Hardware in the loop simulator for the Piccolo avionics

8 Fourth Street
PO Box 1500
Hood River, OR 97031
(541) 387-2120 phone
(541) 387-2030 fax
CCT@gorge.net

Bill Vaglienti
Marius Niculescu

Version 1.3.0
April 20, 2005

Table of contents

1 Introduction

Piccolo is an avionics system for flying small unmanned aircraft. A cornerstone of Piccolo's powerful development environment is a hardware-in-the-loop (HIL) simulator. The simulator allows the aircraft control laws and mission functionality to be tested without risking hardware in flight test. Although HIL simulation can not replace flight testing, it measurably reduces the likelihood of failure by detecting bugs and deficiencies in the lab.

To facilitate this function the *Piccolo* developers kit includes a hardware in the loop simulator which can be used to test the performance of a *Piccolo* implementation. The HIL simulator is based upon the external CAN interface. Piccolo sends servo control information over the CAN bus, and accepts external sensor data on the CAN bus. A PC with a CAN interface card closes the loop by reading the actuator positions, applying them to an aircraft dynamics model, calculating new sensor data, and putting that data on the CAN bus.

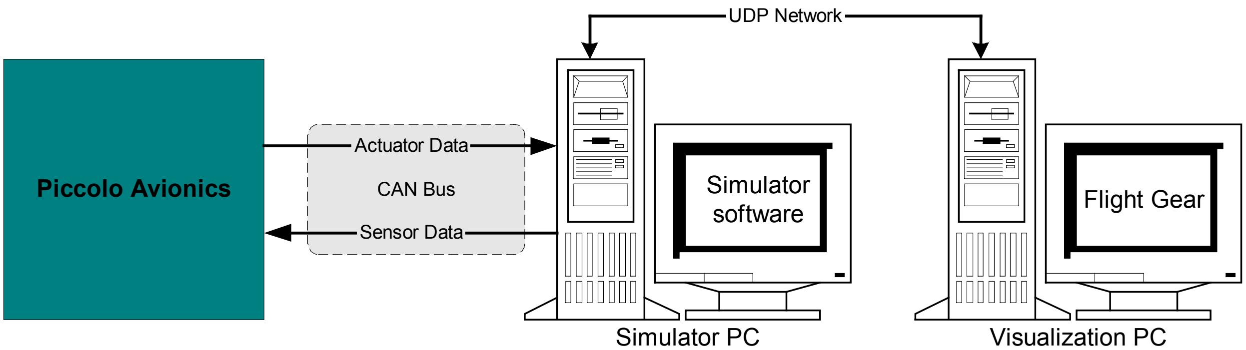

Figure 1: Hardware in the loop simulation

In order to visualize the state of the aircraft the simulator sends UDP network packets to another PC running the open source Flight Gear flight simulator. Flight Gear accepts the simulation state packets in favor of its own dynamics model, when properly configured for this. This provides an attractive graphics interface for visualizing the performance of the aircraft.

2 Dynamics model overview

The heart of the simulator is the dynamics model. It is the sum of all the components which affect the dynamics and kinematics of the aircraft, including: aerodynamics, propulsion, and inertia effects. The dynamics model contains an integration method which estimates the state of the aircraft based on the previous state and new control surface signals.

The dynamics model for Piccolo is based on parametric and table look up data contained in a set of files. Hence the details of the simulation model can be changed without altering the simulator software. The parameters that affect the simulation model are contained in an ASCII file. Each line of the file contains the name of a parameter, and its value. For example the wing area is specified as:

```
Wing_Area=X
```

Where `X` is the actual value of the wing area. The rest of the document will describe the separate components of the system, and the parameters that affect them. Each parameter can appear in the main file in any order. Some parameters are required, and others have default values. Note that all parameters appear in SI units, i.e. meters for length. Angles are given in degrees.

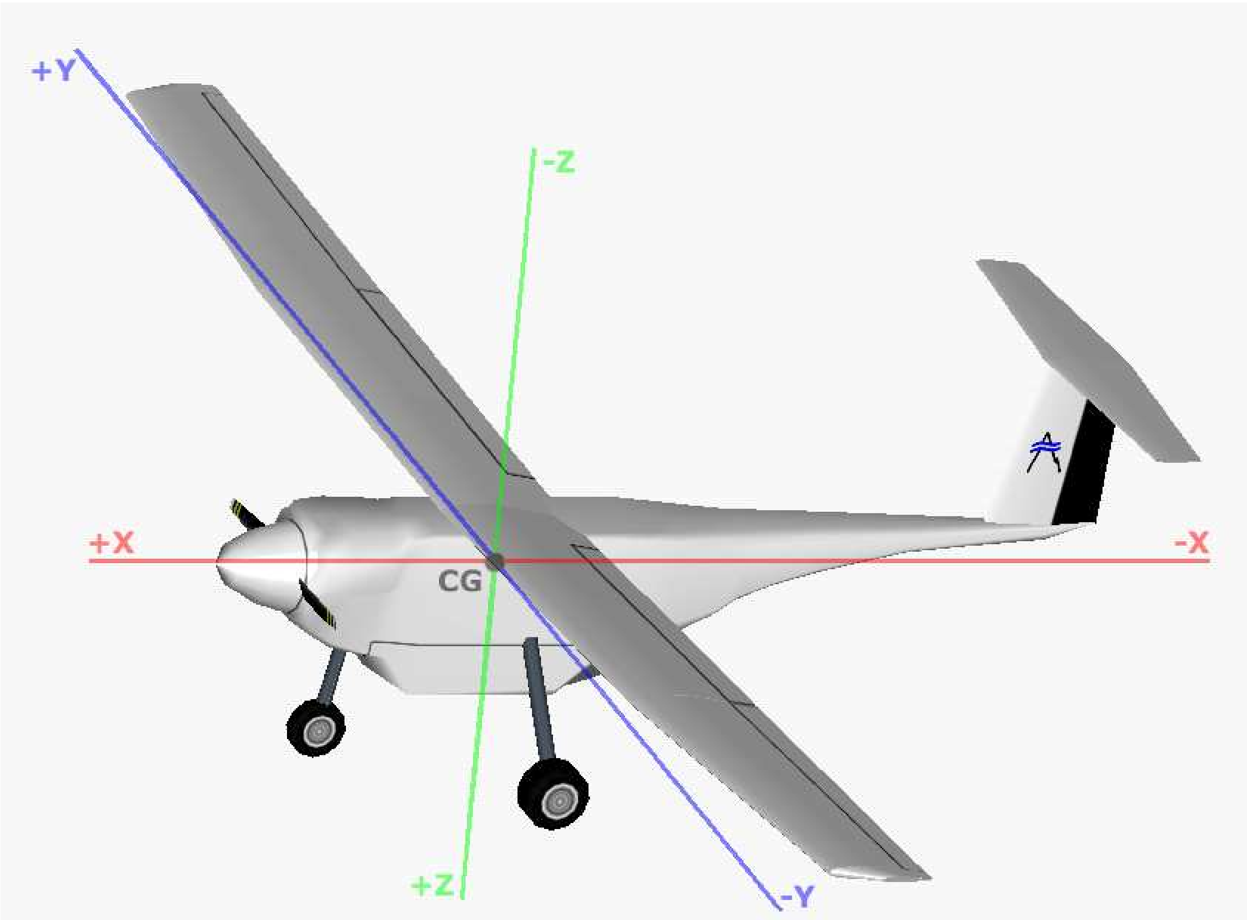

Figure 2. The aircraft body axes used by the Simulator.

