Direct Generation of 3D Overset Grids from Solid Models

John F. Dannenhoffer, III

L. C. Smith College of Engineering & Computer Science

Syracuse University

11^{th} Symposium on
Overset Composite Grids and Solution Technology
October 2012

Overview

- Background & Objective
- Basic strategy
- Sample configuration
- Scalability study
- Conclusions

Background & Objective

- Background
  - overset grids can be used to produce high-accuracy, structured-grid viscous flow solutions
  - biggest limitation is the labor needed to:
    - decompose configuration into surface-sets on which to generate the component grids
    - identify regions in which collar grids need to be generated
  - fortunately, many configurations are currently modeled in a solid-modeling CAD system
- Objective
  - automate the overset grid generation process

Feature Tree Example

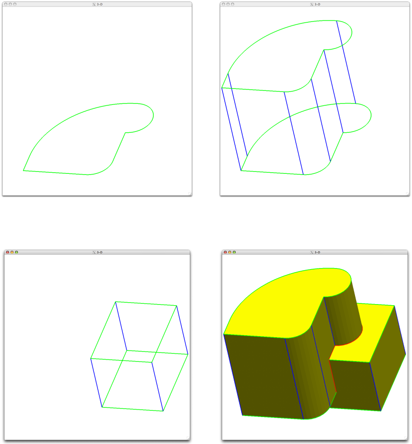
```
skbeg      0.0  0.0  0.0
   linseg  3.0  0.0  0.0
   cirarc  3.7  0.3  0.0  4.0  1.0  0.0
   linseg  4.0  3.0  0.0
   cirarc  5.0  4.0  0.0  4.0  5.0  0.0
   cirarc  1.2  3.8  0.0  0.0  1.0  0.0
skend

extrude    0.0  0.0  5.0

box        3.0  2.0 -1.0  3.0  .0  3.0

union

end
```

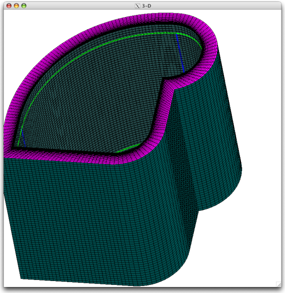


Overall Strategy

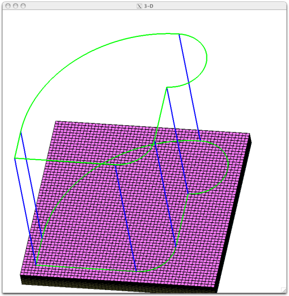
- Exploit duality between feature tree and overset grid system
  - for each primitive solid, generate basic grid(s)
  - for each primitive spine, generate collar grid
  - for each Boolean spine, generate collar grid
  - generate global grid
  - cut holes & trim grids
  - set up donor information

Generate Basic Grids — Exterior

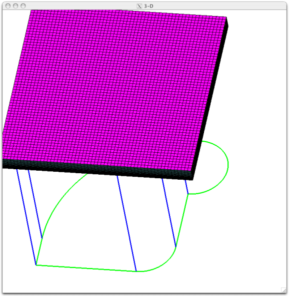
wrap-around grid



lower cap grid



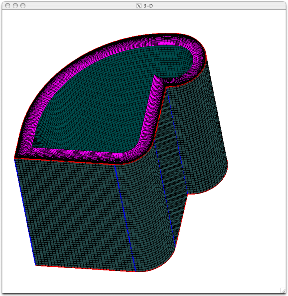
upper cap grid



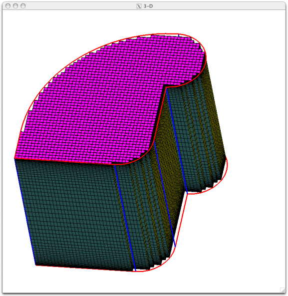
- Based upon user-specified:
  - nominal on-surface spacing
  - nominal off-body spacing
  - layer thickness
  - stretching ratio

Generate Basic Grids — Interior

wrap-inside grid

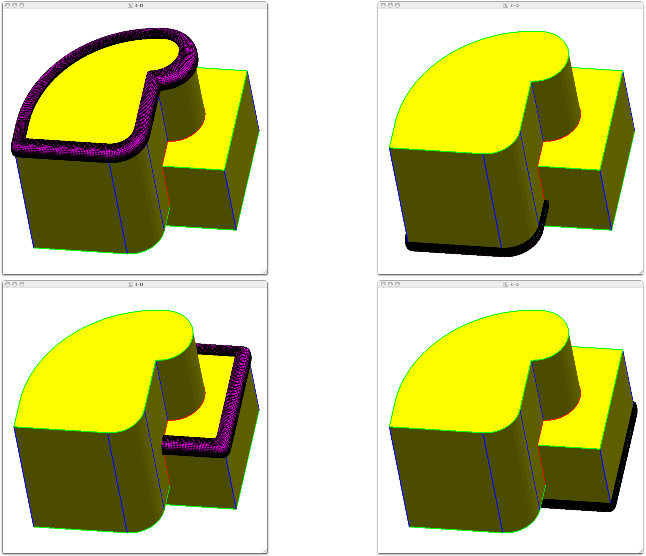


plug grid (after cutting)

