

Engineering Sketch Pad (ESP)



Advanced User Training Session 03 Using OpenVSP Models in ESP

John F. Dannenhoffer, III

john@geocentrictech.com

Geocentric Technologies LLC

updated for v1.29+

- Static OpenVSP models in ESP
 - use case and user-facing process
 - behind the scenes
- Parametric OpenVSP models in ESP
 - use case and user-facing process
 - behind the scenes
 - sensitivities
- Summary
- Homework exercises

- Robust boolean operations
- Fillets (and Flends)
- Add non-airplane features
- Add components imported via STEP or IGES files
- Linked multi-disciplinary and multi-fidelity models
- Sensitivities
- Multi-user collaboration (“pair-programming”)
- Embedded Python interpreter (pyscript)
- Linkage to analysis tools (via CAPS)
- Incorporation of laser-scan data (via Plugs)

- Use case:
 - user has an **OpenVSP** model and wants to combine it into a single body, apply fillets, add other components, or use any analysis supported by **CAPS**
 - the Bodys have fixed geometry (i.e., are not parametric)
- User process:
 - the **.vsp3** file is generated by **OpenVSP** and must exist before the process starts
 - write a **csm** script that calls the **vsp** UDP with the name of the **.vsp** file:

```
UDPRIM vsp3 filename $$/transport.vsp3
```
 - execute **serveESP** on the **.csm** file

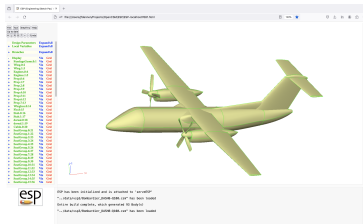
```
serveESP transport
```

- `udpVsp3`
 - writes an `angelscript` that directs `OpenVSP` to run the model and dump a STEP file that contains all (untrimmed) surfaces in the `OpenVSP` model
 - executes `vspscript`
 - imports STEP file
 - combines surfaces in each `OpenVSP` component into a `SolidBody` (or `SheetBody` if open)
- Note: process for transforming `OpenVSP`'s input parameters to geometry is done by `OpenVSP` — not ESP

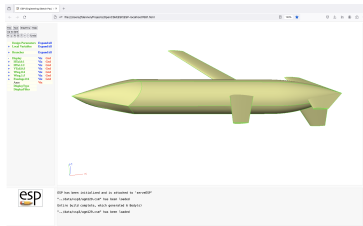


Static Model Examples

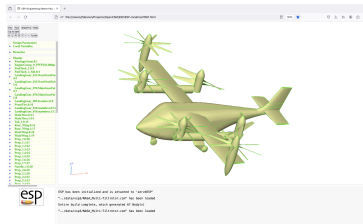
Note: one ESP Body for each OpenVSP Component



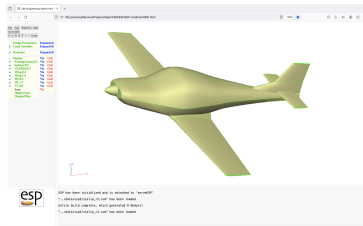
Bombardier_DASH8-Q100



agm129



NASA_Multi-Tiltrotor



sialia_r2

- Create simple ESP script

```
# transport_raw
# written by John Dannenhoffer

UDPRIM    vsp3    filename $$/transport_raw.vsp3

END
```

- Open ESP

- serveESP transport_raw

- Use case:
 - user has an **OpenVSP** model and wants to perform parametric variations, design optimization, or sensitivity analysis
 - the **OpenVSP** model should have “User Parms” in the “ESP_Group”
- User process:
 - every time the user modifies the **.vsp3** file:
 - in **serveESP**, use **Tool**→**VspSetup**, to write a **.udc** file that contains **DESPMTR**, **LBOUND**, and **UBOUND** statements from the tagged variables in the **.vsp3** file, and a **UDPRIM vsp3** statement to generate an updated configuration
 - write a **.csm** file that calls the UDC:
UDPRIM /transport
 - execute **serveESP** on the **.csm** file
serveESP transport