3 Aerodynamics model

The aerodynamics model is built upon a simple aircraft model that uses a mix of linear and non-linear parameters. Contributions due to the wing, fuselage, horizontal tail, and one or two vertical tails are considered.

3.1 Contributions due to the wing

The wing contributes body axis components of: lift, drag, sideforce, pitching moment, yawing moment, and rolling moment. In addition to the wing geometry the state variables which affect these force are: dynamic pressure, angle of attack, angle of sideslip, roll rate, pitch rate, and yaw rate. The model first calculates all the forces in the wind axis and then rotates them to the body axis.

3.1.1 Lift

The wing lift depends on the geometry, angle of attack, and dynamic pressure. The model first estimates the wing lift coefficient, and then calculates the actual lift based on the dynamic pressure and wing area. The wing lift coefficient (C_L) depends only on the angle of attack. The angle of attack is determined from the angle of attack of the whole airplane, the incidence angle of the wing, and the pitch rate of the airplane combined with the wing location. Wing lift coefficient is estimated from the angle of attack by look up table, or by the linear parameter $dC_L/d\alpha$.

Estimating the wing lift by look up table is done by reading in data from the file indicated in the parameter list. This file must contain four columns of ASCII data in which the first column is the angle of attack in degrees, the second column is the corresponding lift coefficient, the third column is the corresponding drag coefficient, and the last column is the pitch moment coefficient. All the coefficients must represent the entire three-dimensional performance of the wing. An example file is given below. Note that lines beginning with "//" are ignored

```
//alpha     CL     CD     CM
-4.49 -.083449716 0.017006564 -0.08
-2.96 0.040597159 0.013196221 -0.08
-1.06 0.181935417 0.011632473 -0.08
0.37  0.298464299 0.014600722 -0.08
2.11  0.438298958 0.021215528 -0.08
4     0.589410606 0.031482185 -0.08
5.19  0.676619318 0.039028072 -0.08
6.43  0.760069034 0.04772756  -0.08
8.11  0.85404394  0.061483296 -0.08
10.1  0.884867708 0.083212559 -0.08
11    0.884867708 0.085712559 -0.08
12    0.751799243 0.112997619 -0.08
13    0.52625947  0.116168833 -0.08
20    0           0.1         -0.08
```

If the look up file is unavailable the simulator will estimate the $dC_L/d\alpha$ parameter. This is done by using the wing area and span geometry data to determine the aspect ratio, and then apply the elliptical wing estimate for the three-dimensional lift curve slope. The estimate assumes that the two-dimensional lift curve slope is 2π. The estimate is then corrected for the dihedral angle.

The estimate can be bypassed by supplying a suitable value in the aircraft description, see Table 1.

Table 1. Wing parameter values for lift estimation

Value	Meaning	Default
Wing_Area	The area of the main wing in meters^2.	Required
Wing_Span	The span of the main wing in meters.	Required
Wing_LUT	The name of the wing look up table file	Optional
Wing_Incidence	The angle of the wing at AOA = 0, in degrees.	0
Wing_Dihedral	The dihedral angle of the wing, in degress.	0
Wing_Lift_Slope	The lift curve slope of the main wing, per radian	Calculated

3.1.2 Drag

The wing drag depends on the geometry, angle of attack, and dynamic pressure. The model first estimates the wing drag coefficient, and then calculates the actual drag based on the dynamic pressure and wing area. The wing drag coefficient (C_D) depends only on the angle of attack, and is estimated by look up table, or by a combination of fixed parasitic drag and calculated induced drag.

Estimating the wing drag by look up table is done in the same way that the wing lift estimate is done, except that the third column of the table is used. The drag coefficient must represent the entire three-dimensional performance of the wing.

If the look up file is unavailable the simulator will estimate the drag parameter. This is done by using the wing area and span geometry data to determine the aspect ratio, and then apply the elliptical wing estimate of for induced drag, plus a constant offset term. The induced drag estimate can be adjusted with Oswald's span efficiency factor.

Table 2. Wing parameter values for drag estimation

Value	Meaning	Default
Wing_Area	The area of the main wing in meters^2.	Required
Wing_Span	The span of the main wing in meters.	Required
Wing_Span_Efficiency	Oswald's span efficiency factor	1.0
Wing_Parasitic_Drag	Constant value for the wing parasitic drag	0.0

3.1.3 Side force

The wing side force results from the dihedral angle of the wing and the sideslip angle. The sideslip angle depends on the sideslip of the whole aircraft, and the location of the wing combined with the yaw rate. The simulator calculates the side force at run time based on the lift curve slope, the dihedral angle, and the sideslip angle.

3.1.4 Pitching moment

The wing pitching moment depends on the geometry, angle of attack, and dynamic pressure. The model first estimates the wing pitching moment coefficient, and then calculates the actual pitching moment based on the dynamic pressure, wing area, mean chord, and wing location. The pitching moment coefficient is relative to the aerodynamic center of the wing, but the final pitching moment is given relative to the center of gravity. When transferring the pitching moment from the aerodynamic center to the center of gravity the model also adds in the pitching moment due to lift and drag.

The wing pitching moment coefficient (C_M) depends only on the angle of attack, and is estimated by look up table, or by a fixed parameter. Estimating the wing pitching moment by look up table is done in the same way that the wing lift estimate is done, except that the fourth column of the table is used. The pitching moment coefficient must represent the entire three-dimensional performance of the wing. If the look up file is unavailable the simulator will use a fixed pitching moment coefficient which it gets from the wing parameter data.

Table 3. Wing parameter values for pitching moment estimation

Value	Meaning	Notes
Wing_Area	The area of the main wing in meters^2.	Required
Wing_Span	The span of the main wing in meters.	Required
Wing_Pitching_Moment	The pitching moment coefficient about the aerodynamic center.	0.0
Wing_X	Body axis X Distance between the wing aerodynamic center and the center of gravity, in meters. Positive if the wing is ahead of the CG	0.0
Wing_Z	Body axis Z Distance between the wing aerodynamic center and the center of gravity, in meters. Positive if the wing is below the CG	0.0

3.1.5 Rolling moment

The wing rolling moment depends on the geometry, rolling rate, sideslip angle, life coefficient, and dynamic pressure. The rolling moment as a function of roll-rate, called roll-damping, depends on the taper ratio and aspect ratio of the wing. The simulator will calculate this value (using data from a Perkins and Hage chart) or it can be entered in the parameter list.

Rolling moment as a function of sideslip, called dihedral effect, depends on the dihedral angle and taper ratio of the wing. As with the roll damping the simulator can calculate this value, or it can be entered in the parameter list. It should be noted that there are actually two components to this term; one which is due primarily to the dihedral angle, and another which is due to the wing sweep. The latter component depends on wing lift coefficient, and is calculated at runtime, therefore the dihedral effect parameter should only include the component due to dihedral angle and wing location.

Rolling moment as a function of yaw rate depends on the lift coefficient of the wing, and is calculated at runtime.

Table 4. Wing parameter values for rolling moment estimation

Value	Meaning	Default
Wing_Area	The area of the main wing in meters^2.	Required
Wing_Span	The span of the main wing in meters.	Required
Wing_Sweep	The sweep back angle of the quarter chord of the wing, in degrees.	0.0
Wing_Roll_Damping	Rolling moment coefficient due to due to roll rate, per dimensionless roll rate (P/2V).	Calculated

Wing_Dihedral	The dihedral angle of the wing, in degress.	0.0
Wing_Dihedral_Effect	Rolling moment coefficient due to slideslip, per radian. Does not include wing sweep back effects	Calculated
Wing_Taper	Wing taper ratio, tip/root	1.0

3.1.6 Yawing moment

The wing yawing moment depends on the geometry of the wing, the lift coefficient, the drag coefficient, and the yaw rate. It is calculated at runtime. Note that classical aerodynamics indicates that there should be a yawing moment which results from roll rate. The simulator does not yet include this term.