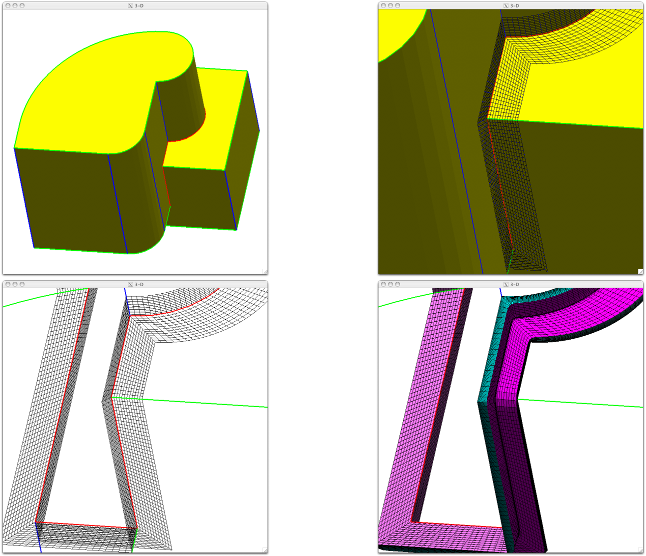


- Based upon user-specified:
  - nominal on-surface spacing
  - nominal off-body spacing
  - layer thickness
  - stretching ratio

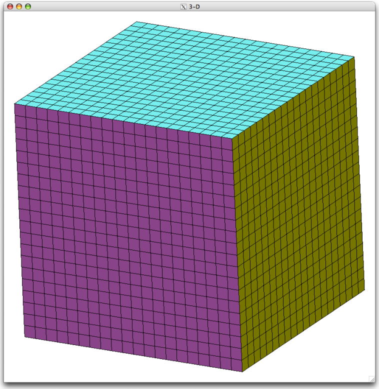
Generate Collar Grids — Primitives



Generate Collar Grids — Boolean



Generate Global Grid



- Cartesian grid based upon user-specified:
  - global spacing (or number of points)
  - buffer around component grids

Cut Holes and Trim Grids

- For each test FIELD point $(x, y, z)^T_{\text{test}}$ in each grid
  - loop through each primitive solid
    - transform test point to unit coordinates $(\hat{x}, \hat{y}, \hat{z})^T$ using pre-computed homogeneous coordinate transformation matrix $M \cdot (x, y, z, 1)^T_{\text{test}} = (\hat{x}, \hat{y}, \hat{z})^T$
    - for simple primitives (such as box), use tests such as $-1 \leq \hat{x} \leq +1$, $-1 \leq \hat{y} \leq +1$, and $-1 \leq \hat{z} \leq +1$
    - for primitives based upon sketches, use 2-D ray-crossing method for $(\hat{x}, \hat{y})$ applied to the sketch and $-1 \leq \hat{z} \leq +1$ in other direction
  - mark "inside" points as HOLE for primitives added to the model
  - mark "outside" points as HOLE for primitives subtracted from the model
- Trim grids by removing bounding grid planes that contain only HOLE points

Setup Donor Information — 1

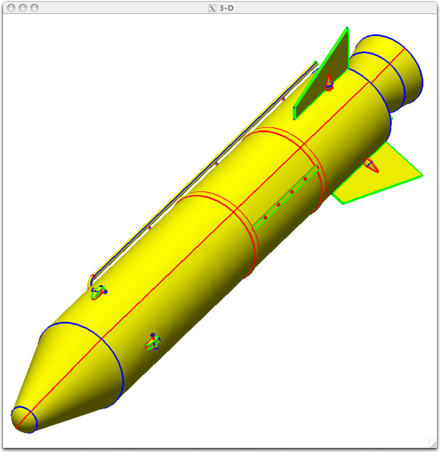
- Loop through all grids
  - create a *regional octree* that contains the range of cells associated with each octant
- Loop through all FIELD points
  - convert each to a HOLE if it is surrounded by other HOLES
- Loop through all SURFACE points
  - convert each to a HOLE if its off-body neighbor is a HOLE
- Loop through all FIELD points
  - convert each to a FRINGE if one of its neighbors is a HOLE
- Loop through all FIELD points
  - convert each to a FRINGE2 if one of its neighbors is a FRINGE

