Walk-in Slide: AU 2014 Social Media Feed

1. Click on the link below, this will open your web browser

http://aucache.autodesk.com/social/visualization.html

2. Use “Extended Display” to project the website on screen if you plan to work on your computer. Use “Duplicate” to display same image on screen and computer.

Complex Modelling and Analysis Using Robot Structural Analysis and the API

Rob May

Associate Director, Buro Happold Beijing

@rwemay

Class summary

This class will show you how to use the **Robot Structural Analysis** software's comprehensive **API** to simulate complex structural problems and thus improve efficiency and provide feedback loops. We will review several examples where the API has been used to link Robot Structural Analysis software to other software in order to speed up model generation and help **create complicated geometry models with a high level of accuracy**. We will show you tips, tricks, and effective practices for generating models that run efficiently, and you will discover how to use the API to extract results in order to provide **optimisation feedback loops**. We will also show you how to use such functions as Result Query and Cache in Microsoft Visual Basic to access data within Robot Structural Analysis software. We will use at least 1 example of this type of workflow to demonstrate how quickly you can create scripts to link software such as Grasshopper to Robot Structural Analysis software, and then continue through to **Revit** software to produce project documentation and images.

Key learning objectives

At the end of this class, you will be able to:

- Discover the benefits and efficiencies that can result from using customized access to control Robot Structural Analysis software using the API
- Understand the concept of structural optimization using feedback loops between Robot Structural Analysis software and parametric scripts
- Gain some detailed knowledge of advanced API methods, such as using Result Query to improve data access times
- Gain beginning knowledge of how to connect Robot Structural Analysis to other software using a simple example in visual basic language

Presentation Contents

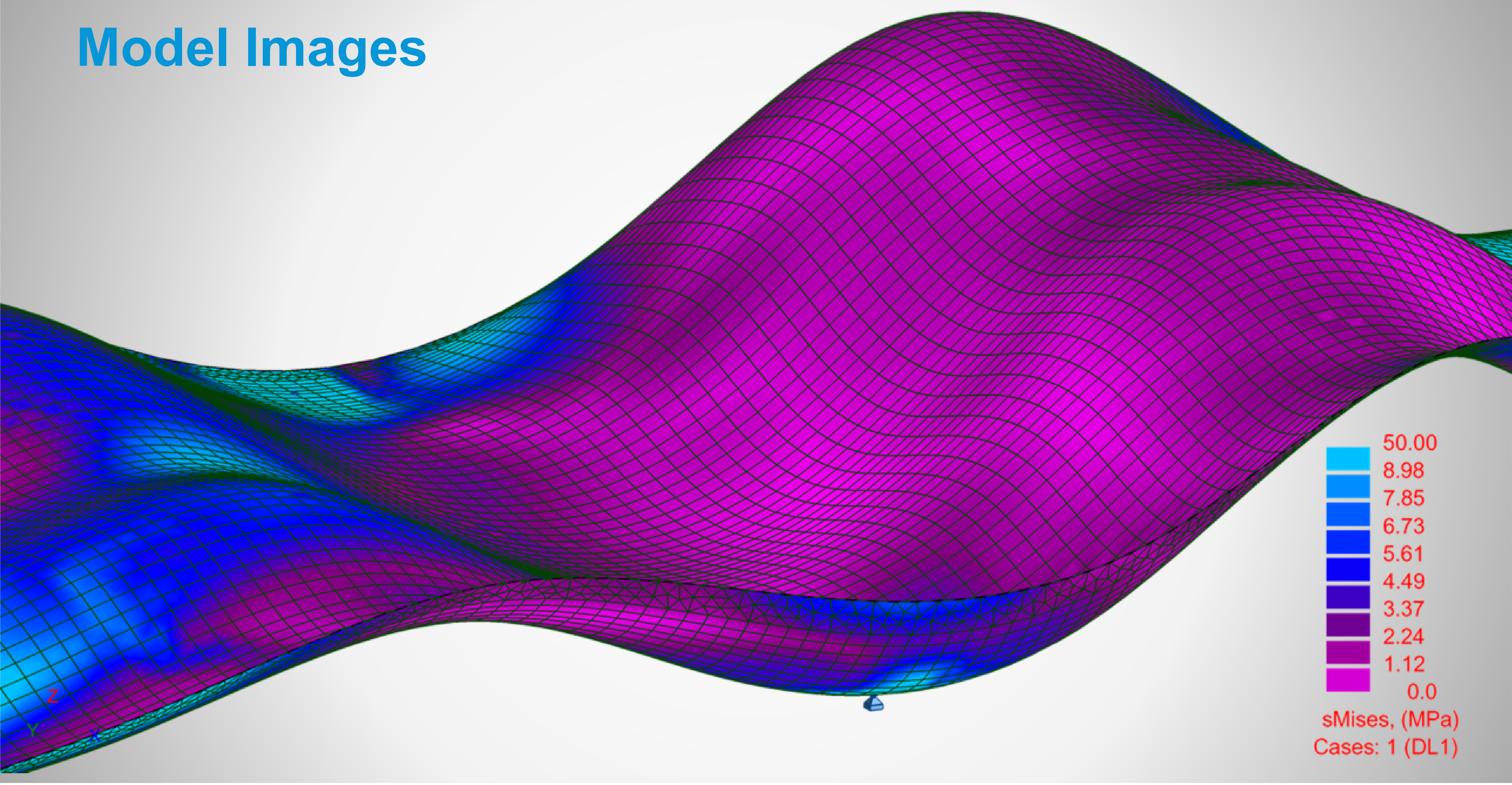
- Engineers as Programmers
- Controlling your Robot – API 101 – a brief introduction into connecting Grasshopper to Robot
- Large and Complex Models – Examples and Tips
- Example Models – Spatial and Cable Structures
- Example Models – Towers
- Example Models – Complex Steel Connections

Presentation Format

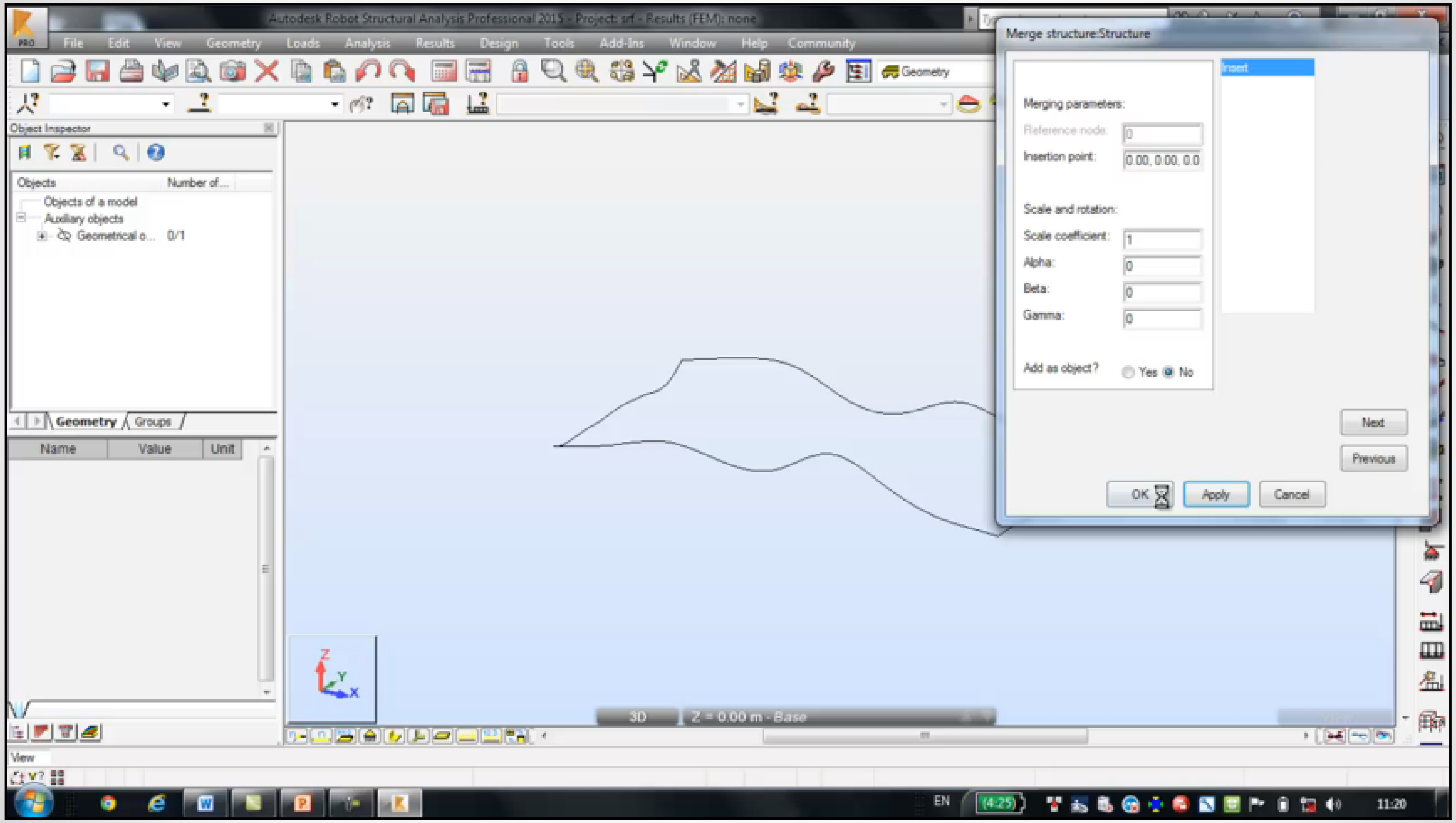
Code

```
Private Sub RunScript(ByVal Bar_Centrelines As DataTree(Of Curve), ByVal Gamma_Angle As DataTree(Of Object), ByVal Type_Name As DataTree(Of String), ByVal Sec
  If activate = True Then
    'Set Robot as a 'RobotApplication' - this means every time I type 'Robot.' from now on options appear
    Dim robot As New RobotApplication

    robot.Project.Structure.ResultsFreeze = False

    'Delete all the node and bar objects in the model (don't use clear as this clears loadcases etc.)
    If delete_bars = True Then
      Dim prev_bar_sel As robotselection
      Dim prev_nod_sel As robotselection
      prev_bar_sel = robot.Project.Structure.Selections.CreateFull(IRobotObjectType.I_OT_BAR)
      prev_nod_sel = robot.Project.Structure.Selections.CreateFull(IRobotObjectType.I_OT_NODE)
      robot.Project.Structure.Bars.DeleteMany(prev_bar_sel)
      robot.Project.Structure.Nodes.DeleteMany(prev_nod_sel)
    End If

    'Create a 'data tree' of bar numbers - this is basically a list of lists
    Dim bar_nums As New Grasshopper.DataTree(Of int32)

    'Create member types that don't exist in the model currently
    For i As int32 = 0 To Type_Name.BranchCount - 1
      For j As int32 = 0 To Type_Name.Branch(i).Count - 1
        Dim Member_Type As IRobotLabel
        If Not robot.Project.Structure.Labels.Exist(IRobotLabelType.I_LT_MEMBER_TYPE, Type_Name.Branch(i).item(j))Then
          Member_Type = robot.Project.Structure.Labels.Createlike(IRobotLabelType.I_LT_MEMBER_TYPE, Type_Name.Branch(i).item(j), "Beam")
          robot.Project.Structure.Labels.Store(Member_Type)
        End If
      Next j
    Next i

    'Create a 'cache' - this means 'store all the elements in quick/memory before asking Robot to create' (faster)
    Dim robot_cache As RobotStructureCache
    robot_cache = robot.Project.Structure.createcache

    'Tell the computer memory what type of data each object is
    Dim bar_start_point As point3d
    Dim bar_end_point As point3d
    Dim gamma As Double
```

Model Images



Process Animations



Engineers as Programmers

Engineers as Programmers

- **Logic** Both Engineering and software programming are heavily reliant on logic. It follows therefore, that once the notion that programming is difficult is overcome, Engineers can make fantastic programmers.
- **Rigor** Good Engineers are rigorous by nature. Apart from efficiency, one of the most difficult aspects of structural engineering on large and complex projects is maintaining a high level of rigor in the design and calculations. With a small amount of computer programming skill, any Engineer can add a layer of quality management by programming repetitive calculations and tasks, and enabling auto-check and feedback in a reliable, checkable programmatic way.
- **Creativity** Good Engineers create creative solutions to complex problems. Once Engineers are bound by software or processes that do not allow them to think or perform outside the box, tasks becomes repetitive and less thought is applied to each solution. Not only can programming be creative in itself – by allowing creative solutions to problems, by reducing repetitive manual tasks, it also frees up time for thinking, sketching and researching.
- **Efficiency** Good Engineers strive for efficiency – whether it's reducing material in a building, or refining aircraft parts. It should go without saying that by smart solutions using computer programming, efficiency of tasks such as calculations can be increased exponentially.