3.1.7 Control surfaces

The wing supports both flaps and aileron control surfaces. At runtime the force from each surface is calculated and applied to the wing based on the surface deflection, the surface effectiveness, the surface location, the surface span, and the dihedral angle of the wing.

The surface effectiveness is given as a flap effectiveness factor which is the ratio of the force generated from a unit deflection of the control surface to the force generated from a unit deflection of the entire wing section. The surface effectiveness can be calculated by the simulator or supplied in the parameter list.

The control surface can also produce a pitching moment. The pitching moment depends on a linear parameter $C_{M\delta}$ which describes the change in pitching moment of the local wing section as a function of control surface deflection.

Table 5. Wing parameter values for rolling moment estimation

Value	Meaning	Default
Wing_Area	The area of the main wing in meters^2.	Required
Wing_Span	The span of the main wing in meters.	Required
Wing_Dihedral	The dihedral angle of the wing, in degrees.	0.0
Left_Aileron_Inboard	The lateral offset of the inboard end of the left aileron, in meters. This value is usually negative.	0.0
Left_Aileron_Outboard	The lateral offset of the outboard end of the left aileron, in meters. This value is usually negative.	0.0
Left_Aileron_Chord	The average chord of the left aileron, in meters.	0.0
Left_Aileron_Effectiveness	Surface effectiveness factor	Calculated
Left_Aileron_CMdelta	Pitching moment of the local wing section as a function of surface deflection, per radian	Calculated
Left_Aileron_Channel	The channel number (0-9) of the signal used for the left aileron.	0
Left_Aileron_Sign	The sign convention applied to the channel signal. This is typically 1, but in cases where a single signal drives two surfaces (like left and right ailerons) a -1 may be needed	1
Right_Aileron_Inboard	The lateral offset of the inboard end	0.0

	of the right aileron, in meters. This value is usually positive.	
Right_Aileron_Outboard	The lateral offset of the outboard end of the right aileron, in meters. This value is usually positive.	0.0
Right_Aileron_Chord	The average chord of the right aileron, in meters.	0.0
Right_Aileron_Effectiveness	Surface effectiveness factor	Calculated
Right_Aileron_CMdelta	Pitching moment of the local wing section as a function of surface deflection, per radian	Calculated
Right_Aileron_Channel	The channel number (0-9) of the signal used for the right aileron.	5
Right_Aileron_Sign	The sign convention applied to the channel signal. This is typically 1, but in cases where a single signal drives two surfaces (like left and right ailerons) a -1 may be needed	1
Left_Flap_Inboard	The lateral offset of the inboard end of the left flap, in meters. This value is usually negative.	0.0
Left_Flap_Outboard	The lateral offset of the outboard end of the left flap, in meters. This value is usually negative.	0.0
Left_Flap_Chord	The average chord of the left flap, in meters.	0.0
Left_Flap_Effectiveness	Surface effectiveness factor	Calculated
Left_Flap_CMdelta	Pitching moment of the local wing section as a function of surface deflection, per radian	Calculated
Left_Flap_Channel	The channel number (0-9) of the signal used for the left flap.	4
Left_Flap_Sign	The sign convention applied to the channel signal. This is typically 1, but in cases where a single signal drives two surfaces (like left and right ailerons) a -1 may be needed	1
Right_Flap_Inboard	The lateral offset of the inboard end of the right flap, in meters. This value is usually positive.	0.0
Right_Flap_Outboard	The lateral offset of the outboard end of the right flap, in meters. This value is usually positive.	0.0
Right_Flap_Chord	The average chord of the right flap, in meters.	0.0
Right_Flap_Effectiveness	Surface effectiveness factor	Calculated
Right_Flap_CMdelta	Pitching moment of the local wing section as a function of surface deflection, per radian	Calculated
Right_Flap_Channel	The channel number (0-9) of the signal used for the right aileron.	9
Right_Flap_Sign	The sign convention applied to the channel signal. This is typically 1, but in cases where a single signal drives two surfaces (like left and right ailerons) a -1 may be needed	1

3.2 Contributions due to the horizontal tail

The tail of the aircraft is treated in almost identical fashion to the wing. The only difference is its physical location and size. In addition the tail has only elevator control surfaces not flaps and ailerons. Note that the parameter names for the tail are the same as those for the wing except that "`Wing_`" is replaced with '`Tail_`", and "`Flap_`" is replaced with "`Elevator_`". All coefficients are with respect to the area and chord of the tail.

3.3 Contributions due to the vertical fin

The simulator allows the aircraft to have two fins, a left and a right. The fin is similar to a wing, except it is oriented vertically. The fin contributes body axis components of: sideforce, drag, yawing moment, pitching moment, and rolling moment. In addition to the fin geometry the state variables which affect these force are: dynamic pressure, angle of sideslip, and yaw rate. Each of the parameter names given in the following sections should be preceded by either a "`Left_`" or a "`Right_`" depending on which surface is intended. All coefficients are with respect to the area and chord of the fin.

3.3.1 Side force

The fin side force depends on the geometry, angle of sideslip, and dynamic pressure. The model first estimates the side force coefficient, and then calculates the actual side force based on the dynamic pressure and fin area. The side force coefficient (C_Y) depends only on the angle of sideslip, and is estimated by look up table, or by the linear parameter $dC_Y/d\beta$. Note that this coefficient is typically negative

Estimating the fin sideforce by look up table is done by reading in data from the file indicated in the parameter list. This file must contain four columns of ASCII data in which the first column is the angle of sideslip in degrees, and the second column is the corresponding sideforce coefficient. The coefficient must represent the entire three-dimensional performance of the fin.

If the look up file is unavailable the simulator will estimate the $dC_Y/d\beta$ parameter. This is done by using the fin area and span (the vertical height) geometry data to determine the aspect ratio, and then apply the elliptical wing estimate for the three-dimensional lift curve slope. The estimate assumes that the two-dimensional lift curve slope is 2π. The estimate can be bypassed by supplying a suitable value in the aircraft description, see Table 6. Note that since there is a left find and a right fin, each parameter is preceded by either `Left_` or `Right_`.

Table 6. Fin parameter values for side force estimation

Value	Meaning	Default
Fin_Area	The area of the fin in meters^2.	0.0
Fin_Span	The vertical extent of the fin in meters.	0.0
Fin_SideForce_Slope	The side force coefficient slope of the fin, per radian	Calculated

3.3.2 Drag

The fin drag depends on the geometry, angle of sideslip, and dynamic pressure. The model first estimates the fin drag coefficient, and then calculates the actual drag based on the dynamic pressure and fin area. The fin drag coefficient (C_D) depends only on the angle of sideslip, and is estimated by look up table, or by a combination of fixed parasitic drag and calculated induced drag.

Estimating the fin drag by look up table is done in the same way that the wing lift estimate is done, except that the third column of the look up table is used. The drag coefficient must represent the entire three-dimensional performance of the fin.

If the look up file is unavailable the simulator will estimate the drag parameter. This is done by using the fin area and span geometry data to determine the aspect ratio, and then apply the elliptical wing estimate for induced drag, plus a constant offset term. The induced drag estimate can be adjusted with Oswald's span efficiency factor.