Parametric OpenVSP Model Demo — 1

- Show transport built in ESP
 - `serveESP transport_ref`
- Show transport built in OpenVSP
 - `vsp transport.vsp3`
 - **Model**→**User Params**
- Set up linkage between OpenVSP and ESP
 - `serveESP`
 - **Tool**→**VspSetup**
 - `transport.vsp3`
 - `transport.udc`
 - **File**→**Edit:** <new file>
 - `UDPRIM /transport`
 - **Save**
 - `transport`

- Show/change DESPMTRs in ESP
 - (serveESP continued)
 - change wing:sweep to 15
 - **Press to Re-build**
 - change wing:sweep back to 35
 - **Press to Re-build**
- Combine Bodys into single Body
 - (serveESP continued)
 - **Branches→Union**
 - toMark = 1
 - **Press to Re-build**



- Show sensitivities
 - (serveESP continued)
 - wing:dihebral
 - **Compute geom sens**
 - **Compute tess sens**
- Add wing/fuselage fillets
 - **File→Edit: transport.csm**
add FILLETs at the wing/fuselage junctions
DESPMTR filrad 2.5

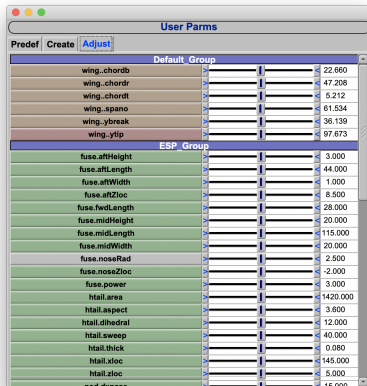
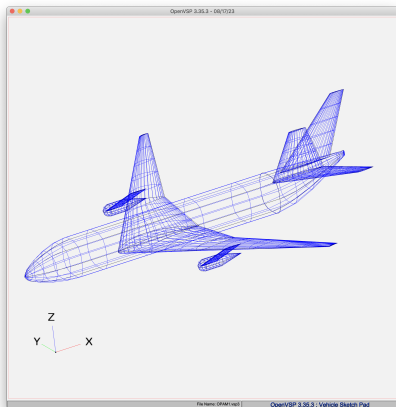
SELECT EDGE 1 0 3 0 0
FILLET filrad @sellist 1

SELECT EDGE 2 0 3 0 0
FILLET filrad @sellist 1
 - **Press to Re-build**



Parametric OpenVSP Model Demo — 4

Screen-shots of OpenVSP





Parametric OpenVSP Model Demo — 5

Note: one ESP Body for each OpenVSP

ESP (Engineering Sketch Pad) X

file:///Users/fdannenhoffer/Projects/OpenCSM/ESP/ESP-localhost7681.html

120%

File Edit Display/Hide Help

Up to date

Design Parameters Collapse All

- face:
 - aftHeight 3
 - aftLength 44
 - aftWidth 1
 - aftZloc 8.5
 - ftailLength 28
 - midftailHeight 20
 - midftailLength 115
 - midWidth 20
 - noseRad 2.5
 - noseZloc -2
 - power 3
- tail:
 - area 1420
 - aspect 3.6
 - dihedral 12
 - overlap 40
 - thick 0.08
 - sloc 145
 - zloc 5
- pod:
 - diameter -15
 - diameter -5
 - length 25
 - thick 0.25
 - y 0.5
- pylon:
 - diameter 1
 - diameter 1
 - length 9.5
 - thick 0.1
- vial:
 - area 610
 - aspect 1.8
 - overlap 45
 - taper 0.28
 - thick 0.08
 - sloc 150
 - zloc 9
- wing:
 - aspect 1

ESP has been initialized and is attached to 'serveESP'

"../data/vsp3/OPAW1_raw.csm" has been loaded
uses: "../data/vsp3/OPAW1.udc"

