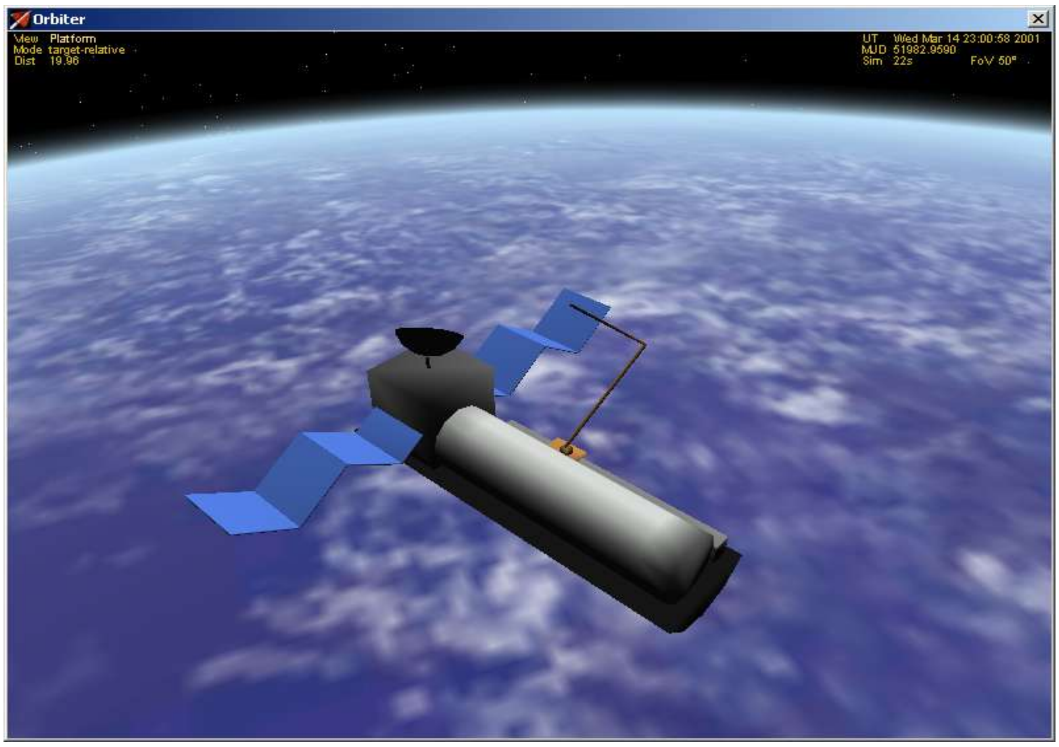
Spacecraft support
Version : 050630
Author : Vinka
Orbiter Version compatibility : 050216



Introduction
The spacecraft.dll module gives support to spacecraft that can be configured through initialisation file.
Most of the conventional spacecrafts will be supported by this module. You will be able to create your own spacecraft without the need to write a specific DLL.

Limitation
- maximum of 10 spacecrafts in the same scenario.
- maximum of 16 thrusters exhaust rendering for main engines
- maximum of 16 thrusters exhaust rendering for hover engines
- maximum of 16 thrusters exhaust rendering for retro engines
- maximum of 64 thrusters exhaust rendering for attitude engines
- maximum of 10 payloads
- maximum of 16 docking ports
- no limitation of animation sequence (dynamically allocated)
- no limitation on components in an animation sequence (dynamically allocated)
- maximum of 1024 groups for all the animation sequences

Module Command keys :
'j' : jettison the payloads

Keys for animation sequence must be defined in the ini file (recognized keys are K, G, left-shift 0,1,2, …, 9

THE BEST WAY TO UNDERSTAND IT ALL, IS TO HAVE A LOOK AT THE EXAMPLE DELIVERED.

Files needed :

Modules\spacecraft.dll : the module implementing the generic spacecraft class
Config\spacecraft\spacecraft.cfg : the class configuration file

Config\spacecraft\<vessel name>.ini : the ini file which define the parameters of the spacecraft called "vessel name" in the scenario file (see below)

Scenario File format

The scenario file must define the ship as follow :

```
…
BEGIN SHIP
vessel name:spacecraft\spacecraft
…
```

The spacecraft module will search a ini file with the name of the vessel defined in the scenario file only in the config\spacecraft directory.

