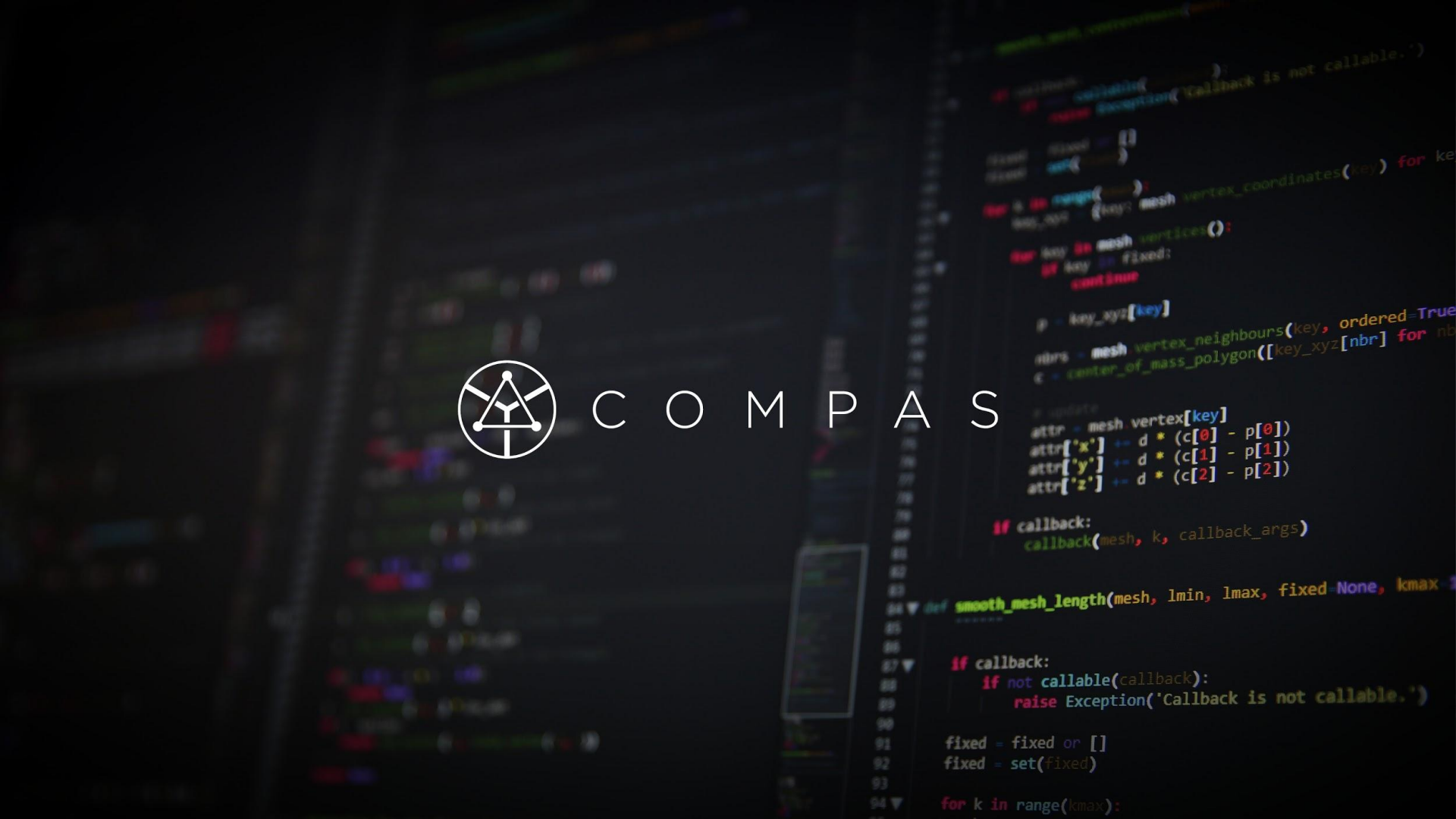
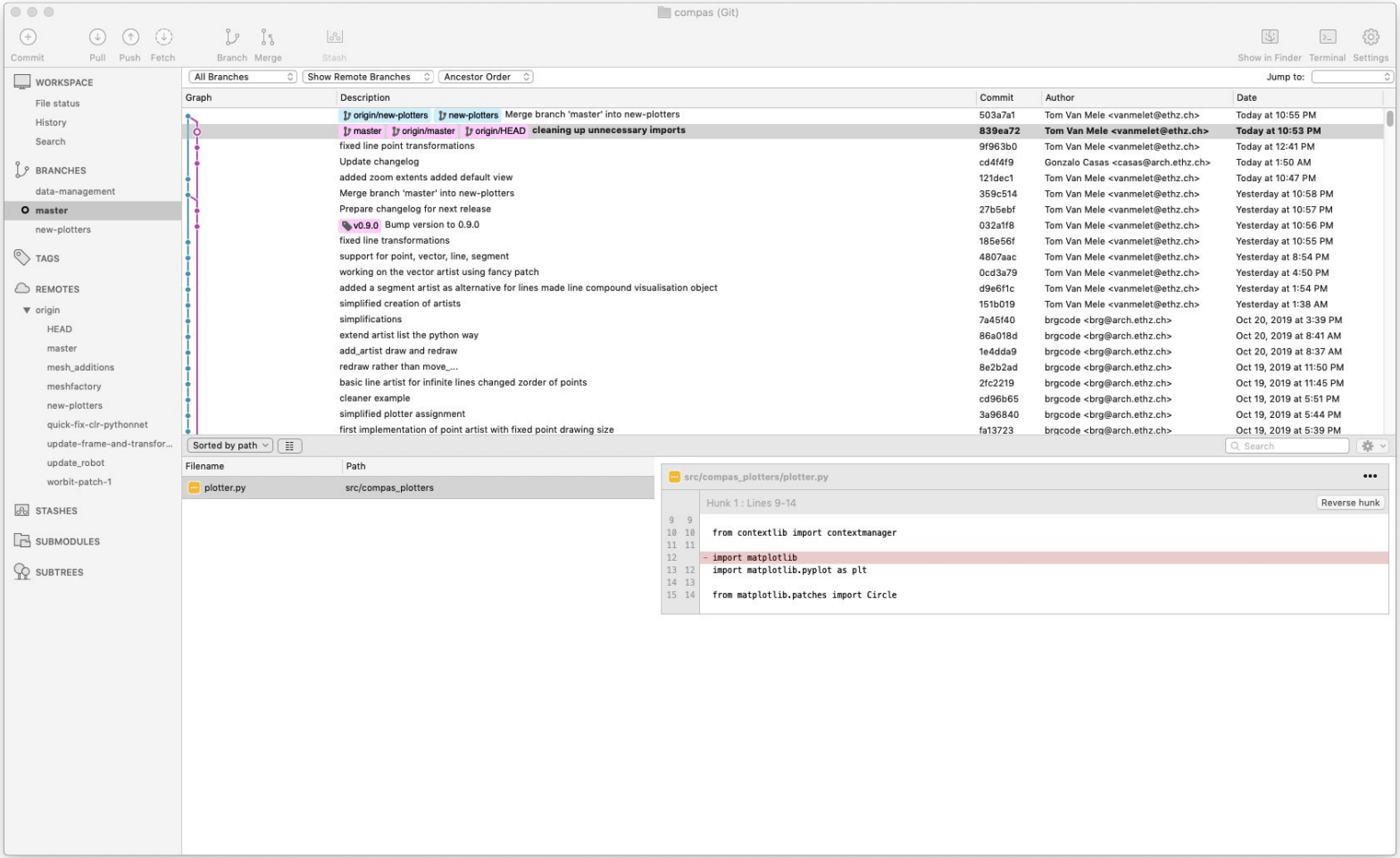
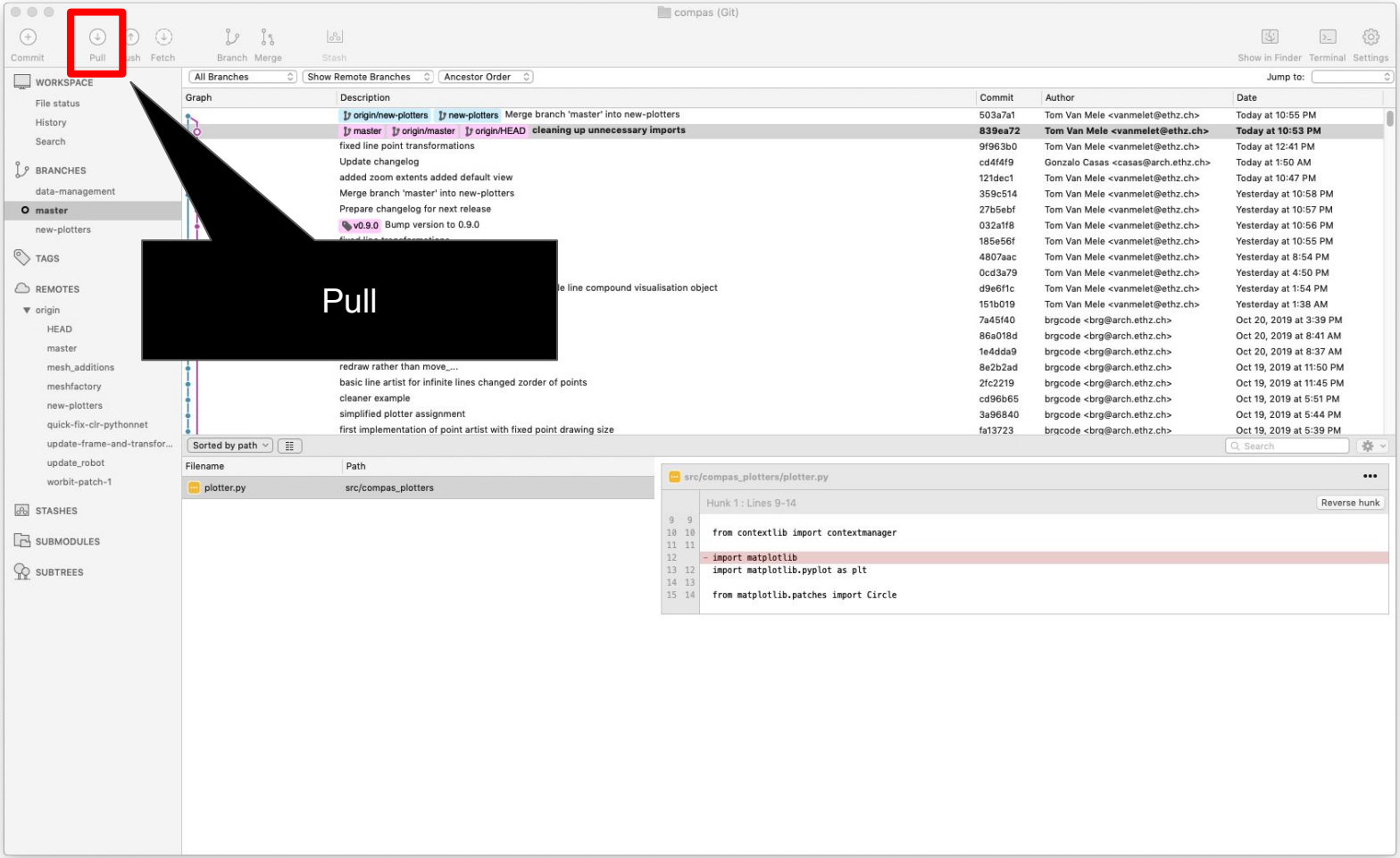
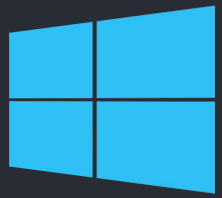

COMPAS

Update COMPAS repo


compas (Git)
Commit
Pull
Push
Fetch
Branch
Merge
Stash
Show in Finder
Terminal
Settings
WORKSPACE
File status
History
Search
BRANCHES
data-management
master
new-plotters
TAGS
REMOTES
origin
HEAD
master
mesh_additions
meshfactory
new-plotters
quick-fix-clr-pythonnet
update-frame-and-transfor...
update_robot
worbit-patch-1
STASHES
SUBMODULES
SUBTREES
All Branches
Show Remote Branches
Ancestor Order
Jump to:
Graph
Description
Commit
Author
Date
origin/new-plotters
new-plotters
Merge branch 'master' into new-plotters
503a7a1
Tom Van Mele <vanmelet@ethz.ch>
Today at 10:55 PM
master
origin/master
origin/HEAD
cleaning up unnecessary imports
839ea72
Tom Van Mele <vanmelet@ethz.ch>
Today at 10:53 PM
fixed line point transformations
9f963b0
Tom Van Mele <vanmelet@ethz.ch>
Today at 12:41 PM
Update changelog
cd4f4f9
Gonzalo Casas <casas@arch.ethz.ch>
Today at 1:50 AM
added zoom extents added default view
121dec1
Tom Van Mele <vanmelet@ethz.ch>
Today at 10:47 PM
Merge branch 'master' into new-plotters
359c514
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 10:58 PM
Prepare changelog for next release
27b5ebf
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 10:57 PM
v0.9.0
Bump version to 0.9.0
032a1f8
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 10:56 PM
fixed line transformations
185e56f
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 10:55 PM
support for point, vector, line, segment
4807aac
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 8:54 PM
working on the vector artist using fancy patch
0cd3a79
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 4:50 PM
added a segment artist as alternative for lines made line compound visualisation object
d9e6f1c
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 1:54 PM
simplified creation of artists
151b019
Tom Van Mele <vanmelet@ethz.ch>
Yesterday at 1:38 AM
simplifications
7a45f40
brgcode <brg@arch.ethz.ch>
Oct 20, 2019 at 3:39 PM
extend artist list the python way
86a018d
brgcode <brg@arch.ethz.ch>
Oct 20, 2019 at 8:41 AM
add_artist draw and redraw
1e4dda9
brgcode <brg@arch.ethz.ch>
Oct 20, 2019 at 8:37 AM
redraw rather than move_...
8e2b2ad
brgcode <brg@arch.ethz.ch>
Oct 19, 2019 at 11:50 PM
basic line artist for infinite lines changed zorder of points
2fc2219
brgcode <brg@arch.ethz.ch>
Oct 19, 2019 at 11:45 PM
cleaner example
cd96b65
brgcode <brg@arch.ethz.ch>
Oct 19, 2019 at 5:51 PM
simplified plotter assignment
3a96840
brgcode <brg@arch.ethz.ch>
Oct 19, 2019 at 5:44 PM
first implementation of point artist with fixed point drawing size
fa13723
brgcode <brg@arch.ethz.ch>
Oct 19, 2019 at 5:39 PM
Sorted by path
Search
Filename
Path
plotter.py
src/compas_plotters
src/compas_plotters/plotter.py
Hunk 1 : Lines 9-14
Reverse hunk
from contextlib import contextmanager
- import matplotlib
import matplotlib.pyplot as plt
from matplotlib.patches import Circle


compas (Git)
Commit
Pull
Fetch
Branch
Merge
Stash
Show in Finder
Terminal
Settings
WORKSPACE
File status
History
Search
BRANCHES
data-management
master
new-plotters
TAGS
REMOTES
origin
HEAD
master
mesh_additions
meshfactory
new-plotters
quick-fix-clr-pythonnet
update-frame-and-transfor...
update_robot
worbit-patch-1
STASHES
SUBMODULES
SUBTREES
All Branches
Show Remote Branches
Ancestor Order
Jump to:
Graph
Description
Commit
Author
Date
origin/new-plotters new-plotters Merge branch 'master' into new-plotters 503a7a1 Tom Van Mele <vanmelet@ethz.ch> Today at 10:55 PM
master origin/master origin/HEAD cleaning up unnecessary imports 839ea72 Tom Van Mele <vanmelet@ethz.ch> Today at 10:53 PM
fixed line point transformations 9f963b0 Tom Van Mele <vanmelet@ethz.ch> Today at 12:41 PM
Update changelog cd4f4f9 Gonzalo Casas <casas@arch.ethz.ch> Today at 1:50 AM
added zoom extents added default view 121dec1 Tom Van Mele <vanmelet@ethz.ch> Today at 10:47 PM
Merge branch 'master' into new-plotters 359c514 Tom Van Mele <vanmelet@ethz.ch> Yesterday at 10:58 PM
Prepare changelog for next release 27b5ebf Tom Van Mele <vanmelet@ethz.ch> Yesterday at 10:57 PM
v0.9.0 Bump version to 0.9.0 032a1f8 Tom Van Mele <vanmelet@ethz.ch> Yesterday at 10:56 PM
185e56f Tom Van Mele <vanmelet@ethz.ch> Yesterday at 10:55 PM
4807aac Tom Van Mele <vanmelet@ethz.ch> Yesterday at 8:54 PM
0cd3a79 Tom Van Mele <vanmelet@ethz.ch> Yesterday at 4:50 PM
d9e6f1c Tom Van Mele <vanmelet@ethz.ch> Yesterday at 1:54 PM
151b019 Tom Van Mele <vanmelet@ethz.ch> Yesterday at 1:38 AM
7a45f40 brgcode <brg@arch.ethz.ch> Oct 20, 2019 at 3:39 PM
86a018d brgcode <brg@arch.ethz.ch> Oct 20, 2019 at 8:41 AM
1e4dda9 brgcode <brg@arch.ethz.ch> Oct 20, 2019 at 8:37 AM
redraw rather than move_... 8e2b2ad brgcode <brg@arch.ethz.ch> Oct 19, 2019 at 11:50 PM
basic line artist for infinite lines changed zorder of points 2fc2219 brgcode <brg@arch.ethz.ch> Oct 19, 2019 at 11:45 PM
cleaner example cd96b65 brgcode <brg@arch.ethz.ch> Oct 19, 2019 at 5:51 PM
simplified plotter assignment 3a96840 brgcode <brg@arch.ethz.ch> Oct 19, 2019 at 5:44 PM
first implementation of point artist with fixed point drawing size fa13723 brgcode <brg@arch.ethz.ch> Oct 19, 2019 at 5:39 PM
Pull
Sorted by path
Search
Filename
Path
plotter.py
src/compas_plotters
src/compas_plotters/plotter.py
Hunk 1 : Lines 9-14
Reverse hunk
from contextlib import contextmanager
- import matplotlib
import matplotlib.pyplot as plt
from matplotlib.patches import Circle