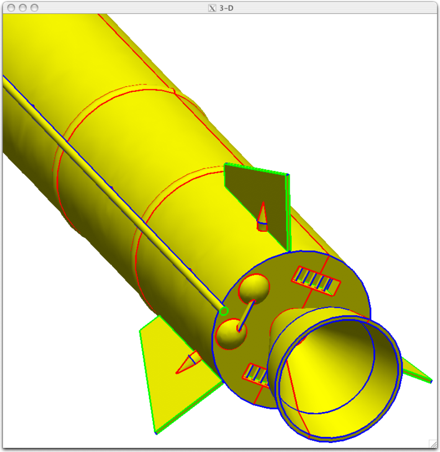
Setup Donor Information — 2

- For each FRINGE or FRINGE2 point in each grid, loop through the other grids and
  - skip the grid if its smallest cell volume is larger than current candidate cell
  - skip the grid if the FRINGE point is not in its bounding box
  - loop through the cells in the octree region that contains the FRINGE point
    - if the cell is supported by a FRINGE(2) or HOLE point, skip it
    - if the cell's volume is larger that the current candidate, skip it
    - remember this candidate cell if it contains the FRINGE point
- If any stencil searches fail
  - modify grid generation parameters for grids in question
  - regenerate grids and repeat

Sample Configuration — JMR3

51 features, 58 primitive solids, 57 Boolean operators

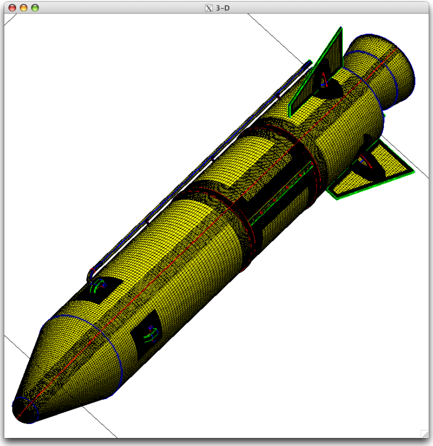


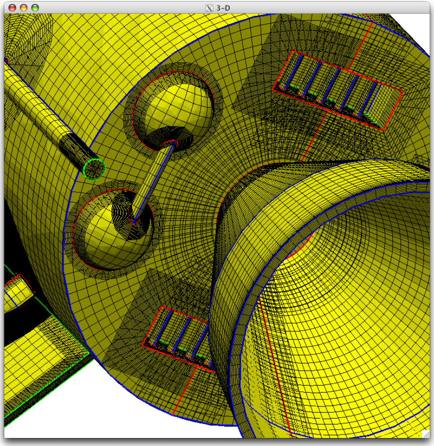


