(-\sin\Phi\cos\Psi + \cos\Phi\sin\Theta\sin\Psi\right)$$

$$\dot{h} = U\sin\Theta - V\sin\Phi\cos\Theta - W\cos\Phi\cos\Theta$$

II.2. 비선형 항공기 모델링

항공기의 운동은 우리가 예측할 수 있는 힘 이외에 주위의 여러가지 영향이 복합적으로 작용하여 이루어진다. 항공기 운동에 영향을 미치는 요소들을 모두 정확히 고려하는 것은 불가능한 일이기 때문에, 중요한 요소를 선별하여 사용하기 편리하도록 모델링을 하게 된다. 이 모델이 실제 현상과 얼마나 유사한가에 따라 시뮬레이션 결과의 신뢰도가 결정되기 때문에 시뮬레이터 제작에 있어서 모델링은 매우 중요한 부분을 차지한다.

가. 공기역학적 계수

이 논문에서는 NASA 의 Dryden 과 Langley 연구 센터에서 F-16 항공기의 축소 모델을 풍동 실험한 결과를 사용한다. 공력 계수들이 일정한 간격에 따라 받음각과 옆미끄럼각, 제어면의 각도에 따른 표로 주어져 있으며, interpolation 을 하여 특정 상태에 따른 공력 계수를 계산한다. 받음각은 -10° 에서 45°, 옆미끄럼각은 -30° 에서 30° 까지 공력 계수가 주어지는데, 이 영역 밖의 상태에서는 extrapolation 을 하여 값을 구하지만 그 신뢰도가 매우 떨어진다. 또, 주어진 공력 계수들은 마하수가 0.6 이하인 경우에만 유효하므로, 시뮬레이션 결과가 모델링의 영역 안에 있는가를 확인해야 한다.

힘과 모멘트에 대한 각 성분의 공기역학적계수는 다음과 같이 모델링 되었다.

$$C_{X_T} = C_X(\alpha, \delta_E) + \frac{\bar{c}Q}{2V_T} C_{X_Q}(\alpha)$$

$$C_{Y_T} = C_Y(\beta, \delta_A, \delta_R) + \frac{bR}{2V_T} C_{Y_R}(\alpha) + \frac{bP}{2V_T} C_{Y_P}(\alpha)$$

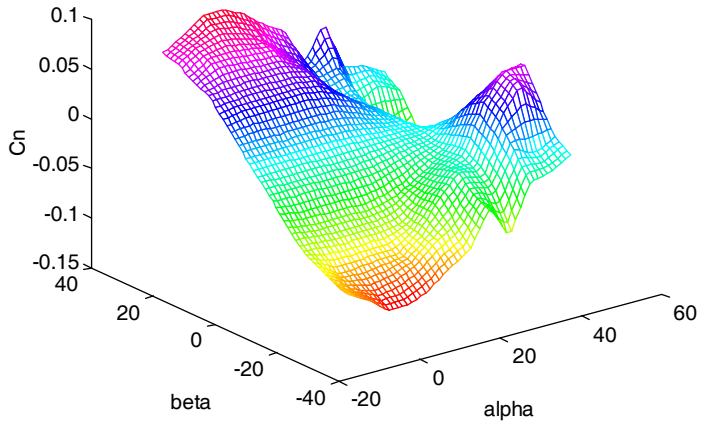
$$C_{Z_T} = C_Z(\alpha, \beta, \delta_E) + \frac{\bar{c}Q}{2V_T} C_{Z_Q}(\alpha)$$

$$C_{l_T} = C_l(\alpha, \beta) + \frac{\partial C_l}{\partial \delta_A}(\alpha, \beta)\delta_A + \frac{\partial C_l}{\partial \delta_R}(\alpha, \beta)\delta_R + \frac{bR}{2V_T} C_{l_R}(\alpha) + \frac{bP}{2V_T} C_{l_P}(\alpha)$$

$$C_{m_T} = C_m(\alpha, \delta_E) + \frac{\bar{c}Q}{2V_T} C_{m_Q}(\alpha) + C_{Z_T}\left(\overline{X}_{CG_r} - \overline{X}_{CG}\right)$$

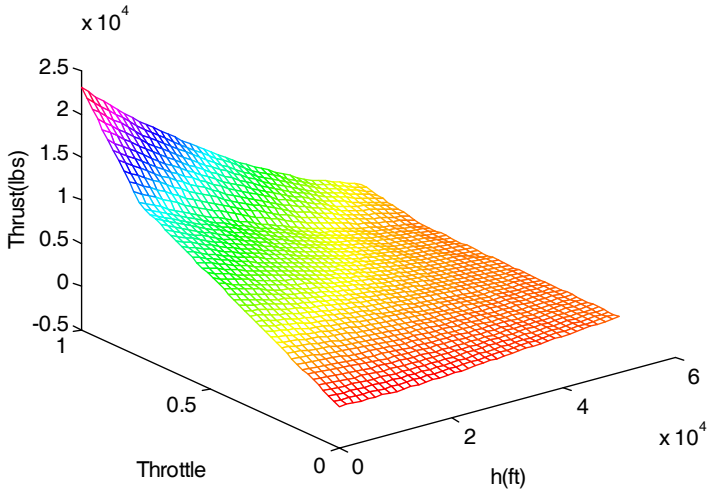
$$C_{n_T} = C_n(\alpha, \beta) + \frac{\partial C_n}{\partial \delta_A}(\alpha, \beta)\delta_A + \frac{\partial C_n}{\partial \delta_R}(\alpha, \beta)\delta_R + \frac{bR}{2V_T} C_{n_R}(\alpha) + \frac{bP}{2V_T} C_{n_P}(\alpha)$$
$$- C_{Y_T}\left(\overline{X}_{CG_r} - \overline{X}_{CG}\right)\frac{\bar{c}}{b}$$

다음 그림은 $\delta_R = \delta_A = 0$ 일 때, 받음각과 옆미끄럼각에 대한 C_{n_T} 의 변화를 나타낸다.