Table 7. Fin parameter values for drag estimation

Value	Meaning	Default
Fin_Area	The area of the fin in meters^2.	0.0
Fin_Span	The vertical extent of the fin in meters.	0.0
Fin_Span_Efficiency	Oswald's span efficiency factor	1.0
Fin_Parasitic_Drag	Constant value for the fin parasitic drag	0.0

3.3.3 Control surfaces

The fin supports a rudder control surface. At runtime the force from the surface is calculated and applied to the fin based on the surface deflection, the surface effectiveness, the surface span, and the surface location.

The surface effectiveness is given as a flap effectiveness factor which is the ratio of the force generated from a unit deflection of the control surface to the force generated from a unit deflection of the entire fin section. The surface effectiveness can be calculated by the simulator or supplied in the parameter list.

The control surface can also produce a yawing moment. The yawing moment depends on a linear parameter $C_{M\delta}$ which describes the change in pitching moment, of the local fin section, as a function of control surface deflection. This parameter is typically positive

Table 8. Wing parameter values for rolling moment estimation

Value	Meaning	Default
Fin_Area	The area of the fin in meters^2.	0.0
Fin_Span	The vertical extent of the fin in meters.	0.0
Rudder_Bottom	The vertical offset of the bottom end of the rudder in meters. Negative if above the cg.	0.0
Rudder_Top	The vertical offset of the top end of the rudder in meters. Negative if above the cg.	0.0
Rudder_Chord	The average chord of the rudder in meters.	0.0
Rudder_Effectiveness	Surface effectiveness factor	Calculated
Rudder_CMdelta	Pitching moment of the local wing section as a function of surface deflection, per radian	Calculated
Rudder_Channel	The channel number (0-9) of the signal used for the rudder.	3(L),8(R)
Rudder_Sign	The sign convention applied to the channel signal. This is typically 1, but in cases where a single signal drives two surfaces	1

	(like left and right ailerons) a -1 may be needed	

3.3.4 Pitch, roll, and yaw moment

The moments from the fin depends on the sideforce, drag, and location of the fin. The fin location is specified in three dimensions, X, Y, and Z

Table 9. Wing parameter values for pitching moment estimation

Value	Meaning	Default
Fin_X	Body axis X Distance between the fin aerodynamic center and the center of gravity, in meters. Negative if the fin is behind of the CG.	0.0
Fin_Y	Body axis Y Distance between the fin and the center of gravity, in meters. Positive if the fin is to the right of the CG.	0.0
Fin_Z	Body axis Z Distance between the fin and the center of gravity, in meters. Positive if the fin is below the CG.	0.0

3.4 Contributions due to the fuselage

The fuselage contributes forces and moments on all three axes. Since fuselage shapes come in many different varieties the simulator does not attempt to calculate the aerodynamic coefficients. Instead they must be supplied in the parameter list. All coefficients are with respect to the cross sectional area and length of the fuselage.

Table 10. Fuselage parameter values

Value	Meaning	Defaults
Fuse_Area	Reference area for the fuselage, usually the frontal area in meters^2.	0.0
Fuse_Length	Reference length for the fuselage in meters.	0.0
Fuse_Lift_Slope	Coefficient of lift of the fuselage as a function of angle of attack, per radian.	0.0
Fuse_Parasitic_Drag	Drag coefficient of the fuselage	0.0
Fuse_SideForce_Slope	The side force coefficient of the fuselage as a function of sideslip angle, should be negative. Per radian.	0.0
Fuse_Pitching_Moment_Slope	The pitching moment coefficient as a function of angle of attack. This does not include contributions due to the lift, drag, or sideforce on the fuselage. Per radian.	0.0
Fuse_Yawing_Moment_Slope	The yawing moment coefficient as a function of angle of attack. This does not include contributions due to the lift, drag, or sideforce on the fuselage. Per radian	0.0
Fuse_X	Body axis X Distance between the fuselage aerodynamic center and the center of gravity, in meters. Negative if the fuselage is behind of the CG.	0.0

Fuse_Y	Body axis Y Distance between the fuselage and the center of gravity, in meters. Positive if the fuselage is to the right of the CG.	0.0
Fuse_Z	Body axis Z Distance between the fuselage and the center of gravity, in meters. Positive if the fuselage is below the CG.	0.0

4 Propulsion model

The propulsion model estimates the loads on the simulated aircraft as a function of the throttle, engine data, propeller data, and aircraft state. The simulator supports left and right propulsion units, and each of the parameter names given in the following sections should be preceded by either a "`Left_`" or a "`Right_`" depending on which propulsion unit is intended.

Note if the propeller diameter is given as `0.0` then the propulsion model is not included and the simulation is that of an unpowered aircraft.

4.1 Engine model

The engine model calculates the torque and power for a given engine based on the throttle, rpm, and ambient pressure. It is based upon a simple one-dimensional look up table which relates wide-open-throttle (WOT) power to RPM. To calculate the power at each time step the engine model uses the current RPM to calculate the WOT power. That power number is then multiplied by the ratio of the ambient pressure to standard sea level pressure, and also by the throttle position (which is a number from 0 to 1). Fuel flow is calculated using a single number, the specific fuel consumption, which relates the fuel consumption to the amount of power produced.

The engine model is admittedly crude, but suffices for purposes of simulation. More advanced models can be constructed, but they require more detailed knowledge of the actual engine implementation. Estimating the engine power by look up table is done by reading in data from the file indicated in the parameter list. This file must contain two columns of ASCII data in which the first column is the RPM, and the second column is the corresponding WOT power, in Watts. The RPM must be given in ascending order.

Table 11. Engine parameter values

Value	Meaning	Default
Engine_LUT	File name for the Look up table of the engine WOT power (in Watts) versus RPM curve.	Required, unless prop diam = 0.0
Engine_BSFC	Brake specific fuel consumption for the engine, in grams of fuel per hour per Kilowatt of power	0.0
Engine_Channel	The channel number of the throttle signal	3(Left) 8(Right)

4.2 Propeller model

The propeller model calculates the thrust and torque of the propeller based on the RPM, and forward speed of the aircraft. To do this it uses a propeller look up table which gives the Coefficient of thrust and Coefficient of power as a function of the advance ratio. The simulator calculates the advanced ratio, and the uses the look up table to determine C_t and C_p. C_t is used to calculate the thrust from the propeller. C_p is used to calculate the power absorbed by the propeller.

The difference in power absorbed by the propeller and power produced by the engine changes the speed of the propeller/engine combination according to a rotational inertia parameter. Finally the thrust and torque of the propeller are applied to the aircraft based on the position of the propeller relative to the cg. Estimating the propeller performance by look up table is done by reading in data from the file indicated in the parameter list. This file must contain three columns of ASCII data in which the first column is the advance ratio, and the second column is the coefficient of power, and the third column is the coefficient of thrust. The advance ratio must be given in ascending order.

Table 12. Propeller parameter values

Value	Meaning	Default
Prop_LUT	File name for the Look up table of the propeller performance.	Required unless diameter = 0.0
Prop_Diameter	Diameter of the propeller in meters.	0.0
Prop_X	Body axis X Distance between the propeller and the center of gravity, in meters. Negative if the propeller is behind of the CG.	0.0
Prop_Y	Body axis Y Distance between the propeller and the center of gravity, in meters. Positive if the propeller is to the right of the CG.	0.0
Prop_Z	Body axis Z Distance between the propeller and the center of gravity, in meters. Positive if the propeller is below the CG.	0.0
Prop_Inertia	The rotational inertia of the propeller/engine combination in kg-m^2.	1.0

5 Inertia model

The inertia model includes the mass and rotational inertia of the whole airplane. The mass depends on the initial mass and the amount of fuel that has been burned. Rotational inertia is based on the inertias of the X, Y, and Z axis. In addition one cross component, X to Z is included.