Verify ita19-dev



Terminal

Anaconda Prompt

```
conda activate ita19-dev
python
```

```
conda activate ita19-dev
python
```

```
>>> import compas
>>>
```

```
>>> import compas
>>> compas.__version__
'0.10.1'
>>>
```

```
>>> import compas
>>> compas.__version__
'0.10.1'
>>> from compas.rpc import Proxy
>>>
```

```
>>> import compas
>>> compas.__version__
'0.10.1'
>>> from compas.rpc import Proxy
>>> p = Proxy()
>>>
```

```
>>> import compas
>>> compas.__version__
'0.10.1'
>>> from compas.rpc import Proxy
>>> p = Proxy()
>>> p.stop_server()
>>>
```

```
>>> import compas
>>> compas.__version__
'0.10.1'
>>> from compas.rpc import Proxy
>>> p = Proxy()
>>> p.stop_server()
>>> p.start_server()
>>>
```

```
>>> import compas
>>> compas.__version__
'0.10.1'
>>> from compas.rpc import Proxy
>>> p = Proxy()
>>> p.stop_server()
>>> p.start_server()
>>> exit()
```

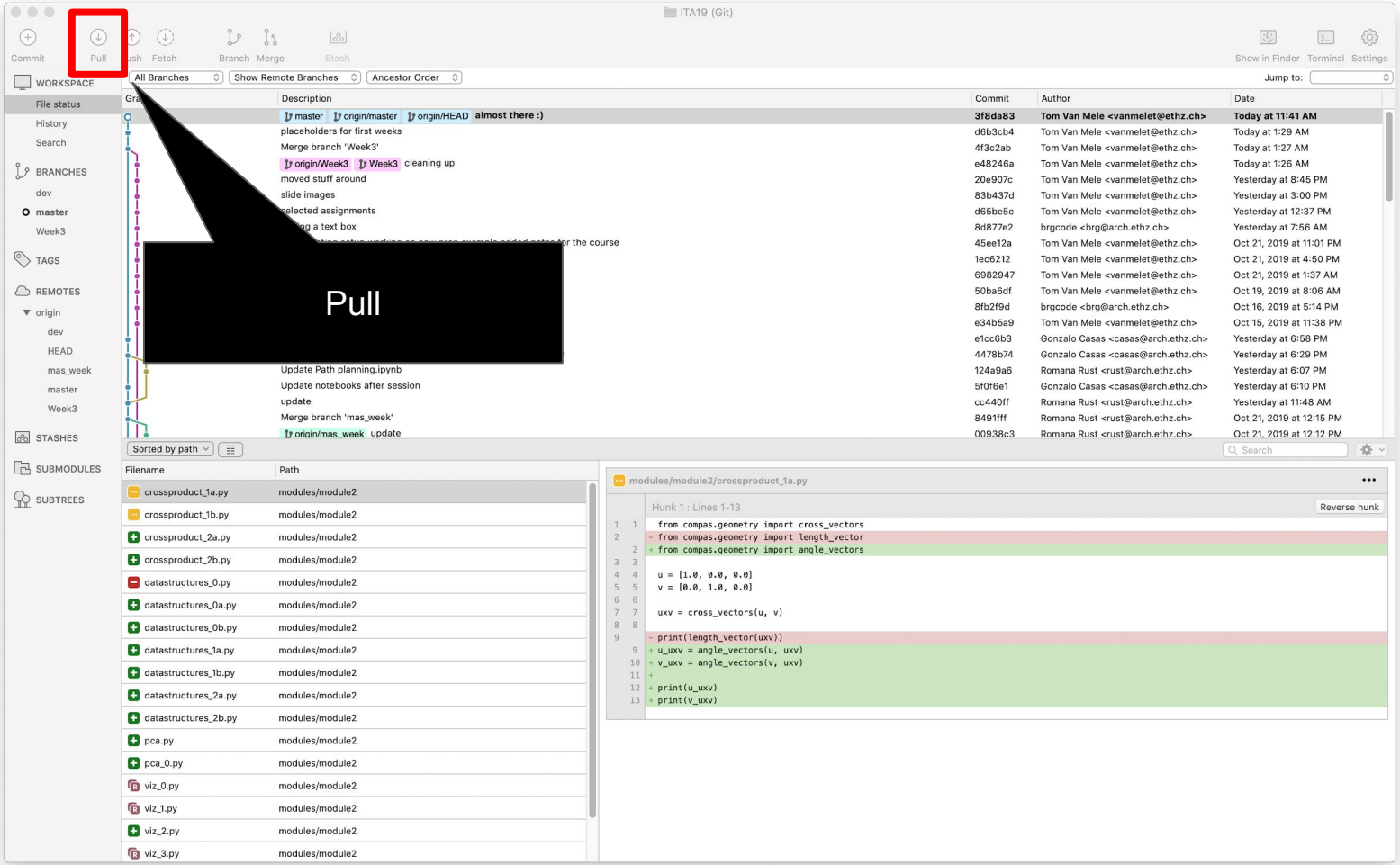
Update Rhino

```
python -m compas_rhino.uninstall
```