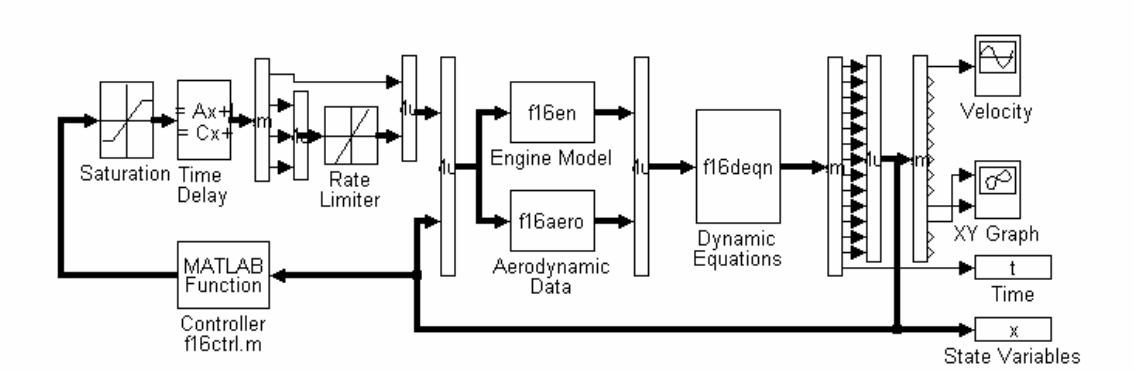
나. Engine

F-16 엔진의 Power 응답은 1 차 lag 형태로 모델링 되었다. Throttle 에 따라 계산된(Throttle Gearing) Power Command 와 현재 엔진의 Power, 마하수, 고도에 따라 추력이 일정한 간격에 따라 표 형태로 주어져 있으며, interpolation 을 하여 특정 상태에 대한 추력을 구한다. 이 표는 Idle, Military, Max 로 구분되며 엔진의 Power 에 따라 선택된다. 마하수는 0 에서 1 까지 0.2 간격으로, 고도는 0 에서 50000ft 까지 10000ft 간격으로 주어지는데, 추력을 구하려는 상태가 이 구간 밖에 있을 경우에는 extrapolation 을 하여 값을 산출하지만 신뢰도는 떨어진다. 다음 그림은 $V_T = 500\ ft/s$ 일 때, Throttle 과 고도에 대한 추력의 변화를 나타낸다.



II.3. Simulator 의 구현

Simulator 는 MATLAB 의 Simulink 를 이용하여 구현하였다. 기본적인 구성은 다음과 같다. Simulink 를 이용하면 Simulator 의 구성을 쉽게 변형할 수 있기 때문에, 제어기나 Gust 에 의한 영향들을 다양한 방법으로 덧붙일 수 있다는 장점이 있다.



크게 Actuator 부분, 공기역학적계수의 산출 부분, 엔진 추력의 산출 부분, 운동 방정식 부분, 제어기 부분으로 나눌 수 있는데, 필요에 따라 Simulink 블록을 추가하여 원하는 형태의 시뮬레이션을 할 수 있다. 항공기의 기본적 자료와, 추력과 공기역학적 계수를 계산하는데 필요한 정적 변수들이 f16def.m 파일에 저장되어 있으며 Simulator 를 실행하는 과정에서 메모리로 옮겨진다.

시뮬레이션 과정은 초기 조건과 제어입력을 통하여 추력과 공기역학적 계수를 얻은 후, 이 자료를 6 자유도 항공기 운동방정식에 대입하여 단계적으로 수치적 적분을 수행함으로써 이루어진다. 각 적분 단계마다 그때의 State vector 에 따라 추력과 공기역학적 계수가 계산되어 항공기 운동방정식에 대입된다.

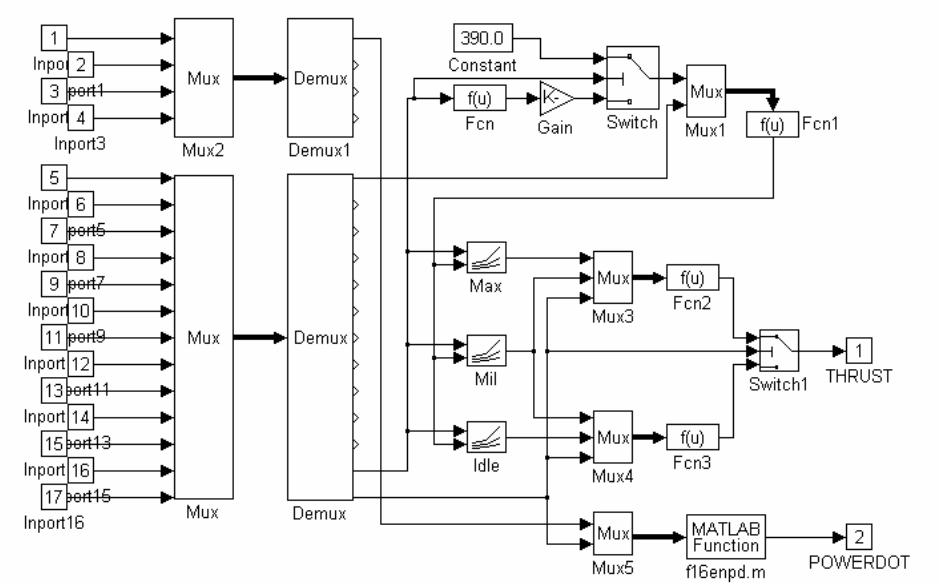
가. Actuator

Elevator, Aileron, Rudder 는 각각 다음과 같은 deflection, rate 한계를 가지며, 시간 지연은 Time Constant 가 0.0495 인 1 차 lag 형태의 미분 방정식으로 표현되어 있다.

Engine 에 관한 모델링은 따로 구분되어 있으므로 이 부분에서 Throttle 은 상한과 하한을 1 과 0 으로 하는 deflection limit 만 주어진다.