Engineers as Programmers

11:18:04 Start of the structure verification

Number of errors: 0

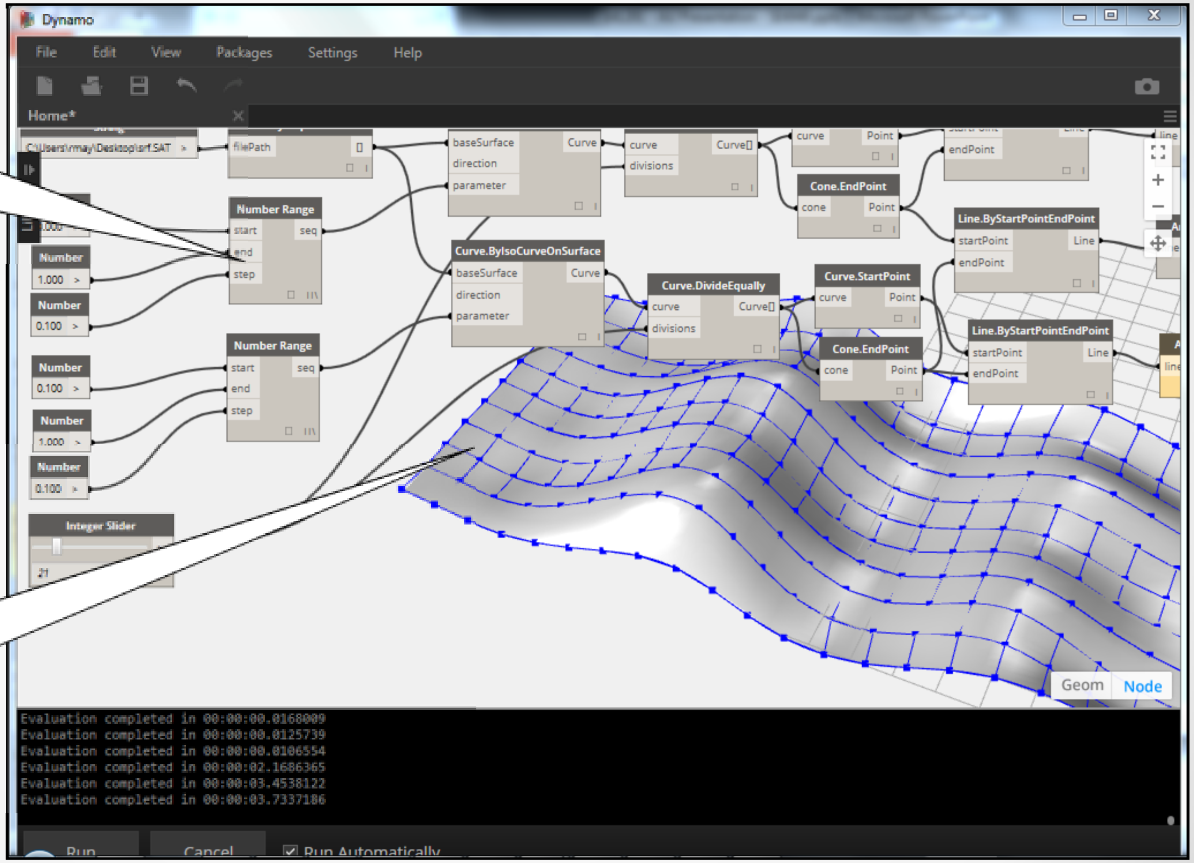
Number of warnings: 0

11:20:31 End of the structure verification

11:20:31 Start of the analysis

Visual Programming

Much easier to get started than text based programming languages

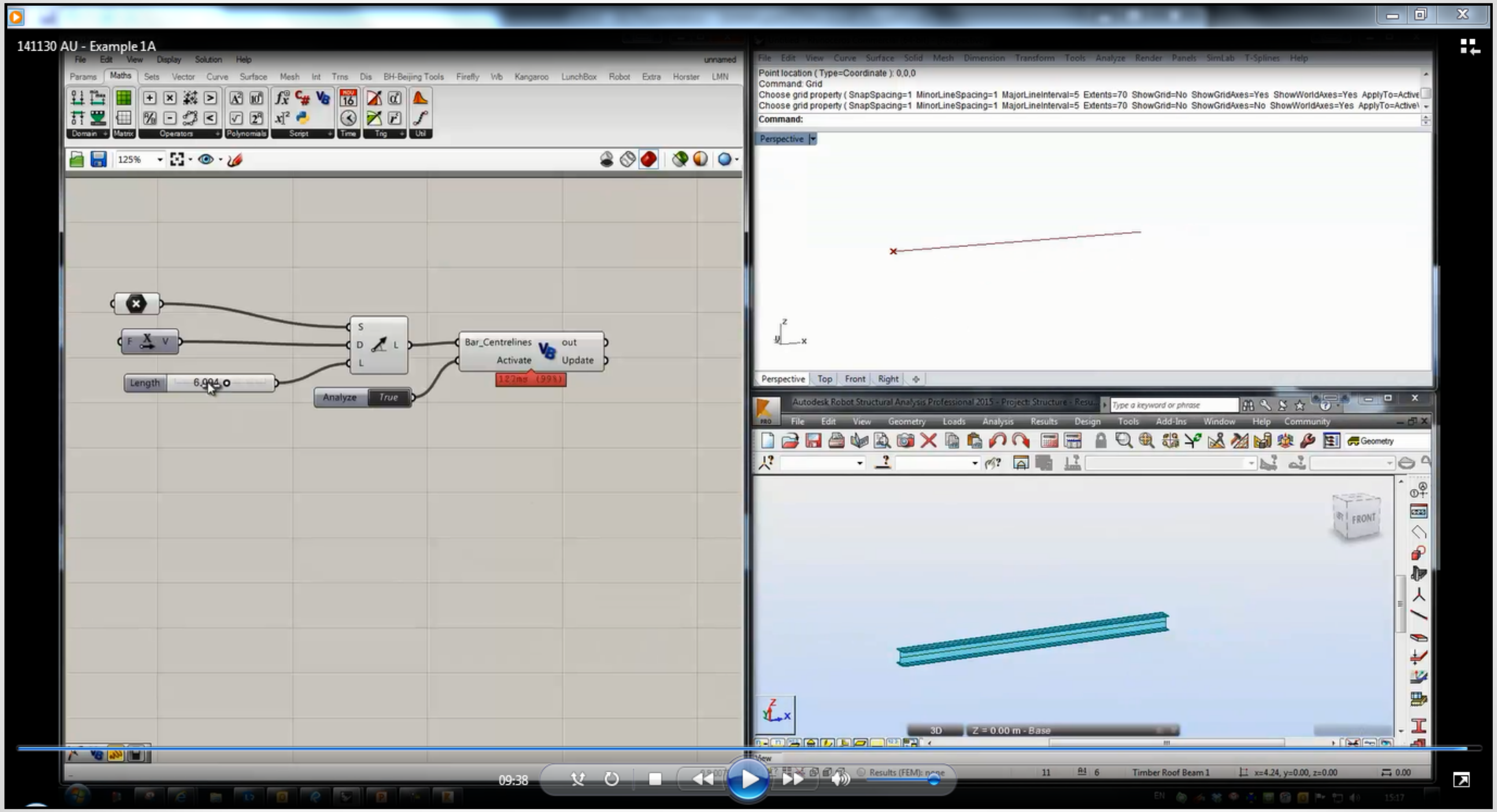


Dynamo and Grasshopper provide constant visual debugging

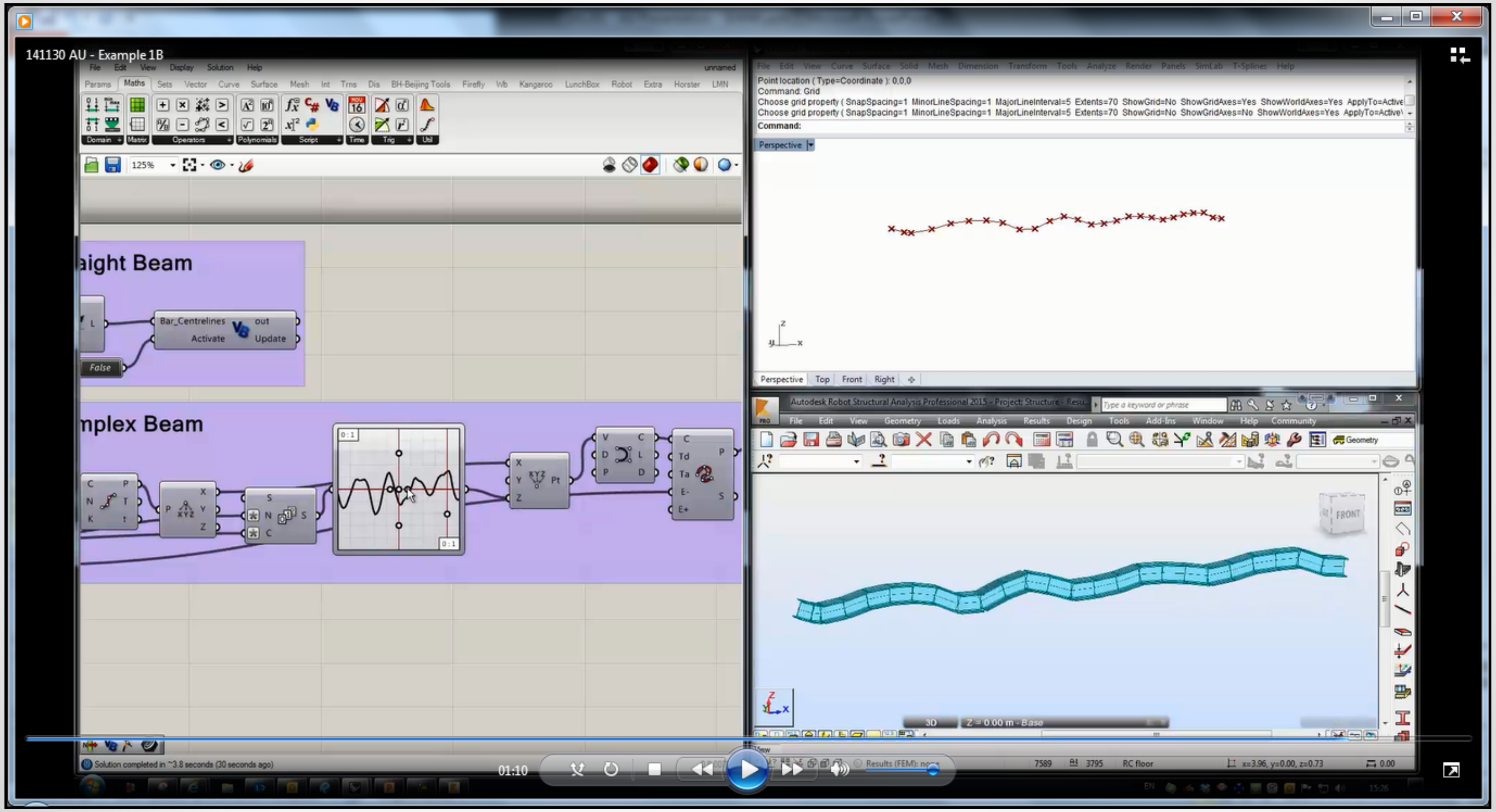
Controlling Your Robot API 101

- Example 1A – how to generate bars and nodes using the ‘cache’
- Example 1B/C – how to add supports and run/rerun analysis for quick feedback
- Example 2 – applying the same simple components to a larger scale grid shell structure
- The full worked examples are available for download/distribution