Table 13. Inertia parameter values

Value	Meaning	Default
Gross_Mass	Mass of the entire aircraft, including fuel, in kg.	Required
Empty_Mass	Mass of the aircraft with no fuel, kg	Required
Roll_Inertia	Inertia about the X axis in kg-m^2.	Required
Pitch_Inertia	Inertia about the Y axis in kg-m^2.	Required
Yaw_Inertia	Inertia about the Z axis in kg-m^2.	Required
Roll_Yaw_Coupled_Inertia	XZ inertia coupling in kg-m^2.	0.0

In addition to mass and moments of inertia, the inertia model also includes the acceleration effects due to the avionics being mounted at a certain distance from the aircraft CG, as well as the frame transformations for inertial measurements due to misalignment between avionics and aircraft body frame. These parameters are listed below:

Table 14. Avionics (or IMU) mounting geometry

Value	Meaning	Default
IMU_Sensor_Roll_Angle	The roll angle of the avionics with respect to the body frame, in deg.	0.0
IMU_Sensor_Pitch_Angle	The pitch angle of the avionics with respect to the body frame, in deg.	0.0
IMU_Sensor_Yaw_Angle	The yaw angle of the avionics with respect to the body frame, in deg.	0.0
IMU_Sensor_Position_X	The X-axis component of the position vector of the avionics' center with respect to the aircraft CG, in m.	0.0
IMU_Sensor_Position_Y	The Y-axis component of the position vector of the avionics' center with respect to the aircraft CG, in m.	0.0
IMU_Sensor_Position_Z	The Z-axis component of the position vector of the avionics' center with respect to the aircraft CG, in m.	0.0

6 Landing gear model

The landing gear model provides the forces and moments that are applied to the airframe when a specific set of points on the aircraft come in contact with the ground. This include not just the landing gear itself but also other parts of the airframe that may come in contact with the ground.

The ground contact points are modeled each with coefficients for stiffness, damping, and longitudinal/lateral friction. There are two types of contact points - simple contact points, and wheels. They differ through the coefficients listed above. The longitudinal friction of a wheel will actually be a roll-friction type (resisting force proportional to the velocity). Also, the wheels can steer - that is, the friction forces are computed in the local frame of the wheel and rotated back to the aircraft frame.

The user can specify the location (with respect to the aircraft CG) of 3 wheels and 9 other contact points. The entire set is represented in the following two figures.

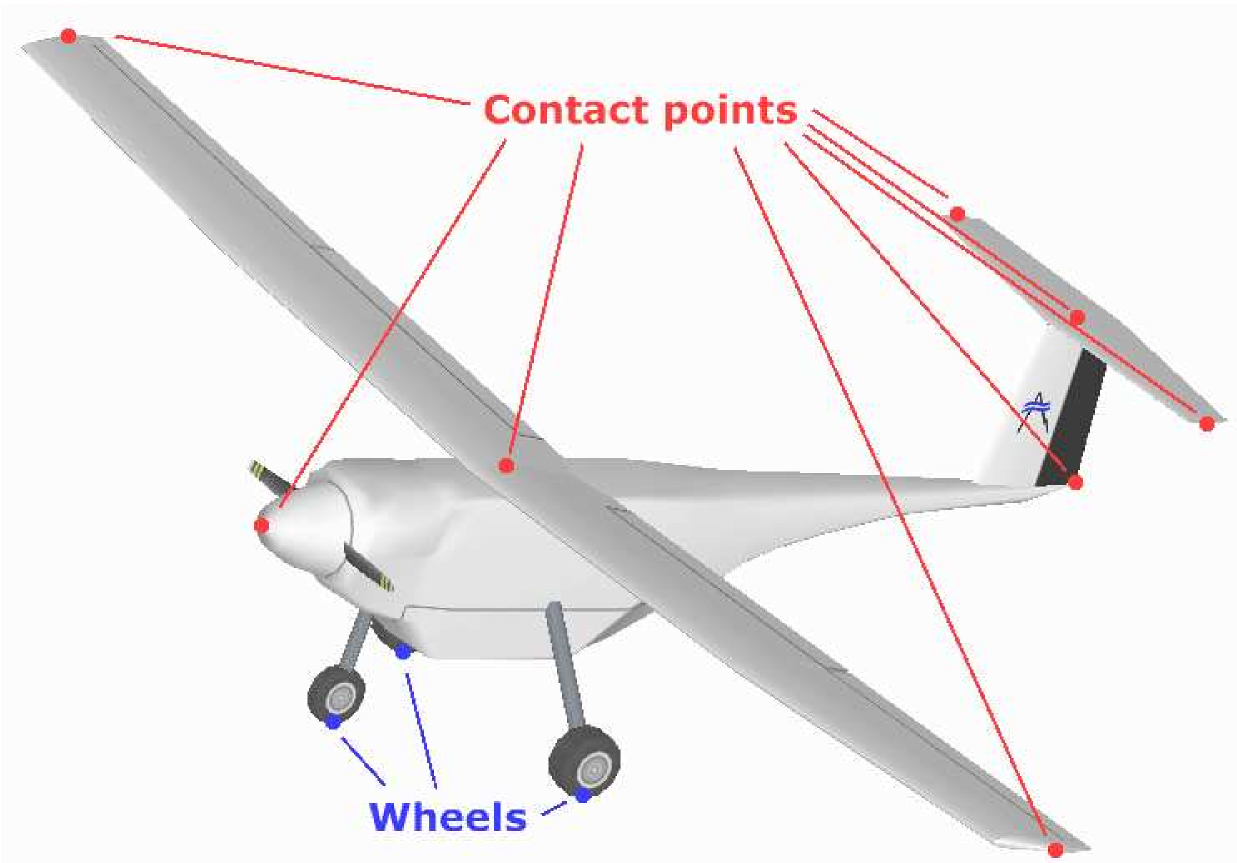

Figure 3. Top view of the ground contact points and wheels.