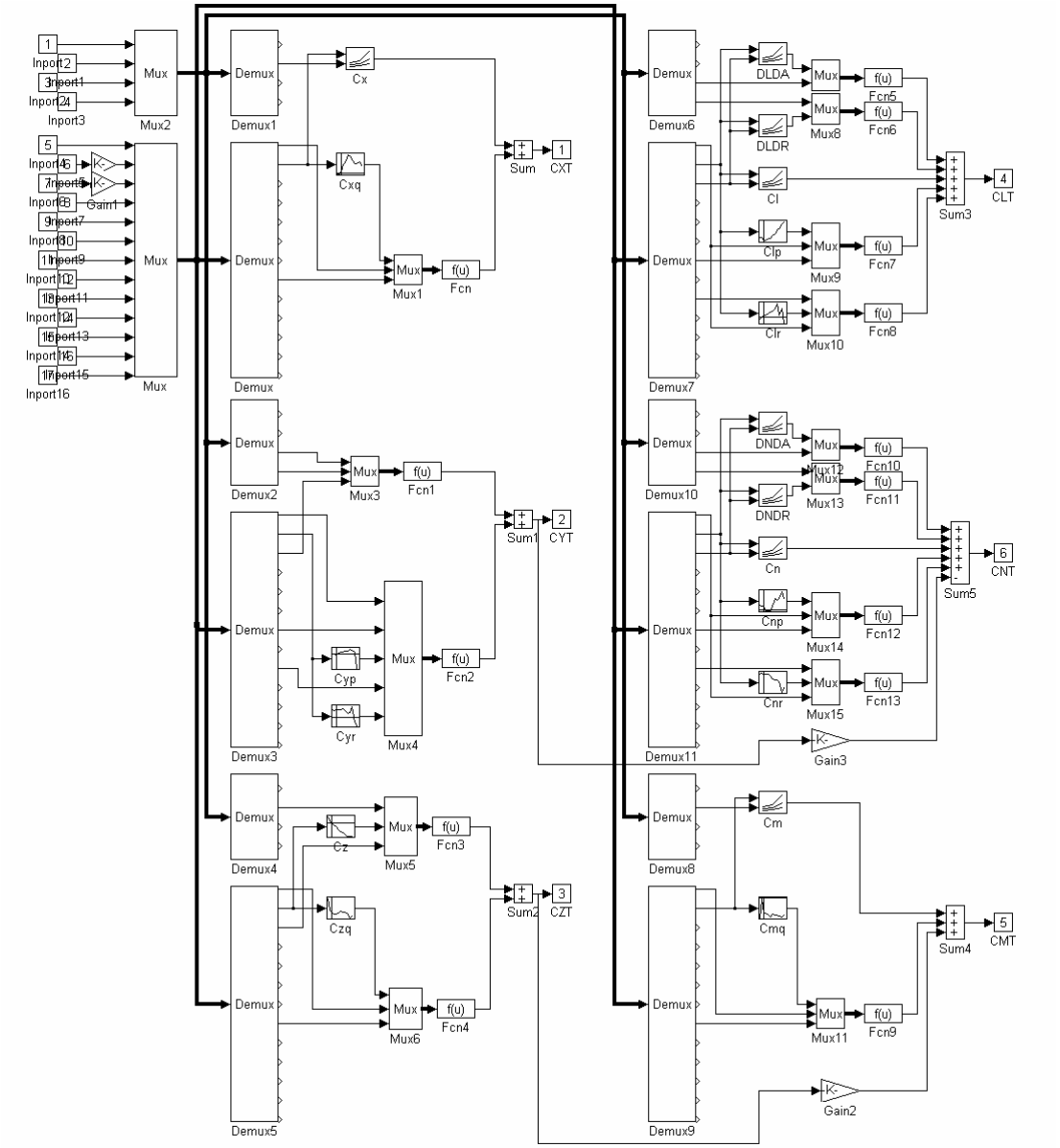
	Deflection Limit	**Rate Limit**	**Time Constant**
Elevator	±25.0°	60°/s	0.0495s lag
Ailerons	±21.5°	80°/s	0.0495s lag
Rudder	±30.0°	120°/s	0.0495s lag

나. Engine : f16en.m



Actuator 부분에서 Control Input 을, 운동 방정식 부분에서 State vector 를 입력 받아 Thrust 와 Engine Power 의 미분값을 출력한다. Throttle 에 따라 Power Command 를 계산하고, 속도와 고도에 따라 마하수를 계산하여 Idle, Military, Max 상태에 따라 추력을 계산한다. Engine Throttle 과 Power 를 입력 받아 Engine Power 의 미분값을 출력하는 f16enpd.m 파일이 MATLAB Function block 에 사용된다.

다. Aerodynamic coefficients : f16aero.m



Actuator 부분에서 Control Input 을, 운동 방정식 부분에서 State vector 를 입력 받아 각 축에 대한 힘과 모멘트 계수 $C_{X_T}, C_{Y_T}, C_{Z_T}, C_{l_T}, C_{m_T}, C_{n_T}$ 를 출력한다. 모델링에 따라 여러 Look Table Block 이 포함되어 있다. 공기역학적 계수를 구하는 과정에서 $\overline{X}_{CG}$ 값이 추가적인 변수로 사용되므로 이 값을 MATLAB Command Window 에서 별로도 입력해야 한다. 변수의 이름은 “Xcg” 이며 f16def.m 에 0.35 로 미리 정의되어 있다.

라. Dynamics Equation : f16deqn.m

위의 두 block 에서 출력한 엔진 자료와 공기역학적 계수를 입력 받아 Flat-Earth, Body Axis, 6-DOF 항공기 운동 방정식을 구성하여 State vector 와 시간을 출력한다.