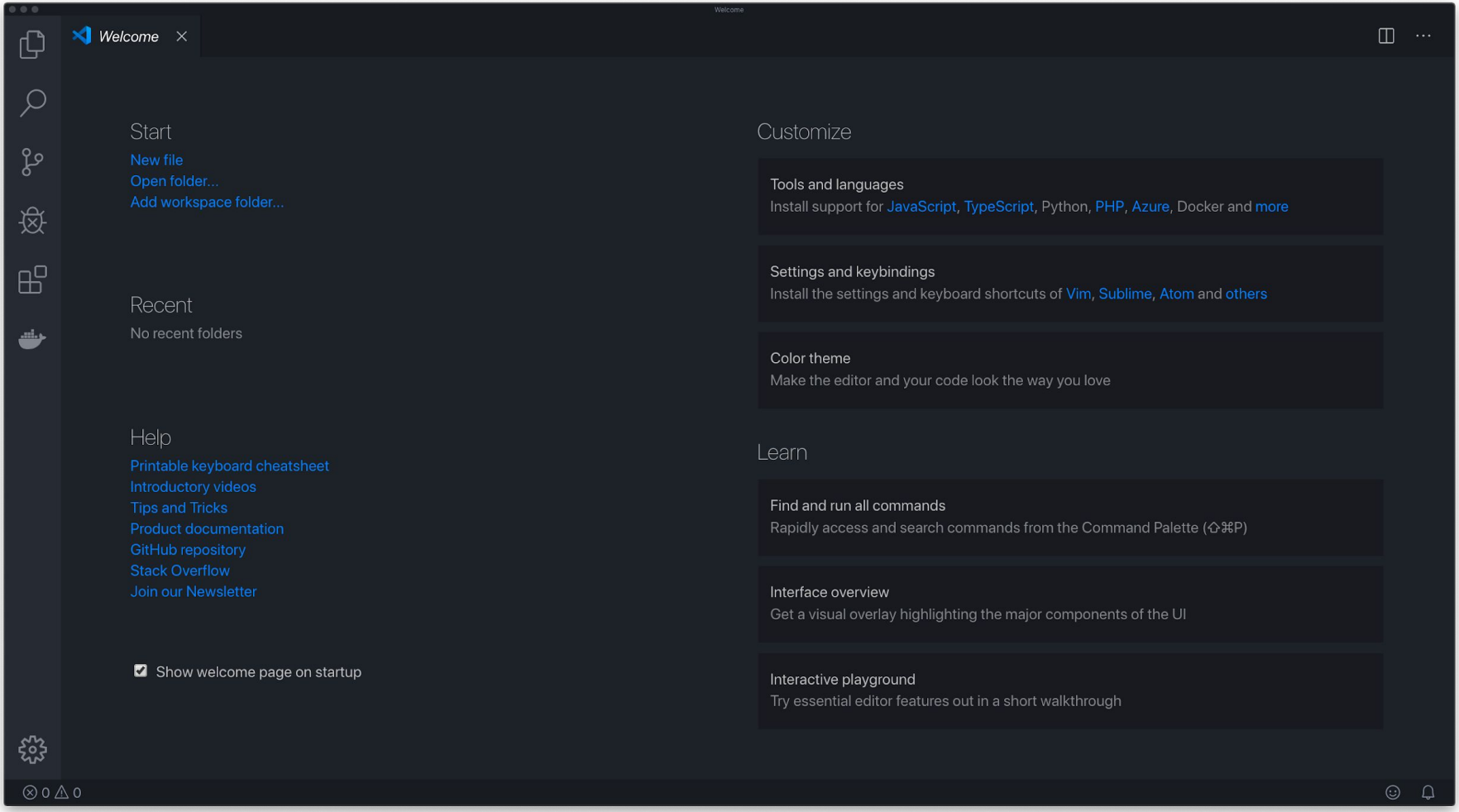
```
python -m compas_rhino.install
```

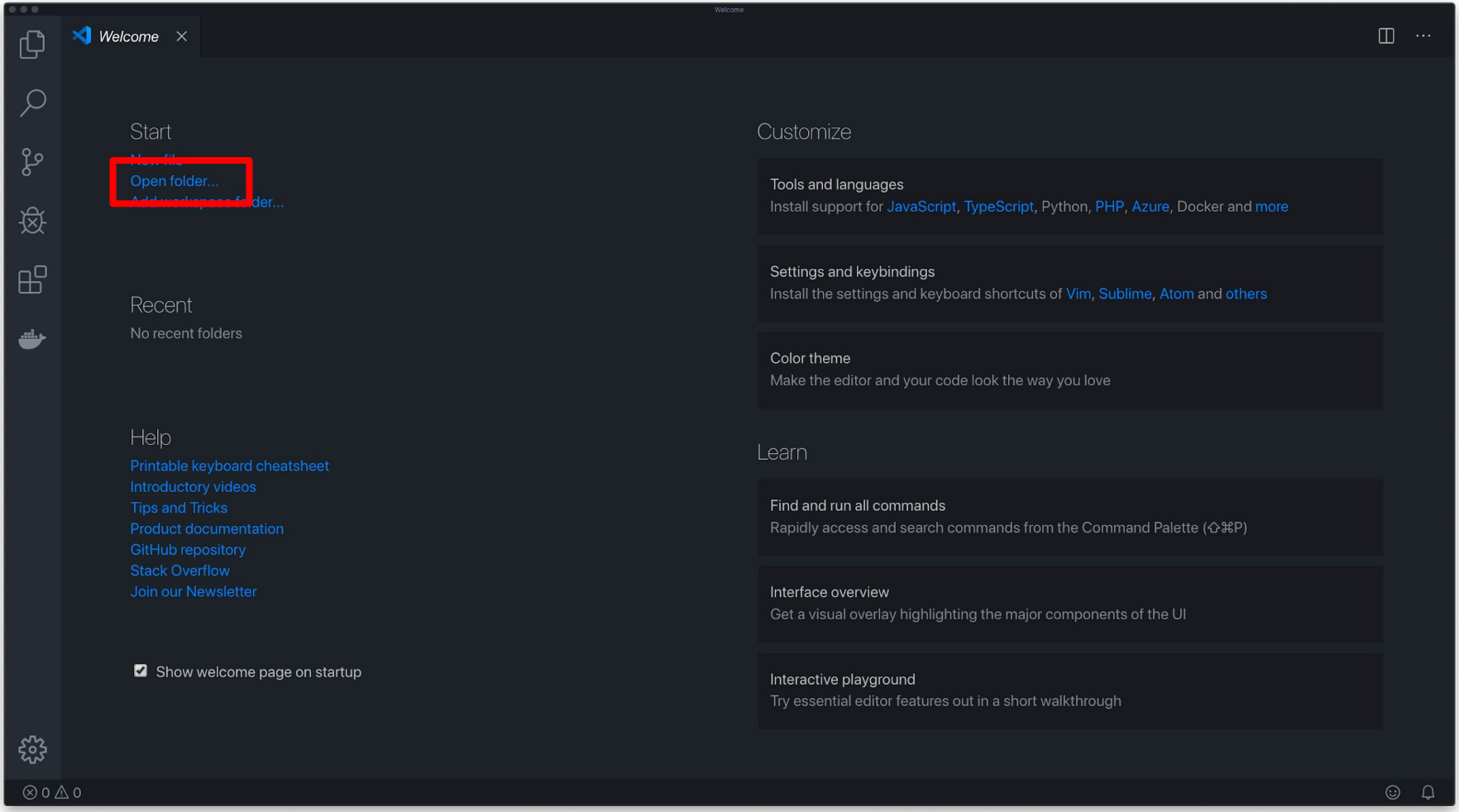

```
python -m compas_rhino.install
```

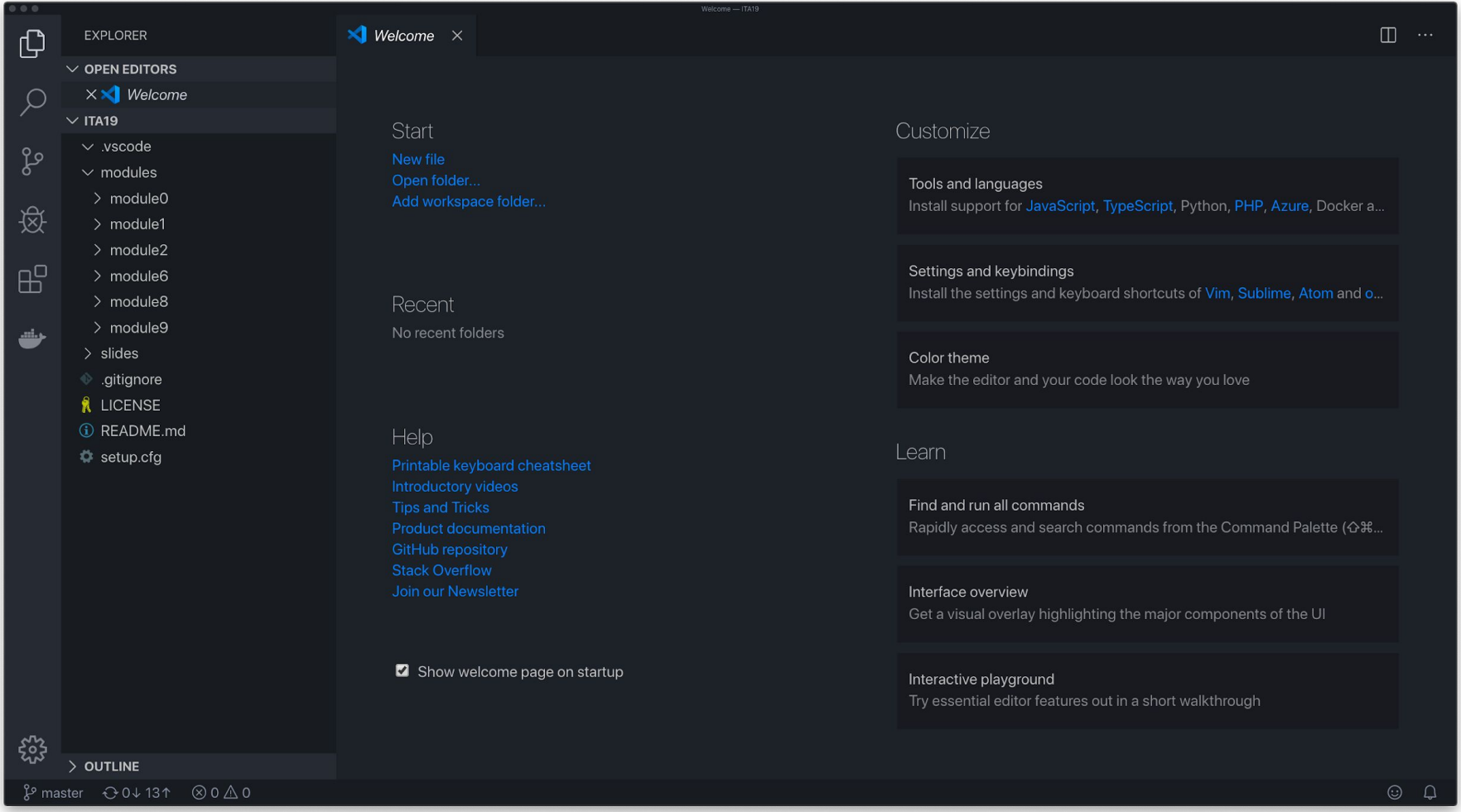
Update ITA19 repo

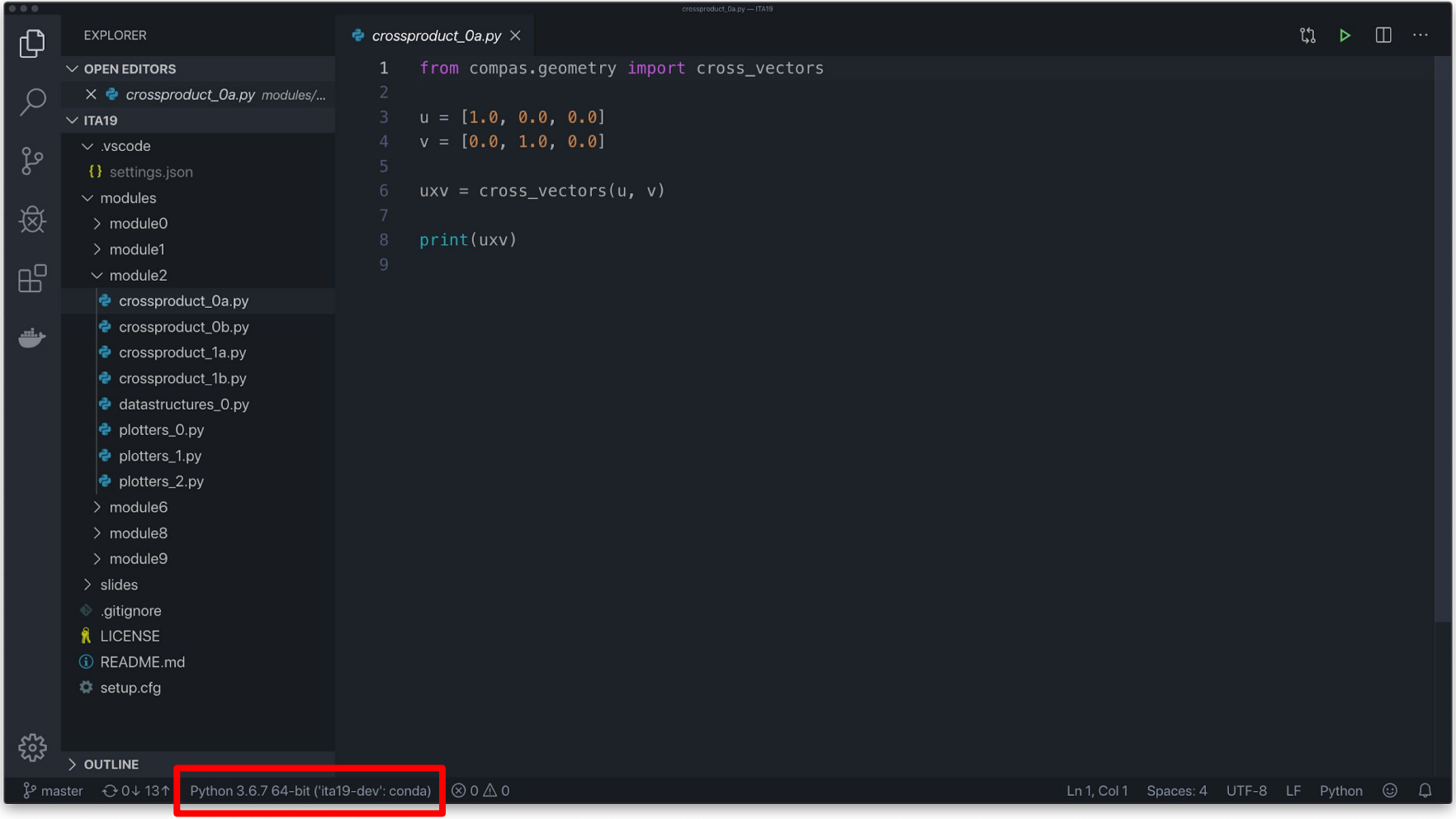

ITA19 (Git)
Commit
Pull
Fetch
Branch
Merge
Stash
Show in Finder
Terminal
Settings
WORKSPACE
File status
History
Search
BRANCHES
dev
master
Week3
TAGS
REMOTES
origin
dev
HEAD
mas_week
master
Week3
STASHES
SUBMODULES
SUBTREES
All Branches
Show Remote Branches
Ancestor Order
Jump to:
Description
Commit
Author
Date
master origin/master origin/HEAD almost there :) 3f8da83 Tom Van Mele <vanmelet@ethz.ch> Today at 11:41 AM
placeholders for first weeks d6b3cb4 Tom Van Mele <vanmelet@ethz.ch> Today at 1:29 AM
Merge branch 'Week3' 4f3c2ab Tom Van Mele <vanmelet@ethz.ch> Today at 1:27 AM
origin/Week3 Week3 cleaning up e48246a Tom Van Mele <vanmelet@ethz.ch> Today at 1:26 AM
moved stuff around 20e907c Tom Van Mele <vanmelet@ethz.ch> Yesterday at 8:45 PM
slide images 83b437d Tom Van Mele <vanmelet@ethz.ch> Yesterday at 3:00 PM
selected assignments d65be5c Tom Van Mele <vanmelet@ethz.ch> Yesterday at 12:37 PM
ng a text box 8d877e2 brgcode <brg@arch.ethz.ch> Yesterday at 7:56 AM
for the course 45ee12a Tom Van Mele <vanmelet@ethz.ch> Oct 21, 2019 at 11:01 PM
1ec6212 Tom Van Mele <vanmelet@ethz.ch> Oct 21, 2019 at 4:50 PM
6982947 Tom Van Mele <vanmelet@ethz.ch> Oct 21, 2019 at 1:37 AM
50ba6df Tom Van Mele <vanmelet@ethz.ch> Oct 19, 2019 at 8:06 AM
8fb2f9d brgcode <brg@arch.ethz.ch> Oct 16, 2019 at 5:14 PM
e34b5a9 Tom Van Mele <vanmelet@ethz.ch> Oct 15, 2019 at 11:38 PM
e1cc6b3 Gonzalo Casas <casas@arch.ethz.ch> Yesterday at 6:58 PM
4478b74 Gonzalo Casas <casas@arch.ethz.ch> Yesterday at 6:29 PM
Update Path planning.ipynb 124a9a6 Romana Rust <rust@arch.ethz.ch> Yesterday at 6:07 PM
Update notebooks after session 5f0f6e1 Gonzalo Casas <casas@arch.ethz.ch> Yesterday at 6:10 PM
update cc440ff Romana Rust <rust@arch.ethz.ch> Yesterday at 11:48 AM
Merge branch 'mas_week' 8491fff Romana Rust <rust@arch.ethz.ch> Oct 21, 2019 at 12:15 PM
origin/mas_week update 00938c3 Romana Rust <rust@arch.ethz.ch> Oct 21, 2019 at 12:12 PM
Pull
Sorted by path
Search
Filename
Path
crossproduct_1a.py modules/module2
crossproduct_1b.py modules/module2
crossproduct_2a.py modules/module2
crossproduct_2b.py modules/module2
datastructures_0.py modules/module2
datastructures_0a.py modules/module2
datastructures_0b.py modules/module2
datastructures_1a.py modules/module2
datastructures_1b.py modules/module2
datastructures_2a.py modules/module2
datastructures_2b.py modules/module2
pca.py modules/module2
pca_0.py modules/module2
viz_0.py modules/module2
viz_1.py modules/module2
viz_2.py modules/module2
viz_3.py modules/module2
modules/module2/crossproduct_1a.py
Hunk 1 : Lines 1-13
Reverse hunk
from compas.geometry import cross_vectors
- from compas.geometry import length_vector
+ from compas.geometry import angle_vectors
u = [1.0, 0.0, 0.0]
v = [0.0, 1.0, 0.0]
uxv = cross_vectors(u, v)
- print(length_vector(uxv))
+ u_uxv = angle_vectors(u, uxv)
+ v_uxv = angle_vectors(v, uxv)
+
+ print(u_uxv)
+ print(v_uxv)

VS Code

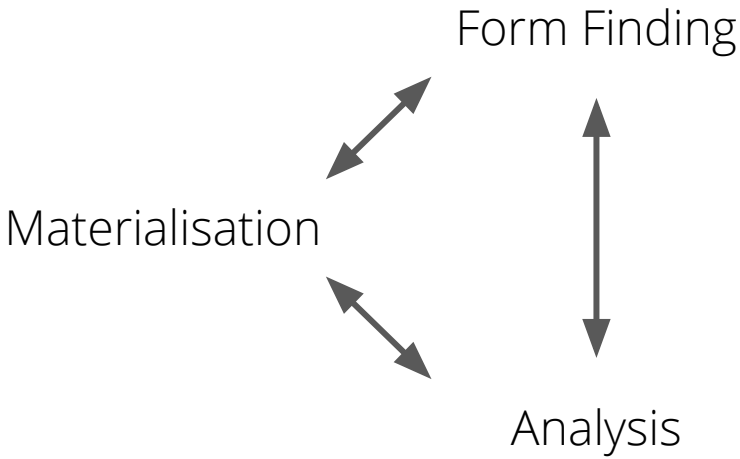

Welcome
Welcome
Start
New file
Open folder...
Add workspace folder...
Recent
No recent folders
Help
Printable keyboard cheatsheet
Introductory videos
Tips and Tricks
Product documentation
GitHub repository
Stack Overflow
Join our Newsletter
Show welcome page on startup
Customize
Tools and languages
Install support for JavaScript, TypeScript, Python, PHP, Azure, Docker and more
Settings and keybindings
Install the settings and keyboard shortcuts of Vim, Sublime, Atom and others
Color theme
Make the editor and your code look the way you love
Learn
Find and run all commands
Rapidly access and search commands from the Command Palette (⇧⌘P)
Interface overview
Get a visual overlay highlighting the major components of the UI
Interactive playground
Try essential editor features out in a short walkthrough
0 0

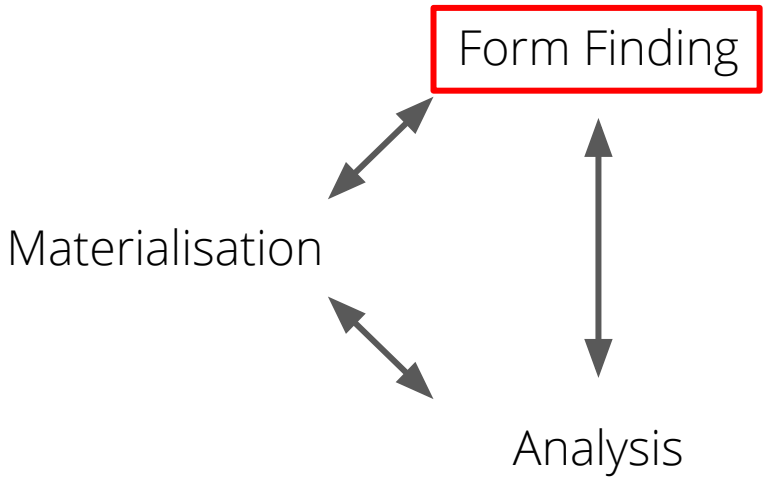

Welcome
Welcome
Start
Open folder...
Recent
No recent folders
Help
Printable keyboard cheatsheet
Introductory videos
Tips and Tricks
Product documentation
GitHub repository
Stack Overflow
Join our Newsletter
Show welcome page on startup
Customize
Tools and languages
Install support for JavaScript, TypeScript, Python, PHP, Azure, Docker and more
Settings and keybindings
Install the settings and keyboard shortcuts of Vim, Sublime, Atom and others
Color theme
Make the editor and your code look the way you love
Learn
Find and run all commands
Rapidly access and search commands from the Command Palette (⇧⌘P)
Interface overview
Get a visual overlay highlighting the major components of the UI
Interactive playground
Try essential editor features out in a short walkthrough
0 0


Welcome — ITA19
EXPLORER
OPEN EDITORS
Welcome
ITA19
.vscode
modules
module0
module1
module2
module6
module8
module9
slides
.gitignore
LICENSE
README.md
setup.cfg
OUTLINE
Welcome
Start
New file
Open folder...
Add workspace folder...
Recent
No recent folders
Help
Printable keyboard cheatsheet
Introductory videos
Tips and Tricks
Product documentation
GitHub repository
Stack Overflow
Join our Newsletter
Show welcome page on startup
Customize
Tools and languages
Install support for JavaScript, TypeScript, Python, PHP, Azure, Docker a...
Settings and keybindings
Install the settings and keyboard shortcuts of Vim, Sublime, Atom and o...
Color theme
Make the editor and your code look the way you love
Learn
Find and run all commands
Rapidly access and search commands from the Command Palette (⇧⌘...
Interface overview
Get a visual overlay highlighting the major components of the UI
Interactive playground
Try essential editor features out in a short walkthrough
master
0↓ 13↑
0 0


crossproduct_0a.py — ITA19
EXPLORER
OPEN EDITORS
crossproduct_0a.py modules/...
ITA19
.vscode
settings.json
modules
module0
module1
module2
crossproduct_0a.py
crossproduct_0b.py
crossproduct_1a.py
crossproduct_1b.py
datastructures_0.py
plotters_0.py
plotters_1.py
plotters_2.py
module6
module8
module9
slides
.gitignore
LICENSE
README.md
setup.cfg
OUTLINE
crossproduct_0a.py
1 from compas.geometry import cross_vectors
2
3 u = [1.0, 0.0, 0.0]
4 v = [0.0, 1.0, 0.0]
5
6 uxv = cross_vectors(u, v)
7
8 print(uxv)
9
master
0↓ 13↑
Python 3.6.7 64-bit ('ita19-dev': conda)
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
LF
Python

Module 1: Structural Design


Form Finding
Materialisation
Analysis


Form Finding
Materialisation
Analysis

compas_ags

Algebraic Graph Statics

compas_tna

Thrust Network Analysis

compas_3gs

3D Graphic Statics

compas_fea

Finite Element Analysis

compas_fab

Robotic Fabrication

compas_vol

Volumetric modelling

compas_knit

Spatial Knitting

compas_dem

Discrete Element Modeling (3DEC)

compas_dr6

6DOF Dynamic Relaxation

compas_rbe

Rigid Block Equilibrium

compas_ml

Machine Learning

compas_fofin

Form Finding

compas_prd

Piecewise Rigid Displacements

compas_assembly

Discrete element assemblies

compas_pattern

Mesh Topology Finding

compas_ags

Algebraic Graph Statics

compas_tna

Thrust Network Analysis

compas_3gs

3D Graphic Statics

compas_fea

Finite Element Analysis

compas_fab

Robotic Fabrication

compas_vol

Volumetric modelling

compas_knit

Spatial Knitting

compas_dem

Discrete Element Modeling (3DEC)