The following lines must be added to the scenario file

```
CONFIGURATION 0 (=launch, 1=in flight, 2=landed)
CURRENT_PAYLOAD (current payload not jettison yet)
SEQ 0 –2 0.0 (animation sequence 0 in state –2 (pause) at value 0.0)
SEQ 1 1  0.0 (animation sequence 1 in state 1 (run) at value 0.0)
```

Initialisation File format

This file follow the format of the standard windows ini files. The following sections and items can/must be defined :

[TEXTURE_LIST]
You must declare the texture files that will be used for engine exhaust rendering, this list is preloaded in the dll.

- TEX_x : the texture name that will be used in one of the ENG_TEX field. x stands for 1, 2, … up to 16.

Note that this texture library is static and common to all spacecraft vessels.This means that a maximum of 16 textures can be used for all spacecraft vessels but different vessels may use the same texture by using the same name.

[PARTICLESTREAM_x]
Definition of a particle stream. "x" stands for 1,2, … up to 13. Particle stream parameters are those defined in the orbiter SDK (see it for details). You must define a particle stream section before using it in the ini file.
Three particule streams are predefined and can be used. They are named "contrail","exhaust" and "rcs" (they are based on the sample definition of Orbiter sample code).

- NAME : the name associated with the particle stream, this name will be used to associate the particle stream with engines. Three default particle stream are already defined in the dll : exhaust, contrail and rcs.
- SRCSIZE : see orbiter API doc for details
- SRCRATE : see orbiter API doc for details
- V0 : see orbiter API doc for details
- SRCSPREAD : see orbiter API doc for details
- LIFETIME : see orbiter API doc for details
- GROWTHRATE : see orbiter API doc for details
- ATMSLOWDOWN : see orbiter API doc for details
- LTYPE : see orbiter API doc for details. Possible values are EMISSIVE, DIFFUSE
- LEVELMAP : see orbiter API doc for details. Possible values are LVL_LIN, LVL_FLAT, LVL_SQRT, LVL_PLIN, LVL_PSQRT
- LMIN : see orbiter API doc for details
- LMAX : see orbiter API doc for details
- ATMSMAP : see orbiter API doc for details. Possible values are ATM_FLAT, ATM_PLIN, ATM_PLOG
- AMIN : see orbiter API doc for details
- AMAX : see orbiter API doc for details
- TEX : texture file name for particle stream rendering

Note that this particle stream library is static and common to all spacecraft vessels.This means that a maximum of 13 particle streams can be used (+3 predefined) for all spacecraft vessels but different vessels may use the same particle stream by using the same name.

[LIFT]
lift section is now obsolete because new atmospheric flight model is used.

[AERODYNAMICS]
In this version, the lift and drag profile are not modifiable and are those of the Deltaglider. Modifiable parameters are listed below

- VAIRFOIL_ATTACK : attack point (x,y,z) in vessel coordinates of the vertical airfoil
- VAIRFOIL_CHORD : chord length of the vertical airfoil
- VAIRFOIL_AREA : vertical airfoil area
- VAIRFOIL_ASPECT : vertical airfoil
- HAIRFOIL_ATTACK : attack point (x,y,z) in vessel coordinates of the horizontal airfoil
- HAIRFOIL_CHORD : chord length of the horizontal airfoil
- HAIRFOIL_AREA : horizontal airfoil area
- HAIRFOIL_ASPECT : horizontal airfoil
- ELEVATOR_ATTACK : attack point of the elevator
- ELEVATOR_AREA : elevator controle surface area
- ELEVATOR_LIFT : elevator maximum change in lift when fully oriented
- ELEVATOR_ANIM : sequence number for animation of elevator
- RUDDER_ATTACK : attack point of the rudder
- RUDDER_AREA : rudder controle surface area
- RUDDER_LIFT : rudder maximum change in lift when fully oriented
- RUDDER_ANIM : sequence number for animation of rudder
- AILERON_ATTACK : attack point of the aileron
- AILERON_AREA : aileron controle surface area
- AILERON_LIFT : aileron maximum change in lift when fully oriented
- AILERON_RIGHT_ANIM : sequence number for animation of the right aileron

- AILERON_LEFT_ANIM : sequence number for animation of the left aileron
- SPEEDBRAKE_ATTACK : speed brake attack point
- SPEEDBRAKE_DRAG : speed brake maximum change in drag when fully deployed
- SPEEDBRAKE_ANIM : sequence number for animation of speed brake

[CONFIG]
spacecraft configuration parameters section, most of these parameters correspond to the definition given by Martin schweiger in is documentation. Not all the parameters must be defined, other parameters will get default values when not specified.