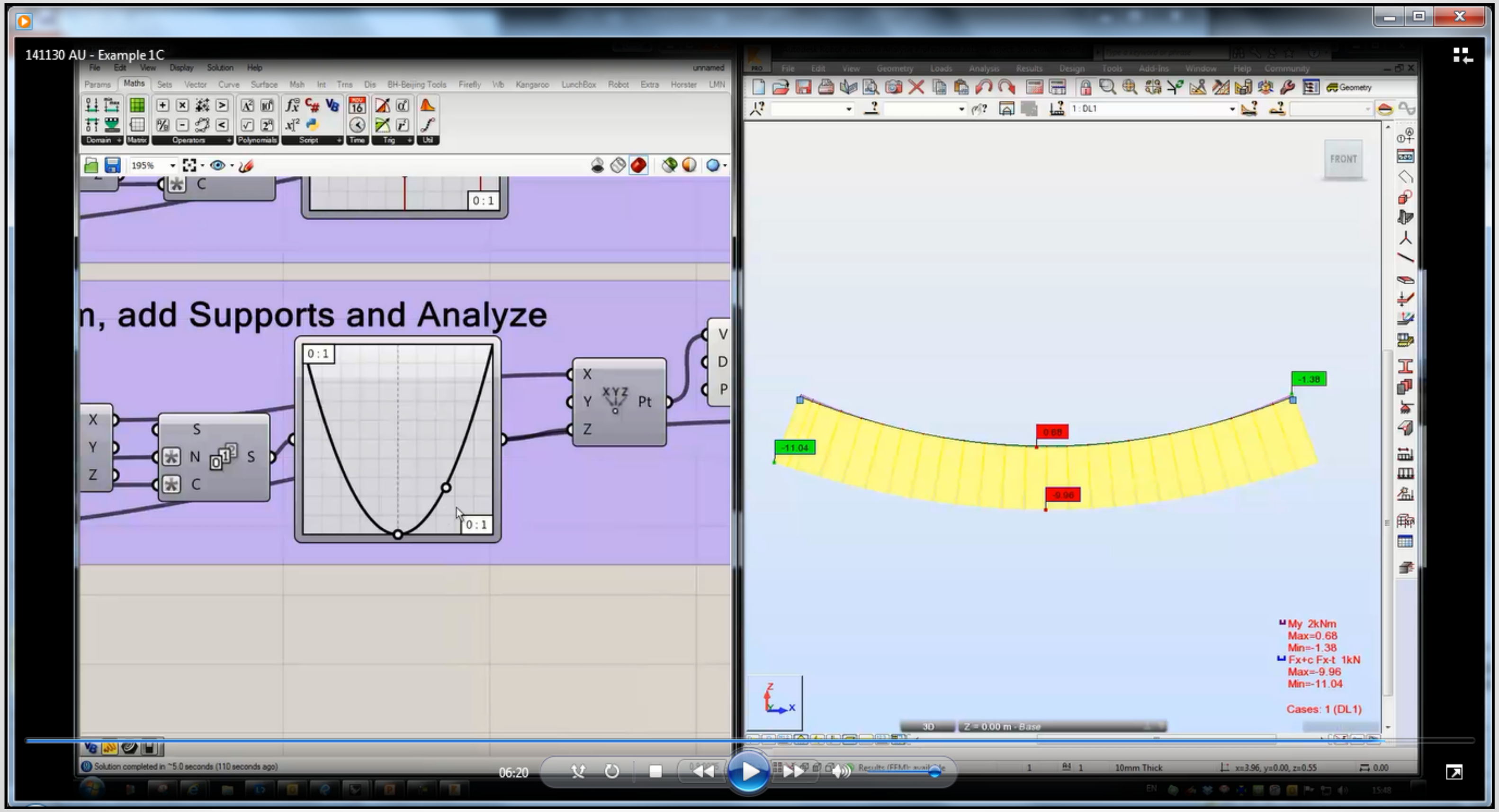
Example 1A – Generate a Simple Bar



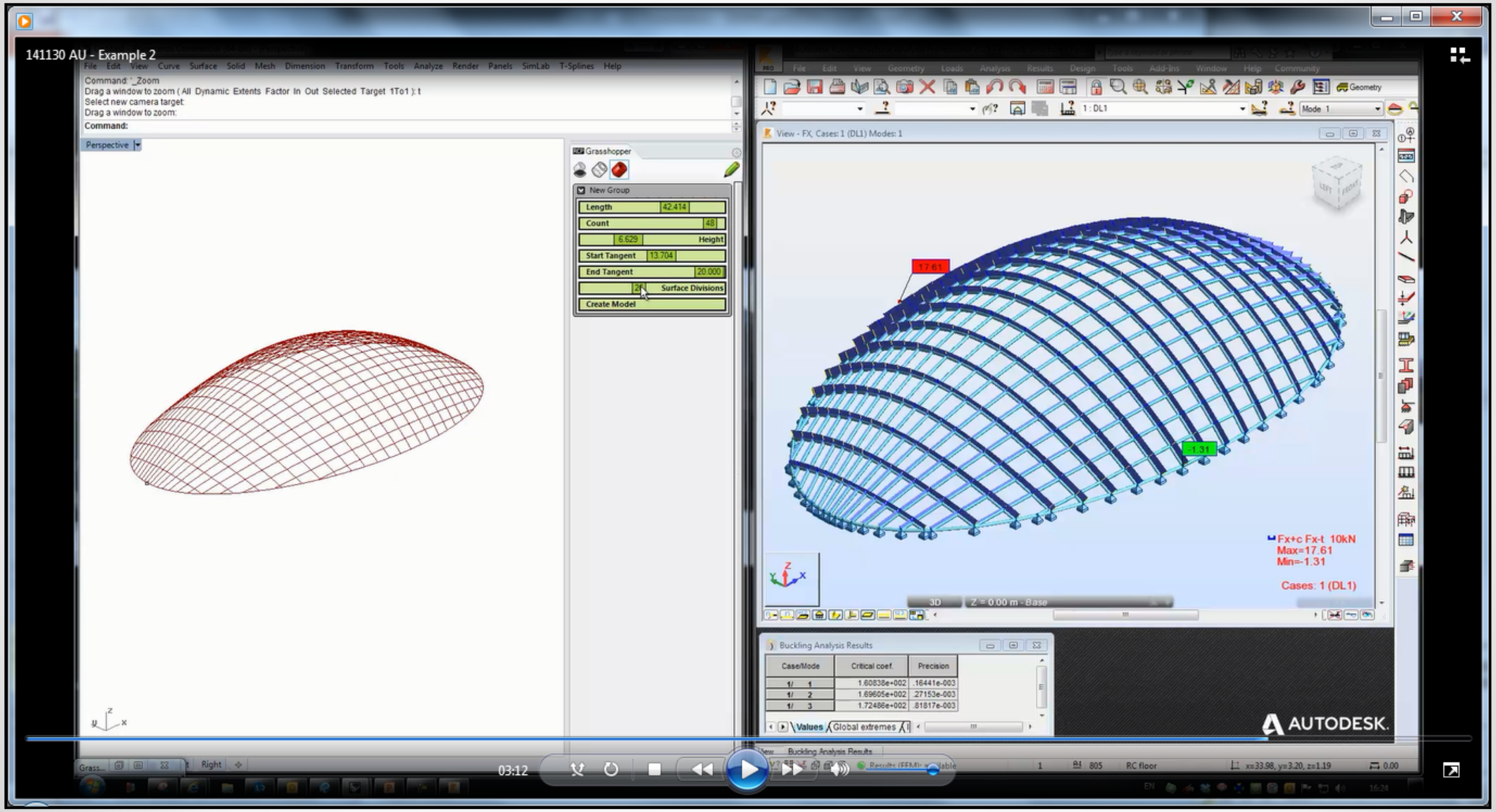
Example 1B – Generate a Chain of Bars



Example 1C – Add Supports and Run Analysis

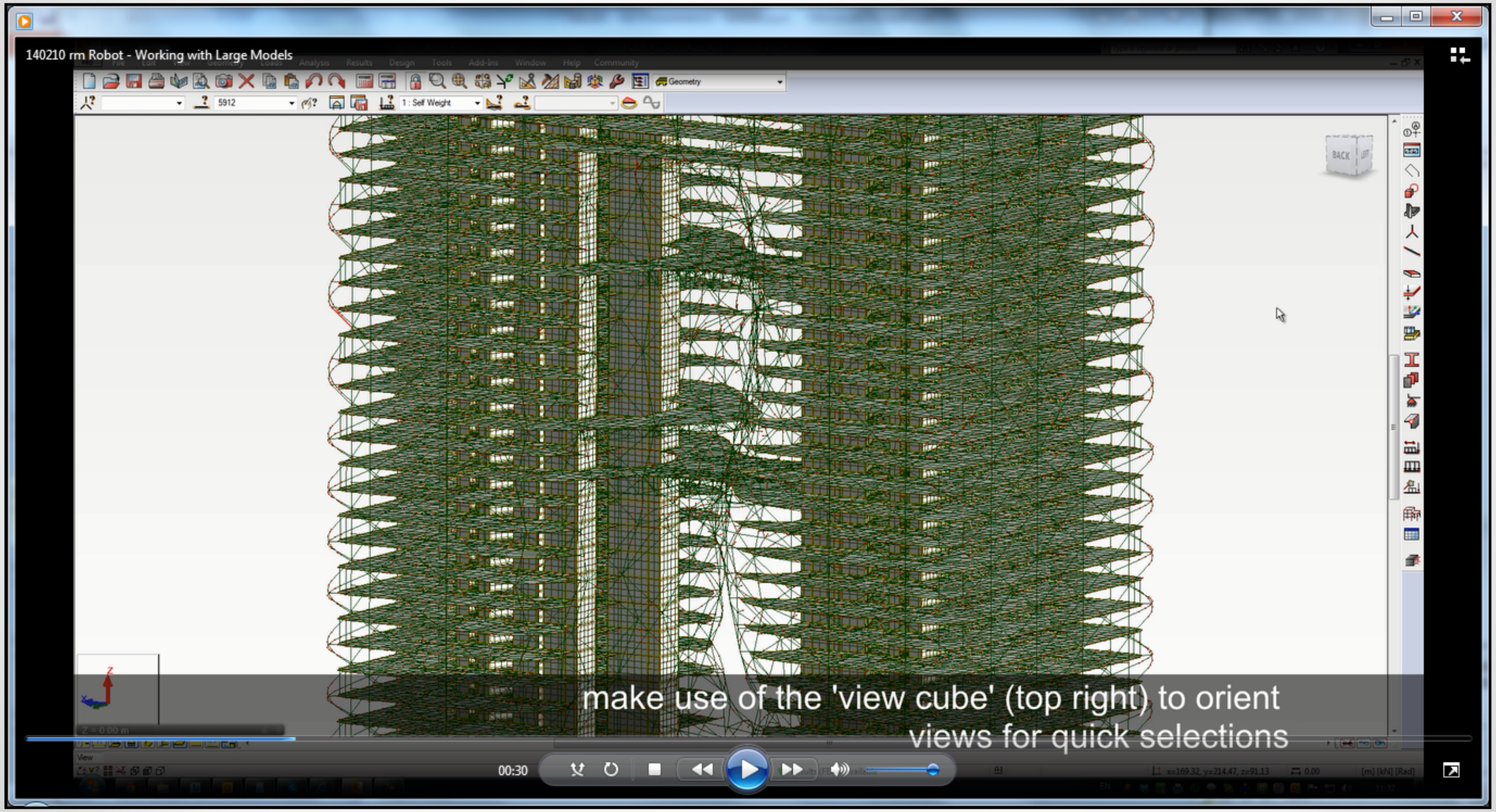


Example 2 – Grid shell Generation and Analysis

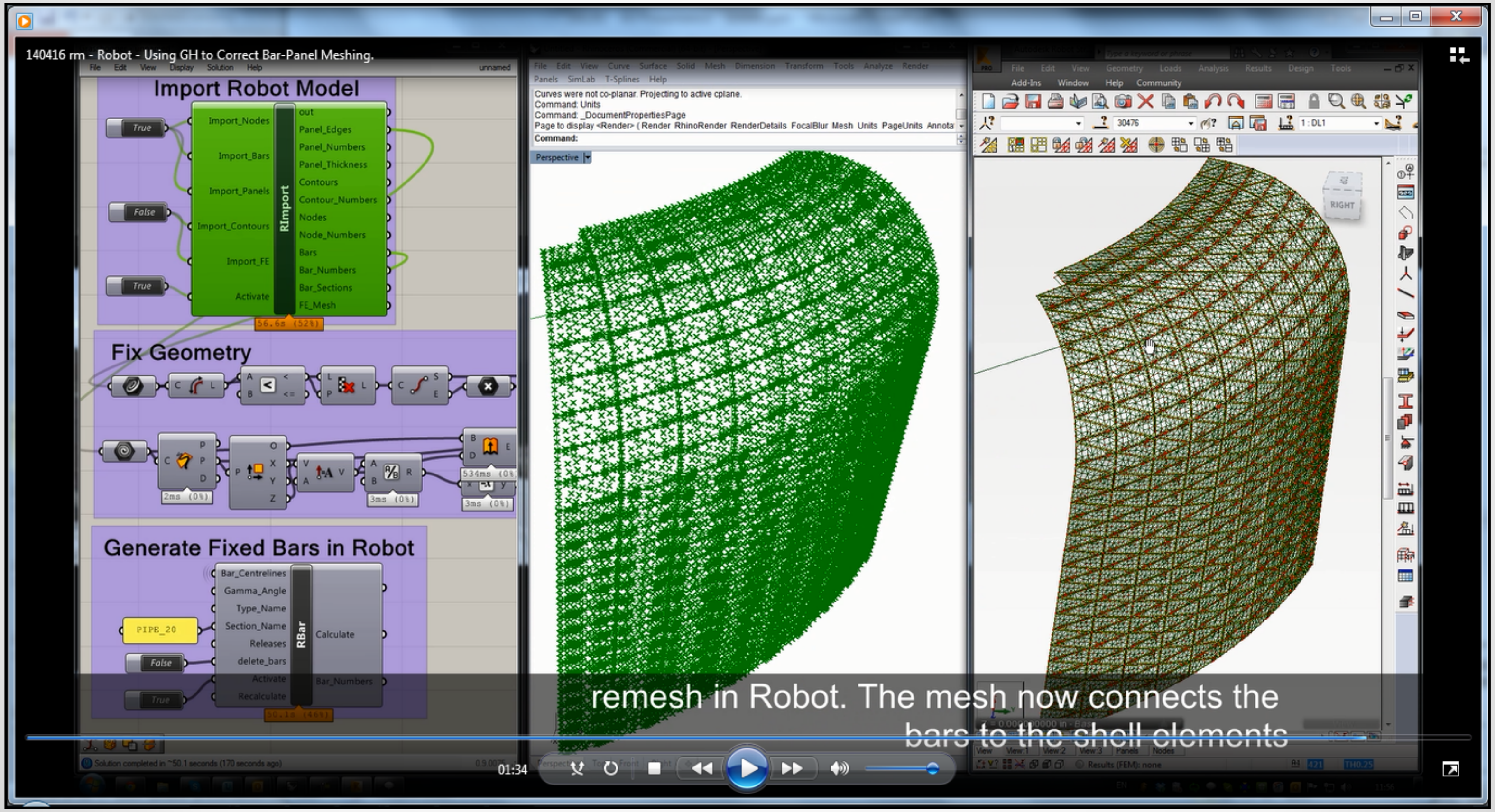


Handling Large & Complex Models – Some Tips

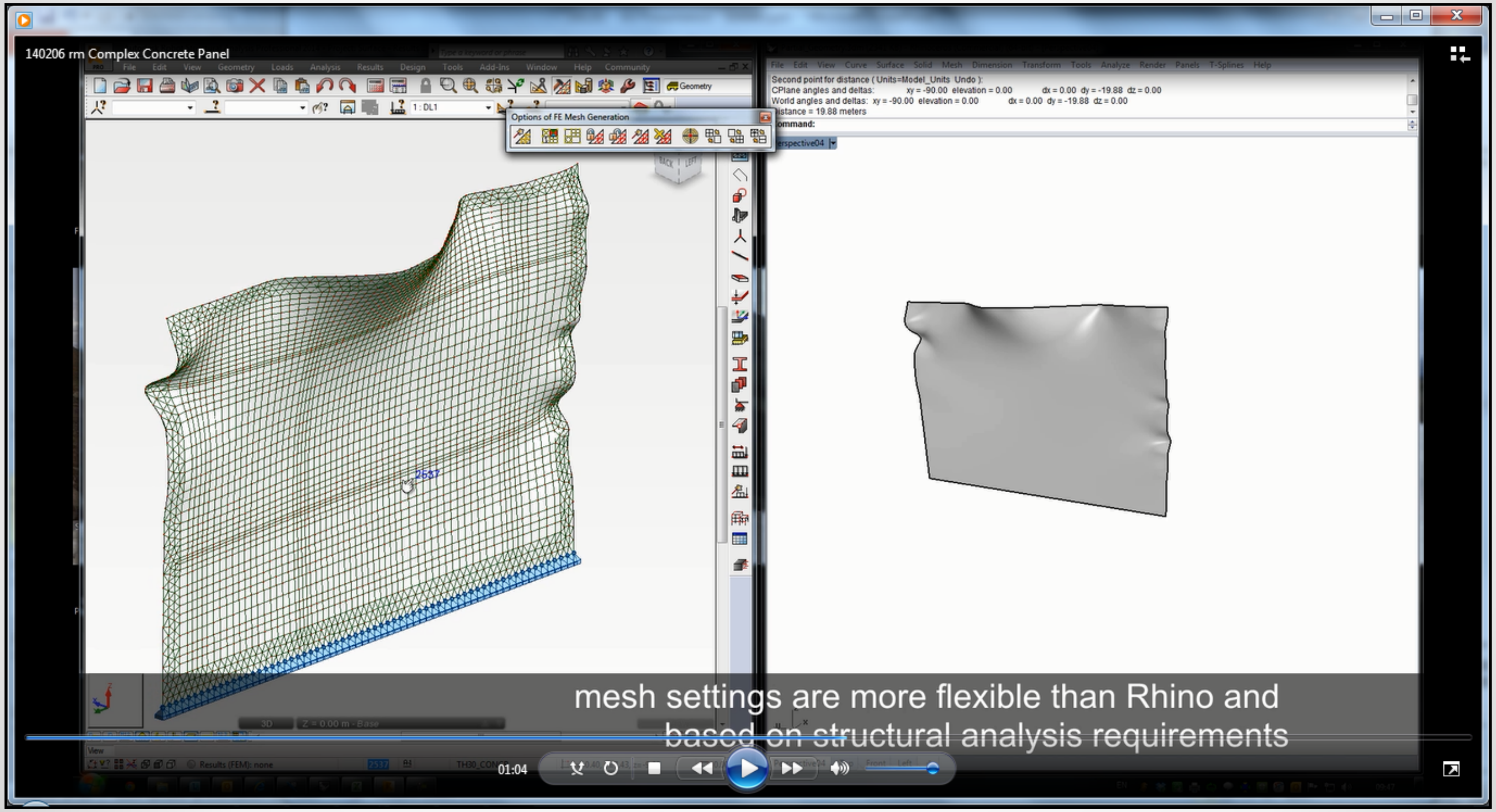
User Interface Tips for Large Models



Using the API to Correct Complex Models



Using SAT Instead of Explicit Generation of Shell FE

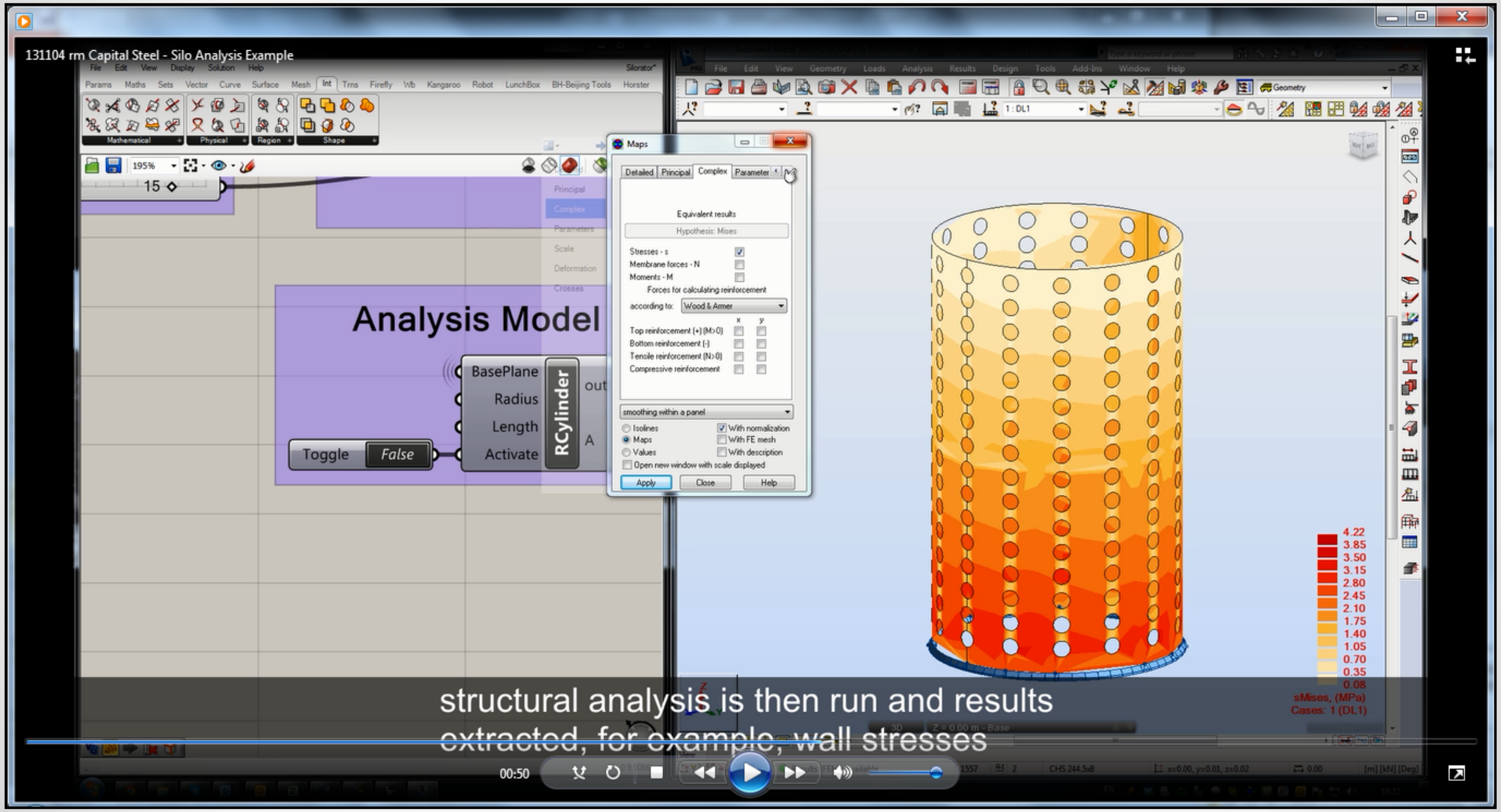