194 faces, 462 edges, 296 nodes

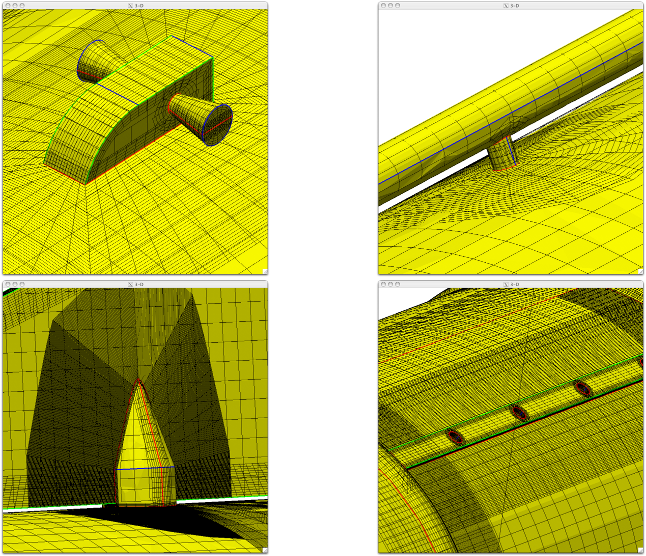
Surface Grids — JMR3

76 basic grids, 152 collar grids, 1 global grid

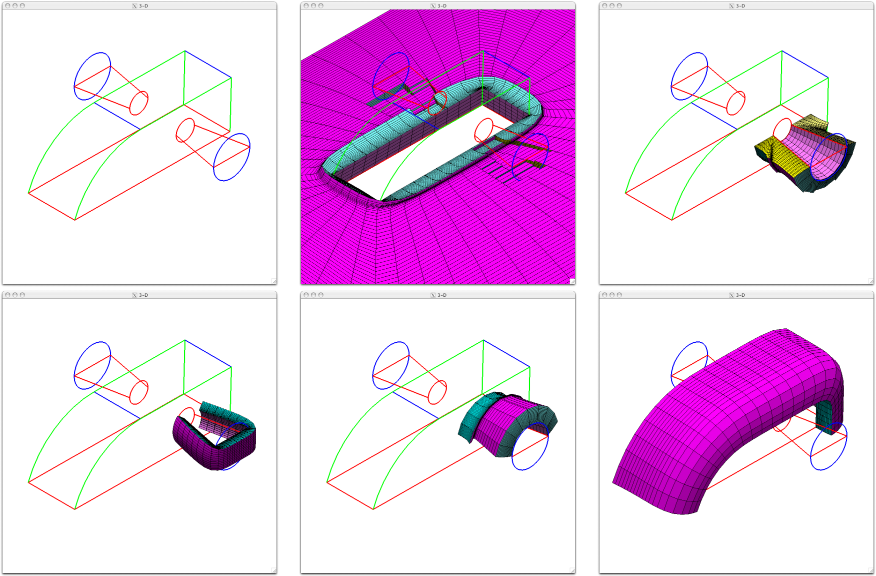




Close-ups of Grids — JMR3



Collar Grids Near a Thruster — JMR3



Summary of JMR3 Grid System

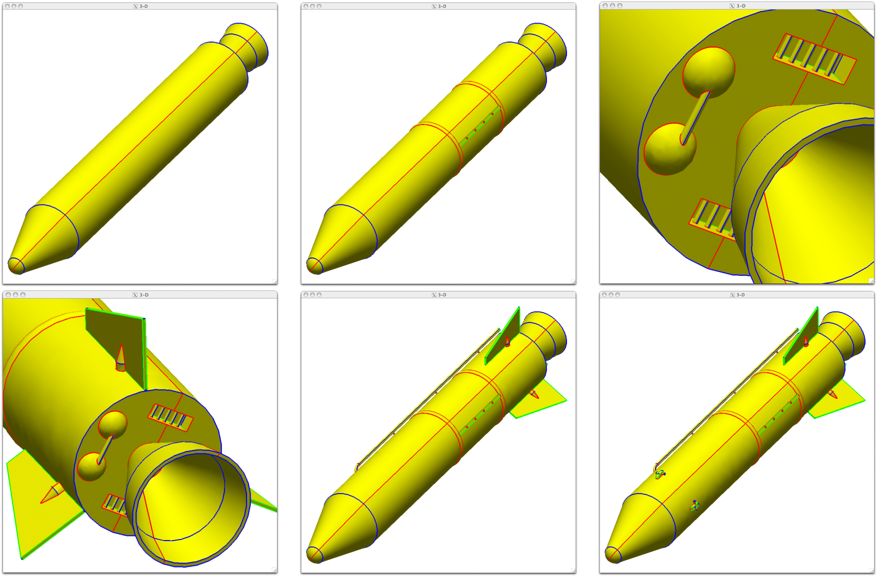
Number of basic grids	76	
Number of collar grids	152	
Number of SURFACE points	399,156	4%
Number of FIELD points	7,324,764	79%
Number of FRINGE points	859,086	9%
Number of HOLE points	710,934	7%

Timings for Generation of JMR3 Grids

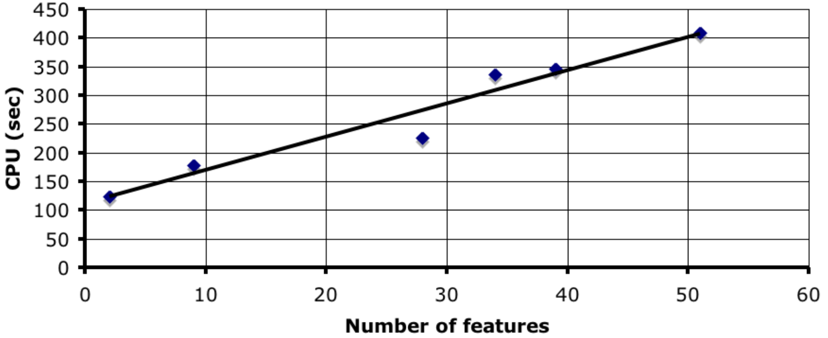
Phase	CPU sec	%
Read feature tree	0.03	0
Build BRep and tessellate	15.03	4
Generate basic grids	9.98	2
Build collar grids	77.84	19
Build global grid	0.43	0
Cut holes	92.51	23
Trim grids	0.04	0
Generate interpolation stencils	211.69	52
Total	407.55	100