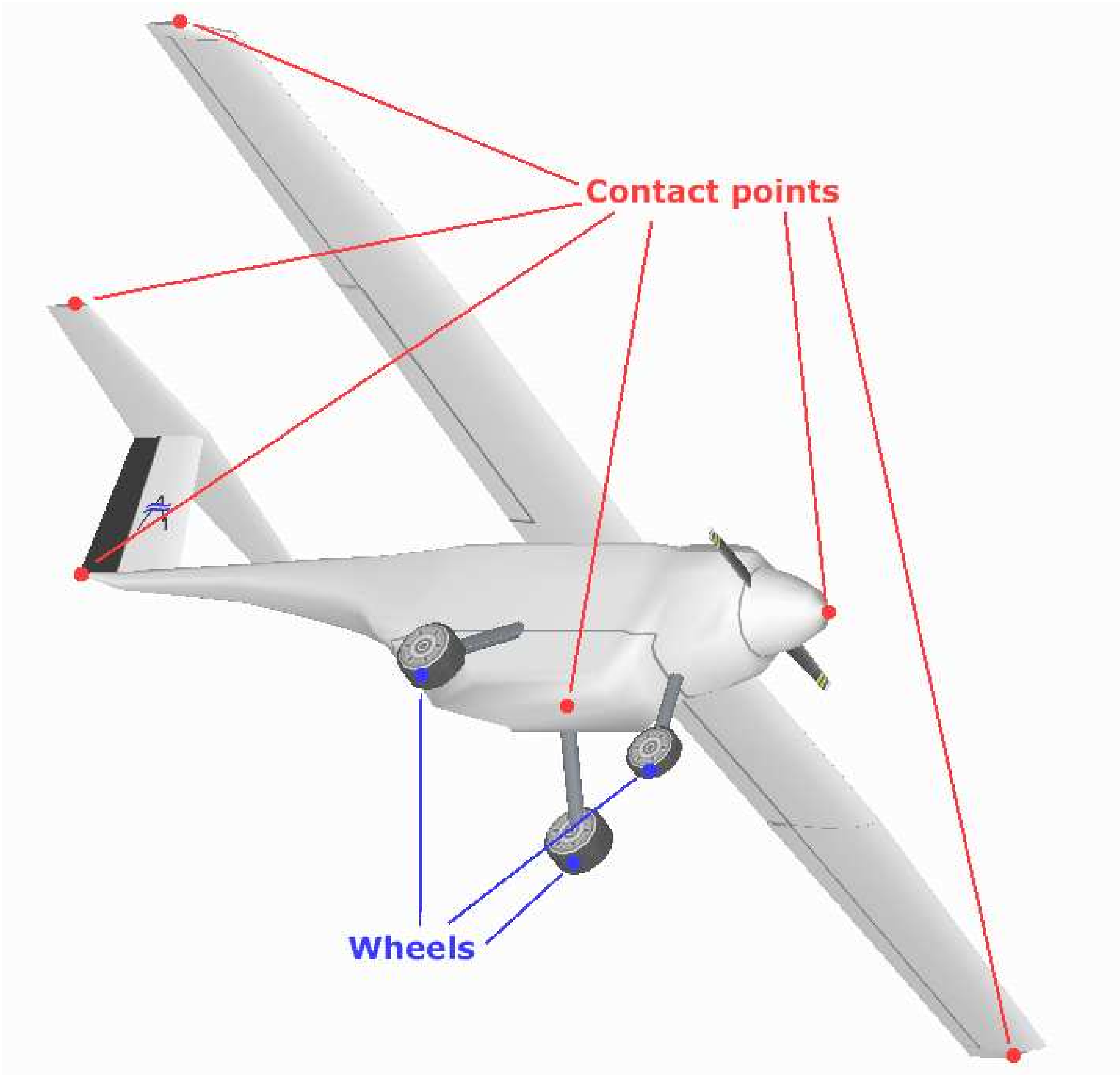

Figure 4. Bottom view of the ground contact points and wheels.

Table 15. Wheel contact points

Value	Meaning	Default
NoseWheel_Position_X	The X-axis component of the position vector of the nose wheel or tail wheel ground contact point with respect to the aircraft CG, in m. (positive if ahead of CG)	0.0
NoseWheel_Position_Y	The Y-axis component of the position vector of the nose wheel or tail wheel ground contact point with respect to the aircraft CG, in m. (positive to the right)	0.0

NoseWheel_Position_Z	The Z-axis component of the position vector of the nose wheel or tail wheel ground contact point with respect to the aircraft CG, in m. (positive if below the CG)	0.0
NoseWheel_RudderWheelRatio	If nose wheel is mechanically linked to the rudder, this is the ratio of steering angle to rudder deflection angle (negative for nose wheel, positive for tail wheel, leave at 0.0 for non-steerable)	0.0
NoseWheel_Steering_Channel	The Piccolo servo channel number for the steering actuator.	0
RightWheel_Position_X	The X-axis component of the position vector of the right wheel of the main gear with respect to the aircraft CG, in m. (positive if ahead of CG)	0.0
RightWheel_Position_Y	The Y-axis component of the position vector of the right wheel of the main gear with respect to the aircraft CG, in m. (positive to the right)	0.0
RightWheel_Position_Z	The Z-axis component of the position vector of the right wheel of the main gear with respect to the aircraft CG, in m. (positive if below the CG)	0.0
LeftWheel_Position_X	The X-axis component of the position vector of the left wheel of the main gear with respect to the aircraft CG, in m. (positive if ahead of CG)	0.0
LeftWheel_Position_Y	The Y-axis component of the position vector of the left wheel of the main gear with respect to the aircraft CG, in m. (positive to the right)	0.0
LeftWheel_Position_Z	The Z-axis component of the position vector of the right wheel of the main gear with respect to the aircraft CG, in m. (positive if below the CG)	0.0

Table 166. Other contact points

Value	Meaning	Default
ContactPoint_Top_Position_X	The X-axis component of the position vector of the top-mid fuselage point w.r.t aircraft CG, in m.	0.0
ContactPoint_Top_Position_Y	The Y-axis component of the position vector of the top-mid fuselage point w.r.t aircraft CG, in m.	0.0
ContactPoint_Top_Position_Z	The Z-axis component of the position vector of the top-mid fuselage point w.r.t aircraft CG, in m.	0.0
ContactPoint_Bottom_Position_X	The X-axis component of the position vector of the bottom-mid fuselage point w.r.t aircraft CG, in m.	0.0
ContactPoint_Bottom_Position_Y	The Y-axis component of the position vector of the bottom-mid fuselage	0.0

	point w.r.t aircraft CG, in m.	
ContactPoint_Bottom_Position_Z	The Z-axis component of the position vector of the bottom-mid fuselage point w.r.t aircraft CG, in m.	0.0
ContactPoint_Nose_Position_X	The X-axis component of the position vector of the fuselage nose point w.r.t aircraft CG, in m.	0.0
ContactPoint_Nose_Position_Y	The Y-axis component of the position vector of the fuselage nose point w.r.t aircraft CG, in m.	0.0
ContactPoint_Nose_Position_Z	The Z-axis component of the position vector of the fuselage nose point w.r.t aircraft CG, in m.	0.0
ContactPoint_Tail_Position_X	The X-axis component of the position vector of the fuselage tail endpoint w.r.t aircraft CG, in m.	0.0
ContactPoint_Tail_Position_Y	The Y-axis component of the position vector of the fuselage tail endpoint w.r.t aircraft CG, in m.	0.0
ContactPoint_Tail_Position_Z	The Z-axis component of the position vector of the fuselage tail endpoint w.r.t aircraft CG, in m.	0.0
ContactPoint_LWing_Position_X	The X-axis component of the position vector of the left wingtip w.r.t aircraft CG, in m.	0.0
ContactPoint_LWing_Position_Y	The Y-axis component of the position vector of the left wingtip w.r.t aircraft CG, in m.	0.0
ContactPoint_LWing_Position_Z	The Z-axis component of the position vector of the left wingtip w.r.t aircraft CG, in m.	0.0
ContactPoint_RWing_Position_X	The X-axis component of the position vector of the right wingtip w.r.t aircraft CG, in m.	0.0
ContactPoint_RWing_Position_Y	The Y-axis component of the position vector of the right wingtip w.r.t aircraft CG, in m.	0.0
ContactPoint_RWing_Position_Z	The Z-axis component of the position vector of the right wingtip w.r.t aircraft CG, in m.	0.0
ContactPoint_LStab_Position_X	The X-axis component of the position vector of the left horizontal stabilizer tip w.r.t aircraft CG, in m.	0.0
ContactPoint_LStab_Position_Y	The Y-axis component of the position vector of the left horizontal stabilizer tip w.r.t aircraft CG, in m.	0.0
ContactPoint_LStab_Position_Z	The Z-axis component of the position vector of the left horizontal stabilizer tip w.r.t aircraft CG, in m.	0.0
ContactPoint_RStab_Position_X	The X-axis component of the position vector of the right stabilizer tip w.r.t aircraft CG, in m.	0.0
ContactPoint_RStab_Position_Y	The Y-axis component of the position	0.0

	vector of the right stabilizer tip w.r.t aircraft CG, in m.	
ContactPoint_RStab_Position_Z	The Z-axis component of the position vector of the right stabilizer tip w.r.t aircraft CG, in m.	0.0
ContactPoint_Fin_Position_X	The X-axis component of the position vector of the top of the vertical tail w.r.t aircraft CG, in m.	0.0
ContactPoint_Fin_Position_Y	The Y-axis component of the position vector of the top of the vertical tail w.r.t aircraft CG, in m.	0.0
ContactPoint_Fin_Position_Z	The Z-axis component of the position vector of the top of the vertical tail w.r.t aircraft CG, in m.	0.0

