OGS Training
October 1, 2018
Erick Gantt
Nick Wyman
Syllabus

• **Meshing and Remediation**
  - Source curves
  - Surface meshing
  - Build Blocks
  - Unstructured block splitting
  - Overset boundary adaption

• **Hierarchical Frameworks**
  - Overview
  - Anchor points
  - Transforms
  - Instancing
  - Naming hierarchy
  - Overset panel

• **Examine enhancements**
  - Object probe
  - Filter
  - Snapshot
  - Persistent cuts

• **Suggar++ integration**

• **Further Overset Learning Resources**
Meshing and Remediation – Source Curves

• New source curves have been added to aid in local grid point (and thus cell edge) alignment on the interior of a domain when needed.

• Create source curves with the standard curve drawing tools: Two Point Curves and Draw Curves.

• Once created, select the source curve and use Grid, Sources to change the Type of source to Align Only.

• Particularly suited to database constrained domains which have a sharp interior feature which do not span the domain, or other features which you wish to align grid points to.
Meshing and Remediation – Source Curves

• Creating a source curve involves creating the curve first, then changing the Entity Type to Source.

• Additionally, creating curves which are database constrained is significantly improved. Curves automatically hop quilts without necessitating use of Segments Add. This is now the default method – Force Drawing in Parametric Space is off by default.
Meshing and Remediation – Build Blocks

• A new Create, Build Blocks command is available to quickly create block topology relative to existing topology.

• Selected domains and connectors become influence entities which drive default spacing values for the new block.

• Excellent for creating farfield or refinement blocks.

• Excellent for overset donor match remediation as refinement and background blocks can be quickly created with appropriate cell sizing.

• Build Blocks command panel options vary based on entity selection and chosen type.

• Only the Farfield type uses the selected topology in the faces of the new block to be created.
Meshing and Remediation – Build Blocks

- A table at top shows the bounding coordinate lengths of the selected entities and spacing values from those entities.
- The Shape frame provides pull-down lists to set the object Symmetry Planes. By default symmetry plane determination is Automatic.
- Margins set block boundaries relative to the centroid of the selected entities.
Meshing and Remediation – Build Blocks

- In the *Spacing Controls* frame choose spacing settings for the *Boundary Spacing*, *Decay* and *Background Spacing*.

- In the *T-Rex* frame you can pre-select settings to be used for creation of T-Rex anisotropic near-body layers. These settings will be used as soon as the new pending block is saved and then initialized via the *Grid* toolbar or *Grid, Solve* command.

- You can preview the pending block boundaries as connectors and domains with *Update Grid Entities*.
WT – Build Blocks Walk Through (p15.9)

• Open ESAV-1.pw
• Select all domains.
• Create menu, Build Blocks, Voxel

Voxel meshes are rectangular-shaped hierarchical Cartesian meshes created about the selected set of entities.

• In the Margins frame, enter 3000 10000 3000 in the Uniform text field.
This is equivalent to specifying the following individual margins: 3000 for -X and +X, 10000 for +Y, and 3000 for -Z and +Z.

• Uncheck Use Default for Transition Layers and enter 3.
The Transition Layers option specifies a target for how many layers of hex cells are used at each level before transitioning to the next size. The range is [1, 100] with a default value of 1.

• OK
A new voxel block has been created; you may need to zoom out to see it. At this point, the block is empty and it needs to be initialized.

• Make sure the new block is selected.
• Click Initialize on the Grid toolbar.
Meshing and Remediation – Unstructured Block Splitting

• It is now possible to split unstructured blocks using a proximity operation.

• This splitting style facilitates separating a T-Rex generated boundary from remaining volume grid allowing the use of a T-Rex boundary as part of a multi-component overset grid.
Meshing and Remediation – Unstructured Block Splitting
Meshing and Remediation – Overset Boundary Adaption – Unstructured

• Under the Boundary Conditions tab for the unstructured Solve command there is a new column, Adapt, for each BC allowing adaption of boundaries to any sources.

• In order for this feature to take effect there must be sources available and enabled for the block being solved, and the Adapt selection on the Boundary Conditions tab of the Solve command must be set to On.