Examples

Concrete Grain Silo Refurbishment

- Generation of cylindrical geometry with proposed new openings
- Simplification of the structure to beam and frame system
- Testing of several opening options using the simple frame system

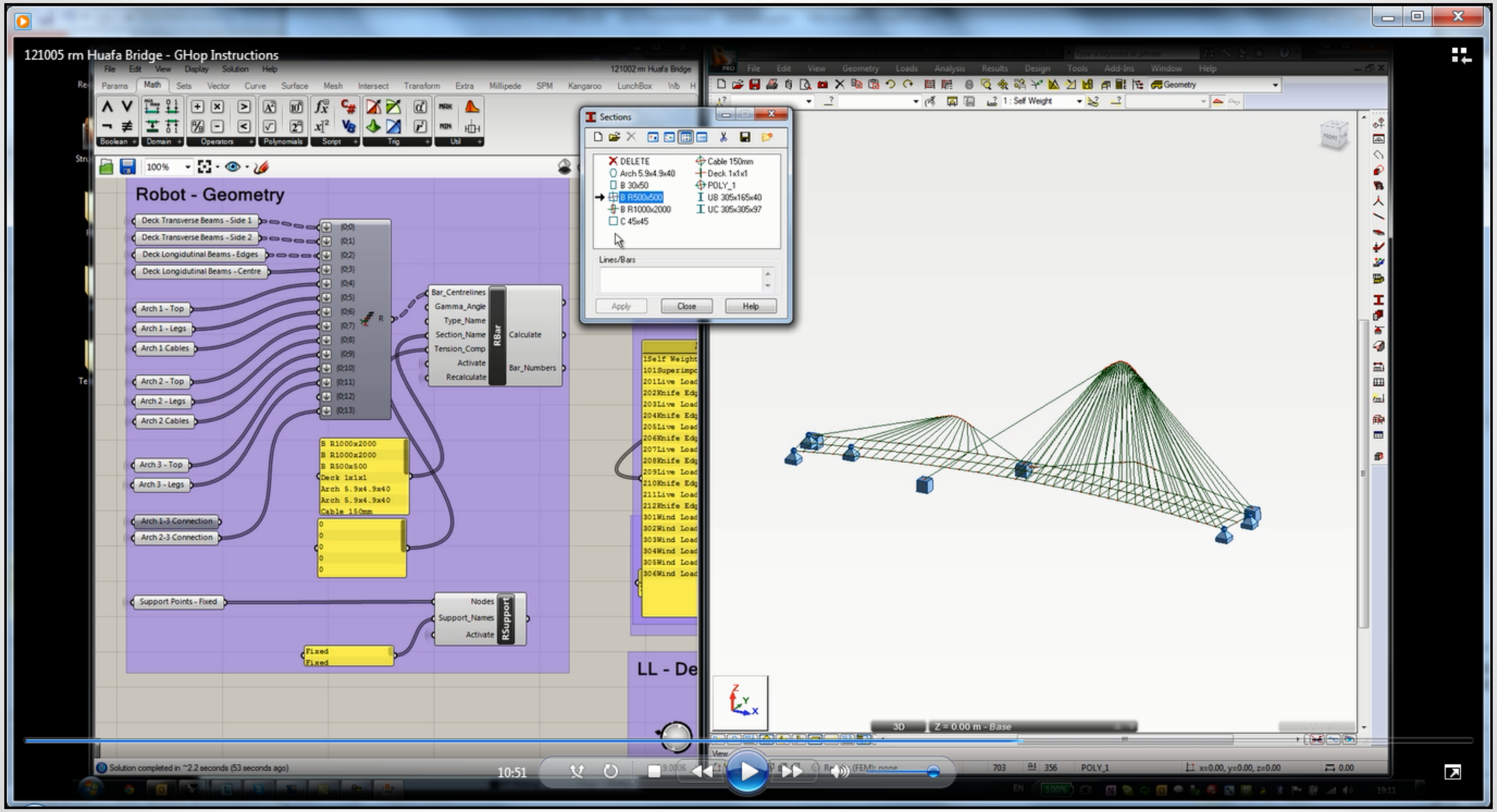
Concrete Grain Silo Refurbishment



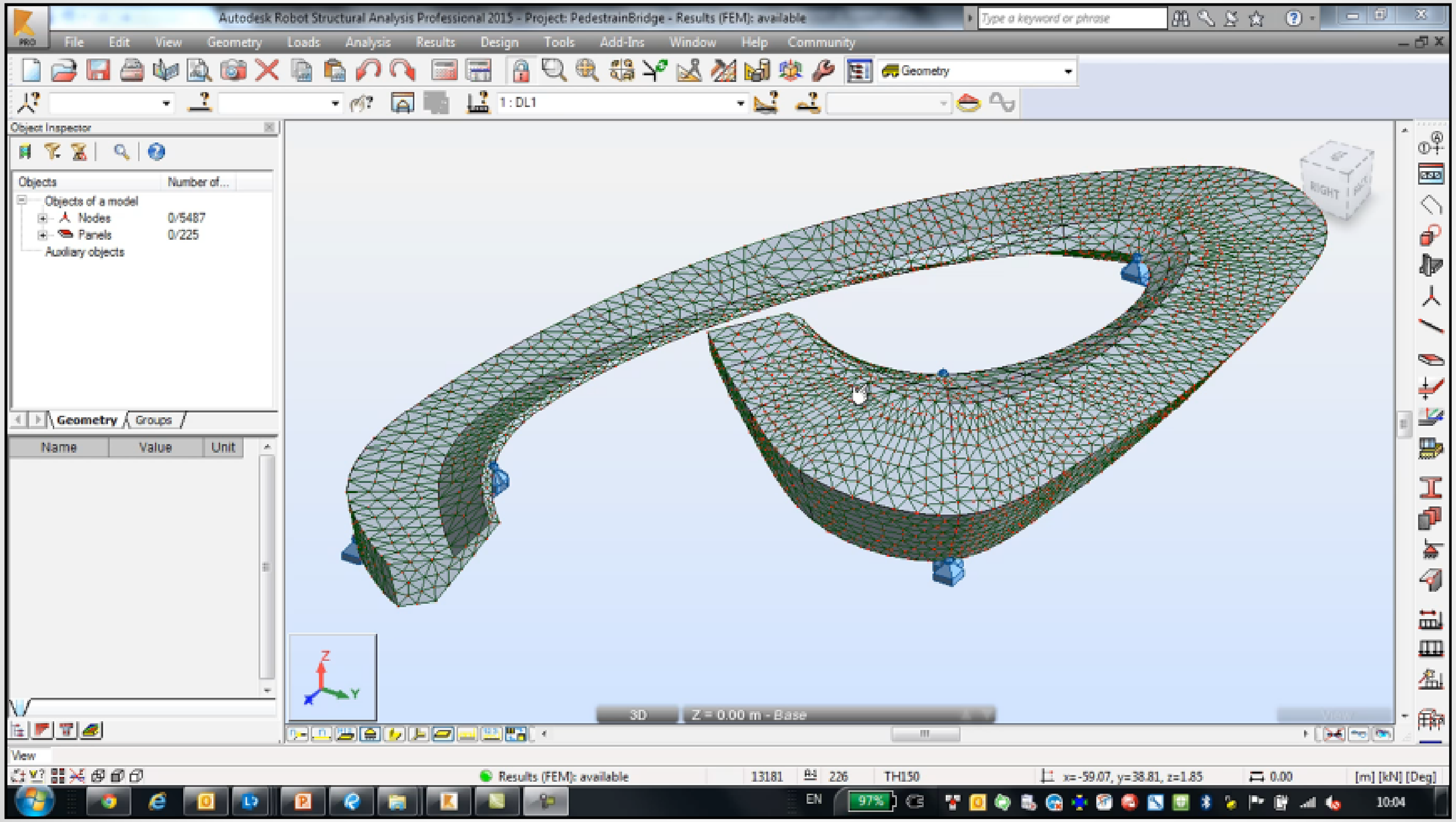
Cable Supported Bridge

- Generation of cable supported complex geometry bridge using Grasshopper
- Feedback of structure deflections and update of cable assembly parameters to find initial lengths for desired assembly case deformation