운동 방정식의 State vector 는 다음과 같이 13 원소로 구성되어 있으며 길이 단위로 ft, 각도의 단위로 radian 을 사용한다.

$$x = \left[V_T, \alpha, \beta, \phi, \theta, \varphi, P, Q, R, p_N, p_E, h, power\right]^T$$

MATLAB m-file S-Function 형식에 따라 초기값, 미분값, State의 개수 등을 출력하도록 프로그램 되어 있기 때문에, 별도의 적분 루틴이 포함되어 있지 않다. Simulink에 적분 프로그램이 내장되어 있으므로 Menu의 Simulation Parameters..부분에서 Euler, 3rd Order Runge-Kutta, 5th Order Runge-Kutta, Adams, Gear, Adams-Gear 의 적분 방법을 선택하면 된다. 적분 과정에 필요한 초기 조건을 블록을 더블 클릭했을 때 나타나는 Function Parameters.. 에 별로도 입력해야 한다.

마. Control inputs 및 Result

Control Input 은 Engine throttle 과 Elevator, Aileron, Rudder deflection 으로 구성된다.

$$U = \left[Throttle, \delta_E, \delta_A, \delta_R\right]^T$$

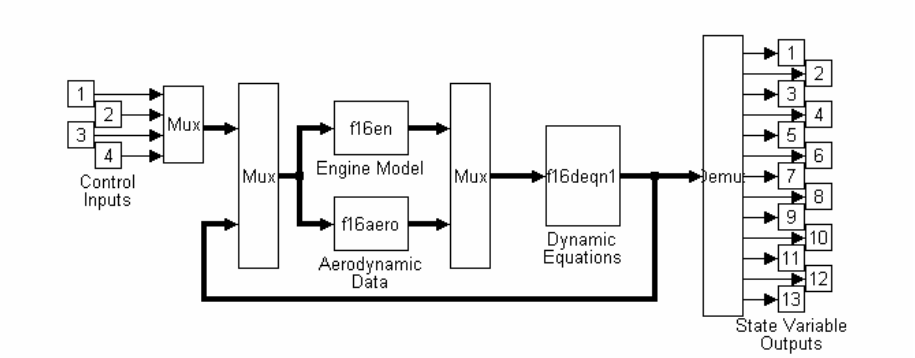
Throttle 은 0 에서 1 사이의 무차원 실수로 입력되며, Control surface deflection 은 Degree 단위로서 다음과 같은 부호 규약을 가진다.

	Deflection	**Sense**	**Effect**
Elevator	Trailing edge down	Positive	Negative pitching moment
Aileron	Right-wing trailing edge down	Positive	Negative rolling moment
Rudder	Trailing edge left	Positive	Negative yawing moment

필요에 따라 state vector 나 시간에 따른 함수로 Control Input 을 구성할 수 있다.

운동 방정식 부분에서 결과는 State vector 와 시간으로 분리되어 MATLAB Command Window 에 각각 "x", "t" 라는 변수명으로 저장되며 그래프로 표시되기도 한다. Simulation 을 수행하면, State vector 의 첫번째 원소인 속도가 시간에 따라 Auto-Scale Graph 에, 10 번째와 11 번째 원소인 수평 변위가 XY Graph 에 그려지는 것을 확인할 수 있다. State vector 는 다음 단계의 엔진과 공기역학적 자료를 얻기 위해 Feedback 된다.

바. Text Mode Simulator : f16text.m



f16text.m 의 기본적인 구조는 위의 것과 일치한다. 다만 Control Input 과 State vector 가 각각 Inport 와 Outport 로 설정되어 있기 때문에, MATLAB Command Windows 에서 필요한 작업을 할 수 있다. 예를 들어 선형화를 할 때 MATLAB 에 내장되어 있는 linmod 함수를 이 f16text 에 이용하면, 쉽게 선형 방정식의 행렬을 얻을 수 있다.

```
[A, B, C, D]=linmod('f16text',x,u)↵
```

MATLAB S-function 형식을 따르면 (help sfunc↵) 특정 시간의 State vector 나 그 미분값을 얻을 수 있기 때문에, 다른 함수에서 이 Simulator 를 이용하려고 할 때 사용하면 편리할 것이다. 물론, RK45, GEAR, ADAMS, EULER 명령을 이용하여 Simulation 을 수행하는 것도 가능하다. 항공기의 기본적인 자료와, 추력과 공기역학적 계수를 구하는 데 필요한 행렬이 f16def.m 파일에 저장되어 있다. 주의할 점은 f16text.m 을 실행하는 과정에서 f16def.m 이 자동으로 메모리에 옮겨지지 않으므로 미리 f16def.m 을 실행해야 한다는 것이다.

f16deqn.m 과 마찬가지로 엔진 자료와 공기역학적 계수를 입력 받아 Flat-Earth, Body Axis, 6-DOF 항공기 운동 방정식을 구성한다. 다만 f16deqn.m 가 State vector 와 시간, 모두 14 개의 원소를 출력하는 반면, 이 f16deqn1.m 은 State vector (13 원소) 만을 출력한다.

이 외에 Engine Model 과 Aerodynamic Data 부분은 모두 앞절의 것과 일치한다.

II.4. Simulation 방법

Elevator Pulse 를 예로 들어 simulation 방법을 설명한다. 이 예제에서 Actuator 부분은 삭제되었다

1. MATLAB Command Window 에서 f16.m 화일을 실행시킨다.

```
f16.m↵
```

2. MATLAB Command Window 에서 변수 Xcg 를 설정한다.

```
Xcg=0.3↵
```

1 번 과정에서 f16.m 파일을 실행하면, f16def.m 파일이 자동으로 실행되는데 이 파일에는 Xcg 가 0.35 로 정의되어 있다. 따라서, 2 번 과정을 먼저 수행한 후 1 번 과정을 수행하면

Xcg 의 값은 f16def.m 에 정의되어 있는대로 0.35 로 설정되므로 순서에 주의해야 한다.