- MESHNAME="atlantis" : the mesh name to be loaded
- SIZE=19.6 : the vessel radius size (m). The sphere defined must enclosed the vessel completely.
- EMPTY_MASS=104326 : the vessel empty mass (kg)
- FUEL_MASS=20000 : the vessel fuel mass (kg)
- MAIN_THRUST=53400 : the vessel main engine thrust (N)
- MAIN_TEX=texture name in the library to render main thrust
- MAIN_PSTREAM1=first particule stream name in the library to render main thrust
- MAIN_PSTREAM2=second particule stream name in the library to render main thrust
- RETRO_THRUST=0 : the vessel retro engine thrust (N)
- RETRO_TEX=texture name in the library to render retro thrust
- RETRO_PSTREAM1=first particule stream name in the library to render retro thrust
- RETRO_PSTREAM2=second particule stream name in the library to render retro thrust
- HOVER_THRUST=0 : the vessel hover engine thrust (N)
- HOVER_TEX=texture name in the library to render main thrust
- HOVER_PSTREAM1=first particule stream name in the library to render hover thrust
- HOVER_PSTREAM2=second particule stream name in the library to render hover thrust
- ATTITUDE_THRUST=7740 : the vessel attitude engine thrust (N), thrust is considered to be applied by thrusters located at 1 meter from the rotation axis, you should adapt the engine thrust according to the real radius.
- ATT_TEX=texture name in the library to render attitude thrust
- ATT_PSTREAM1=first particule stream name in the library to render attitude thrust
- ATT_PSTREAM2=second particule stream name in the library to render attitude thrust
- REENTRY_PSTREAM=particule stream name in the library to render atmospheric reentry
- ISP=5000 : the fuel specific impulsion (N/kg/sec)
- PMI=(78.2,82.1,10.7) : the principal moment of inertia (kgm^2)
- CW_Z_POS=0.2 : the drag coefficient on the z axis when the vessel is moving forward
- CW_Z_NEG=0.5 : the drag coefficient on the z axis when the vessel is moving backward
- CW_X=1.5 : the drag coefficient on the x axis
- CW_Y=1.5 : the drag coefficient on the y axis
- CROSS_SECTION=(234.8,389.1,68.2) : the cross section in x,y,z axis (m^2)
- COG=8 : the center of gravity of the vessel above ground when landed (m)
- PITCH_MOMENT_SCALE=0.00001 : the pitch moment scale
- BANK_MOMENT_SCALE=0.00002 : the bank moment scale
- ROT_DRAG=(0.5,1.0,1.0) : the rotational drag
- WING_ASPECT=0.7 : obsolete
- WING_EFFECTIVENESS=2.5 : obsolete
- TRIM : the trim value
- LAUNCH_PT1=(1,0,0), LAUNCH_PT2=(-1,0,0), LAUNCH_PT3 = (0,0,1) : the coordinates of the touchdown points in launch configuration
- LAND_PT1=(1,0,0), LAND_PT2=(-1,0,0), LAND_PT3 = (0,0,1) : the coordinates of the touchdown points in landing configuration. If you want both launch and land to be the

same, just define LAND_PT1,2,3. The landing configuration is automatically selected after launch when a altitude of 100 meter is reached.

- FOCUS=1 : specify which payload will get the focus when jettisoned, to keep the focus on the mothercraft specify –1. Payloads valid values start at 0 and goes to number of payloads-1.
- VISIBLE=1 : specify if the internal mesh rendering must be done in cockpit view
- CAMERA=(0,0,0) : specify the position of the point of view in cockpit view (should correspond to the cockpit position in the mesh). The default value will set the cockpit at the front of vessel.

[DOCK_0]
...
[DOCK_15]
docking ports list (maximum 16 docking ports per vessel)

- POS=(0,2.44,10.44) : the docking port position
- DIR=(0,1,0) : the docking port approach direction
- ROT=(0,0,-1) : the docking port longitudinal rotation alignment vector

[EX_MAIN_0]
...
[EX_MAIN_15]
main engine exhaust rendering definition sections (as many section as engines rendering are required)

- OFF=(-2.05,3.45,-14.2) : the offset of the rendering
- DIR=(-0.050, 0.099, -0.994) : the direction of the rendering
- LENGTH=4 : the length of the rendering at full thrust
- WIDTH=0.5 : the width of the rendering at full thrust

[EX_RETRO_0]
...
[EX_RETRO_15]
same as EX_MAIN but for retro engines

[EX_HOVER_0]
...
[EX_HOVER_15]
same as EX_MAIN but for hover engines

[EX_ATT_0]
...
[EX_ATT_63]
attitude engines exhaust rendering definition sections

- OFF=(0,1.6,17) : the offset of the rendering
- DIR=(0,1,0) : the direction of the rendering
- LENGTH=0.3 : the length of the rendering at full thrust
- WIDTH=0.3 : the width of the rendering at full thrust
- ROT_AXIS=X : the rotation axis (X, Y or Z)
- ROT_CW=1 : the rotation direction (0=CCW, 1=CW (clock wise)
- LIN_AXIS=Y : the translational axis (X, Y or Z)
- LIN_CW=1 : the translational direction (0=negative axis or 1= positive axis)

Note that the ROT_ parameters or the LIN_ parameters may be missing as the attitude engine can only be used for translation or rotation respectively.