• By default all domains in the Unspecified group of BCs will not be adapted. If this is undesirable for certain domains, a new BC must be create which has Adapt set to On.

• T-Rex wall boundaries can be adapted, but only at the surface. Once layers/points begin marching they follow the standard T-Rex algorithm irrespective of any sources or adapt settings.
Meshing and Remediation – Overset Boundary Adaption – Unstructured

- Double click the field per BC to set Adap to On or Off.
- Enable or disable individual sources on the Size Field tab.
Meshing and Remediation – Overset Boundary Adaption – Structured

• Under the Face Attributes tab for the structured Solve command, there is a new option in the pull-down list for Boundary Conditions Type: Adapt.

• As with the unstructured Solve command, individual sources can be enabled or disabled on the Size Field tab.
WT – Boundary Adaption

- Open GBU12-1.pw
- Select blk-1.
- Grid menu, Solve
- Boundary Conditions tab

Note Adapt is already On by default for the BuildBlocksOff BC.

- Solve tab
- Refine

Note additional clustering in vicinity of the source at the exit boundary.
Syllabus

• Meshing and Remediation
  - Source curves
  - Surface meshing
  - Build Blocks
  - Unstructured block splitting
  - Overset boundary adaption

• Hierarchical Frameworks
  - Overview
  - Anchor points
  - Transforms
  - Instancing
  - Naming hierarchy
  - Overset panel

• Examine enhancements
  - Object probe
  - Filter
  - Snapshot
  - Persistent cuts

• Suggar++ integration

• Further Overset Learning Resources
Hierarchical Frameworks – Overview

• A framework provides a sandbox in which to create and manipulate topology and geometry independently of other frameworks. Therefore no topology duplication checking will take place between frameworks. This check will only be done within a single framework.

• A framework containing topology and/or geometry can be instanced as new frameworks, including a transformation, such that a piece of geometry which is repeated need only be created and meshed once. Frameworks can provide multiple instances, but the entities really only exist once as the original which was instanced. This reduces overhead from having to maintain duplicate entities where repetition is involved.

• Each framework can have entities which are defined in a separate local coordinate system. Those individual frameworks and coordinate systems can be related to a global root framework and coordinate system using attachment points and transformations which are determined automatically.

• These features facilitate meshing of different components in local coordinate systems and combining them easily into a larger complex system under a global coordinate system.
Hierarchical Frameworks – Overview

• A *Legend* at the top shows which frameworks contain Geometry, which have been *Transformed*, which have been *Multiply Instanced* and which one is the *Active* framework.

• The active framework will be rendered normally in the *Display* window. All others and their contents will be shown in transparency.

• Only entities which belong to the active framework can be selected and operated on. There are a few exceptions to this. For instance, when selecting size influencers for a voxel block, it is possible to select across frameworks.

• Below the *Legend* is the frameworks tree showing the entire hierarchy of frameworks.

• **Important**: When the *Frameworks* panel is open, only frameworks can be selected in the tree or *Display* window.
Hierarchical Frameworks – Overview

• Actions commands
  - *Set Active* – makes the currently selected framework the active framework.
  - *New* – creates a new framework immediately below the currently active framework.
  - *Insert* – inserts a new framework immediately above the currently selected framework.
  - *Instantiate Active* – makes a new instance of the contents of the currently active framework inside the currently selected framework.
  - *Isolate* – if the currently selected framework contains an instance, *Isolate* will be available and will force the instanced entities to become unique.
  - *Reparent to Active* – takes the currently selected framework and makes it a child of the active framework.
Hierarchical Frameworks – Overview

• Actions commands continued
  - Flatten to Parent – pushes content of all selected frameworks up to their parent, potentially removing transformations and instances as duplicates.
  - Save As... – provides a file dialog to export the selected frameworks to a new .pw file.
  - Delete – deletes the currently selected frameworks and their contents.