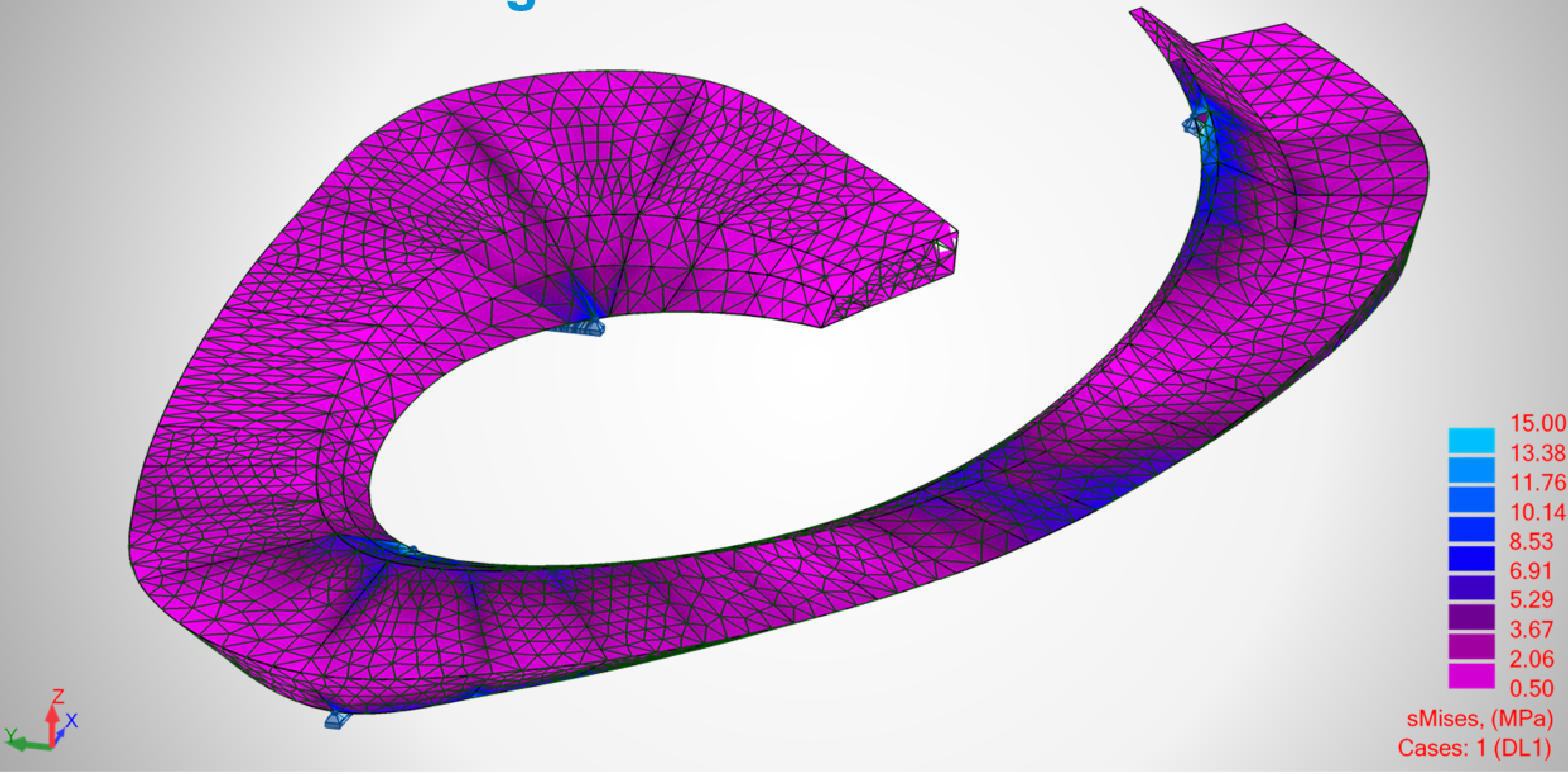
Finding Cable Install Lengths



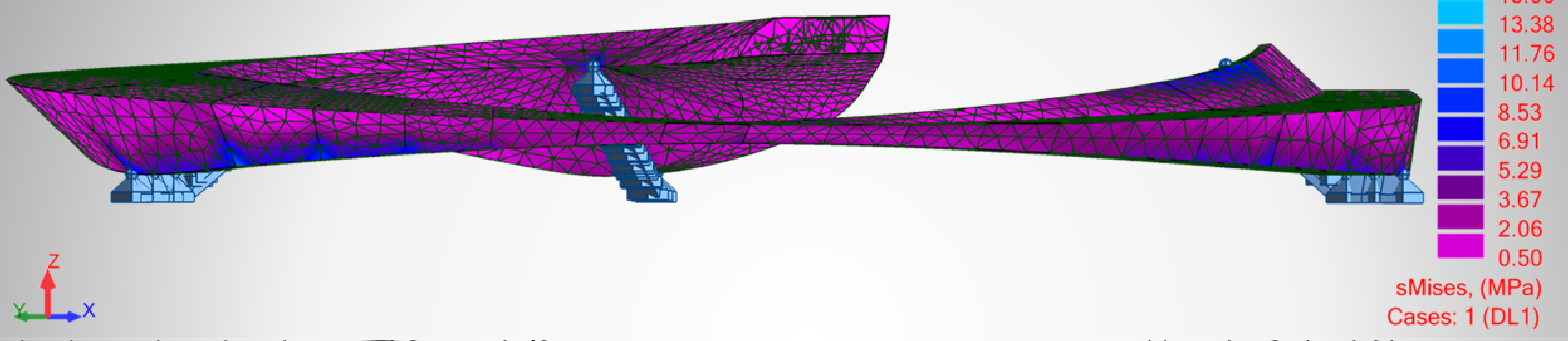
Pedestrian Bridge



Pedestrian Bridge



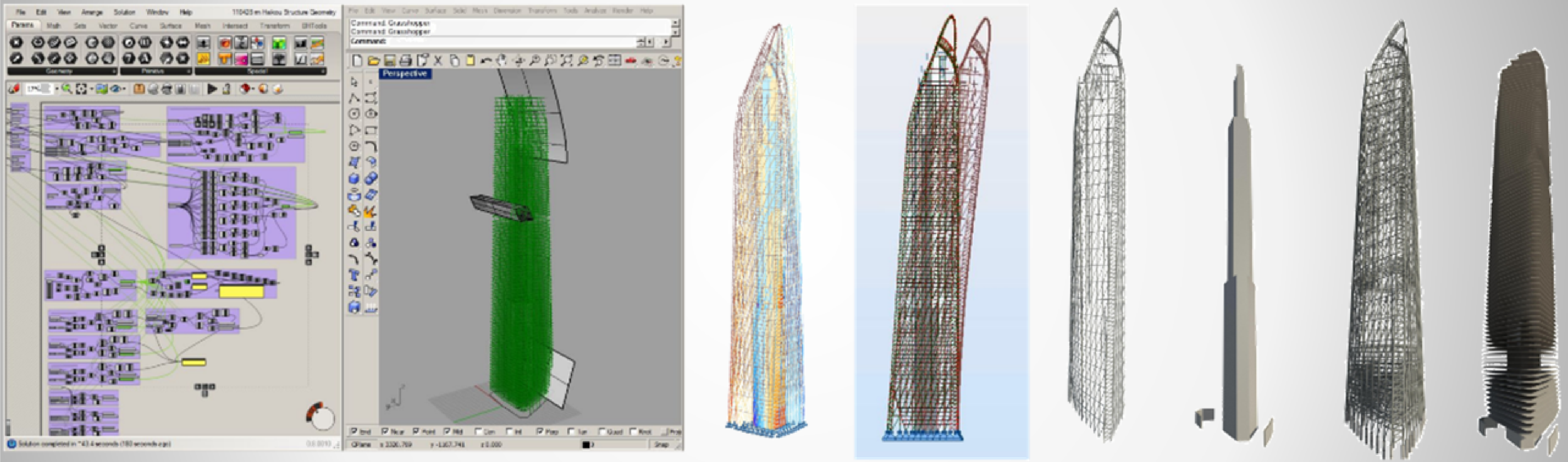
Pedestrian Bridge



Generative Parametric Tall Building Structure Analysis

- Towers are repetitive structures
- Once the tower geometry principles or ‘DNA’ are understood (e.g. diagrid rules) the geometry can be scripted very quickly

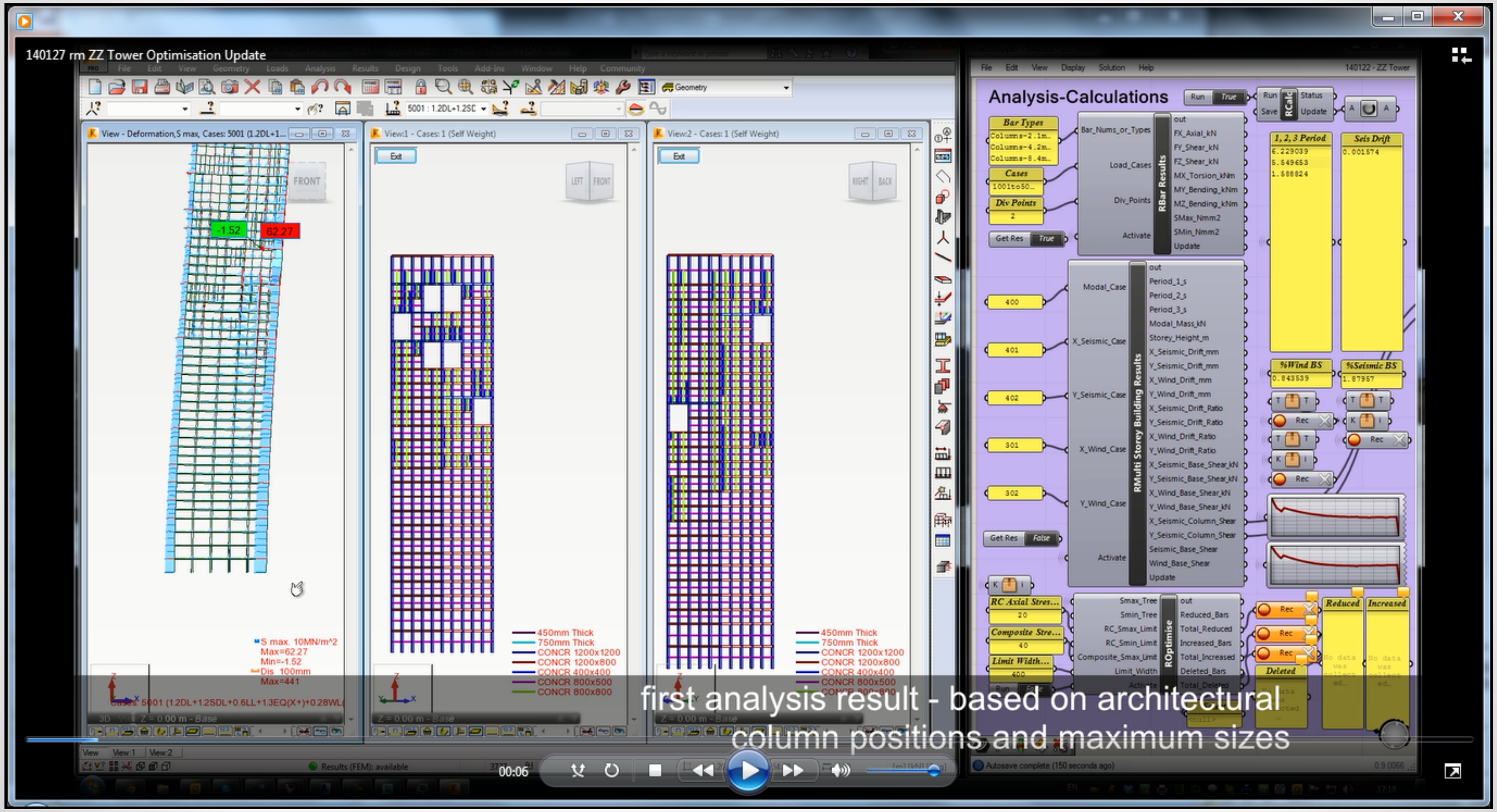
Tower Optimization – Increasing Structure



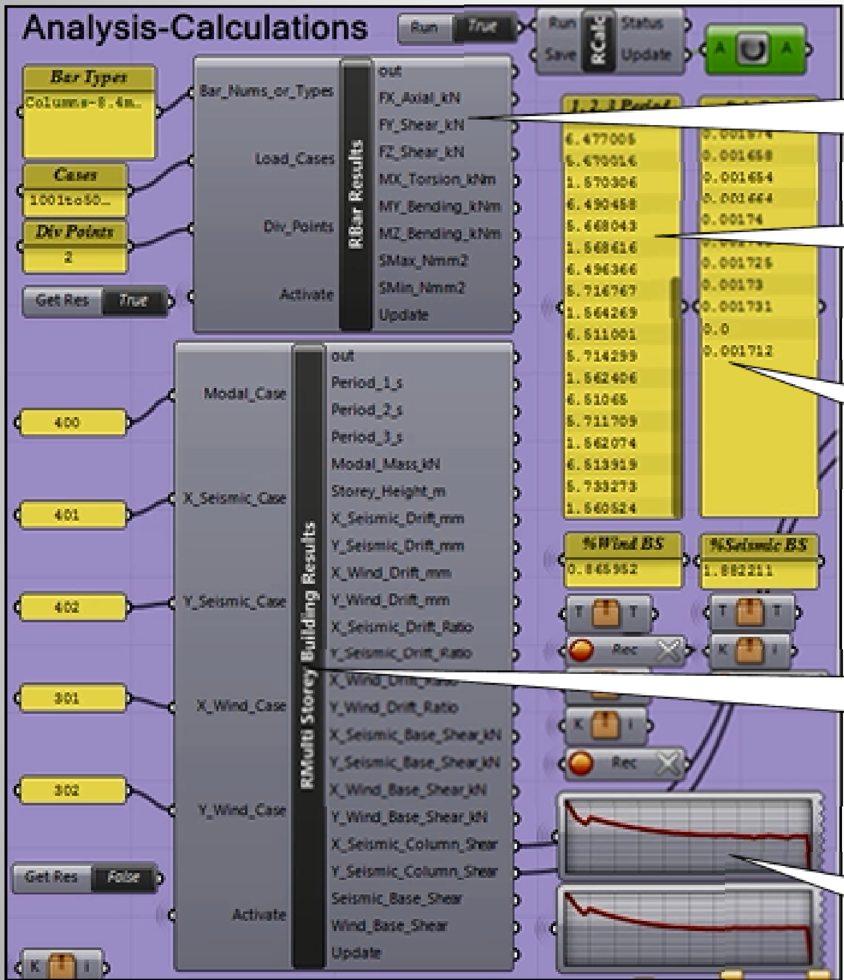
Tower Optimization – Reducing Structure

- Generate the tower structure and simulate it under gravity, seismic and wind loading
- Extract the results and test perimeter structure stresses to measure fitness
- Reduce column sizes to a minimum and then erase
- Start the process again

Tower Optimization – Reducing Structure



Results Extraction



Bar results extraction using Robot API Result Query

Building periods

Seismic drifts

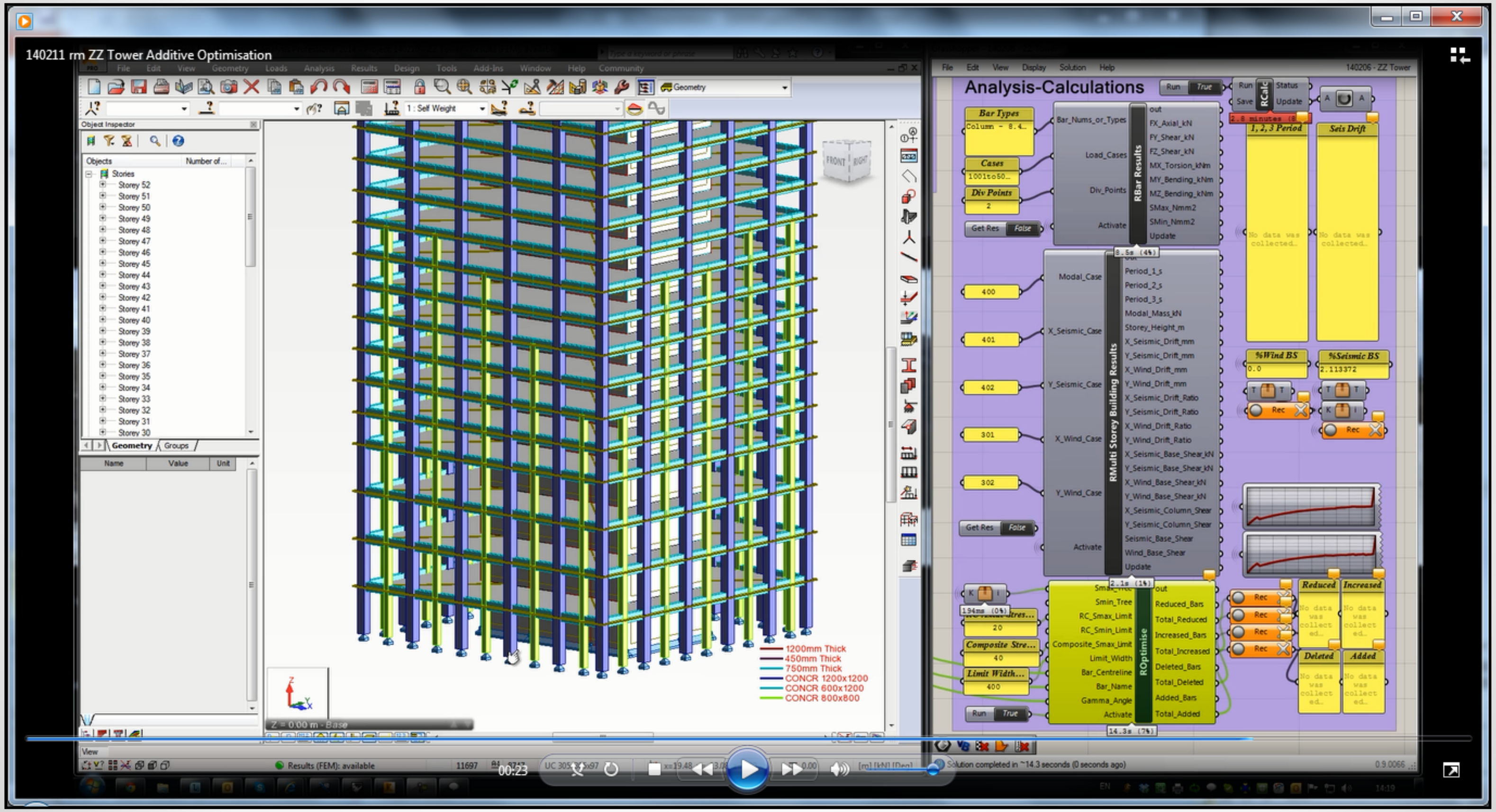
Building results extracted using the Robot API