compas_dr6

6DOF Dynamic Relaxation

compas_rbe

Rigid Block Equilibrium

compas_ml

Machine Learning

compas_fofin

Form Finding

compas_prd

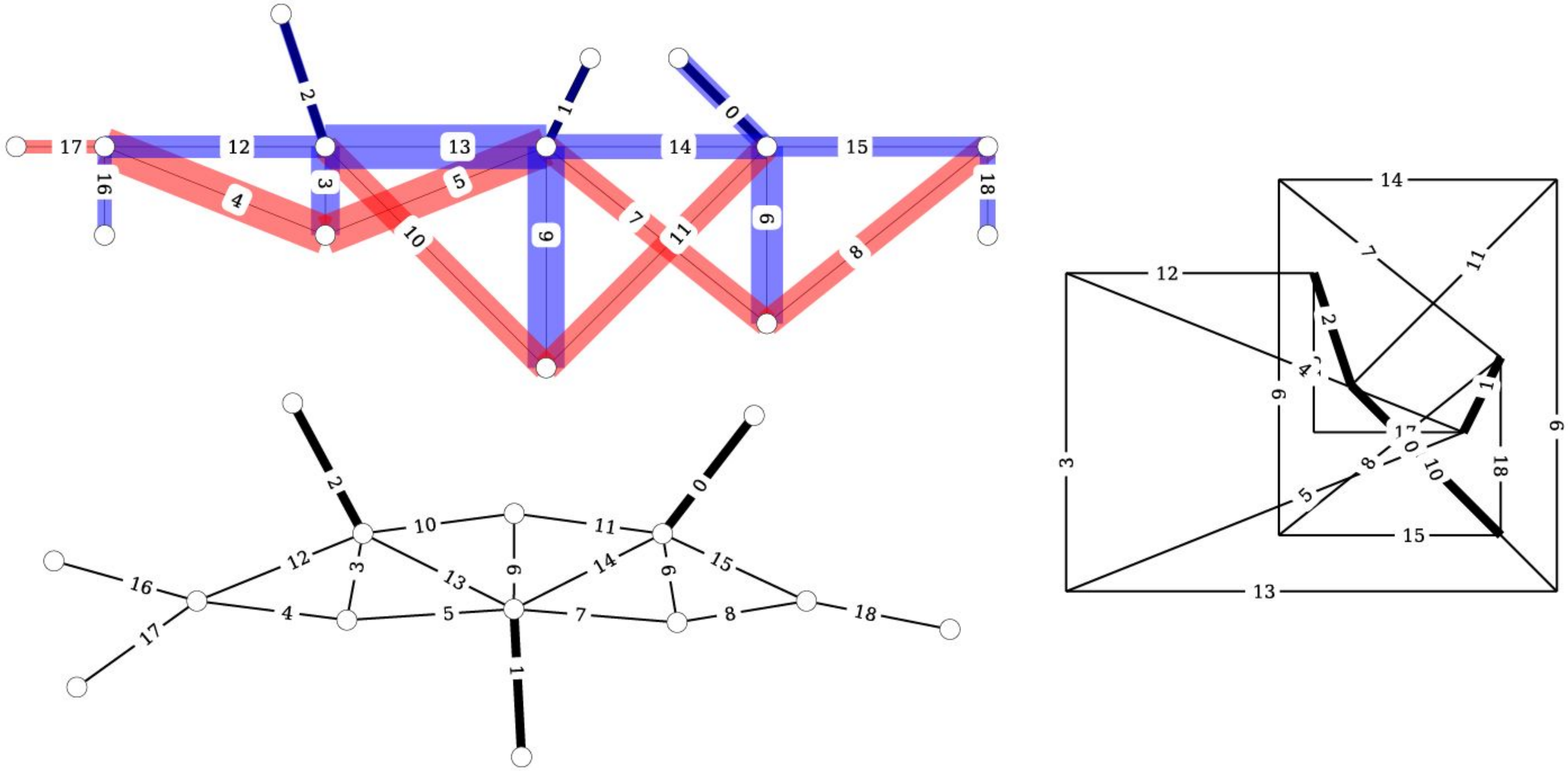
Piecewise Rigid Displacements

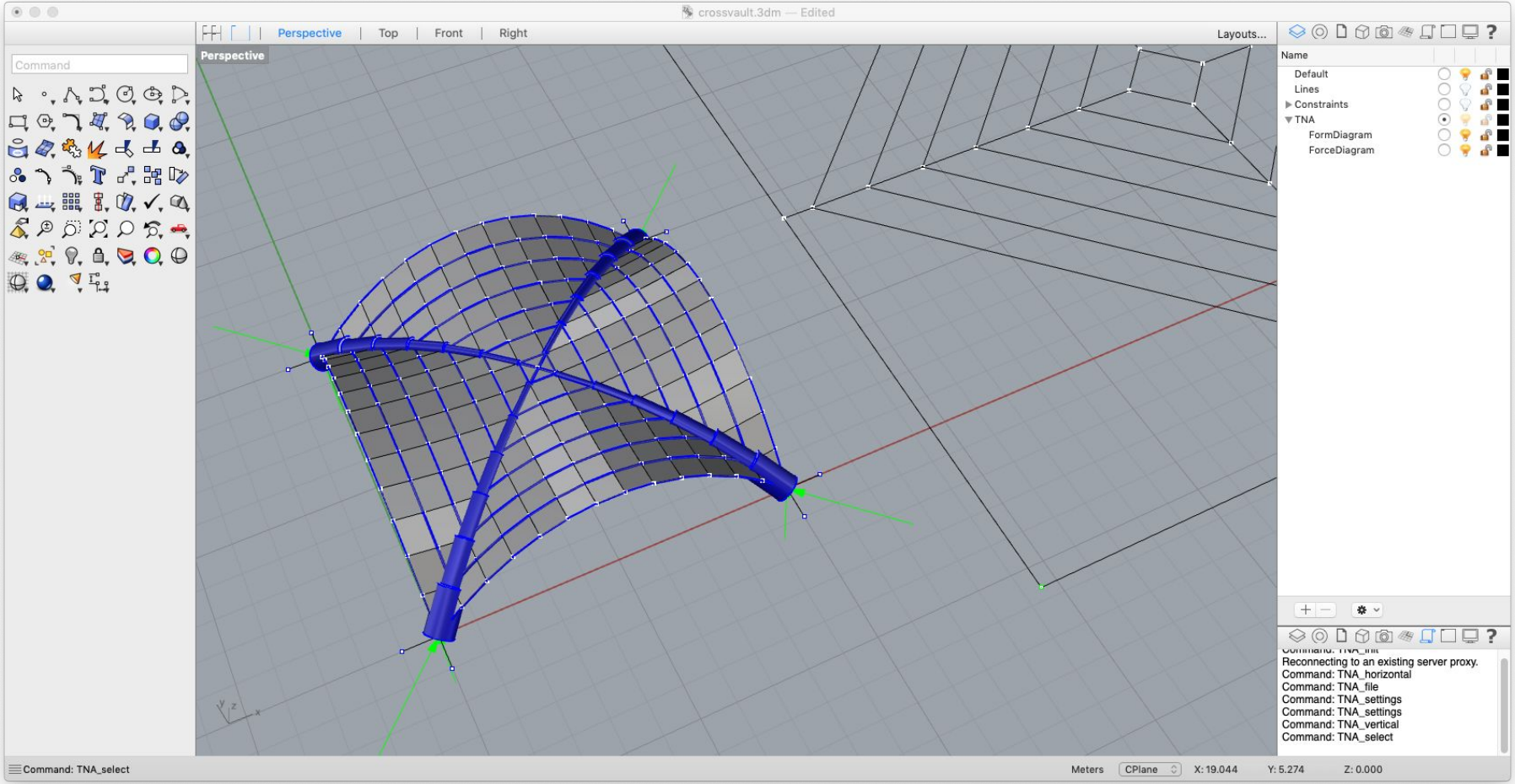
compas_assembly

Discrete element assemblies

compas_pattern

Mesh Topology Finding




crossvault.3dm — Edited
Perspective
Top
Front
Right
Layouts...
Perspective
Command
Name
Default
Lines
Constraints
TNA
FormDiagram
ForceDiagram
Reconnecting to an existing server proxy.
Command: TNA_horizontal
Command: TNA_file
Command: TNA_settings
Command: TNA_settings
Command: TNA_vertical
Command: TNA_select
Command: TNA_select
Meters
CPlane
X: 19.044
Y: 5.274
Z: 0.000

compas_3gs 0.2.0

Search docs

- Introduction
- Getting started
- Overview
- Theoretical background
- **User manual**
  - Polyhedral cell
  - **Multi-cell polyhedron**
    - Datastructure
    - Interpreting diagrams
    - Constructor
    - Duality
    - Reciprocation
    - Planarisation
    - **Arearisation**
    - Modification
    - Transformation
    - Visualisation
    - Unified diagram
  - Cell network
  - Materialisation
- API Reference
- License
- Citing
- Publications
- How to contribute

Arearisation

Arearisation of a `volmesh` is implemented as a special case of the planarisation algorithm. In addition to the planarisation of the faces by projection at each iteration step, they can be re-sized to match target areas.

Example

In this example, four randomly chosen faces are given new target areas. The initial normals of all of the faces are constrained to remain unchanged.

https://compas-dev.github.io/compas_3gs/

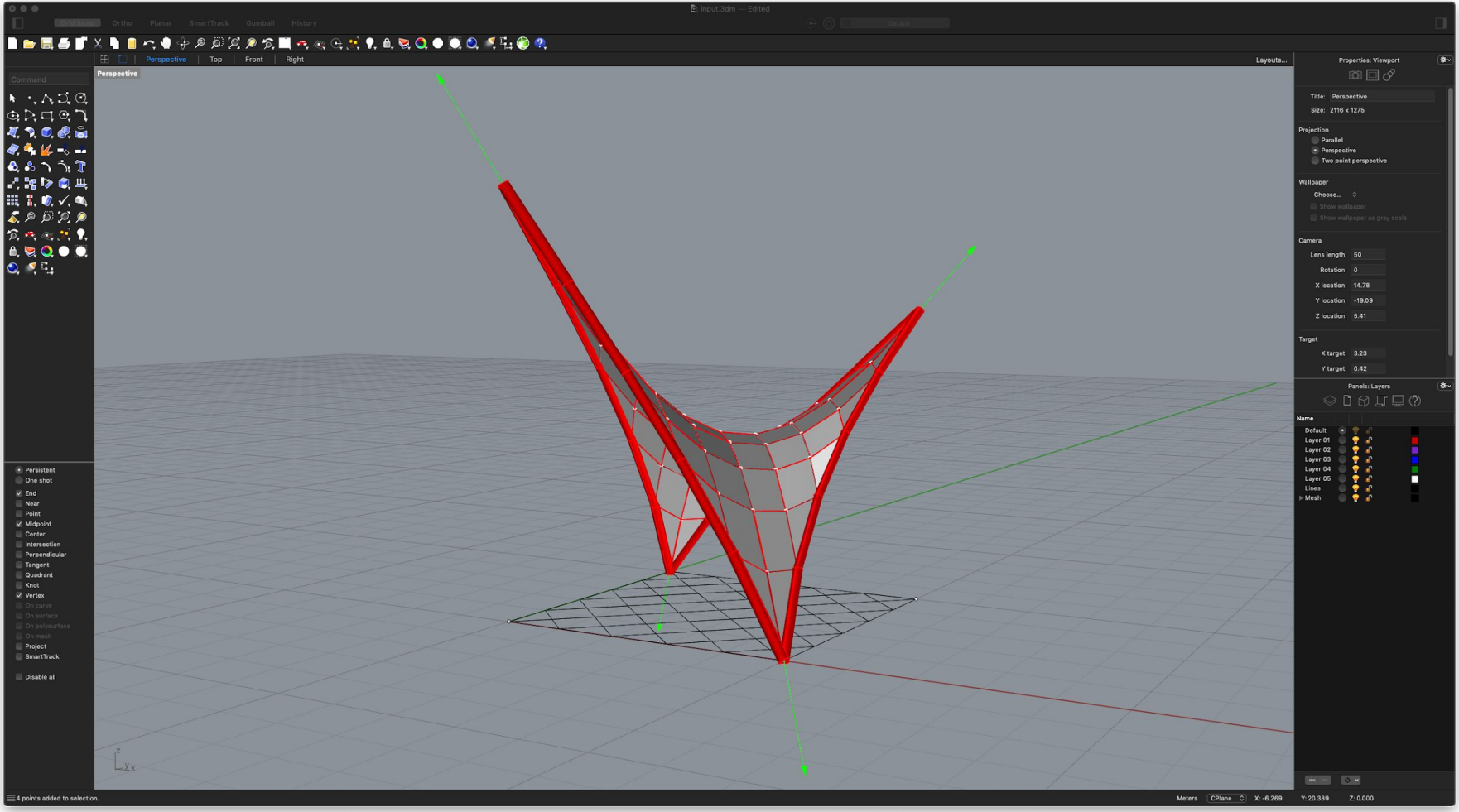
Downloads

- volmesh_mat.3dm

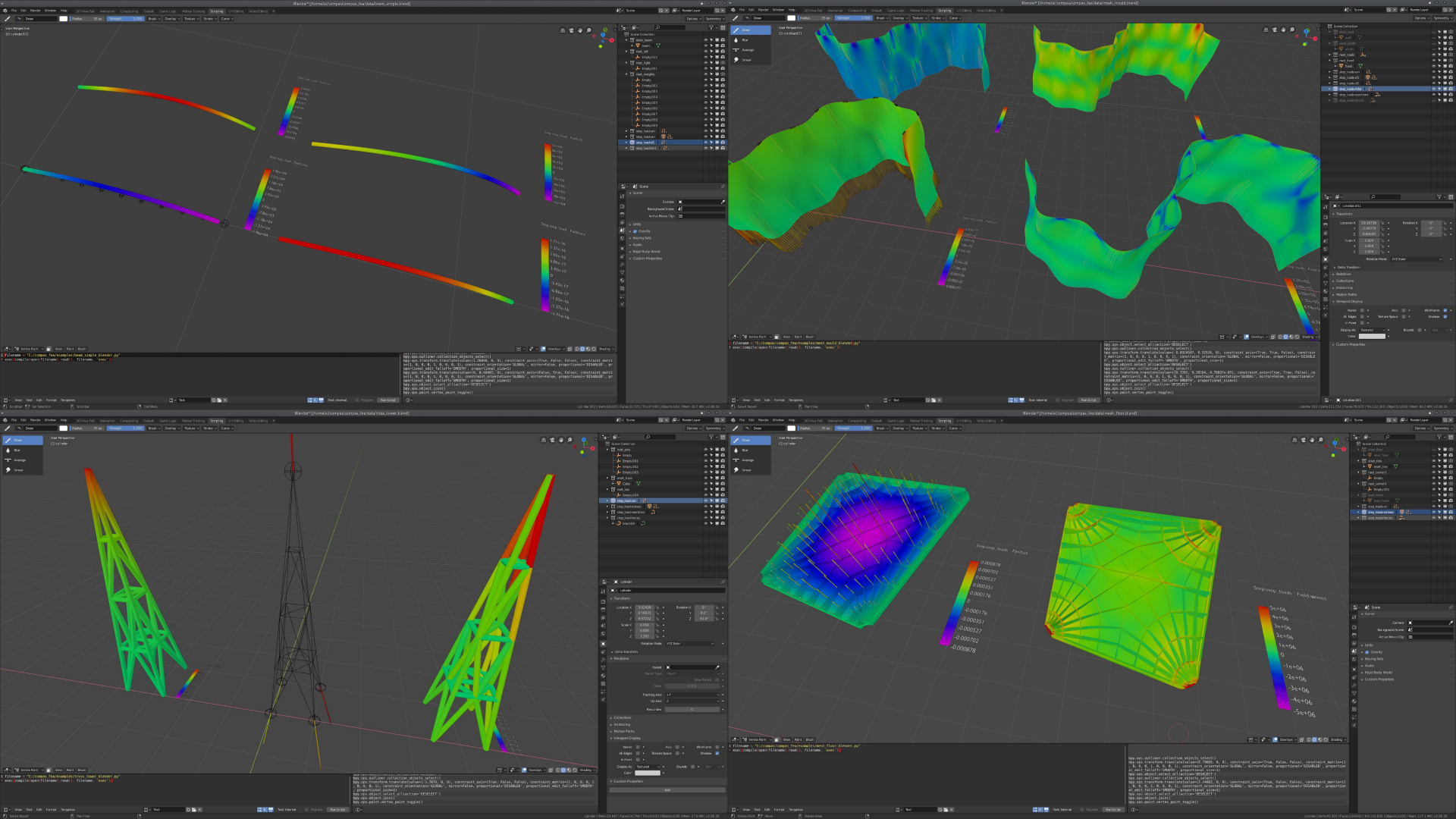
```
from __future__ import absolute_import
from __future__ import print_function
from __future__ import division

import compas
```

Arearisation
Example

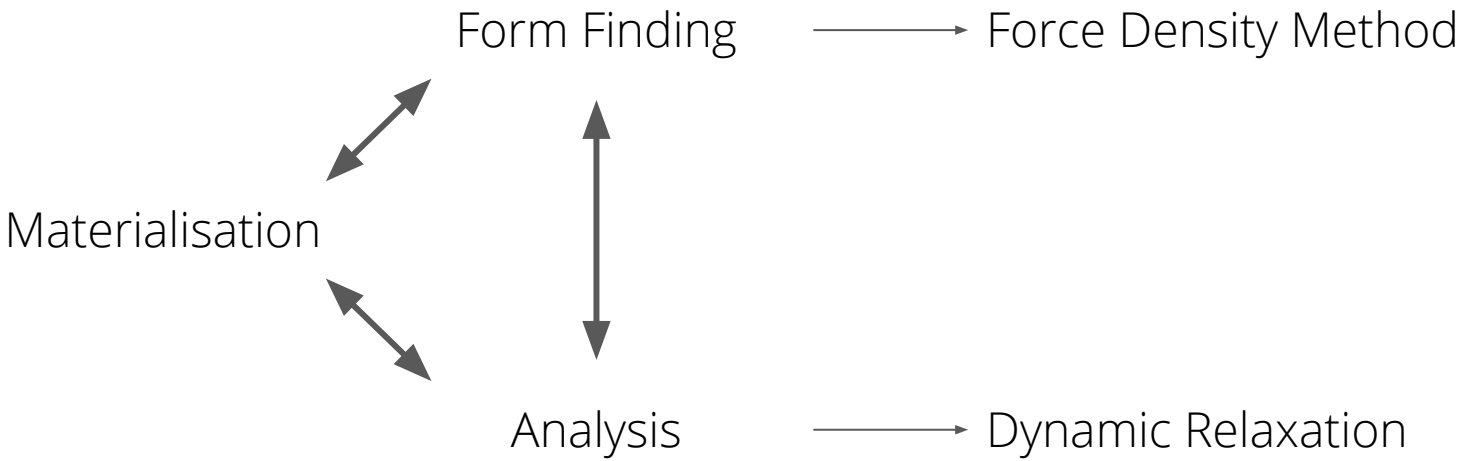
compas_ags Algebraic Graph Statics	**compas_tna** Thrust Network Analysis	**compas_3gs** 3D Graphic Statics	**compas_fea** Finite Element Analysis	**compas_fab** Robotic Fabrication
compas_vol Volumetric modelling	**compas_knit** Spatial Knitting	**compas_dem** Discrete Element Modeling (3DEC)	**compas_dr6** 6DOF Dynamic Relaxation	**compas_rbe** Rigid Block Equilibrium
compas_ml Machine Learning	**compas_fofin** Form Finding	**compas_prd** Piecewise Rigid Displacements	**compas_assembly** Discrete element assemblies	**compas_pattern** Mesh Topology Finding