3. State vetor 의 초기 조건을 Command Window 에 입력한다.

```
x0=[502.0 0.03936 4.1e-9 0 0.03936 0 0 0 0 0 0 0 9.64359]↵
```

F16 Window 의 Dynamic Equation block 을 더블클릭하여 Function Parameter 란에 변수 이름 "x0" 를 입력한다.

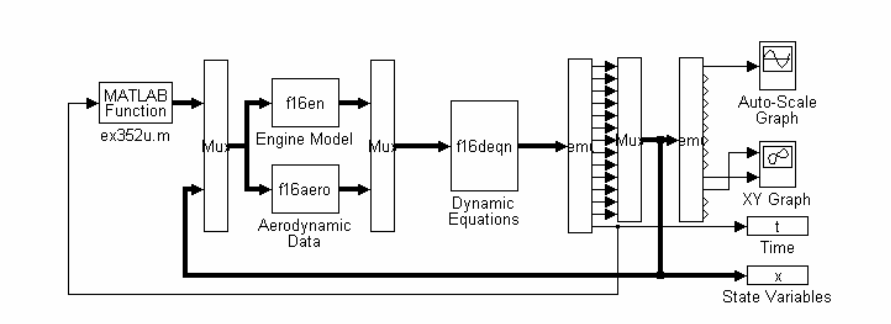
4. Control Input 을 설정한다.

이 예제에서는 Control Input 중 Elevator deflection angle 이 시간에 따라 변화하므로, 시간을 입력 변수로 하는 다음과 같은 MATLAB 함수를 만들어 "ex352u.m" 으로 저장한다.

```
function u=ex352u(t)
u=[0.1485 -1.931 -7.0e-8 8.3e-7];
if  t < 1.0

elseif   t < 1.5
       u(2)=-1.931+2;
elseif   t < 2.0
       u(2)=-1.931-2;
end
```

Simulink 의 Nonlinear library 에 있는 MATLAB Function 을 Drag 하여 MATLAB function: 란에 "ex352u"를, Output width: 란에 "4"를 입력한다. Dynamics Equation 에서 출력된 Time 을 MATLAB Function 의 Input 에 연결하면 다음과 같은 형태가 된다.



5. F16 Window 의 Simulation Menu 에서 Parameters..를 선택하여 적분 방법과 조건등을 설정하고, Start 를 선택하여 simulation 을 수행한다.

Actuator 를 고려한 simulation 에서는 Control Input 의 초기값을 Time Delay Block 에 별도로 입력하는 과정만 추가하면 된다.

II.5. 운동 상태의 표현

Simulation 을 수행하면 시간에 따른 각 State Vector 의 변화가 숫자로 표현된다. MATLAB 의 plot 명령으로 그래프를 그려볼 수 있지만, 항공기의 운동상태를 이해하기는 어려운 점이 있다. 보다 쉽게 운동 상태를 이해하기 위해, 항공기의 운동을 Computer Graphic 으로 표현하는 프로그램을 OpenGL 과 Visual C++/MFC 를 이용하여 작성하였다.

가. OpenGL

OpenGL 은 Silicon Graphics 에서 만든 Graphic Library 의 일종이다. Graphic Library 는 "software interface to graphics hardware"라고 정의할 수 있는데, 사용자가 간단한 명령어를 이용하여 원하는 장면을 기술하면, 실제로 그림을 그리기 위한 정보들을 생성하고 이것을 화면에 표시하는 디스플레이 장치에 전달하는 역할을 하는 것이다. 복잡한 Computer Graphic 의 이론을 모르는 사용자도 쉽게 Computer Graphic 을 응용한 프로그램을 작성할 수 있다.

OpenGL 은 Silicon Graphic 에서 만든 IRISGL 에서 출발하였다. 컴퓨터 그래픽을 위한 전문적인 IRIS Graphics Workstation 에 사용되도록 제작한 IRISGL 을 Hardware 나 OS 에 상관 없이 여러 사람이 쉽게 사용할 수 있도록 "Open"한 것이 바로 OpenGL 이다. 현재 OpenGL 이 새로운 표준으로 자리잡아가고 있으며, Direct3D 라는 별도의 Graphic Library 를 개발하던 Microsoft 도 Silicon Graphics 와 협력할 것을 약속했다.

나. 프로그램의 설명

OpenGL 은 Graphic Library 이기 때문에 그림을 표현하는 윈도우를 생성하고 마우스나 키보드를 통해 사용자와 정보를 교환하는 부분은 별도의 Compiler 에 의해 제작되어야 한다. 이 논문에서는 Microsoft Visual C++/MFC 를 이용하였다.

OpenGL 은 AutoCAD 나 3DS 와 같은 Drawing Tool 이 아니라 프로그램을 작성하기 위한 도구이기 때문에 항공기와 같은 형태를 모델링하기 위해서는 수많은 점과 다각형을 직접 입력해야 한다. 일반 사용자에게 공개된 OpenGL 용 그리기 도구는 없기 때문에 이 논문에서는 3DS 에서 그린 F-16 파일을 OpenGL 에서 사용할 수 있도록 변환하여 이용하였다.

다음은 몇 가지 중요한 파일이다.

- F16.cpp : F16 항공기의 형태를 담고 있는 파일이다. 3DS 를 OpenGL 에서 사용할 수 있도록 변환하는 별도의 프로그램에 의해서 제작되었다. 각 다각형 꼭지점의 3 차원 좌표와, 특정 다각형에 대한 꼭지점의 번호를 저장하는 행렬이 저장되어 있다. F16 은 2153 개의 다각형으로 모델링 되었으며, 좌표계에서 1 은 1ft 를 의미한다.
- F16sdat.cpp : Simulation 의 결과가 0.2 초 단위로 표시되어 있는 행렬을 저장하는 파일이다. 항공기의 자세를 표현하는 Euler Angle Φ, Θ, Ψ 과 위치를 표현하는 p_N, p_E, h 가 시간에 따라 저장된다. 이 파일은 MATLAB 에서 Simulation 을 한 후에, f16save.m 파일을 실행하면 자동으로 생성된다.