• Tasks
  - Transform – Open the Transform Framework panel to perform a transformation of the selected framework.
  - Attachment Points – Open the Attachment Points panel to create new or edit existing attachment points.
Hierarchical Frameworks – Overview

• Visibility
  - *Show* – makes visible again previously hidden frameworks.
  - *Hide* – makes invisible the currently selected frameworks. Note that only non-active frameworks can be hidden and are normally rendered as transparent.
  - *Transparency* – adjusts the transparency level of the currently selected frameworks. Note that changing the transparency of a parent framework does not change the transparency of its children.
Hierarchical Frameworks – Overview

• A new framework can also be created by selecting a set of entities in the List or Display window and going to Create, Framework. This action will produce a new framework containing the selected entities and which is a child of the currently active framework.
Hierarchical Frameworks – Attachment Points

• The Attachment Points command provides tools to define these anchors used to perform transformations of one framework into the coordinate system of a parent framework.

• Simply use New to populate the list with a new attachment point. Define the Location, Normal and Forward orientation of the attachment point.

• Conversely an existing attachment point may be selected in the list box and edited or removed via Delete.
Hierarchical Frameworks – Transforms

• It is recommended practice to create a child framework which simply represents a desired transformation. Afterward an additional child framework of the transformed framework can contain the desired entities or instances of entities.

• Also recommended for complex transformations between different local coordinate systems is that attachment points are pre-defined to facilitate the transformation. Attachment points provide a simple click-click operation to select the two attachments to initiate the transform.

• To use, select the desired framework to transform and use Transform in the Tasks frame of the Frameworks panel.

• Note that there is no binding between attachment points after a transformation is completed.
Hierarchical Frameworks – Transforms

• To perform a transform using attachment points simply select the child attachment followed by the parent attachment.

• The child and parent frameworks will be aligned via their attachment points.

• Animate Transformations is enabled by default so the transformation will be animated.
Hierarchical Frameworks – Transforms

• A framework can also be simply transformed without anchor points through a translation or rotation.

• These are standard translate and rotate operations as found in the Edit, Transform command.
• *Open ESAV-2.pw*
• *View menu, Panels, Frameworks* (if not open already)
• Select the *Frameworks* tab.
• Select the ESAV (global) framework.
• Click *Set Active* in the Actions frame.

This makes the ESAV framework active. Note that the active framework will be fixed during a transformation.

• Select the GBU-12_Transform framework.
• Click *Transform* in the *Tasks* frame.
• Expand the ESAV node in the *Parent Framework* frame and select the ESAV_Attachment attachment point.
• Expand the GBU-12_Transform node in the *Child Framework* frame.
• Expand the GBU-12 node and select the GBU-12_Attachment attachment point.

Note the animation showing the store being placed in the ESAV pylon. Also note how the attachment points snap together: points and forward vectors (yellow) overlap while normal vectors (orange) oppose.

• *OK*
Hierarchical Frameworks – Instancing

Initial

Final
Hierarchical Frameworks – Instancing
WT – Instancing

• Open ESAV-3.pw
• Select the Frameworks tab.
• Select the ESAV framework.
• Click New.
• Double click the framework-1 name and enter GBU-12-2_Transform.
• Click New.
• Double click the framework-1 name and enter GBU-12-2.
• Select the GBU-12 (original) framework.
• Click Set Active.
• Select the GBU-12-2 framework.
• Click Instantiate Active.

Note that the new instance of the store is located at its original position and orientation. This is because a transform has not been applied yet to the intermediate framework. Instances of geometry and grid are not copies. Therefore they do not incur additional resource overhead in memory or on disk.
Hierarchical Frameworks – Naming

• Care should be taken when naming frameworks such that they are unique, consistent, concise, and yet brief. Brevity is desirable as the hierarchy in the framework tree of a component will transfer automatically into the Overset command.

• The tree layers become a path in the statistics table in the Overset panel, including the block name(s) in each framework.