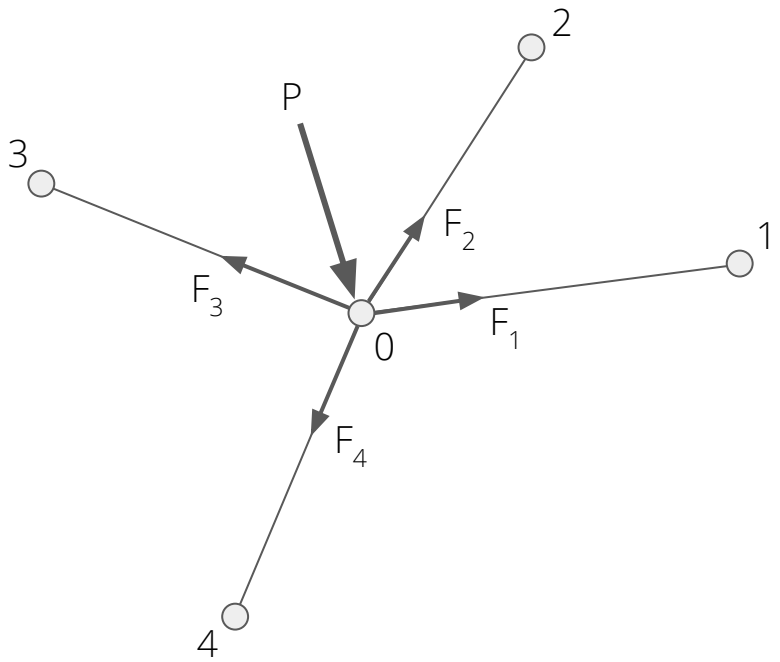

input.3dm — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Command
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.78
Y location: -19.09
Z location: 5.41
Target
X target: 3.23
Y target: 0.42
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Lines
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
4 points added to selection.
Meters
CPlane
X: -6.269
Y: 20.389
Z: 0.000

compas_ags Algebraic Graph Statics	**compas_tna** Thrust Network Analysis	**compas_3gs** 3D Graphic Statics	**compas_fea** Finite Element Analysis	**compas_fab** Robotic Fabrication
compas_vol Volumetric modelling	**compas_knit** Spatial Knitting	**compas_dem** Discrete Element Modeling (3DEC)	**compas_dr6** 6DOF Dynamic Relaxation	**compas_rbe** Rigid Block Equilibrium
compas_ml Machine Learning	**compas_fofin** Form Finding	**compas_prd** Piecewise Rigid Displacements	**compas_assembly** Discrete element assemblies	**compas_pattern** Mesh Topology Finding



compas_fofin

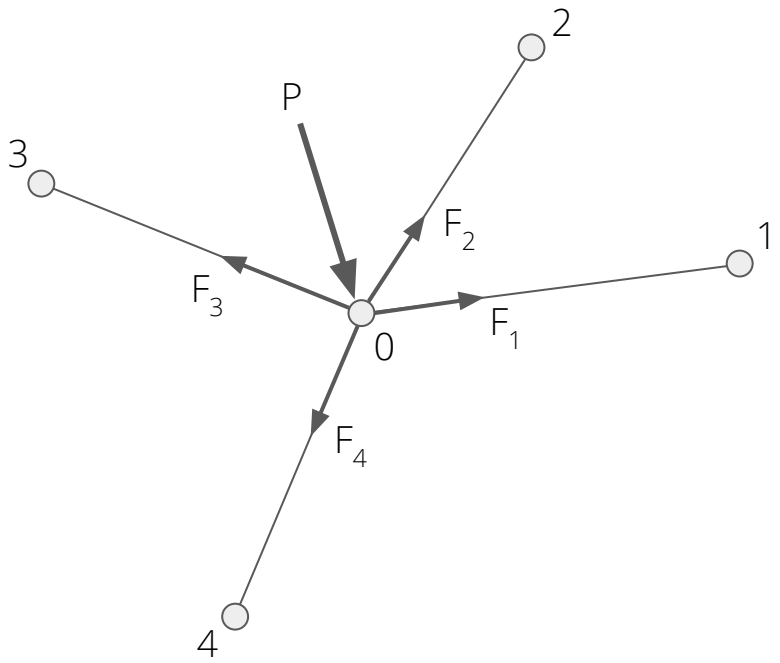

Form Finding
Force Density Method
Materialisation
Analysis
Dynamic Relaxation



$$F_{1,x} + F_{2,x} + F_{3,x} + F_{4,x} = P_{0,x}$$

$$F_{1,y} + F_{2,y} + F_{3,y} + F_{4,y} = P_{0,y}$$

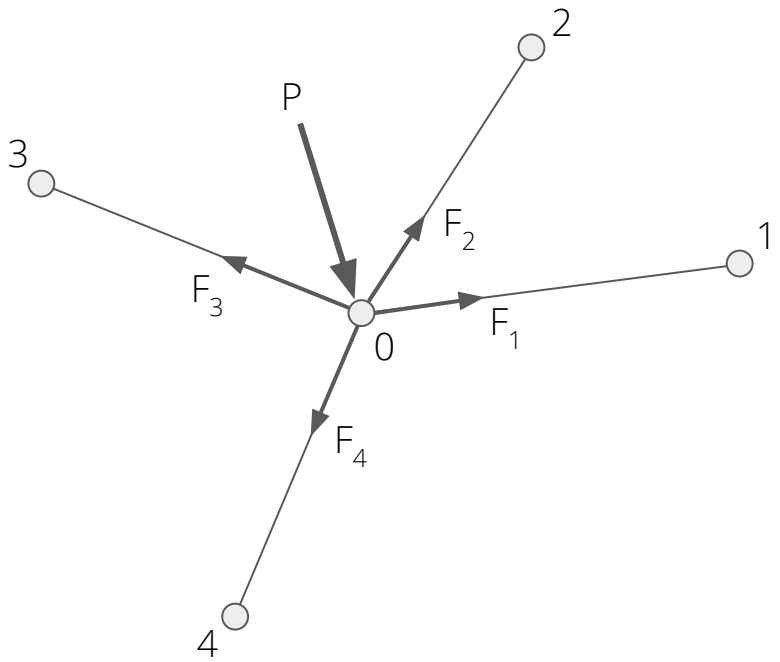
$$F_{1,z} + F_{2,z} + F_{3,z} + F_{4,z} = P_{0,z}$$



$$\sum_{j=1}^{4} F_{j,x} = P_{0,x}$$

$$\sum_{j=1}^{4} F_{j,y} = P_{0,y}$$

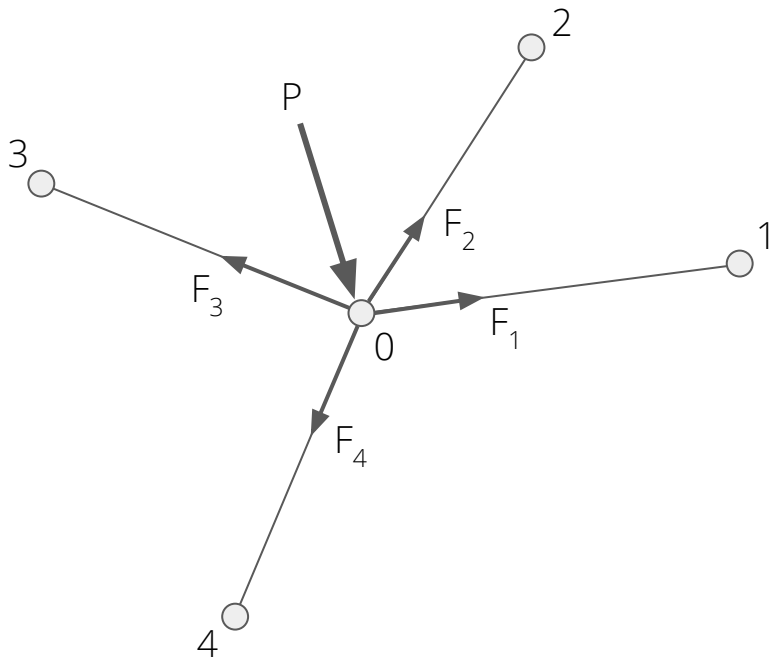
$$\sum_{j=1}^{4} F_{j,z} = P_{0,z}$$



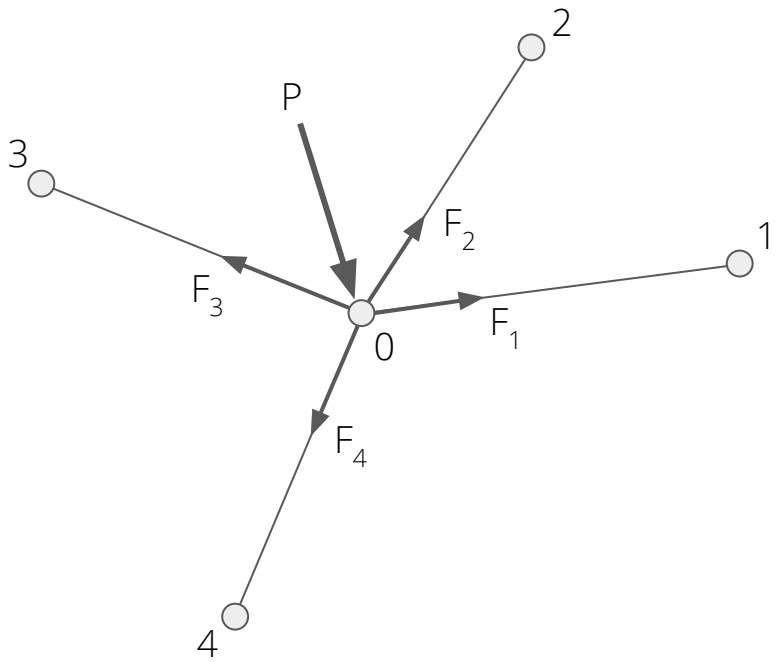
$$\sum_{j=1}^{4} \frac{F_j}{L_j}(x_j - x_0) = P_{0,x}$$

$$\sum_{j=1}^{4} \frac{F_j}{L_j}(y_j - y_0) = P_{0,y}$$

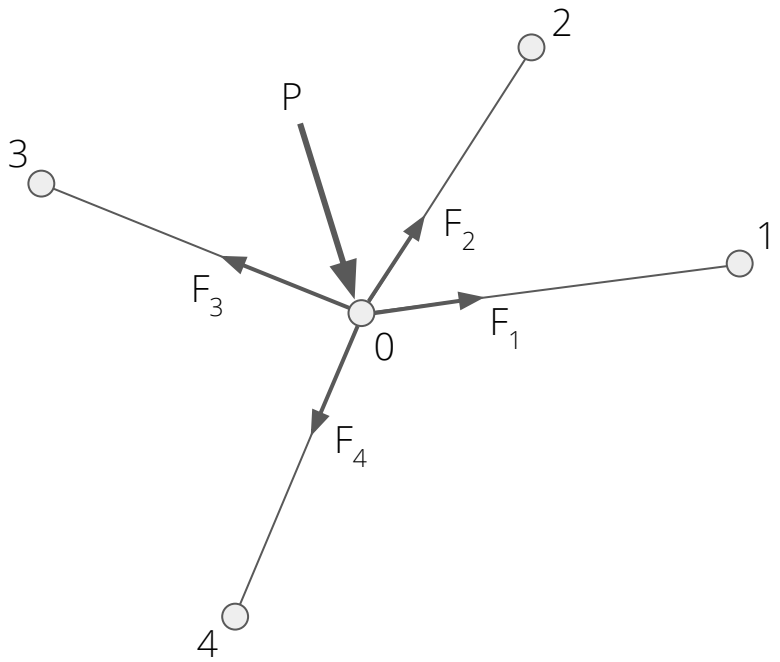
$$\sum_{j=1}^{4} \frac{F_j}{L_j}(z_j - z_0) = P_{0,z}$$



$$L_j = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2}$$



$$Q_j = \frac{F_j}{L_j}$$



$$\sum_{j=1}^{4} Q_j(x_j - x_0) = P_{x,0}$$

$$\sum_{j=1}^{4} Q_j(y_j - y_0) = P_{y,0}$$

$$\sum_{j=1}^{4} Q_j(z_j - z_0) = P_{z,0}$$

```
from compas.numerical import fd_numpy
```

RightClick
+
Go to Definition

```python
from compas.numerical import fd_numpy
```

RightClick
+
Go to Definition

```python
def fd_numpy(vertices, edges, fixed, q, loads):
    v    = len(vertices)
    free = list(set(range(v)) - set(fixed))
    xyz  = asarray(vertices, dtype=float).reshape((-1, 3))
    q    = asarray(q, dtype=float).reshape((-1, 1))
    p    = asarray(loads, dtype=float).reshape((-1, 3))
    C    = connectivity_matrix(edges, 'csr')
    Ci   = C[:, free]
    Cf   = C[:, fixed]
    Ct   = C.transpose()
    Cit  = Ci.transpose()
    Q    = diags([q.flatten()], [0])
    A    = Cit.dot(Q).dot(Ci)
    b    = p[free] - Cit.dot(Q).dot(Cf).dot(xyz[fixed])

    xyz[free] = spsolve(A, b)

    l = normrow(C.dot(xyz))
    f = q * l
    r = p - Ct.dot(Q).dot(C).dot(xyz)

    return xyz, q, f, l, r
```

```python
from compas.numerical import fd_numpy
```

RightClick
+
Go to Definition

```python
def fd_numpy(vertices, edges, fixed, q, loads):
    v    = len(vertices)
    free = list(set(range(v)) - set(fixed))
    xyz  = asarray(vertices, dtype=float).reshape((-1, 3))
    q    = asarray(q, dtype=float).reshape((-1, 1))
    p    = asarray(loads, dtype=float).reshape((-1, 3))
    C    = connectivity_matrix(edges, 'csr')
    Ci   = C[:, free]
    Cf   = C[:, fixed]
    Ct   = C.transpose()
    Cit  = Ci.transpose()
    Q    = diags([q.flatten()], [0])
    A    = Cit.dot(Q).dot(Ci)
    b    = p[free] - Cit.dot(Q).dot(Cf).dot(xyz[fixed])

    xyz[free] = spsolve(A, b)

    l = normrow(C.dot(xyz))
    f = q * l
    r = p - Ct.dot(Q).dot(C).dot(xyz)

    return xyz, q, f, l, r
```

```python
from compas.numerical import fd_numpy
```

RightClick
+
Go to Definition

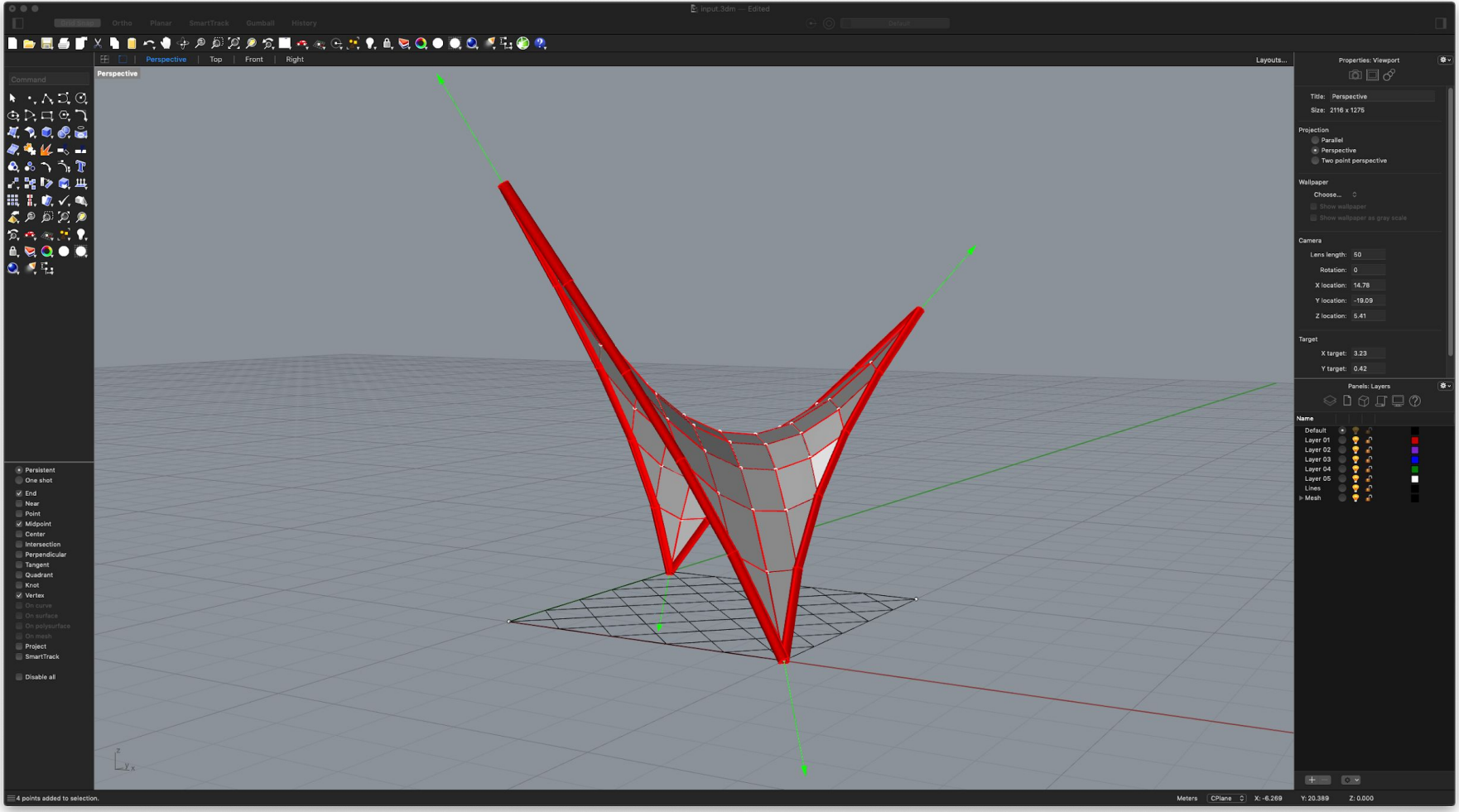
```python
def fd_numpy(vertices, edges, fixed, q, loads):
    v    = len(vertices)
    free = list(set(range(v)) - set(fixed))
    xyz  = asarray(vertices, dtype=float).reshape((-1, 3))
    q    = asarray(q, dtype=float).reshape((-1, 1))
    p    = asarray(loads, dtype=float).reshape((-1, 3))
    C    = connectivity_matrix(edges, 'csr')
    Ci   = C[:, free]
    Cf   = C[:, fixed]
    Ct   = C.transpose()
    Cit  = Ci.transpose()
    Q    = diags([q.flatten()], [0])
    A    = Cit.dot(Q).dot(Ci)
    b    = p[free] - Cit.dot(Q).dot(Cf).dot(xyz[fixed])

    xyz[free] = spsolve(A, b)

    l = normrow(C.dot(xyz))
    f = q * l
    r = p - Ct.dot(Q).dot(C).dot(xyz)

    return xyz, q, f, l, r
```

Implementation

1_fofin_simple.py 2_fofin_custom.py

3_fofin_input.py 4_fofin_output.py

5_fofin_interactive.py


input.3dm — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Command
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.78
Y location: -19.09
Z location: 5.41
Target
X target: 3.23
Y target: 0.42
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Lines
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
4 points added to selection.
Meters
CPlane
X: -6.269
Y: 20.389
Z: 0.000