Wind drift

Result Query

```
Dim result_params As IRobotResultQueryParams
result_params = robot.CmpntFactory.Create(IRobotComponentType.I_CT_RESULT_QUERY_PARAMS)

Dim bar_sel As IRobotSelection
If bar_input_type = "bar_nums" Or bar_nums = "all" Then
  bar_sel = robot.Project.Structure.Selections.Create(IRobotObjectType.I_OT_BAR)
  bar_sel.FromText(bar_nums)
Else
  bar_sel = robot.Project.Structure.Selections.Create(IRobotObjectType.I_OT_BAR)
  For i As int32 = 0 To bar_nums_or_types.count - 1
    bar_sel.Add(robot.Project.Structure.Selections.CreateByLabel(IRobotObjectType.I_OT_
  Next i
End If
Dim cas_sel As IRobotSelection
cas_sel = robot.Project.Structure.Selections.Create(IRobotObjectType.I_OT_CASE)
cas_sel.FromText(Load_Cases)
Dim FX_Tree,FY_Tree,FZ_Tree,MX_Tree,MY_Tree,MZ_Tree,SMax_Tree,Smin_Tree,SX_Tree,Positio
With result_params
  .ResultIds.SetSize(9)
  .ResultIds.Set(1, IRobotExtremeValueType.I_EVT_FORCE_BAR_FX)
  .ResultIds.Set(2, IRobotExtremeValueType.I_EVT_FORCE_BAR_FY)
  .ResultIds.Set(3, IRobotExtremeValueType.I_EVT_FORCE_BAR_FZ)
  .ResultIds.Set(4, IRobotExtremeValueType.I_EVT_FORCE_BAR_MX)
  .ResultIds.Set(5, IRobotExtremeValueType.I_EVT_FORCE_BAR_MY)
  .ResultIds.Set(6, IRobotExtremeValueType.I_EVT_FORCE_BAR_MZ)
  .ResultIds.Set(7, IRobotExtremeValueType.I_EVT_STRESS_BAR_SMAX)
  .ResultIds.Set(8, IRobotExtremeValueType.I_EVT_STRESS_BAR_SMIN)
  .ResultIds.Set(9, IRobotExtremeValueType.I_EVT_STRESS_BAR_FX_SX)
  .SetParam(IRobotResultParamType.I_RPT_BAR_DIV_COUNT, Div_Points)
  .Selection.Set(IRobotObjectType.I_OT_BAR, bar_sel)
  .Selection.Set(IRobotObjectType.I_OT_CASE, cas_sel)
  .SetParam(IRobotResultParamType.I_RPT_MODE, 0)
  .SetParam(IRobotResultParamType.I_RPT_MODE_CMB, IRobotModeCombinationType.I_MCT_CQC)
  .SetParam(IRobotResultParamType.I_RPT_MULTI_THREADS, True)
  .SetParam(IRobotResultParamType.I_RPT_THREAD_COUNT, 4)
End With

Dim row_set As New RobotResultRowSet
Dim query_return As New IRobotResultQueryReturnType
```

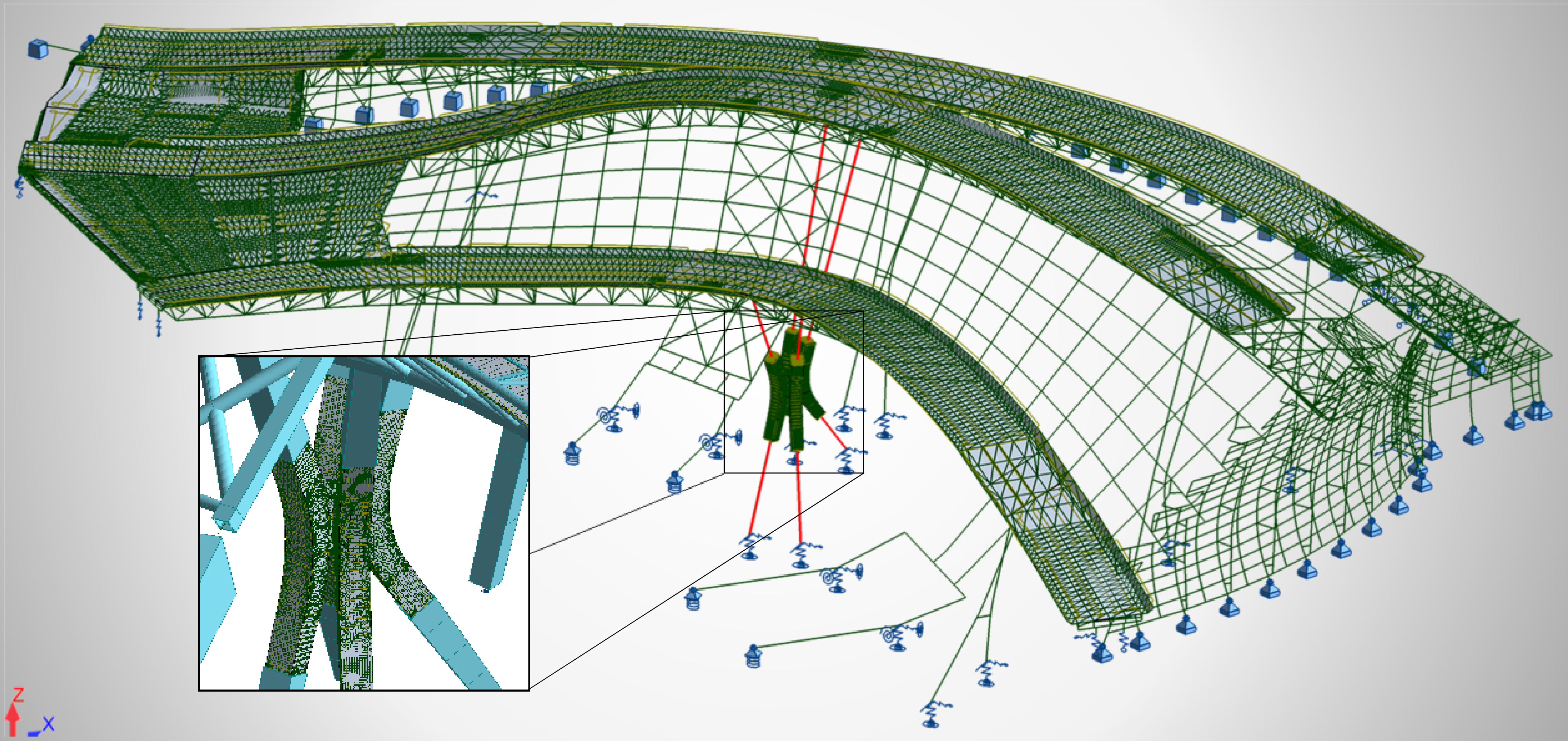
Tower Optimization – Increasing Structure



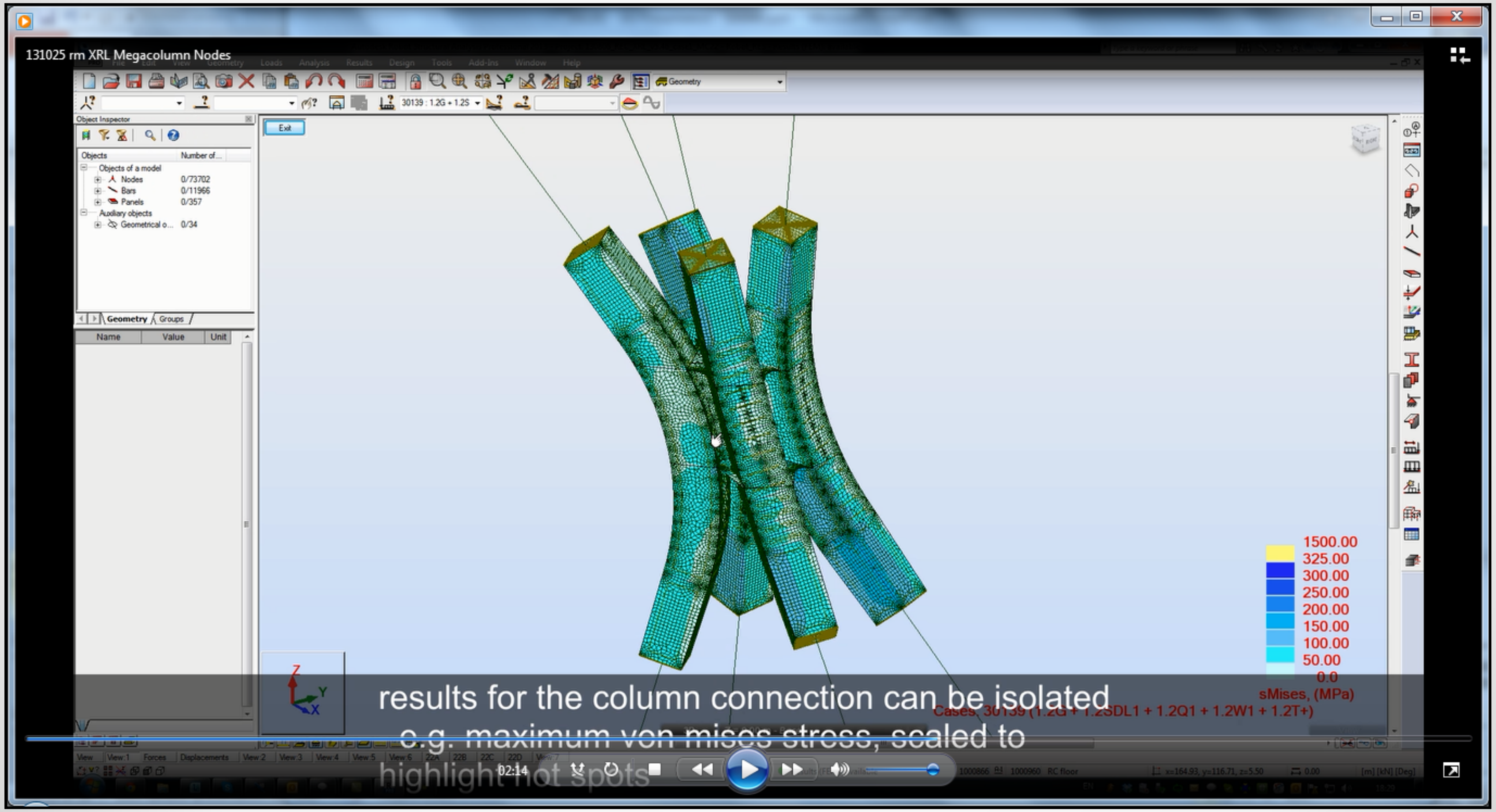
Curved Steel Column Connections Analysis

- Generate geometry from a Tekla fabrication model for analysis of contractor geometry
- Faceting of geometry using Grasshopper to create flat panels for Robot
- Insertion of a detailed shell finite element node into a global roof analysis model