7 Sensor models

The simulator supports sensor models which can be used to corrupt the sensor information sent to the avionics. The sensor data are included in a separate file, which is referenced from the main file using the flag `Sensors=` followed by the path name to the sensors file. The sensors that are modeled are given in Table

Table 17. Sensor names for the simulator sensor model

Value	Meaning
Latitude_Sensor_	Latitude of the vehicle from the GPS
Longitude_Sensor_	Longitude of the vehicle from the GPS
Height_Sensor_	Height of the vehicle from the GPS
VNorth_Sensor_	North component of ground speed from the GPS
VEast_Sensor_	East component of ground speed from the GPS
VDown_Sensor_	Down component of ground speed from the GPS
PDynamic_Sensor_	Dynamic pressure sensor
PStatic_Sensor_	Static pressure sensor
Roll_Rate_Sensor_	Avionics axis roll rate sensor
Pitch_Rate_Sensor_	Avionics axis pitch rate sensor
Yaw_Rate_Sensor_	Avionics axis yaw rate sensor
X_Accel_Sensor_	Avionics x-axis acceleration sensor
Y_Accel_Sensor_	Avionics y-axis acceleration sensor
Z_Accel_Sensor_	Avionics z-axis acceleration sensor

Each parameter in the file is formed by concatenating the sensor name with the parameter name. Each sensor can be modeled according to the following parameters:

Table 18. Sensor modeling parameters

Value	Meaning	Units
Offset	Sensor output when the signal being sensed is at zero	Output
Order	Order of the butterworth low pass filter that the sensor output is passed through. Use 0 for no filter	N/A
Bandwidth	Cutoff frequency in Hz of the low pass filter.	Hz
Gain	Gain of the sensor signal.	N/A
Resolution	Resolution of the sensor signal	Output
Noise	Sensor noise, used to scale random number added to the sensor output before going through the low pass filter	Output
Min	Minimum saturation limit.	Output
Max	Maximum saturation limit.	Output
Drift_Rate	Maximum offset drift rate.	Output/s
Max_Drift	Maximum offset drift value.	Output
Drift_Hold	Amount of time to spend at a specific offset drift value before allowing the sensor to drift again.	s

In addition to the individual sensor parameters there are also gobal sensor parameters controlling the GPS update rates.

Table 18. Sensor modeling parameters

Value	Meaning	Units

GPS_Period	The update rate of the GPS This value should 1000 for the M12 GPS and 250 for the uBlox GPS (used in Piccolo II)	ms
GPS_Position_Lag	The time lag of the position output, prior to the velocity projection. This should be 500 for the M12 and 125 for the uBlox	ms
GPS_Velocity_Lag	The time lag of the velocity output. Should be 1100 for the M12 and 250 for the uBlox.	ms

8 Actuator models

The simulator supports actuator models which can be used to limit the responsiveness of the control surface actuators. The actuator data are included in a separate file, which is referenced from the main file using the flag `Actuators=` followed by the path name to the actuators file. The actuators that are modeled are given in the table below.

Table 19. Actuator names for the simulator actuators model

Actuator	Channel number	Output units
Left_Aileron_	0	Radians
Left_Elevator_	1	Radians
Left_Throttle_	2	0-1
Left_Rudder_	3	Radians
Left_Flap_	4	Radians
Right_Aileron_	5	Radians
Right_Elevator_	6	Radians
Right_Throttle_	7	0-1
Right_Rudder_	8	Radians
Right_Flap_	9	Radians

Each parameter in the file is formed by concatenating the actuator name with the parameter name. The actuators can be modeled according to the following parameters:

Table 20. Actuator modeling parameters

Value	Meaning	Units	Defaults
Bandwidth	Bandwidth of the actuator	Hz	5
Rate_Limit	Maximum rate of actuator	output/s	100
Min_Limit	Minimum deflection of the actuator	output	-1
Max_Limit	Maximum deflection of the actuators	output	1
Error	Output error of the actuator	output	0
Backlash	Backlash error of the actuator	output	0

Each actuator model in the Simulator includes the following elements:

- Input saturation
- Scaling
- Linear dynamics
- Output saturation
- Rate limiting

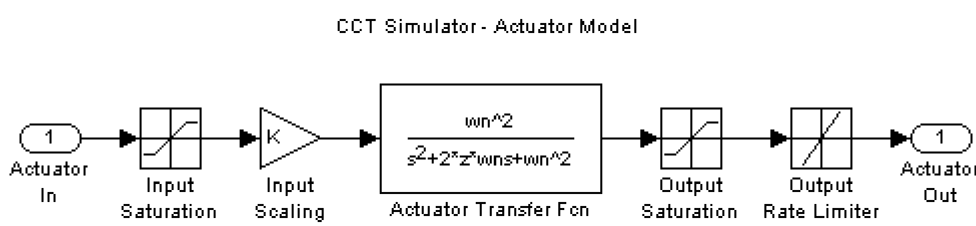

Figure 5. Actuator Model block diagram

The linear dynamics of the actuator are described by the transfer function:

$$G(s) = \frac{\omega_n^{\ 2}}{s^2 + 2\varsigma\omega_n + \omega_n^{\ 2}}$$

where the natural frequency and the damping are:

$$\omega_n = 2\pi B$$

$$\varsigma = \frac{1}{2}10^{-\frac{G}{20}}$$

where B is the actuator bandwidth in Hz, at a gain G = -3dB.

The input saturation block limits the input signal between two values $I_{\min}$ and $I_{\max}$.

The output saturation block limits the output signal between two values $O_{\min}$ and $O_{\max}$.

The scaling block scales the actuator input linearly using the following equation:

$$out = O_{\min} + \frac{O_{\max} - O_{\min}}{I_{\max} - I_{\min}} \cdot (in - I_{\min})$$

The rate limiter limits block limits the rate of variation of the output signal between two values $\left.\frac{dO}{dt}\right|_{\min}$ and $\left.\frac{dO}{dt}\right|_{\max}$.

9 Launcher model

The simulator supports a rail launcher. This is intended to model simple catapult launch systems in which the vehicle is accelerated along a fixed rail. The rail length, force, heading, and pitch angle can be controlled with parameters in the dynamics file.

Table 17. Launcher parameters values

Value	Meaning	Defaults
Launcher_Length	The length of the launch rail in meters	10.0
Launcher_Initial_Force	The force applied to the vehicle in the direction of the rail, a the start of the launch	160.0
Launcher_Final_Force	The force applied to the vehicle in the direction of the rail, a the end of the launch. Between the beginning and the end of the launch the force applied is scaled linearly from the initial force to the final force	160.0
Launcher_Heading	The heading of the launch rail in degrees. This determines the direction the vehicle will move as it accelerates on the rail	0.0
Launcher_Pitch	The pitch angle of the launch rail in degrees. This determines how steeply the vehicle will be climbing when it leaves the rail.	20.0

10 Initialization

After a model is loaded the simulator will initialize all its state variables with zero. However the initialization can be overridden using inputs available on the screen. Further more the state of the model can be saved to a file and loaded later as initialization data. The form of the file is similar to the dynamics model files, i.e each variable is named in the file. The table below gives a list of the state variables that can be initialized.