1_fofin_simple.py

```python
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division

import os

from compas.datastructures import Mesh
from compas.geometry import add_vectors
from compas.geometry import scale_vector

import compas_rhino
from compas_rhino.artists import MeshArtist

# ==============================================================================
# Proxy
# ==============================================================================

from compas.rpc import Proxy
numerical = Proxy('compas.numerical')
fd_numpy = numerical.fd_numpy
```

```python
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division

import os

from compas.datastructures import Mesh
from compas.geometry import add_vectors
from compas.geometry import scale_vector

import compas_rhino
from compas_rhino.artists import MeshArtist

# ==============================================================================
# Proxy
# ==============================================================================

from compas.rpc import Proxy
numerical = Proxy('compas.numerical')
fd_numpy = numerical.fd_numpy
```

```
# ==============================================================================
# Input file
# ==============================================================================

HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'faces.obj')

# ==============================================================================
# Mesh
# ==============================================================================

mesh = Mesh.from_obj(FILE)

mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})
```

```
# ==============================================================================
# Input file
# ==============================================================================

HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'faces.obj')

# ==============================================================================
# Mesh
# ==============================================================================

mesh = Mesh.from_obj(FILE)

mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})
```

```
# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = list(mesh.vertices_where({'vertex_degree': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', 7.0, keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = list(mesh.vertices_where({'vertex_degree': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', 7.0, keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = list(mesh.vertices_where({'vertex_degree': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', 7.0, keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
# ==============================================================================
# FoFin input
# ==============================================================================

xyz = mesh.get_vertices_attributes('xyz')
fixed = list(mesh.vertices_where({'is_fixed': True}))
loads = mesh.get_vertices_attributes(('px', 'py', 'pz'))

edges = list(mesh.edges())
q = mesh.get_edges_attribute('q')

# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
```

```
# ==============================================================================
# FoFin input
# ==============================================================================

xyz = mesh.get_vertices_attributes('xyz')
fixed = list(mesh.vertices_where({'is_fixed': True}))
loads = mesh.get_vertices_attributes(('px', 'py', 'pz'))

edges = list(mesh.edges())
q = mesh.get_edges_attribute('q')

# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
```

```
# ==============================================================================
# FoFin input
# ==============================================================================

xyz = mesh.get_vertices_attributes('xyz')
fixed = list(mesh.vertices_where({'is_fixed': True}))
loads = mesh.get_vertices_attributes(('px', 'py', 'pz'))

edges = list(mesh.edges())
q = mesh.get_edges_attribute('q')

# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
```

```
# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)

# ==============================================================================
# Fofin update
# ==============================================================================

for key, attr in mesh.vertices(True):
    attr['x'] = xyz[key][0]
    attr['y'] = xyz[key][1]
    attr['z'] = xyz[key][2]
    attr['rx'] = r[key][0]
    attr['ry'] = r[key][1]
    attr['rz'] = r[key][2]

for index, (u, v, attr) in enumerate(mesh.edges(True)):
    attr['f'] = f[index][0]
```

```
# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)

# ==============================================================================
# Fofin update
# ==============================================================================

for key, attr in mesh.vertices(True):
    attr['x'] = xyz[key][0]
    attr['y'] = xyz[key][1]
    attr['z'] = xyz[key][2]
    attr['rx'] = r[key][0]
    attr['ry'] = r[key][1]
    attr['rz'] = r[key][2]

for index, (u, v, attr) in enumerate(mesh.edges(True)):
    attr['f'] = f[index][0]
```

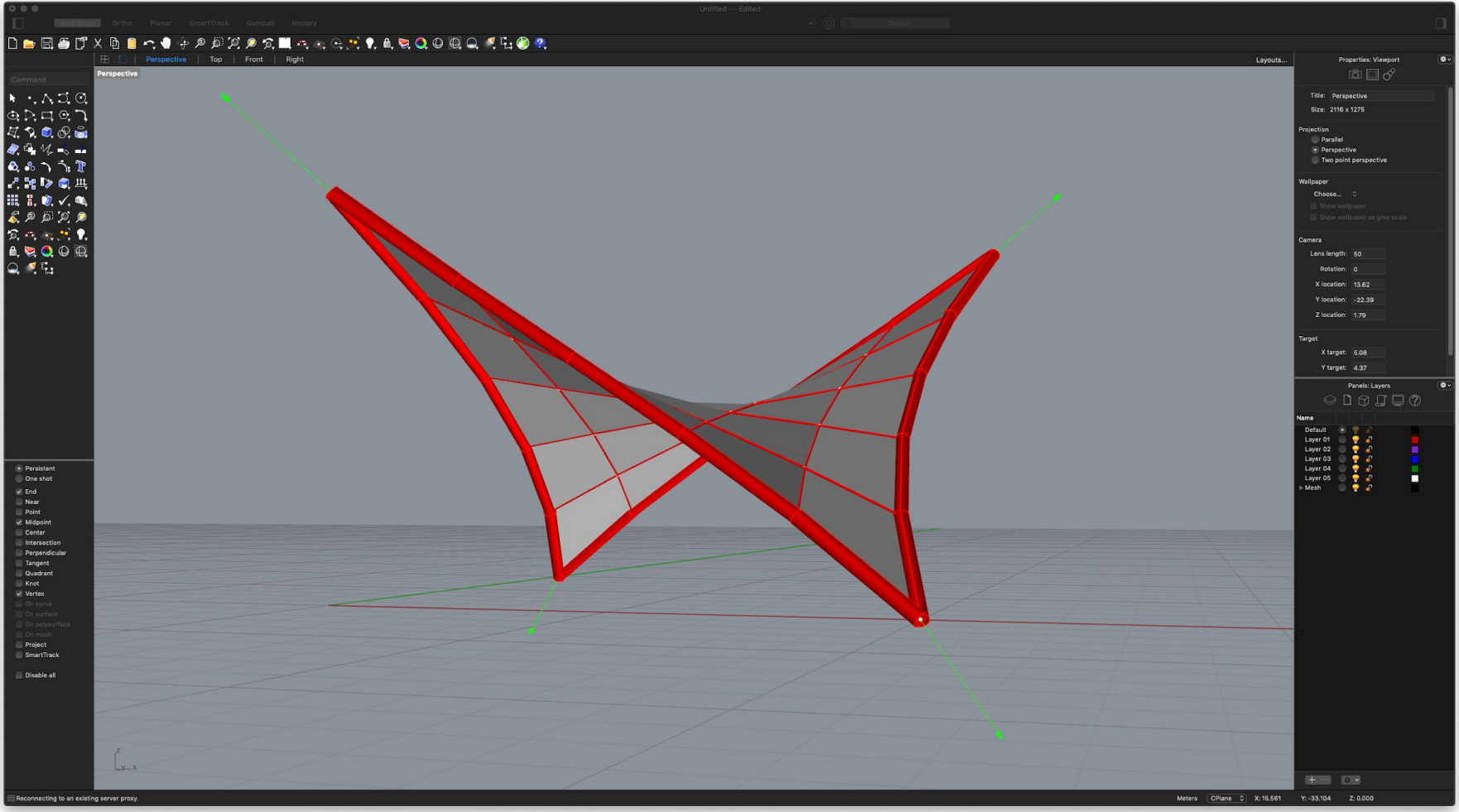
```python
# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)

# ==============================================================================
# Fofin update
# ==============================================================================

for key, attr in mesh.vertices(True):
    attr['x'] = xyz[key][0]
    attr['y'] = xyz[key][1]
    attr['z'] = xyz[key][2]
    attr['rx'] = r[key][0]
    attr['ry'] = r[key][1]
    attr['rz'] = r[key][2]

for index, (u, v, attr) in enumerate(mesh.edges(True)):
    attr['f'] = f[index][0]
```

```python
# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)

# ==============================================================================
# Fofin update
# ==============================================================================

for key, attr in mesh.vertices(True):
    attr['x'] = xyz[key][0]
    attr['y'] = xyz[key][1]
    attr['z'] = xyz[key][2]
    attr['rx'] = r[key][0]
    attr['ry'] = r[key][1]
    attr['rz'] = r[key][2]

for index, (u, v, attr) in enumerate(mesh.edges(True)):
    attr['f'] = f[index][0]
```

```
# ==============================================================================
# Visualize result
# ==============================================================================

artist = MeshArtist(mesh, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
artist.draw_faces()
```

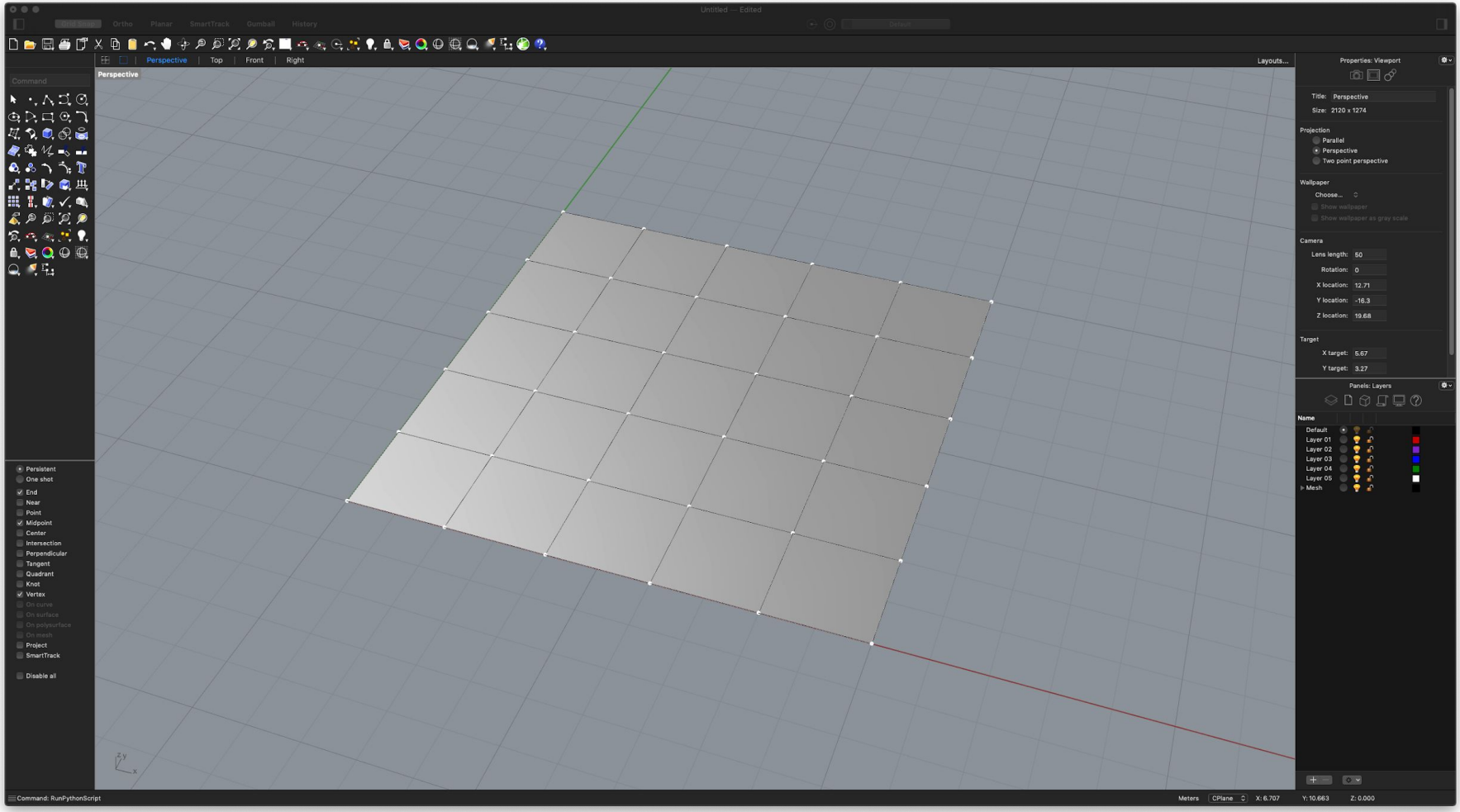
```
forces = []
for u, v, attr in mesh.edges(True):
    force = attr['f']
    start = mesh.vertex_coordinates(u)
    end = mesh.vertex_coordinates(v)
    radius = 0.01 * force
    forces.append({
        'start': start,
        'end': end,
        'radius': radius,
        'color': (255, 0, 0)})

compas_rhino.draw_cylinders(forces, layer="Mesh::Forces", clear=True)
```

```
reactions = []
for key, attr in mesh.vertices_where({'is_fixed': True}, True):
    reaction = [attr['rx'], attr['ry'], attr['rz']]
    vector = scale_vector(reaction, -0.1)
    start = mesh.vertex_coordinates(key)
    end = add_vectors(start, vector)
    reactions.append({
        'start': start,
        'end': end,
        'arrow': 'end',
        'color': (0, 255, 0)})

compas_rhino.draw_lines(reactions, layer="Mesh::Reactions", clear=True)
```


Untitled — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 13.62
Y location: -22.39
Z location: 1.79
Target
X target: 5.08
Y target: 4.37
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
Reconnecting to an existing server proxy.
Meters
CPlane
X: 15.561
Y: -33.104
Z: 0.000

Visualise faces.obj

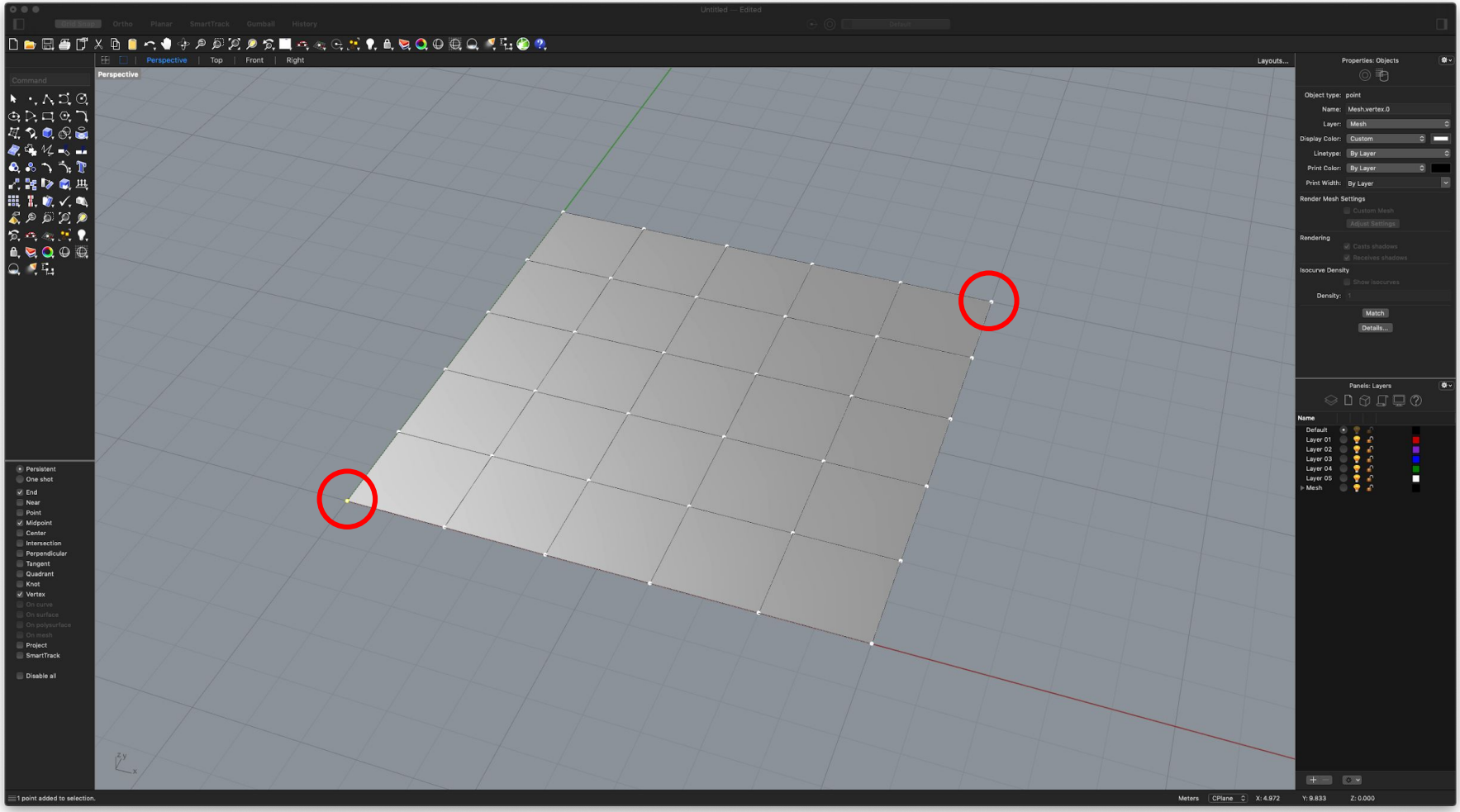
1a_fofin_simple.py

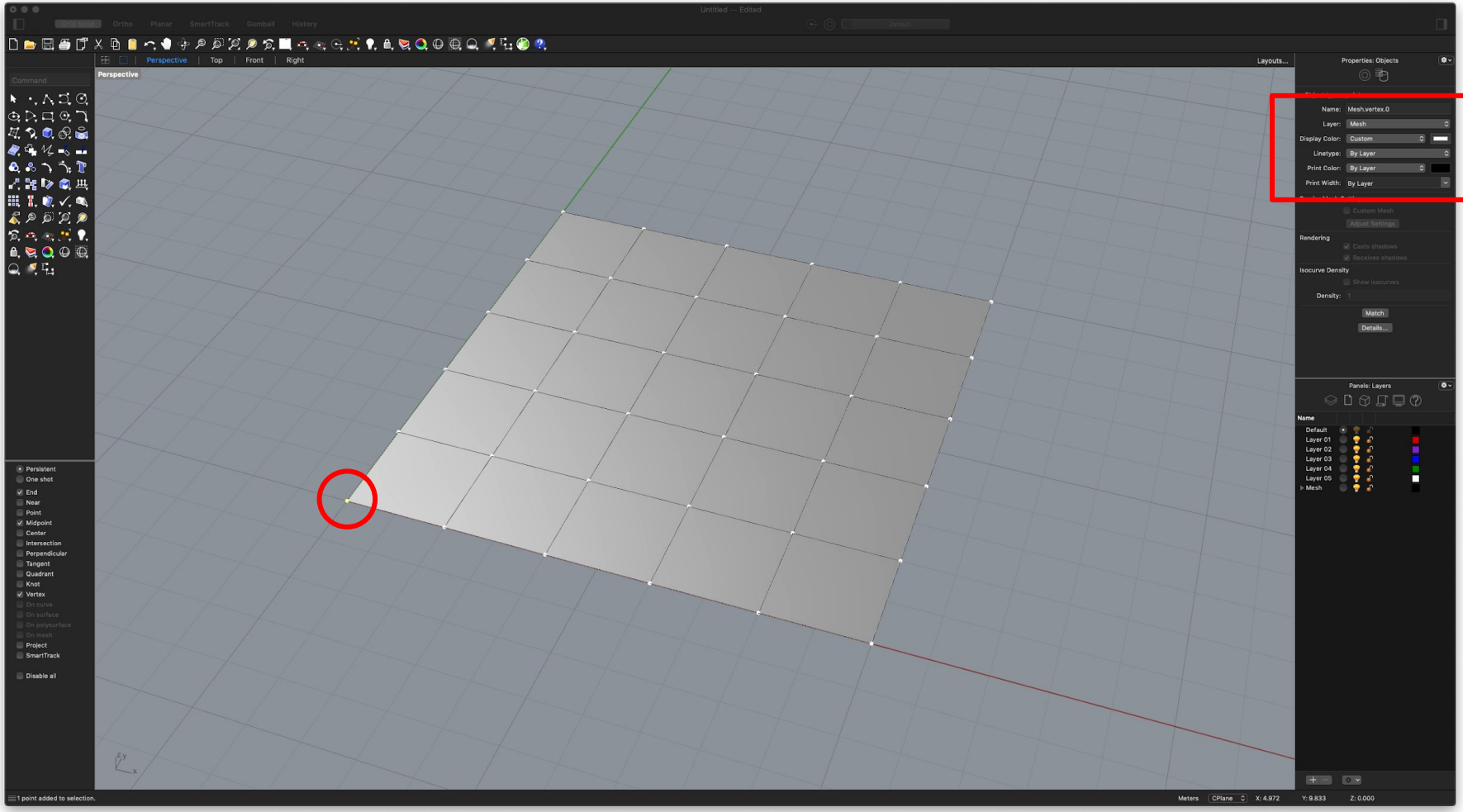

Untitled — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Command
Properties: Viewport
Title: Perspective
Size: 2120 x 1274
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 12.71
Y location: -16.3
Z location: 19.68
Target
X target: 5.67
Y target: 3.27
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
Command: RunPythonScript
Meters
CPlane
X: 6.707
Y: 10.663
Z: 0.000