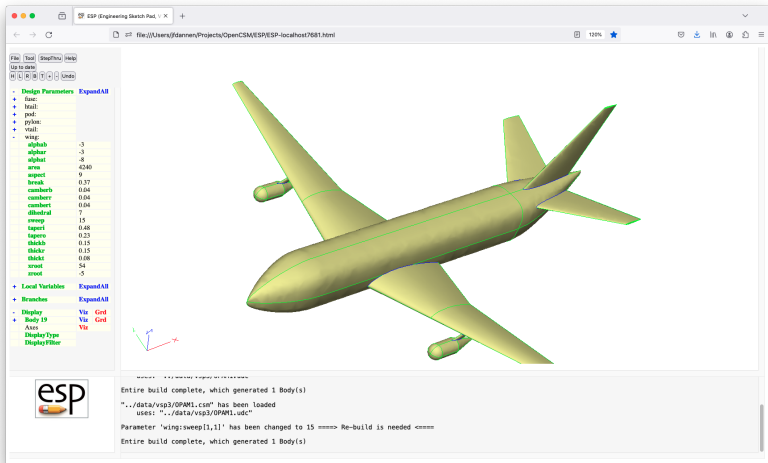
Entire build complete, which generated 10 Body(s)

"../data/vsp3/OPAW1_raw.csm" has been loaded
uses: "../data/vsp3/OPAW1.udc"



Parametric OpenVSP Model Demo — 6

Wing sweep changed to 15°





Parametric OpenVSP Model Demo — 7

Note: single Body with wing/fuselage fillets

File Edit Structure Help
Up to date
[Icons]

Design Parameters Collapse All

- Root
 - airlight 3
 - airlength 44
 - airwidth 1
 - airzinc 8.5
 - fuelLength 28
 - midairlight 20
 - midairlength 115
 - midairwidth 20
 - noseRad 2.5
 - noseZinc -2
 - perme 3
- head:
 - area 1420
 - aspect 3.6
 - dihedral 12
 - sweep 40
 - thick 0.08
 - slat 145
 - slat 5
- pod:
 - diameter -15
 - diameter -5
 - length 25
 - thick 0.25
 - yl 0.5
- pylon:
 - draped 1
 - drawing 1
 - length 9.5
 - thick 0.1
- tail:
 - area 610
 - aspect 1.8
 - sweep 45
 - upper 0.28
 - thick 0.08
 - slat 150
 - slat 9
- wing:
 - ...

ESP has been initialized and is attached to 'serveESP'
"../data/vsp3/OPAWL_fillet.csm" has been loaded
uses: "../data/vsp3/OPAWL.udc"
Entire build complete, which generated 1 Body(s)
"../data/vsp3/OPAWL_fillet.csm" has been loaded
uses: "../data/vsp3/OPAWL.udc"

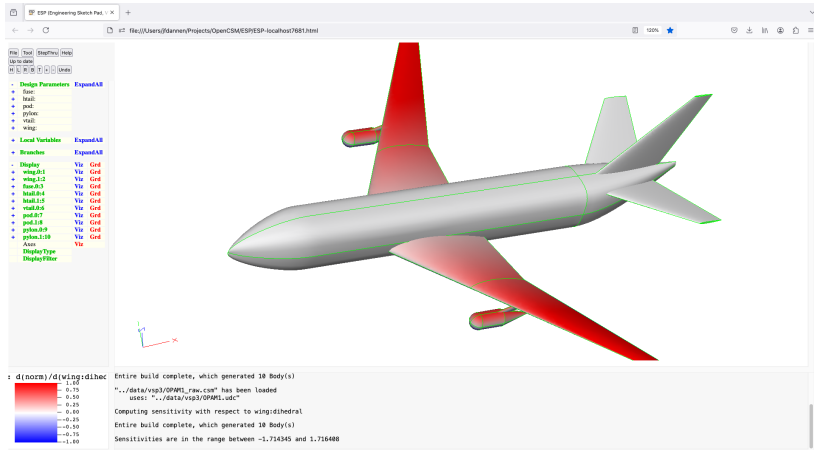
- ESP has the ability to compute two types of sensitivities:
 - configuration sensitivities tell derivative of the normal component at any point with respect to any group of design parameters (and is exact)
 - tessellation sensitivities tell derivative of tessellation point movement with respect to any group of design parameters (and is approximate since point placement algorithm is unknown to ESP)
- These sensitivities can be applied to any to any design parameter in a parametric OpenVSP model