Table 21. State initialization file

Value	Meaning
Alpha	Angle of attack of the model, in degrees
Beta	Angle of sideslip of the model, in degrees
Roll	Euler roll angle, in degrees
Pitch	Euler pitch angle, in degrees
Yaw	Euler heading angle, in degrees
P	Body axis roll rate, in degrees per second
Q	Body axis pitch rate, in degrees per second
R	Body axis yaw rate, in degrees per second
TAS	True air speed in meters per second
Latitude	Latitude in degrees
Longitude	Longitude in degrees
Altitude	Altitude in meters
Left_Engine_RPM	RPM of the left engine
Right_Engine_RPM	RPM of the right engine

11 Installation and operation

The simulator works as a standalone executable for which no special installation is required. However the CAN interface driver must be installed. This is done using the USBCanModule driver disc. In addition when starting the simulator the USB to CAN module must be plugged in. The simulator will detect that it is plugged in and configure itself to use it as its source of control surface data. If the CAN module is not connected the simulator will attempt to connect to a software simulator source, and if it fails to do that it will use any installed joysticks for the control surface data.

11.1 Installing FlightGear

The current version of the Simulator supports FlightGear 0.9.2, 0.9.4, and 0.9.8.

11.1.1 FlightGear version 0.9.2

Flight Gear flight simulator version 0.9.2 should be installed by unzipping its archives. First, make sure that the directory C:\Flightgear on your computer does not exist already. If it exists and you would like to keep that version as well, rename that directory. Otherwise, delete it before installing the newer version.

- Unzip the file "fgfs-base-0.9.2.zip" to the root directory of your C: drive. This will create a new directory called Flightgear and its directory tree. This archive contains most of theFlightgear data.
- Unzip the file "fgfs-win32-msvc-bin-0.9.2.zip to the root directory of your C: drive. This will add more files to the Flightgear directory and its tree. This archive contains the binary executables of Flightgear.
- Copy the two batch files "runfgfsnet-c172.bat" and "runfgfsnet-j3cub.bat" to the C:\Flightgear directory. These two batch files can be used to launch the Flightgear program in "external flight dynamics" mode, which is exactly what we want for interfacing with our Simulator. The only difference between the two files is that they load different visual models - one of them loads a Cessna 172, and the other one a Piper Cub.
- Optionally, unzip the files "wXXXnXX.tar.gz" to C:\Flightgear\Data\Scenery. The 4 files included provide scenery and airports for the west coast of the United States, the highlighted area in the map shown in Figure 2.

11.1.2 FlightGear version 0.9.4 and 0.9.8

The newer versions of FlightGear for Windows platform come as a self-installing archive. Execute "fgsetup0.9.4.exe" program and the Setup will guide you through the installation. When completed copy the two batch files "runfgfsnet-c172.bat" and "runfgfsnet-j3cub.bat" to the directory where FlightGear was installed. These two batch files can be used to launch the FlightGear in the correct mode.

11.2 Running the Simulator

- Start flight gear by using one of the supplied batch files. This will start FlightGear in external flight dynamics mode. The program will accept aircraft state data (position, velocity and attitude) from a UDP network connection, on port 5500. If you prefer to use the default starting batch file, the exact command line for launching FGFS in external dynamics mode is:

```
runfgfs.bat --native-fdm=socket,in,30,,5500,udp --fdm=external
```

- Then, on the simulator computer launch the Simulator application and select File-Open to open the text file which contains you simulation model.
- From the top menu, select External/FlightGear… This will bring up a dialog box in which you type the network name of the computer running Flightgear. Once this is done, you should see the aircraft in FlightGear exactly at the location at which it is set in the Simulator.
- The aircraft states can now be initialized properly in the Simulator, after which you can start the simulation anytime by pressing the Start button. The Stop button will halt the simulation but it will maintain the aircraft states to the last computed values. The Reset button will reset the aircraft states to the program defaults.

Since we are using FlightGear as a visualization tool for the simulation, a brief presentation of the FlightGear view options is necessary. When you first start the program, you are presented with a graphics window which looks similarly to Figure 3.

Figure 8. The FlightGear cockpit view.

Most of the instruments in the cockpit view are not useful in our application since they are designed for manned aircraft, and they are not connected to our external flight dynamics model anyways. We can un-clutter the screen by removing the cockpit using the key "S". Now the view should be similar to Figure 4. The cockpit can be enabled by presing the key "S" again.

Figure 9. Forward view with cockpit removed.

The forward-looking pilot view is not the only one available in FlightGear. Views are fully configurable, so any view is theoretically possible. A mini how-to document about configuring the views can be found here: http://www.flightgear.org/Docs/fgfs-viewer-howto.html

You can navigate between different views using the "V" key.

There is a set of default views that are already configured when you started FlightGear. One of them is the pilot view which we started with. We also have two external views, one of them has fixed orientation with respect to the airplane, the other one has fixed orientation with respect to the ground (inertial frame). An example of the external view is shown in Figure 5.

Figure 10. External view in FlightGear.

For the external views, we can change the location of the camera with respect to the airplane using the SHIFT + arrow keys. Also the zoom can be adjusted using X (zoom in) and SHIFT+X (zoom out).

Another type of view is the ground/tower view, which is particularly useful for flying UAVs/RPVs under manual control. An example is shown in Figure 6.

Figure 11. Tower view in FlightGear.

11.3 Displaying Multiple Aircraft

If there are multiple Piccolo Simulators running on the local network, all aircraft can be displayed in a single FlightGear window. Each instance of the Simulator application will still have to connect to individual FlightGear computers, but with the appropriate command line parameters, one or multiple FlightGear computers can display all of the aircraft simultaneously, see Figure 7.

On the client FGFS computer run FlightGear with the following command line parameters:

```
runfgfs.bat --aircraft=j3cub --native-fdm=socket,in,30,,5500,udp
--fdm=external --multiplay=out,10,XXX.XXX.XXX.XXX,5501 --callsign=player1
```

where XXX.XXX.XXX.XXX is the IP address of the server.

On the server FGFS computer (where all aircraft will be displayed) run FlightGear with the following command line parameters:

```
runfgfs.bat --aircraft=j3cub --native-fdm=socket,in,30,,5500,udp
--fdm=external --multiplay=in,10,XXX.XXX.XXX.XXX,5501 --callsign=player2
```

where XXX.XXX.XXX.XXX is the IP address of the server.

Figure 12. Multiple Aircraft Displayed in FlightGear.

11.4 Displaying custom 3-D Aircraft Models

FlightGear can accept models in various 3-D formats; a document which explains how to use custom aircraft visual models in FlightGear is presented at:

http://www.flightgear.org/Docs/fgfs-model-howto.html

As an example, we will run a Microsoft Flight Simulator 2000 model, the Predator UAV. First we copy the 3-D model file "predator.mdl" to the C:\Flightgear\data\Models\ directory. In the same directory we also create a small XML file, which defines the path property to the 3-D model file. The content will be in this case:

```
<?xml version="1.0"?>
<PropertyList>
 <path>Models/predator.mdl</path>
</PropertyList>
```

We save this file as "predator.xml". Finally, we edit the batch file which we used to start Flight Gear, by adding a new command line parameter, --prop:/sim/model/path= which we set to point to the newly created XML file. The line in the batch file will have:

```
%TOP_ROOT%\BIN\WIN32\FGFS.EXE
--prop:/sim/model/path=%FG_ROOT%\Models\predator.xml
--native-fdm=socket,in,30,,5500,udp --fdm=external
```

Launching FlightGear with this batch file will result in loading the Predator FS2000 model, as shown in Figure 9.

Figure 13. A Microsoft Flight Simulator 2000 Model loaded in FlightGear.