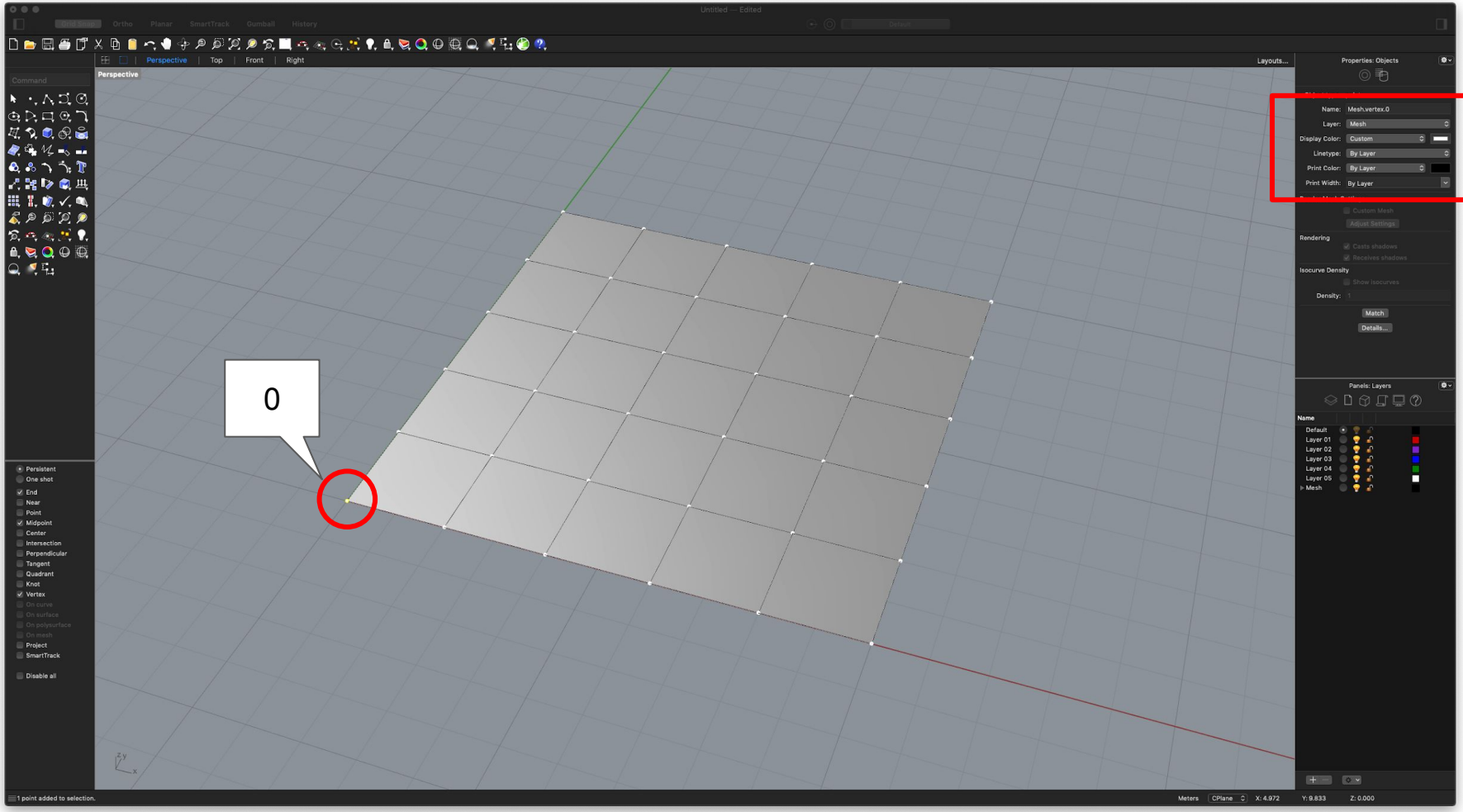
```
HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'faces.obj')

# ==============================================================================
# Mesh
# ==============================================================================

mesh = Mesh.from_obj(FILE)

# ==============================================================================
# Visualize result
# ==============================================================================

artist = MeshArtist(mesh, layer="Mesh")
artist.clear_layer()
artist.draw_...()
artist.draw_...()
artist.draw_...()
```

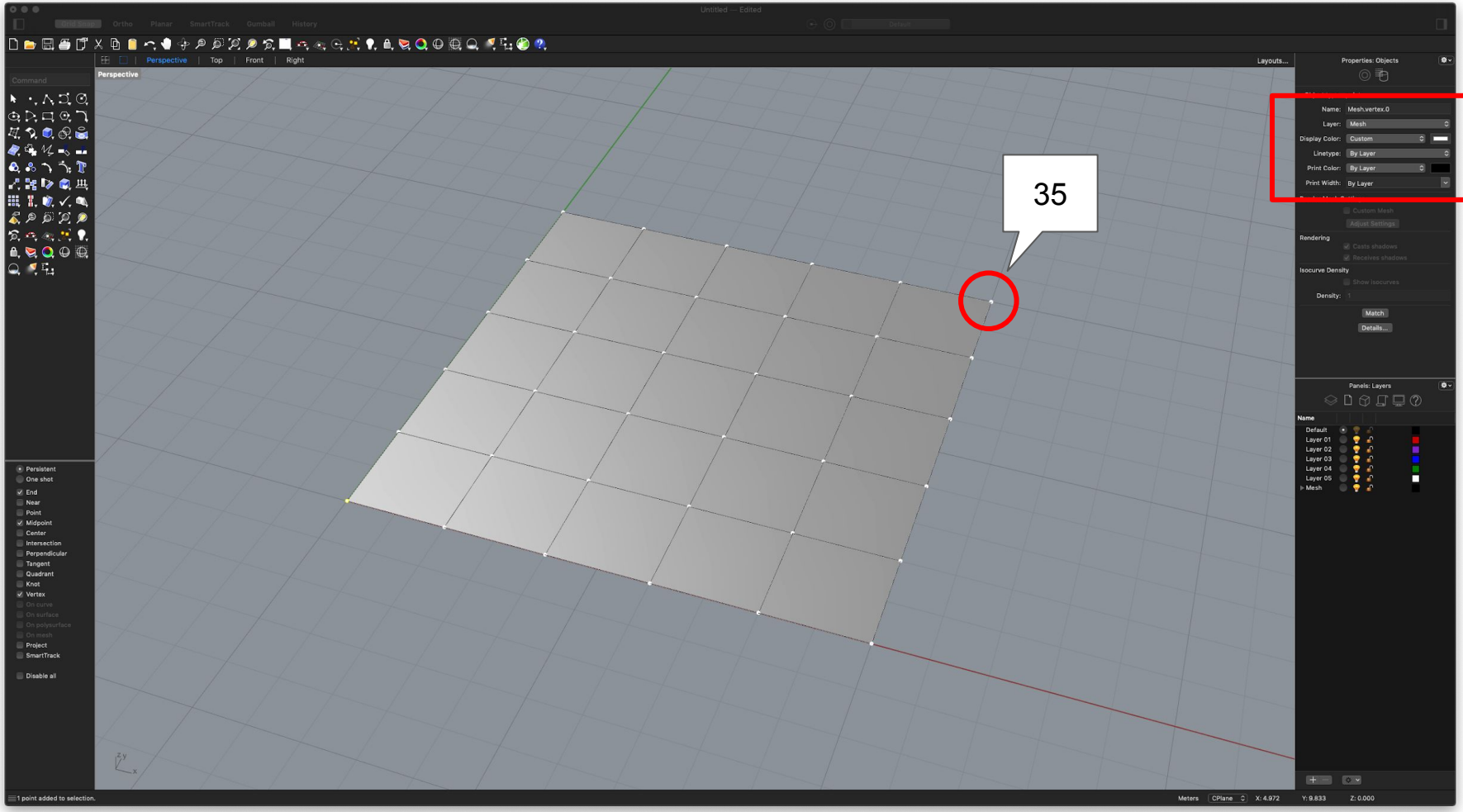
```
HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'faces.obj')

# ==============================================================================
# Mesh
# ==============================================================================

mesh = Mesh.from_obj(FILE)

# ==============================================================================
# Visualize result
# ==============================================================================

artist = MeshArtist(mesh, layer="Mesh")
artist.clear_layer()
artist.draw_...()
artist.draw_...()
artist.draw_...()
```

```
HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'faces.obj')

# ==============================================================================
# Mesh
# ==============================================================================

mesh = Mesh.from_obj(FILE)

# ==============================================================================
# Visualize result
# ==============================================================================

artist = MeshArtist(mesh, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
artist.draw_faces()
```


Untitled — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Command
Properties: Objects
Object type: point
Name: Mesh.vertex.0
Layer: Mesh
Display Color: Custom
Linetype: By Layer
Print Color: By Layer
Print Width: By Layer
Render Mesh Settings
Custom Mesh
Adjust Settings
Rendering
Casts shadows
Receives shadows
Isocurve Density
Show Isocurves
Density:
Match
Details...
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
1 point added to selection.
Meters
CPlane
X: 4.972
Y: 9.833
Z: 0.000

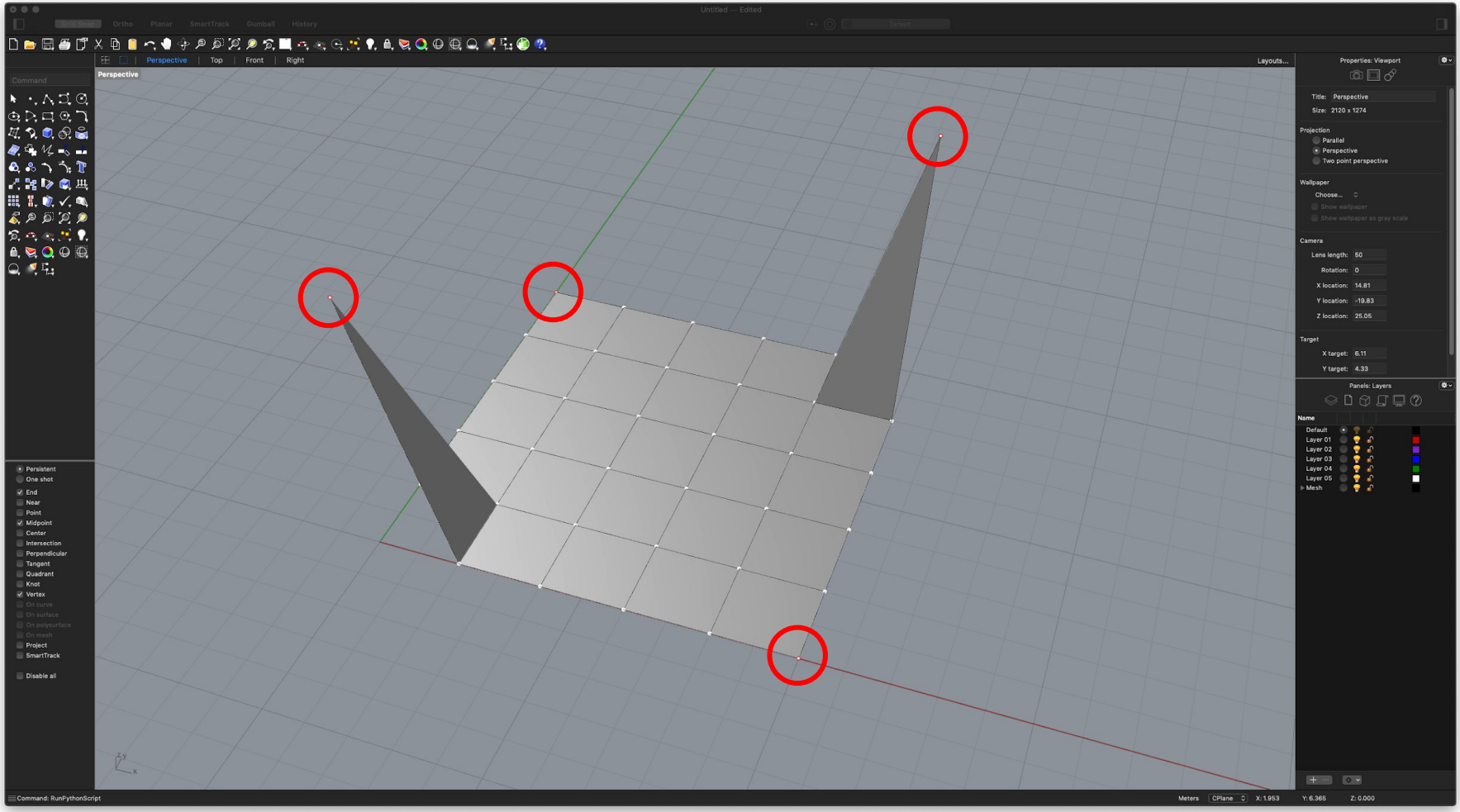

Perspective
Top
Front
Right
Layouts...
Properties: Objects
Name: Mesh.vertex.0
Layer: Mesh
Display Color: Custom
Linetype: By Layer
Print Color: By Layer
Print Width: By Layer
Custom Mesh
Adjust Settings
Rendering
Casts shadows
Receives shadows
Isocurve Density
Show isocurves
Density:
Match
Details...
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
1 point added to selection.
Meters
CPlane
X: 4.972
Y: 9.833
Z: 0.000