• The Node Tree under attributes does not update automatically.
Syllabus

• Meshing and Remediation
  - Source curves
  - Surface meshing
  - Build Blocks
  - Unstructured block splitting
  - Overset boundary adaption

• Hierarchical Frameworks
  - Overview
  - Anchor points
  - Transforms
  - Instancing
  - Naming hierarchy
  - Overset panel

• Examine enhancements
  - Object probe
  - Filter
  - Snapshot
  - Persistent cuts

• Suggar++ integration

• Further Overset Learning Resources
Examine Tools – Object Probe

• Use RMB click to select any overset object to produce the *Show Overset Data* command. Data must be up to date.
• Click *Show Overset Data* to produce a detailed information table in the *Messages* window.
Examine Tools – **Object Probe**

- Example fringe point results shown
- Details provided for the fringe point, donor and donor candidate
- Details include block names, point indices relative to the owning block, vertex data, cell volume, average edge length, etc.
WT – Object Probe

- Open ESAV-4.pw
- Select block *near-body*.
- Examine menu, *Volume*

Note that the block is immediately rendered with yellow points representing overset fringe points.
- Right click any yellow fringe point to see the *Show Overset Data* option.
- Click *Show Overset Data*.

You can now view the various overset and point data printed to the *Messages* window.
- Close
Examine Tools – Filters

• Use filters to control the display of rendered grid cells or overset objects.

• Similar to the Color Bar Max. and Min. controls with much greater granularity.

• Filters provide the means to pinpoint exactly the cells/objects you wish via multiple decisions and conditions.

• Filters act side-by-side with other static rendering tools, such as the color bar. In other words, cells matching both color bar criteria and filter criteria will all be rendered simultaneously.

• Cells qualifying for multiple filters and/or static controls are only rendered once.
Examine Tools – Filters

[Diagram showing a software interface for examining tools with a focus on filters.]
Examine Tools – Filters

- The table at top lists all filters created.
- Select a filter to view/edit settings.
- Use the Name text frame to change the name of the selected filter.
- Use the Type pull-down to choose either a Grid or Overset filter.
- New populates the list with a new filter.
- Delete is available to remove the currently selected filter.
- A newly created filter includes immediately a base decision in the tree below...
Examine Tools – Filters

• Choose from one of three decisions:
  - *All* – Boolean AND
  - *Any* – Boolean OR
  - *None* – Boolean NOT
Examine Tools – Filters

• For the base decision, add a condition(s) by clicking “…”
  - Add Condition
  - Add Decision
  - Toggle Enabled
  - Delete Decision (unavailable for root decision)
  - Enable All Children
  - Remove All Children

• Conditions are made up of:
  - An attribute
  - A comparison
  - An input value

• Overset attributes include: Overset Element Type, Donor Entity Name, Donor Candidate Entity Name, Donor Volume Ratio and Donor Edge Length Ratio.
Examine Tools – Filters

• Finally, visibility of filters is controlled by the Filters frame on the primary Examine tab.
• Toggle filters on and off via the checkbox.
• Filters are off by default upon entering the Examine command.
Examine Tools – Create Point Cloud

- The *Create Point Cloud* command frame provides tools for selection of *Examine* objects to be saved for later use as a point cloud source.
- This frame enters a selection mode disabling the remainder of the *Examine* panel.
Examine Tools – **Create Point Cloud**

• In the *Spacing Type* frame choose how you wish the new source spacings to be determined:
  - *Constant* – creates a constant spacing and decay source. Use options in the *Constant Values* frame to specify *Spacing* and *Decay*. Or leave as the default settings.
  - *General* – creates a source point cloud with varying target spacing. Use the *General Values* frame to specify use of an *Average Edge Length* taken from the local cell and applied to the current source point. Otherwise use *Current Function Value* to directly use length based function diagnostic values.

• Use *Create* to save the new source.
Examine Tools – Persistent Cuts

- Persistent cuts (PCs) provide a diagnostic tool for persistent display of a desired *Examine* diagnostic.

- Persistent cuts update automatically following any updates to affected blocks.

- Multiple PCs can be created and can overlap as desired.

- Only one diagnostic value can be displayed on all PCs at any given time.

- It is possible to create PCs which always auto-locate at either of the diagnostic extrema locations.