Curved Steel Columns – Connection Analysis



Curved Steel Columns – Connection Analysis

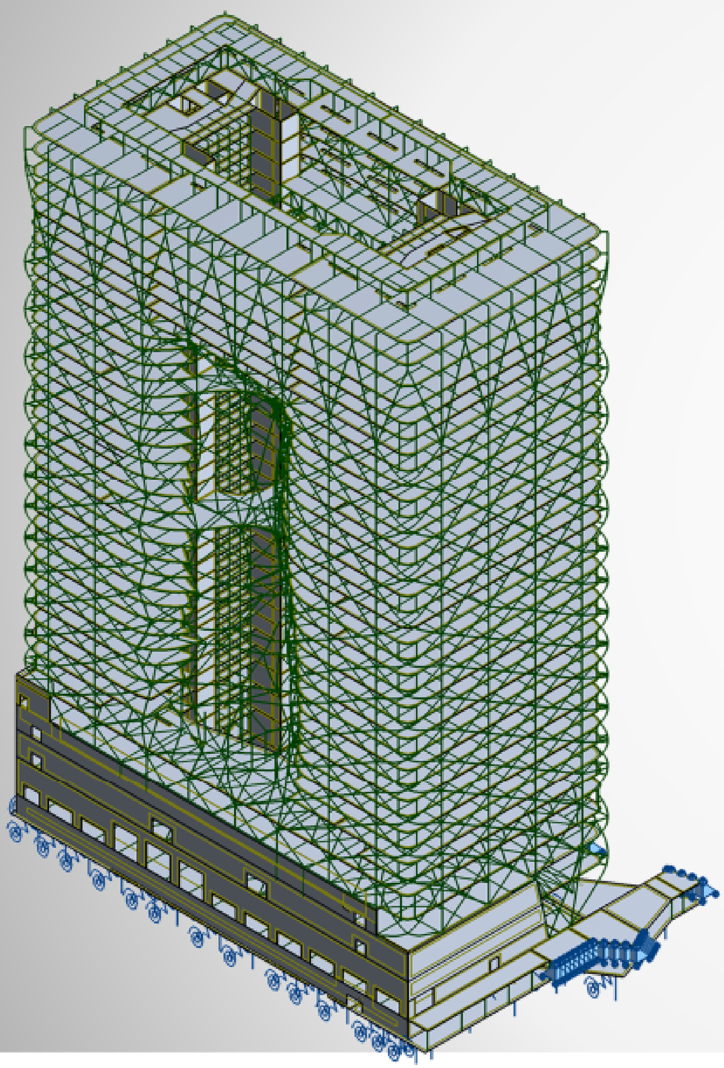


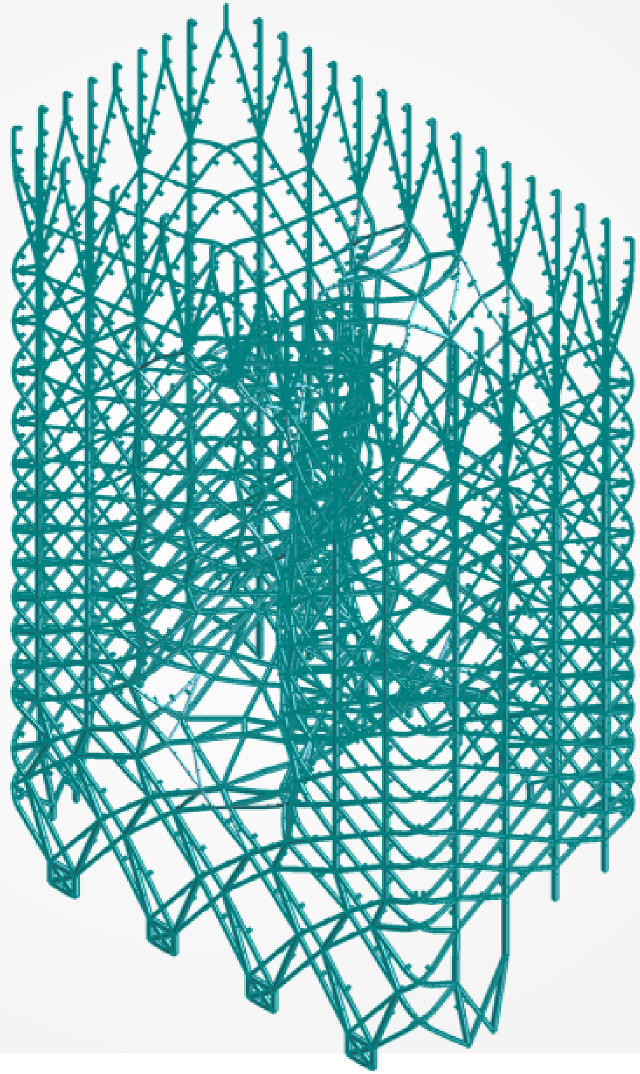
Complex Steel Connections

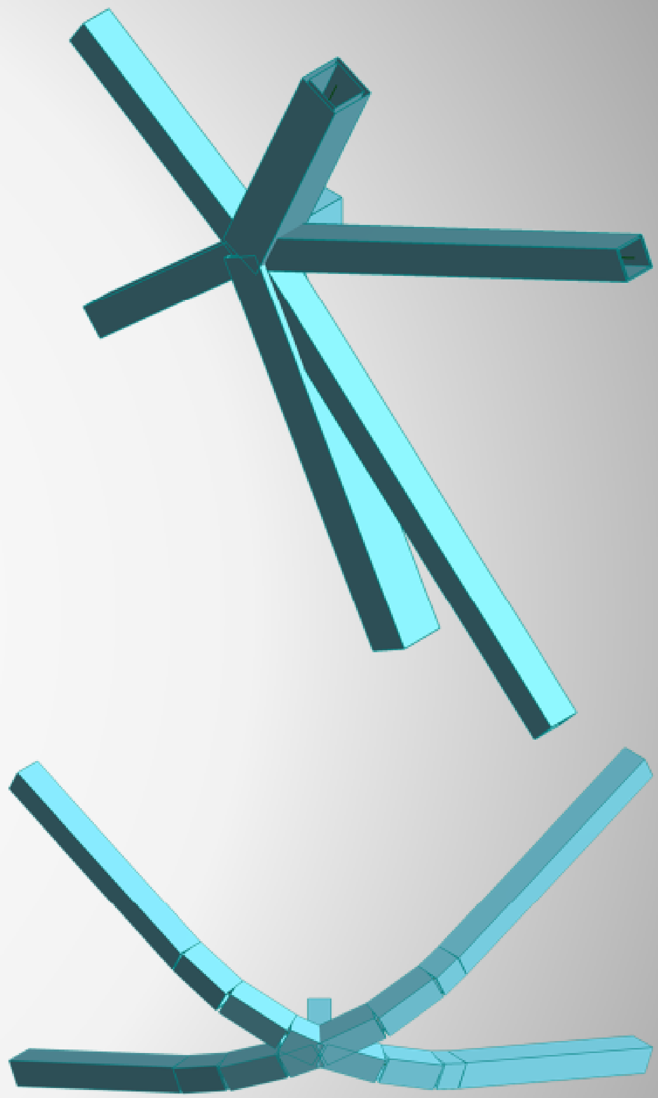
Steel Exoskeleton Connections Analysis

- Extract approximately 6GB of bar forces from Robot and Midas
- Use visual basic to analyze structure for similar connections (e.g. angles, curvature).
- Map and superpose forces to similar connections to reduce number of analysis models
- Generate Robot shell and bar geometry accurately to ensure fast and efficient analysis