- MainFrm.cpp : 기본적인 윈도우를 생성하는 파일이다.
- OpenGL.cpp : Main Application Source 로서 Document 와 View 의 구조로 나뉜다.
- OpenGLDoc.cpp : Main Application Source 의 Document 부분으로 파일 입출력에 관한 내용을 다룬다. 이 프로그램에서는 기본적인 Class 구조 이외에 파일 입출력에 관한 부분이 포함되지 않았기 때문에, 다른 Simulation 결과를 확인하기 위해서는 새로운 F16sdat.cpp 를 추가하여 Compile 을 다시 해야 하는 번거로움이 있다. 가장 시급한 개선 사항이라고 생각한다.
- OpenGLView.cpp : Main Application Source 의 View 부분으로, 화면에 나타나는 내용을 다루기 때문에 이 프로그램의 가장 중요한 부분이다. 다음과 같은 함수로 이루어진다.

  calcNormal(); : 그림자를 생성하기 위해 다각형의 수직 단위벡터를 구한다.

  COpenCLView(); : 변수들을 초기화 한다.

  F16MakeList(); : F16 의 다각형의 위치를 읽어들인다.

  GLRenderScene(); : OnDraw();에서 설정된 좌표에 따라 F16 을 화면에 그린다.

  GLSetupRC(); : 빛과 재질, 색깔을 설정한다.

  OnDraw(); : 관성 좌표계와 지면을 그리고 항공기 자세와 위치에 따라 좌표계를 변환한 후에 GLRenderScene();을 호출한다.

  OnEyePosition(); : 보는위치와 보이는 위치를 설정하는 Dialogue Box 를 생성하고 정보를 교환한다.

  OnSimulationStart(); : 0.2 초 간격으로 WN_TIMER Message 를 발생시켜 Simulation 을 시작한다.

  OnSimulationStop(); : Simulation 을 종료한다.

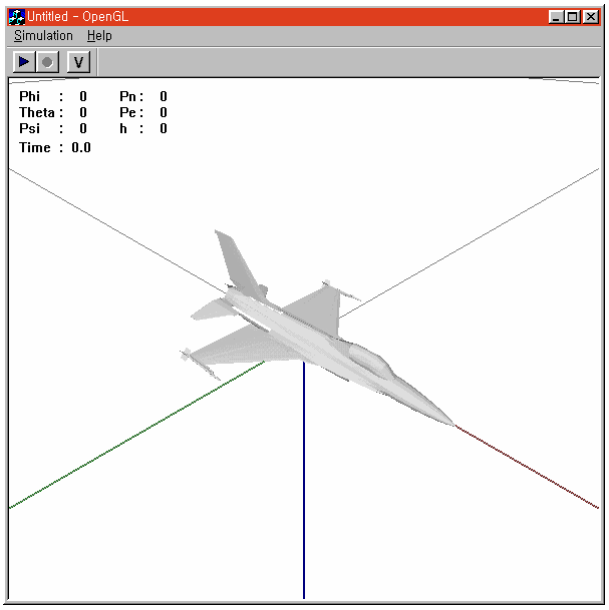
  OnTimer(); : WN_TIMER Message 를 받으면, 다음 단계의 항공기 자세와 위치 정보를 얻은 후에 OnDraw();함수를 호출하여 새로운 그림을 그린다.

  OnSize(); : 윈도우의 크기가 변했을 때 Viewport 를 새롭게 정의한다.

  OnUpdateSimulationStart(); : Simulation 을 하는 도중에 OnSimulationStart(); 함수가 호출되지 않도록 조절한다.
- OpenGL.rc : 메뉴, 단축기, Dialogue Box, Tool Bar 를 생성한다.

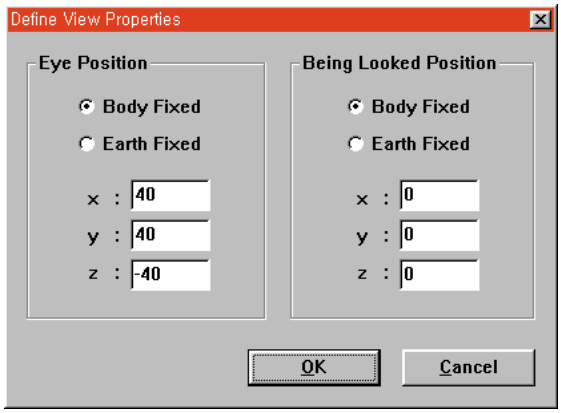
다. Simulation 방법

다음은 프로그램의 실행 화면이다.



화면의 왼쪽 위에는 Euler 각(degree)과 위치(ft), 시간(sec)이 표시된다. z=0 인 평면에 400 ft 간격으로 Grid 가 그려져 있으며, x,y,z 축은 각각 빨간색(R), 녹색(G), 파란색(B)으로 표현된다. (F-16 의 span 은 30ft 이다.)

Tool Bar 의 V 버튼 또는, 메뉴의 View Properties..를 선택하거나 “Ctrl-V”를 누르면 눈의 위치와 보이는 위치를 설정하는 다음 Dialogue Box 가 표시된다.