[ANIM_SEQ_0]
[ANIM_SEQ_1]
...
animation sequence definition sections, this corresponds to the definition of the ANIM_SEQ given by Martin in the sdk documentation

- INIT_POS : value between [0,1] which correspond to the position of the original mesh group in the mesh file (default is 0)
- KEY=3 : the keypad key that will start the animation sequence. Recognized keys are K (usually used for payload bay open/close), G (for gear up/down), left-shift 0, 1, 2, …, 9
- DURATION=10 : the duration of the sequence
- REPEAT : repeat mode (0=no repeat (default), 1=repeat by restarting cycle (0->1;0->1;0->…), 2=repeat by reversing cycle (0->1->0->1->0->…)

In normal mode, the animation sequence will generate a value from 0.0 to 1.0 while in running mode (1). The value will change in 'duration' seconds. Once the value 1.0 is reached, the simulation will go in pause mode (-2). Restarting the animation by pressing a key will turn the pause mode (-2) in running mode (-1). The value will now change from 1.0 to 0.0. Pressing the key will in running mode will revert the running mode (1->-1 or –1->1). In the scenario file you specify the sequence number, the running mode (1,2,-1,-2) and the initial value between 0.0 and 1.0. The mesh design must always represent the moving parts in the specified initial position. If you want to have the original mesh position at Orbiter start, you should configure the scenario file with <seq num> 2 0.0. If you want the other extreme position, you should configure the scenario file with <seq num> -2 1.0

The same key can be used to start/stop one or more sequences (simultaneously).

[ANIM_COMP_0]
[ANIM_COMP_1]
...
animation component definition sections.

FOR ROTATION :

- SEQ=0 : the animation sequence containing this component
- GROUPS=91, 92, 93 : the mesh group list that must be animated
- RANGE=(0.0,1.0) : the range value when the animation takes place within the sequence (between 0 and 1)
- TYPE=ROTATE : specify that the animation is a rotation
- ROT_PNT=(-2.8,1.35,0.0) : the point of rotation (one point where the axis of rotation passes through it)
- ROT_AXIS=(0,0,1) : the rotation axis
- ANGLE=160. : the angle of rotation (negative value for rotation in other direction)
- PARENT= : the ANIM_COMP number of the parent if any. The parent must always be declared before the child.

FOR TRANSLATION

- SEQ=3 : the animation sequence containing this component
- GROUPS=6 : the mesh group list that must be animated
- RANGE=(0.,1.) : the range value when the animation takes place within the sequence (between 0 and 1)
- TYPE=TRANSLATE : specify that the animation is a translation

- SHIFT=(0,-0.8,0) : the translation value that is applied.
- PARENT= : the ANIM_COMP number of the parent if any. The parent must always be declared before the child.

FOR SCALING
- SEQ=3 : the animation sequence containing this component
- GROUPS=6 : the mesh group list that must be animated
- RANGE=(0.,1.) : the range value when the animation takes place within the sequence (between 0 and 1)
- TYPE=SCALE : specify that the animation is a scaling
- SCALE=(0.1,0.1,1) : the scale values that are applied.
- REF=(-0.340,-3.1855,0) : the reference point to apply the scaling (fixed point where the mesh seems to be scaled around).
- PARENT= : the ANIM_COMP number of the parent if any. The parent must always be declared before the child.

Composition of movements :
the animation components are now managed with reference to parent components (fully compliant with the Orbiter definition).