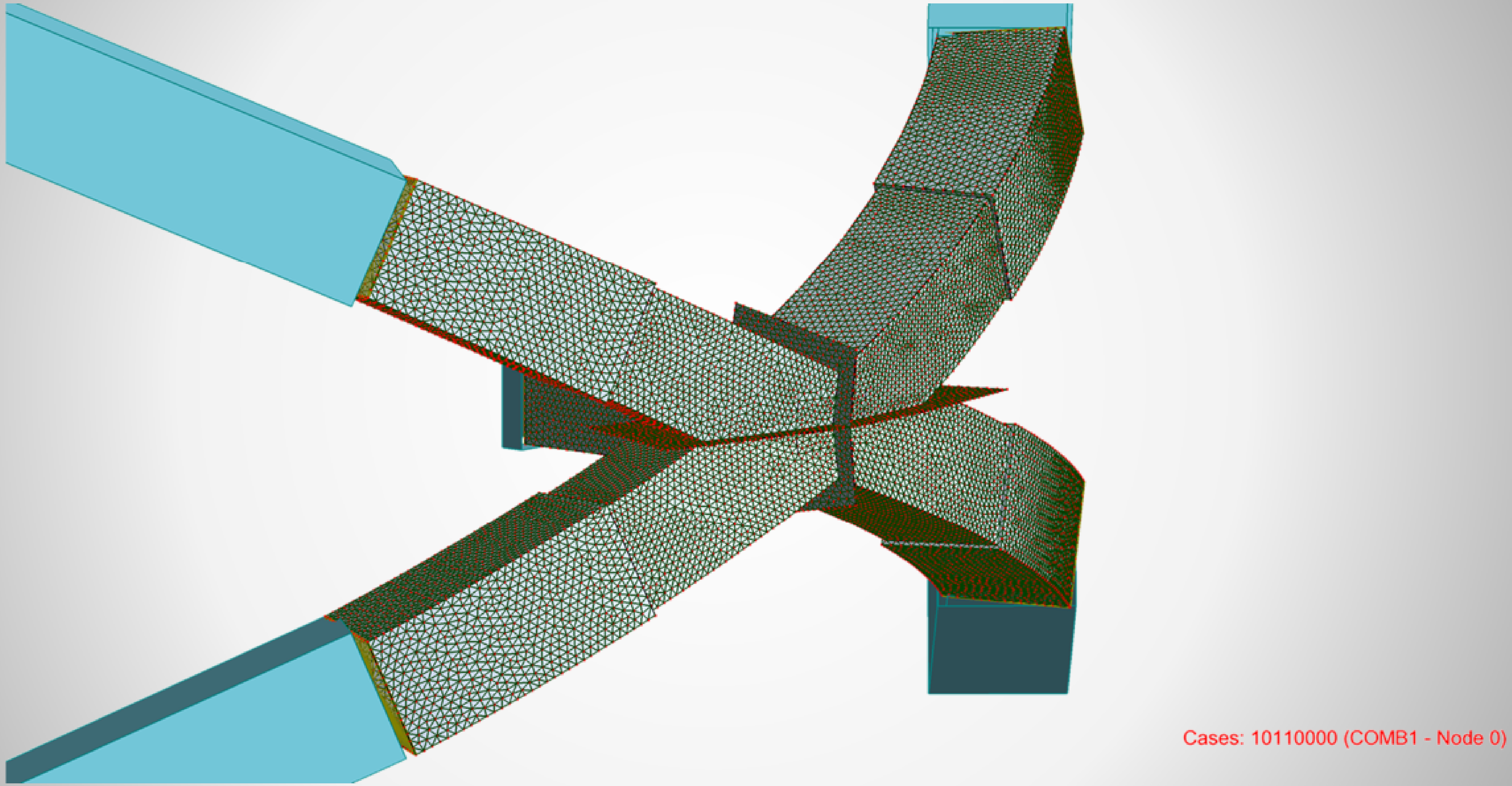
Steel Exoskeleton Connections



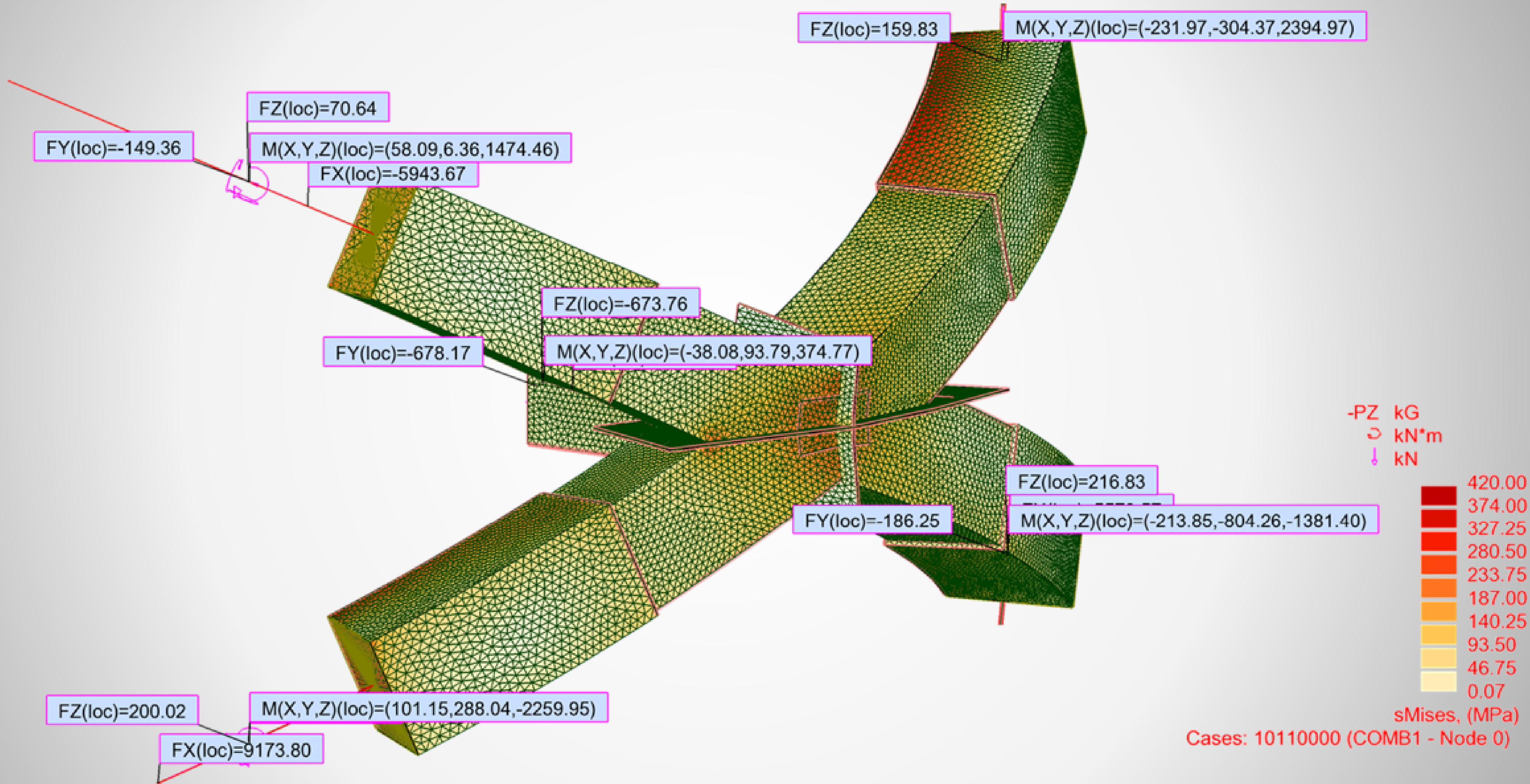




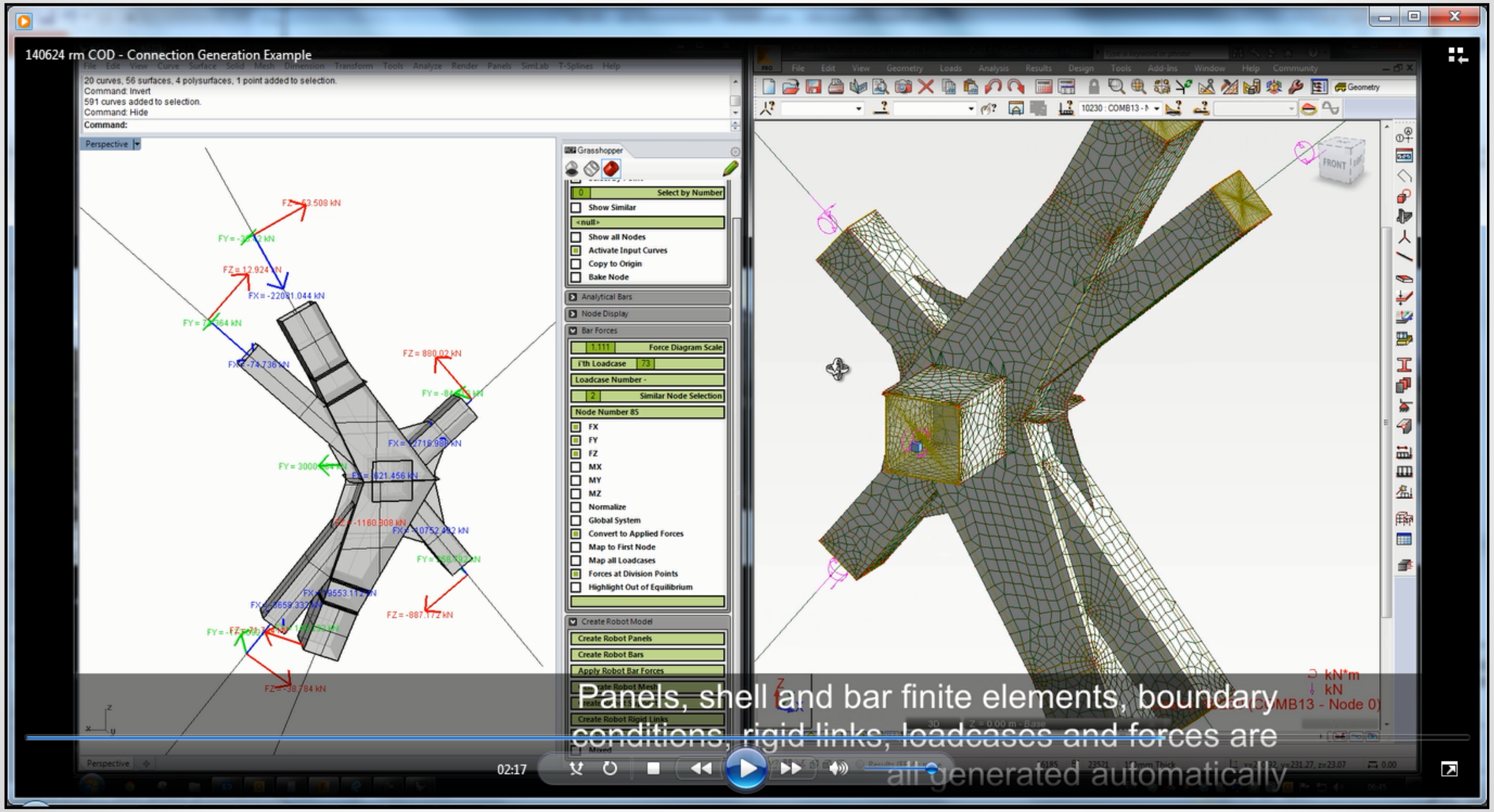
Steel Exoskeleton Connections



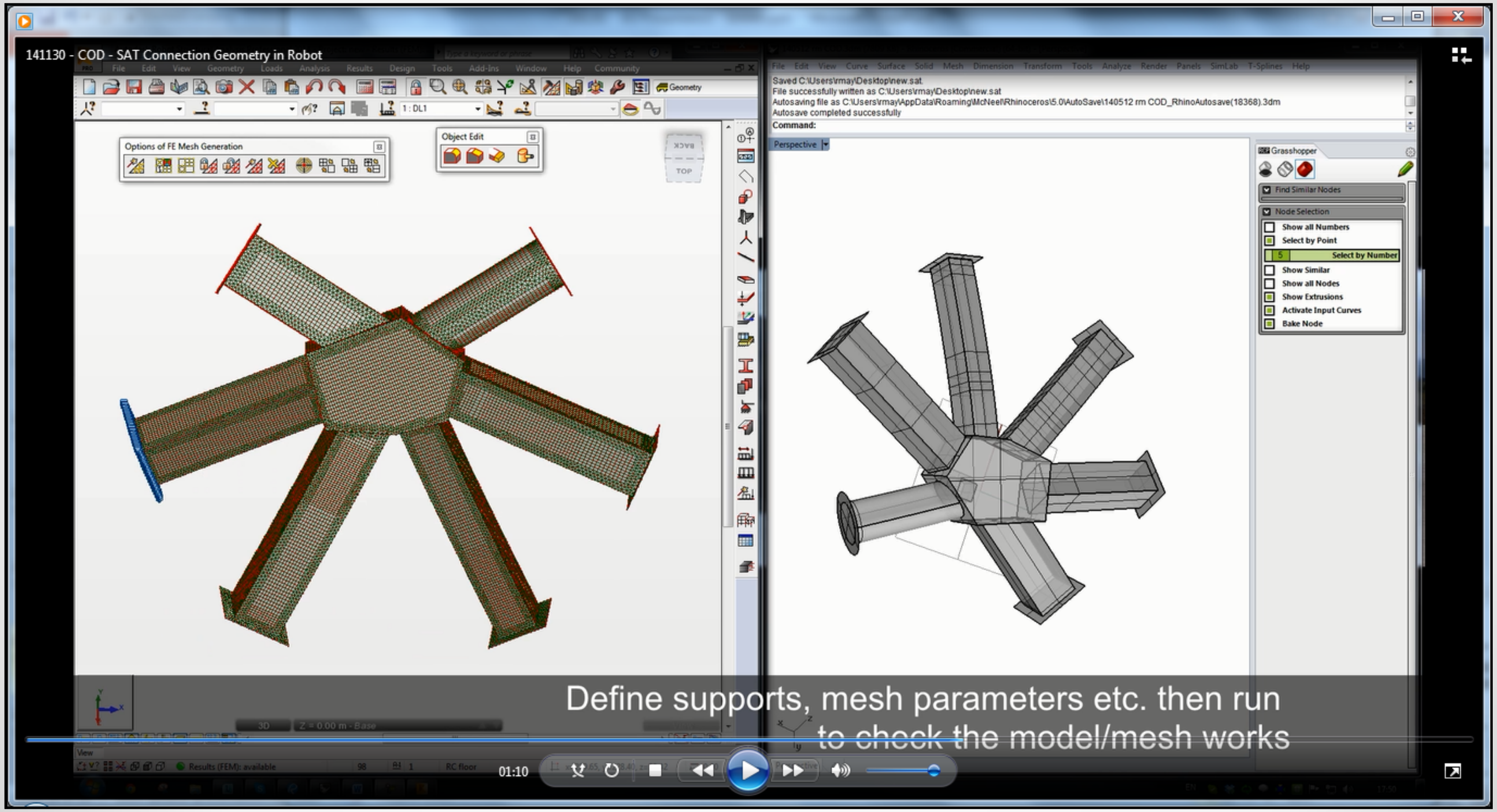
Steel Exoskeleton Connections



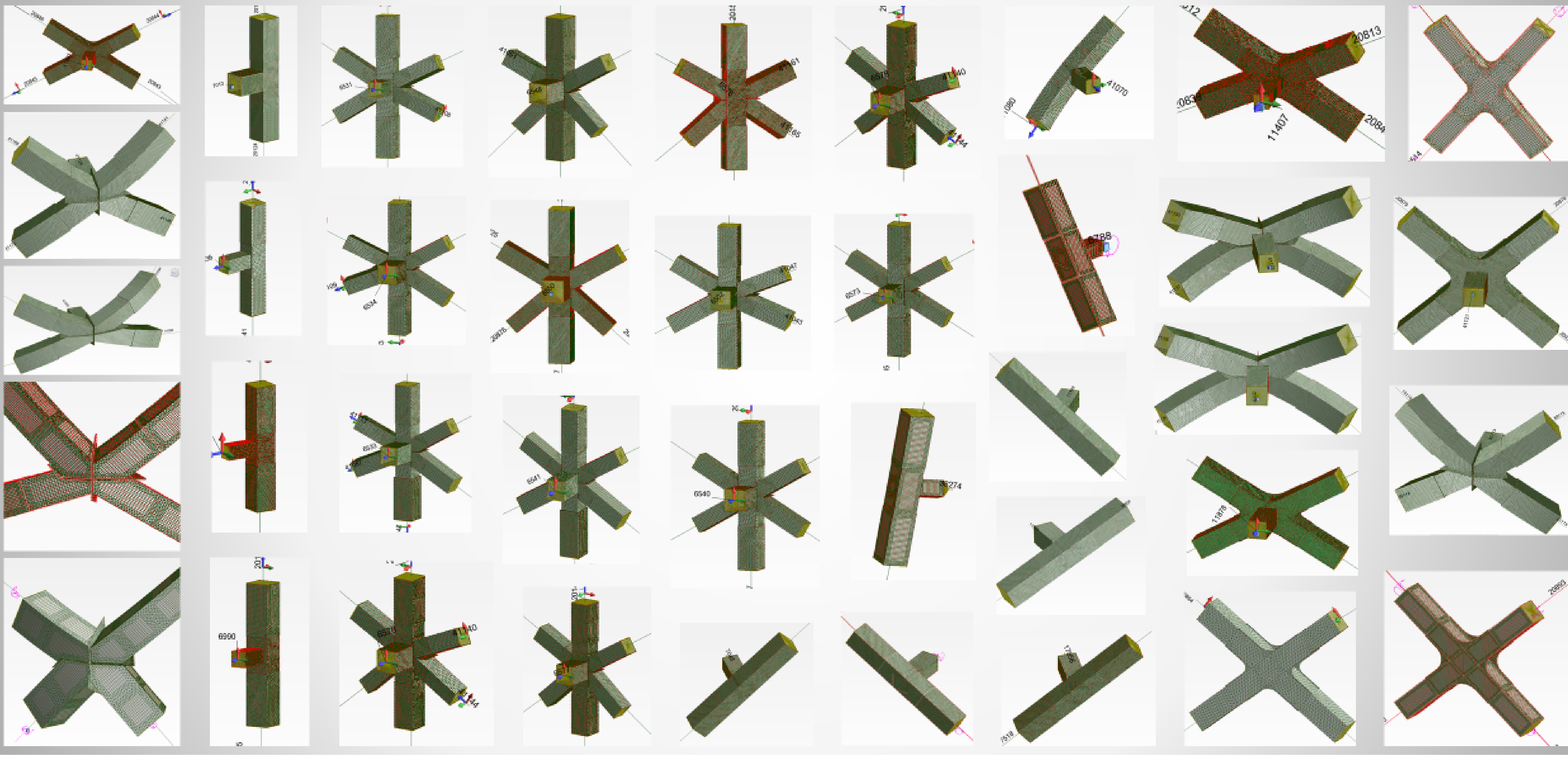
Exoskeleton Nodes – Generation Example



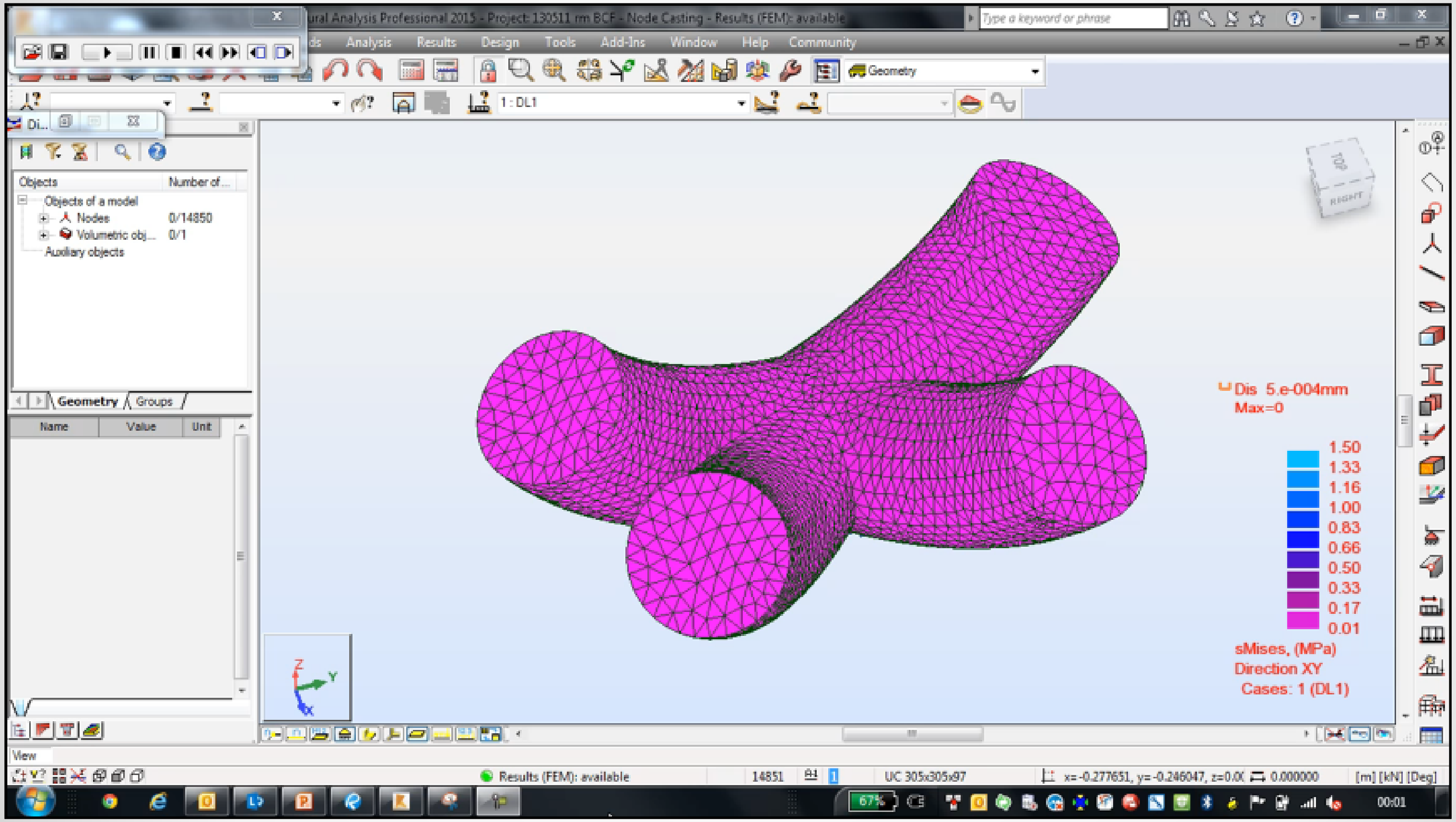
Exoskeleton Nodes – SAT Geometry



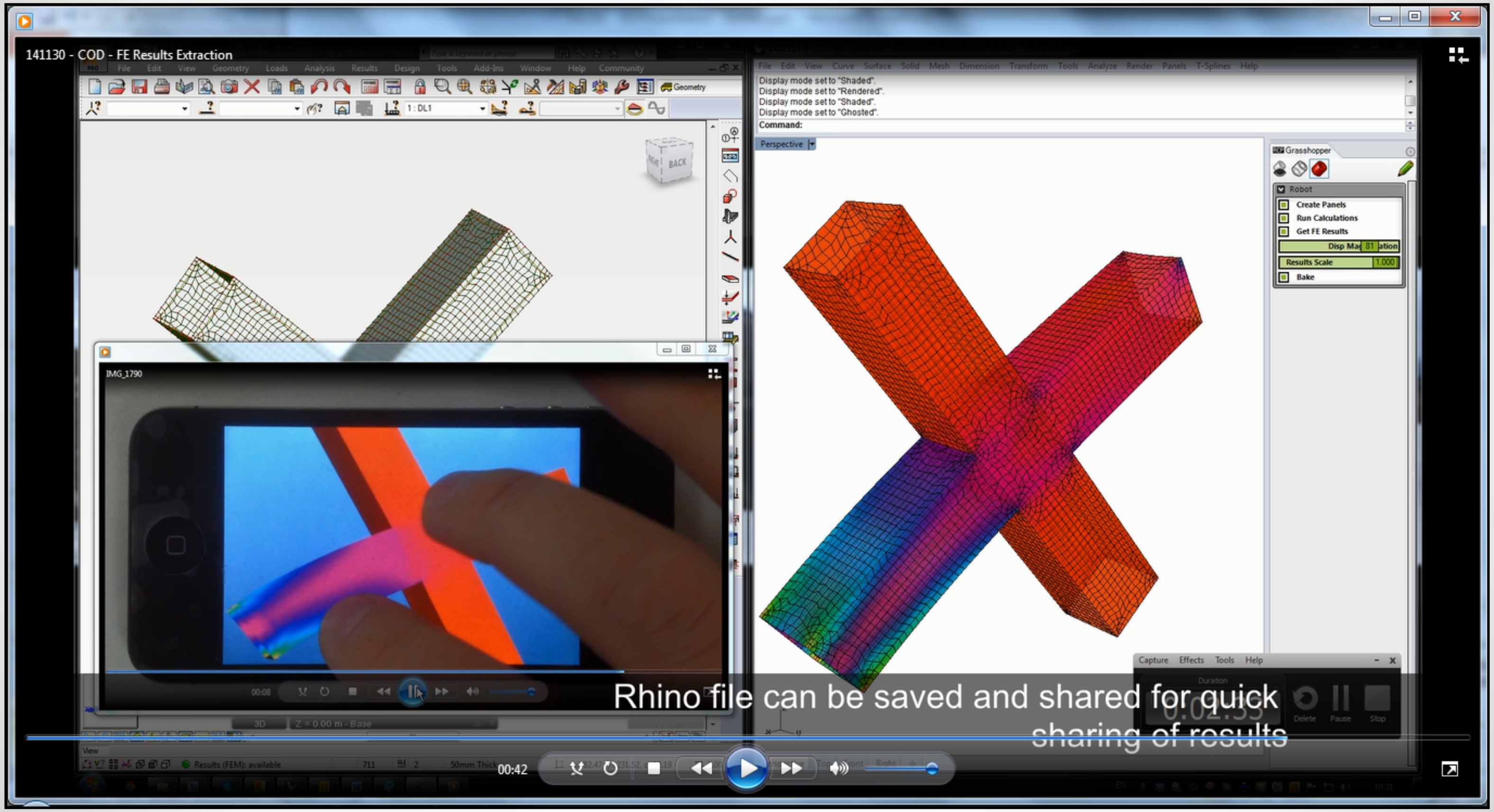
COD Batch 2 of 9: 40 Models – 210 Connections



Volumetric Models – SAT Geometry



Exoskeleton Nodes – Results Extraction



Dynamo - Robot