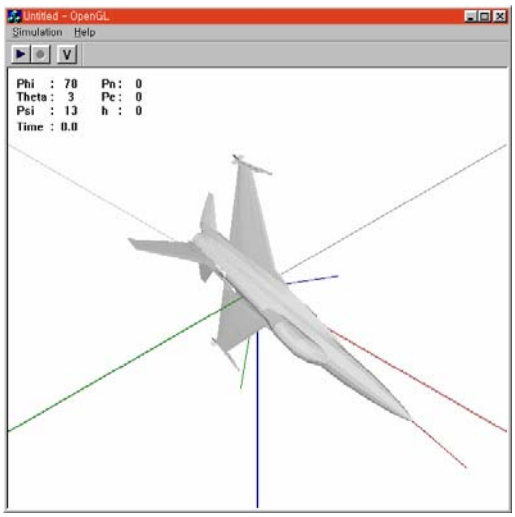


Eye Position 은 눈의 위치를, Being Looked Position 은 보이는 위치를 설정하는 부분이다. Earth Fixed 와 Body Fixed 중에 한가지를 선택할 수 있는데, 각각 관성 좌표계와 항공기의 무게 중심을 원점으로 하고 항공기를 따라 평행 이동하는 좌표계를 의미한다. x,y,z 는 선택한 좌표계에 대한 ft 단위의 위치이다. 예를 들어 눈의 위치와 보이는 위치를 모두 Body Fixed 로 설정하면 항공기를 따라가면서 항공기를 보게 된다. 다만, Body Fixed 좌표계는 항공기의 운

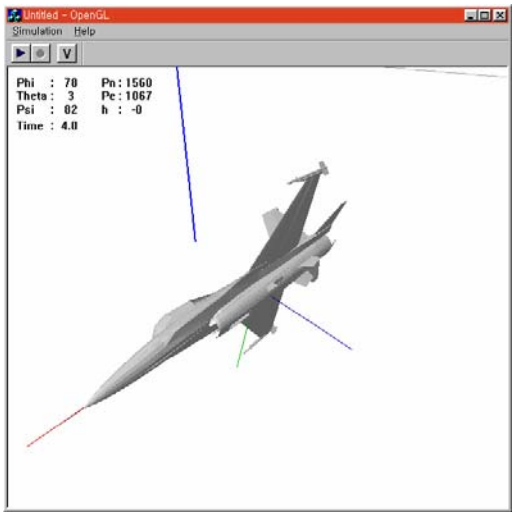
동 방정식에서 정의된 좌표계와는 달리 회전을 하지 않고 평행 이동만을 한다. 만약에 눈의 위치를 결정하는 좌표계가 항공기를 따라 회전한다면 Simulation 을 하는 동안 항공기의 한 쪽 면만을 보게 될 것이기 때문이다.

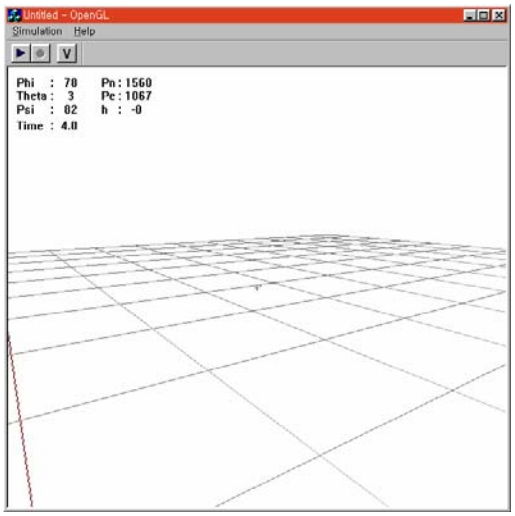
Toolbar  의 버튼 또는, 메뉴의 Start 를 선택하거나 “Ctrl-T”를 누르면 Simulation 이 시작된다. 다음은 Eye Position 을 각각 Body Fixed 와 Earth Fixed 로 선택했을 때, $r = 0.3 rad/s$ 의 Steady Turn 을 Simulation 한 결과이다.