CPU times on MacBook Pro 2.6 GHz Intel Core 2 Duo computer

Scalability Study: Configurations

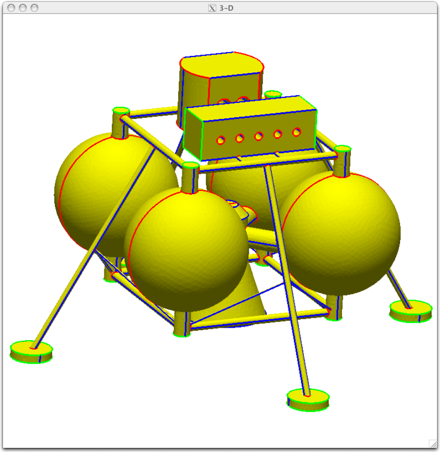


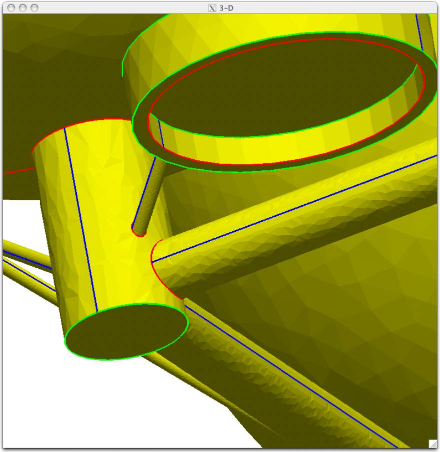
Scalability Study: Results



Sample Configuration — Lander

53 features, 56 primitive solids, 55 Boolean operators

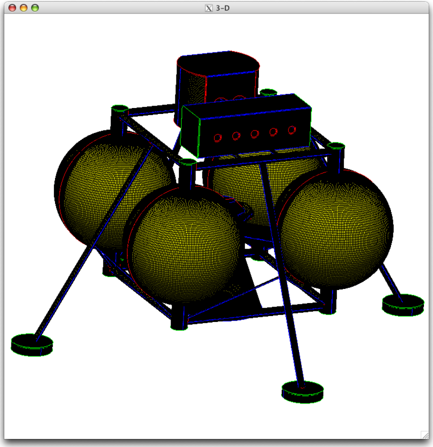


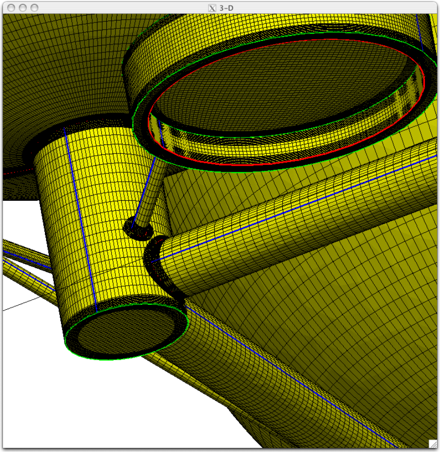


158 faces, 426 edges, 278 nodes

Surface Grids — Lander

74 basic grids, 114 collar grids, 1 global grid





Summary of Lander Grid System

Number of basic grids	74	
Number of collar grids	114	
Number of SURFACE points	1,341K	5%
Number of FIELD points	23,588K	80%
Number of FRINGE points	2.610K	9%
Number of HOLE points	1,836K	6%
Number of ORPHAN points	0K	0%

Timings for Generation of Lander Grids

Phase	CPU sec	%
Read feature tree	0	0
Build BRep and tessellate	20	0
Generate basic grids	2495	42
Build collar grids	493	8
Build global grid	0	0
Cut holes	193	3
Trim grids	0	0
Generate interpolation stencils	2492	42
Write grid and Xray files	253	4
Total	5946	100

CPU times on MacBook Pro 2.6 GHz Intel Core 2 Duo computer

Conclusions

- Overset grids can be generated *automatically* from a solid model
  - component grids generated for each feature tree primitive
  - collar grids generated for each primitive and each Boolean spine
  - holes cut using homogeneous transformations of solids
  - interpolation stencils created through bounding box, cell volume, and regional octree tests
- For a configuration with more than 50 features, more than 125 overset grids are generated, holes cut, and donors found in less than 7 minutes
- CPU time is found to be nearly linear with the complexity of the configuration — and NO user interaction is needed
- Effect of 100s of grids on solver accuracy and efficiency still needs to be determined