[PAYLOAD_0]
...
[PAYLOAD_9]
payloads definition sections

- MESHNAME=Carina : mesh name of payload. You can define up to 5 meshes for one payload, they must be declared in the same string, with each mesh name separated by a ; (semi-column). Ex: mesh1;mesh2
- OFF=(0,10,-5) : (x,y,z) vector position of payload mesh offset in launcher (meters). If you have declared more than one mesh for the payload, you must declare the same number of offset in the same order. Offset positions are separated by ; (semi-columns). Ex: (0,0,25);(0,0,27)
- MASS=1200 : payload mass in kg
- MODULE=carina : module name to be called at vessel create. This is the config file name of the vessel type of the payload.
- NAME : the name that will be given to the vessel at payload release time
- SPEED : translation speed for payload jettison relative to spacecraft (in m/s)
- ROT_SPEED : rotation speed for payload jettison relative to spacecraft (in rad/s)

[SOUND]
The sound section gives support to OrbiterSound version 3.0

- MAIN_THRUST : specify the sound file to be played for main thrust
- HOVER_THRUST : specify the sound file to be played for hover thrust
- RCS_THRUST_ATTACK : specify the sound file to be played for RCS thrust
- RCS_THRUST_SUSTAIN : specify the sound file to be played for RCS thrust
- AIR_CONDITIONNING : specify the sound file to be played for air conditionning
- COCKPIT_AMBIENCE_1 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_2 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_3 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_4 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_5 : specify the sound file to be played for cockpit ambience

- COCKPIT_AMBIENCE_6 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_7 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_8 : specify the sound file to be played for cockpit ambience
- COCKPIT_AMBIENCE_9 : specify the sound file to be played for cockpit ambience

Examples

The following examples can be downloaded with the spacecraft add-on, their purpose is to illustrate the possibilies of spacecraft.dll, they are probably more usefull for developpers than for end users.

Animations

a space platform demonstrating various animations possibilities.
shift+0 : antenna rotation. The antenna is rotating permanently over 360° (repeat mode 1)
shift+1 : robotic arm activation. Demonstrat composition of movements (translation of the base and rotations of the arms). The movement is permanently cycling from one end to the other (repeat mode 2)
shift+2 : solar panel deployment. Demonstrate composition of movements (rotations only)
shift+3 : inflatable structure. Demonstrate the use of the scale function and composition of movements (translation and scaling).
shift+4 : rotation of the arms

Atlantis Final Approach

The Atlantis Space Shuttle in final approach at KSC. The new aerodynamic model is implemented, use shift+ Numpad 1 to deploy air brake

Atlantis

The space shuttle Atlantis in orbital configuration. The payload bay contains three small payloads that can be jettison (J key). Various ejection speed (translation & rotation) are demonstrated. The payload bay is commanded with 'K' key and the landing gear by 'G' key. At second payload jettison, the focus change to the payload (as specified in ini file), use F3 and select sts-101 to return the focus on to the shuttle.

Deltaglider final approach

The Deltaglider in final approach at KSC. The new aerodynamic model is implemented with control surface animations. Use 'G' key to deploy landing gear, Use shift+Numpad 4 to deploy air braje.

Deltaglider landed

The Delta glider landed at the Cape.'K' key commands the nose cone, 'G' key commands the landing gear, shift + Numpad 0 key commands the air lock, shift + Numpad 1 commands the radiator, shift + Numpad 2 commands the escape ladder, shift + Numpad 3 commands the top hatch,shift + Numpad 4 commands the air brake.

Deltaglider

The Delta glider in orbital configuration. See previous example for the command keys.

Multiple docking

Two deltagliders docked to a station module with 2 docking ports. All three vessels are defined from spacecraft.dll.

Russian Dolls

This example illustrates the possibility to define payloads composed of multiple meshes. Each ship is carrying the smaller ones. Jettison the small ships then change the focus and jettison again until all vessels are separated. You should end this scenario with 6 ships, all of class spacecraft

Vertical take off Horizontal landing

The Delta glider landed vertically, ready for take off. Going over 100 m altitude will arm the horizontal landing.

Distribution

The distribution is taking advantage of the orbiter support for sub-directories. no file will be placed in the root directories. This should result in better classification and in no existing file being replaced.

Base package (always required)
module\spacecraft.dll : module implementation of the spacecraft vessel class
config\spacecraft\spacecraft.cfg : spacecraft vessel class configuration file
add-on docs\spacecraft.pdf : this documentation file

For developers
The source code can be obtained upon request.

Credits/Copyright

Space shuttle, DeltaGlider : from the original Orbiter package
Source code : all source code by Vinka inspired from the various sample code by Martin Schweiger.
Thanks to BigJimW and Roger “Frying Tiger” Long for beta testing this version.

Known bug

There is not much protection against wrong ini file therefore this will result in Orbiter crash. My advice is to start from a working ini file and modify it gradually.

Support

Any questions, help, request or bug report to vinka@swing.be

What will come next

- attachment management
- robotic arm management