t=0

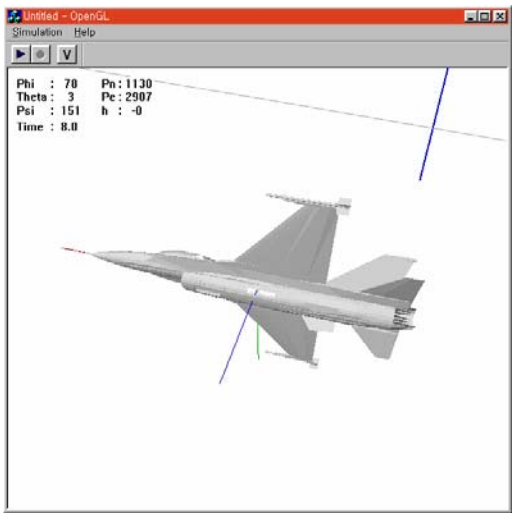


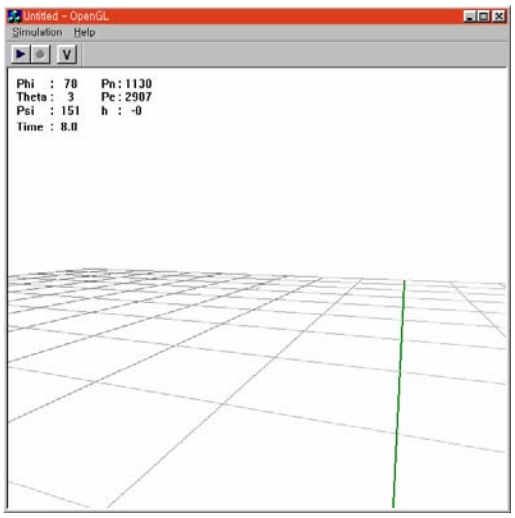
t=4



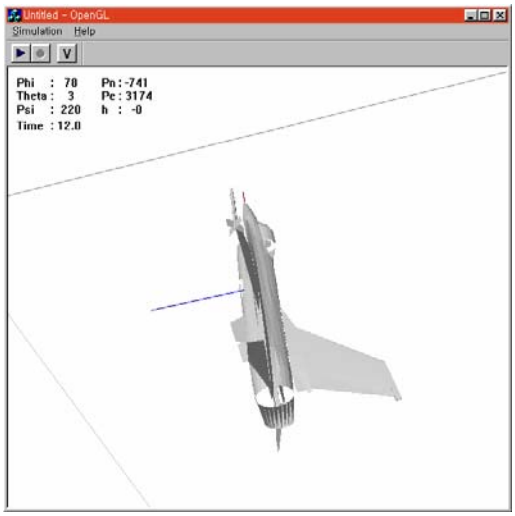


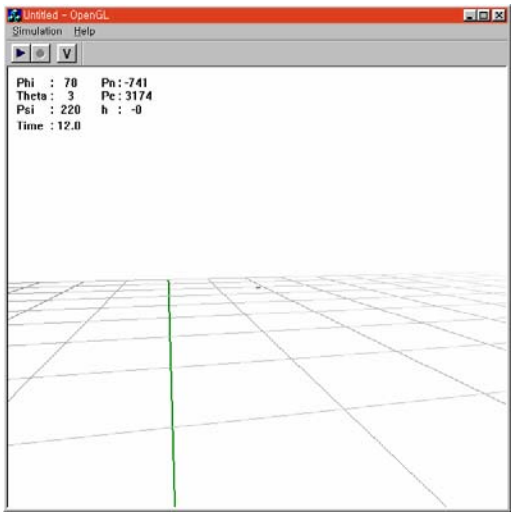
t=8


Untitled - OpenGL
Simulation Help
Phi : 70 Pn : 1130
Theta : 3 Pe : 2907
Psi : 151 h : -0
Time : 8.0

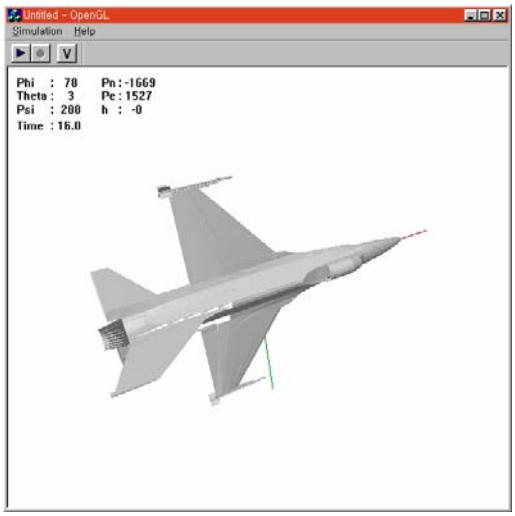

Untitled - OpenGL
Simulation Help
Phi : 70 Pn : 1130
Theta : 3 Pe : 2907
Psi : 151 h : -0
Time : 8.0

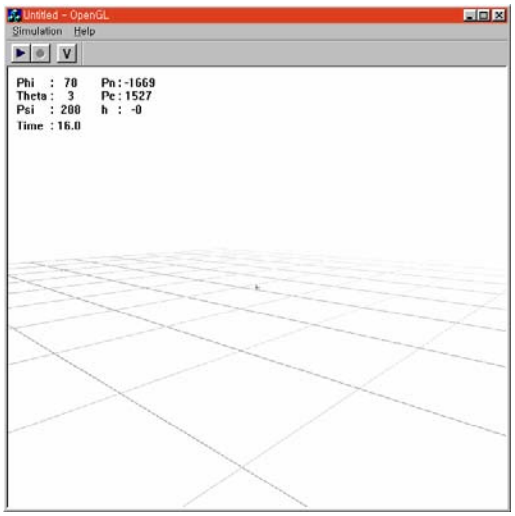
t=12


Untitled - OpenGL
Simulation Help
Phi : 70 Pn : -741
Theta : 3 Pe : 3174
Psi : 220 h : -0
Time : 12.0

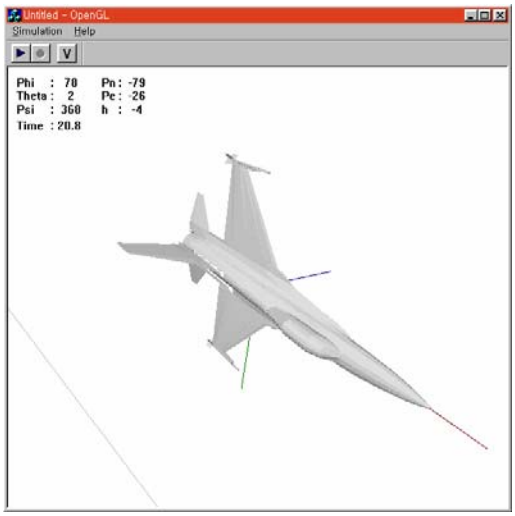

Untitled - OpenGL
Simulation Help
Phi : 70 Pn : -741
Theta : 3 Pe : 3174
Psi : 220 h : -0
Time : 12.0

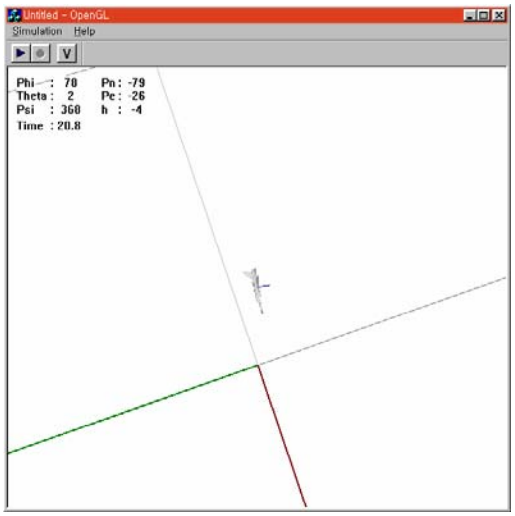
t=16


Untitled - OpenGL
Simulation Help
Phi : 70 Pn : -1669
Theta : 3 Pe : 1527
Psi : 208 h : -0
Time : 16.0


Untitled - OpenGL
Simulation Help
Phi : 70 Pn : -1669
Theta : 3 Pe : 1527
Psi : 208 h : -0
Time : 16.0

t=20.8


Untitled - OpenGL
Simulation Help
Phi : 70 Pn: -79
Theta : 2 Pe: 26
Psi : 360 h : -4
Time : 20.8


Untitled - OpenGL
Simulation Help
Phi : 70 Pn: -79
Theta : 2 Pe: 26
Psi : 360 h : -4
Time : 20.8