- The derivative of the B-spline control point locations are computed via finite differences (using a base and perturbed model)
- The sensitivities at any location on the final model are computed analytically using ESP's normal sensitivity mechanism



Parametric OpenVSP Model Demo — 8

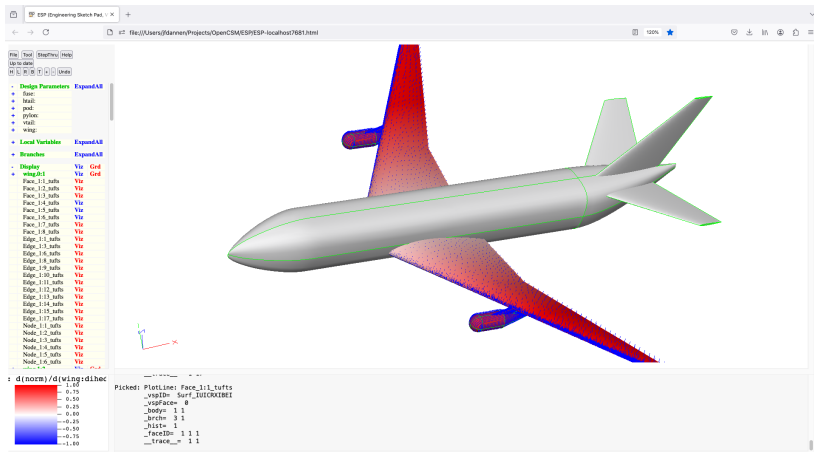
Geometric Sensitivities wrt Wing Dihedral





Parametric OpenVSP Model Demo — 9

Tessellation Sensitivities wrt Wing Dihedral



- OpenVSP models can now be imported into ESP
 - static models are represented as fixed-shape Bodys
 - parametric models can be generated for any change in an ESP design parameter
- Once in ESP, users can:
 - combine the Bodys into a single Body, apply fillets, add other components, or use any analysis supported by CAPS
 - can compute sensitivities (for parametric models)

- Execute `serveESP` on `exercises/session03/transport_ref` to see the original transport that was written in ESP
- Execute `vsp` on `exercises/session03/transport.vsp3` to see the transport reimplemented in OpenVSP
 - the `VSP3_ROOT` environment variable must point to OpenVSP's top-level directory
- Write and execute `transport_raw.csm` that executes the `vsp3` UDC to see OpenVSP's raw geometry represented in ESP
- Write and execute `transport_unioned.csm` to UNION all of OpenVSP's BODYS combined into one Body
- Write and execute `transport_fillet.csm` that starts with `transport_unioned.csm` and then creates a FILLET between the right wing (Body 1) and fuselage (Body 3) and another between the left wing (Body 2) and the fuselage (Body 3)

- Execute the VspSetup tool in serveESP to start from `exercises/session03/transport.vsp3` to generate the UDC file `transport.udc`
- Write and execute `transport_despmtrs.csm` by starting with `transport_unioned.csm` and incorporating `transport.udc`
 - Change the `wing:sweep` to 15
 - Change the `wing:sweep` back to 35
 - Examine the geometric sensitivity of the configuration with respect to `wing:dihedral`
 - this may be slow since the sensitivities are computed via finite differences and so it requires a complete rebuild
 - Examine the tessellation sensitivities of the configuration with respect to `fuse:midWidth`
 - if the tufts are too small, set the velocity associated with `fuse:midWidth` to 10 and **PressToRe-build**