- PCs can be defined on the *Cuts* tab of the *Examine* command or via the *Create, Cut Planes* command.

- A new *Cuts* panel is available to control PC visibility, change PC display attributes and delete PCs, among other things.
Examine Tools – Persistent Cuts

• Use *Make Persistent Cut* in the *Cutting* frame on the *Cuts* tab of the *Examine* command to save a cut as persistent so that it can be viewed outside of the examine command.

• Or, use the *Create, Cut Planes* command to create PCs directly...
Examine Tools – Persistent Cuts

• On the *Cut Planes* panel:
  - Use standard plane creation tools to define the cut location.
  - If an extrema auto-locate PC is desired, choose *Minimum* or *Maximum* in the *Extrema* frame.
  - Choose display attributes from standard tools in the *Display* frame.

• Specific to overset grids:
  - Control visibility of fringe and fringe donors in the *Overset Show Fringe/Donors* frame.
  - Additionally, toggle on *Hole Grid*, *Hole Symbol*, *Orphan* and *Orphan Donor Candidates* in the *Overset Show* frame.
Examine Tools – Persistent Cuts

• Change the diagnostic for all cuts and toggle the color bar via the Global frame at top.

• Display and Overset attribute settings are identical to the Create, Cut Planes panel.

• Select only a cut to use Edit Cut Definition (looks just like Create, Cut Planes) or to Delete Cut Definition.

• Select only a block to Remove Block Cut, effectively take the selected block out of the cut’s scope.

• Other than the two controls in the Global frame, all attributes can be made on a per cut or even per block (within a cut) basis.
Examine Tools – Persistent Cuts

• Using *Edit Cut Definition* we can see the original settings used to create the selected cut.

• Note the *Edit Cut Definition* option also provides the additional *Blocks* frame to revise the block selection.
WT – Persistent Cut Creation (p15.21)

- Open ESAV-5.pw
- Select the List tab.
- Select the off-body voxel block.
- Examine menu, Volume
- In the Overset Options, Show Fringe/Donors frame, select Off.
- Switch to the Cuts tab.
- Toggle on Enable Cutting
- Select Y for the cut direction.
- Toggle on Select Point.
- Pick the lower forward corner of the GBU12 block.
- For Style select Crinkle.
- Click Make Persistent Cut in the Cutting frame.

Note that the Message windows shows that the cut plane cut-1 was created.
- Close
WT – Persistent Cut Creation (continued)

- View menu, Panels, Persistent Cuts
- In the Global frame use the Function pull-down to select a diagnostic, for instance, Donor Edge Length Ratio.
- Toggle on Show Color Bar.

Note that any changes to the off-body block will now trigger an update to the persistent cut automatically. This makes it easy to confirm grid improvements after performing volume refinements and additional overset assembly.
Syllabus

• Meshing and Remediation
  - Source curves
  - Surface meshing
  - Build Blocks
  - Unstructured block splitting
  - Overset boundary adaption

• Hierarchical Frameworks
  - Overview
  - Anchor points
  - Transforms
  - Instancing
  - Naming hierarchy
  - Overset panel

• Examine enhancements
  - Object probe
  - Filter
  - Snapshot
  - Persistent cuts

• Suggar++ integration

• Further Overset Learning Resources
Suggar++ Integration

• Support has been added for flex (Suggar++ native) format.
• Celeritas has created a plugin API capability to support custom grid formats allowing direct mesh export from Suggar++ to the flow solver.
• Plot3D export to the assembler carries CAE BC naming through for proper transfer of BC information downstream of assembly.
• Pointwise now supports multiple versions:
  - Suggar++ 2.1
  - Suggar++ 2.6.0
  - Suggar++ 2.9.1
Further Overset Learning Resources

• To further your knowledge of the overset tools available in Pointwise, please work the full tutorials found by accessing the Help menu and selecting Tutorial Workbook.

• Have a look at these specific tutorials:
  - Chapter 13, “Wing-Body: Overset Meshing”
  - Chapter 14, “Wing-Body: Overset Assembly”
  - Chapter 15, “ESAV Generic Fighter” (examples including page number)