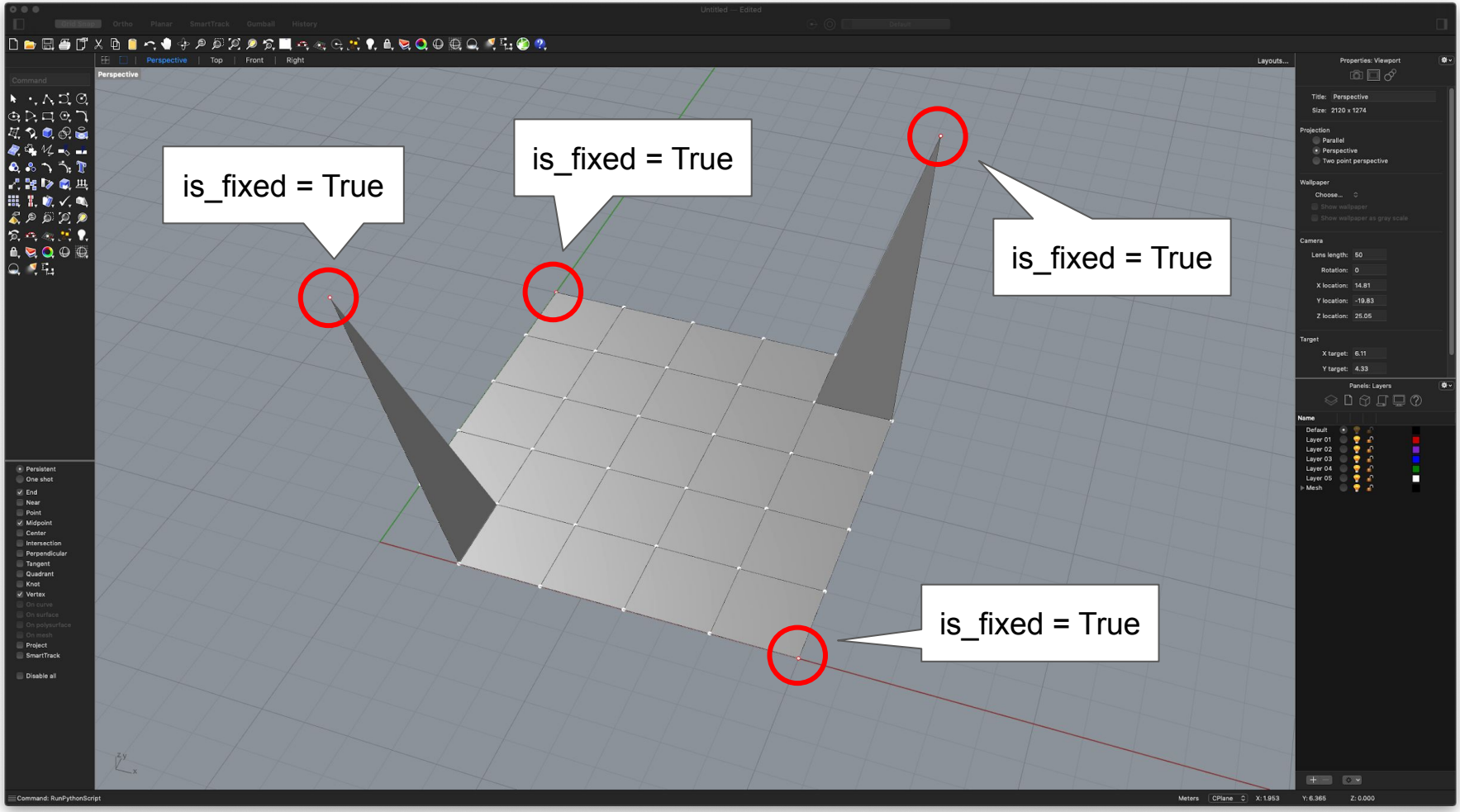

0
Perspective
Top
Front
Right
Layouts...
Properties: Objects
Name: Mesh.vertex.0
Layer: Mesh
Display Color: Custom
Linetype: By Layer
Print Color: By Layer
Print Width: By Layer
Custom Mesh
Adjust Settings
Rendering
Casts shadows
Receives shadows
Isocurve Density
Show isocurves
Density:
Match
Details...
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
1 point added to selection.
Meters
CPlane
X: 4.972
Y: 9.833
Z: 0.000

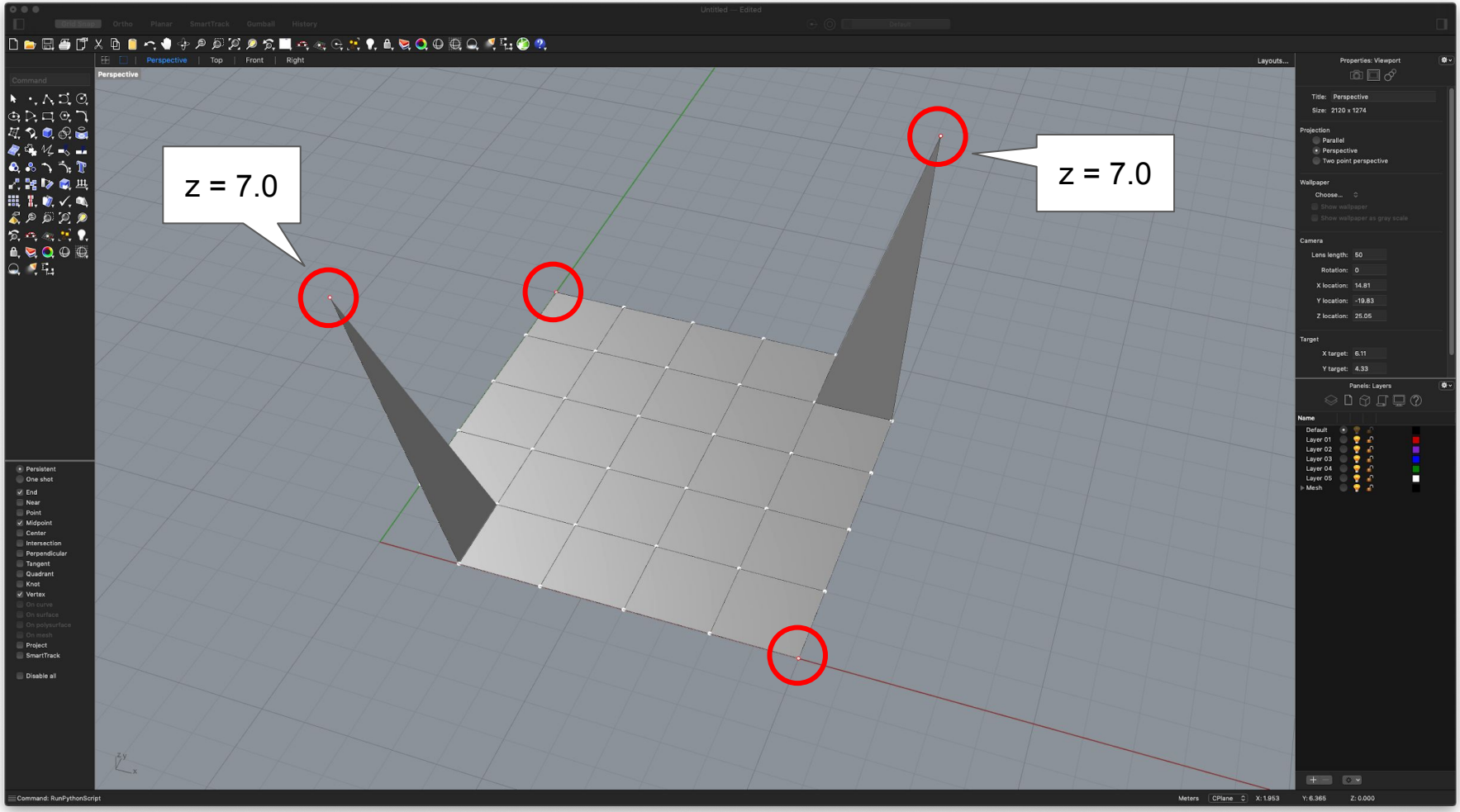

35
Perspective
Top
Front
Right
Layouts...
Properties: Objects
Name: Mesh.vertex.0
Layer: Mesh
Display Color: Custom
Linetype: By Layer
Print Color: By Layer
Print Width: By Layer
Custom Mesh
Adjust Settings
Rendering
Casts shadows
Receives shadows
Isocurve Density
Show isocurves
Density: 1
Match
Details...
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
1 point added to selection.
Meters
CPlane
X: 4.972
Y: 9.833
Z: 0.000

Update Attributes

1b_fofin_simple.py


Untitled — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2120 x 1274
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.81
Y location: -19.83
Z location: 25.05
Target
X target: 6.11
Y target: 4.33
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
Command: RunPythonScript
Meters
CPlane
X: 1.953
Y: 6.366
Z: 0.000


is_fixed = True
is_fixed = True
is_fixed = True
is_fixed = True


z = 7.0
z = 7.0
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2120 x 1274
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Camera
Lens length: 50
Rotation: 0
X location: 14.81
Y location: -19.83
Z location: 25.05
Target
X target: 6.11
Y target: 4.33
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
Project
SmartTrack
Disable all
Command: RunPythonScript
Meters
CPlane
X: 1.953
Y: 6.366
Z: 0.000

```
mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = ...(mesh.vertices_where({'...': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', ..., keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = ...(mesh.vertices_where({'...': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', ..., keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = ...(mesh.vertices_where({'...': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', ..., keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = ...(mesh.vertices_where({'...': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', ..., keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = ...(mesh.vertices_where({'...': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', ..., keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
mesh.update_default_vertex_attributes({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
mesh.update_default_edge_attributes({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = list(mesh.vertices_where({'vertex_degree': 2}))
high = [0, 35]

mesh.set_vertices_attribute('is_fixed', True, keys=corners)
mesh.set_vertices_attribute('z', 7.0, keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(mesh.edges_on_boundary())

mesh.set_edges_attribute('q', 5.0, keys=boundary)
```

```
# ==============================================================================
# Visualize result
# ==============================================================================

artist = MeshArtist(mesh, layer="Mesh")
artist.clear_layer()
artist.draw_vertices(
    color={key: (..., 0, 0) for key in mesh.vertices_where({'...': True})})
artist.draw_edges()
artist.draw_faces()
```

```
# ==============================================================================
# Visualize result
# ==============================================================================

artist = MeshArtist(mesh, layer="Mesh")
artist.clear_layer()
artist.draw_vertices(
    color={key: (255, 0, 0) for key in mesh.vertices_where({'is_fixed': True})})
artist.draw_edges()
artist.draw_faces()
```

FoFin

1c_fofin_simple.py

```
# ==============================================================================
# Proxy
# ==============================================================================

# from compas.numerical import fd_numpy
from compas.rpc import Proxy
numerical = Proxy('...')
fd_numpy = numerical.fd_numpy
```

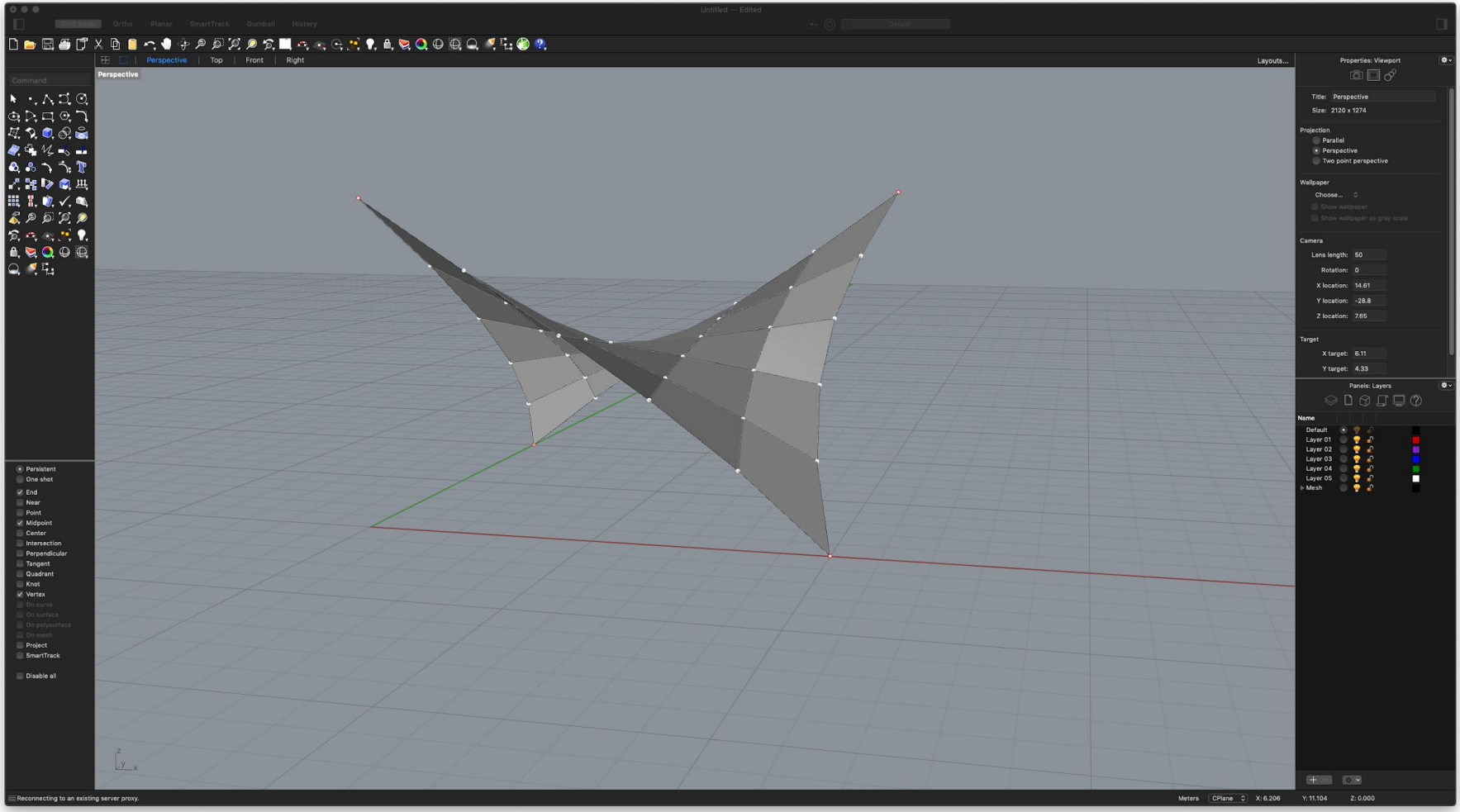
```
# ==============================================================================
# Proxy
# ==============================================================================

# from compas.numerical import fd_numpy
from compas.rpc import Proxy
numerical = Proxy('compas.numerical')
fd_numpy = numerical.fd_numpy
```

```
# ==============================================================================
# FoFin input
# ==============================================================================

xyz = mesh.get_vertices_...('xyz')
fixed = ...(mesh.vertices_where({'is_fixed': ...}))
... = mesh.get_vertices_attributes(('...', '...', '...'))

edges = ...(mesh.edges())
... = mesh.get_..._attribute('q')

# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
```

```
# ==============================================================================
# FoFin input
# ==============================================================================

xyz = mesh.get_vertices_attributes('xyz')
fixed = list(mesh.vertices_where({'is_fixed': True}))
loads = mesh.get_vertices_attributes(('px', 'py', 'pz'))

edges = list(mesh.edges())
q = mesh.get_edges_attribute('q')

# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
```

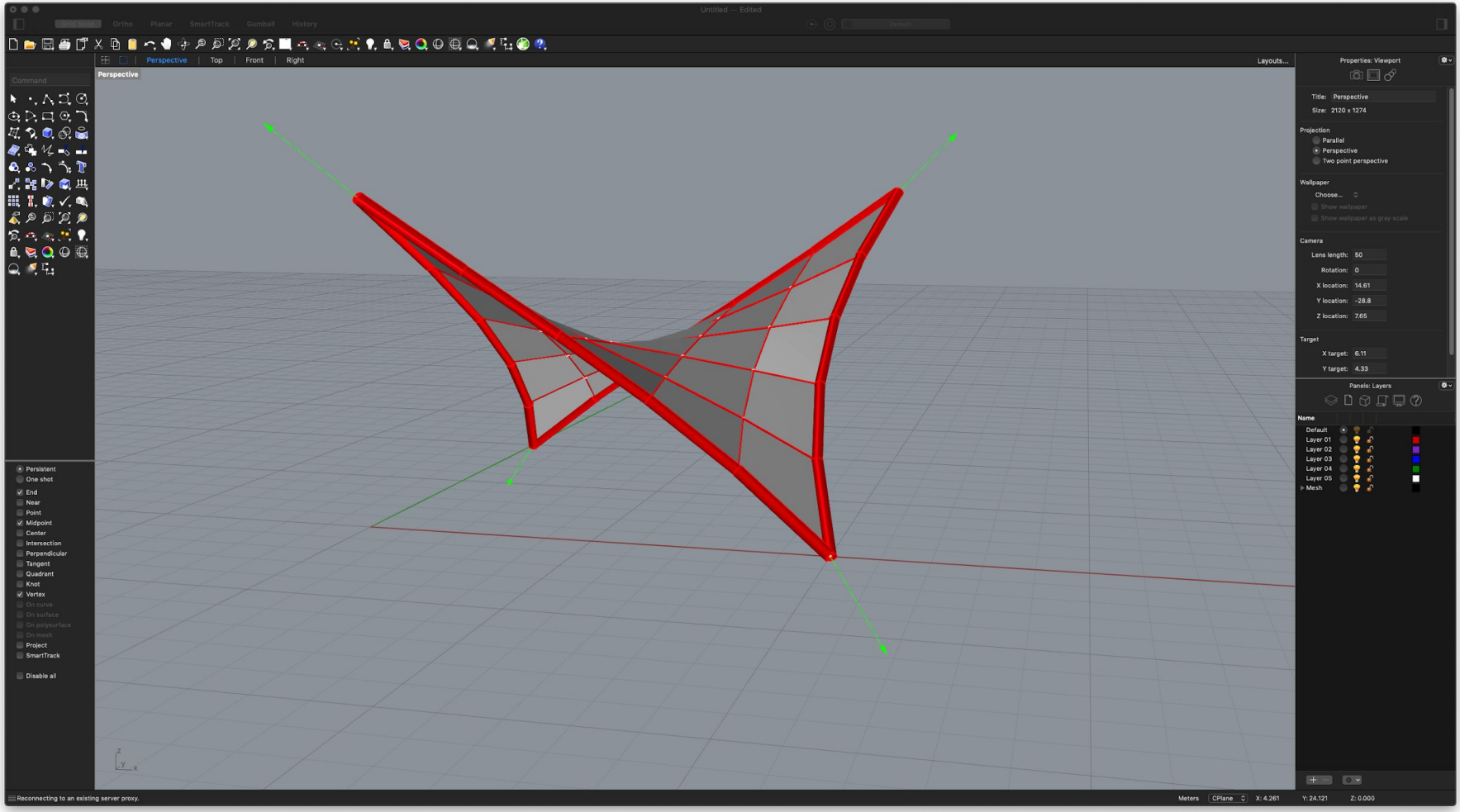
```python
# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)

# ==============================================================================
# Fofin update
# ==============================================================================

for key, attr in mesh.vertices(True):
    attr['x'] = xyz[key][0]
    attr['y'] = xyz[key][1]
    attr['z'] = xyz[key][2]
    attr['...'] = r[key][...]
    attr['...'] = r[key][...]
    attr['...'] = r[key][...]

for ..., (u, v, attr) in enumerate(mesh.edges(True)):
    attr['f'] = ...[index][0]
```

```
# ==============================================================================
# Fofin run
# ==============================================================================

xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)

# ==============================================================================
# Fofin update
# ==============================================================================

for key, attr in mesh.vertices(True):
    attr['x'] = xyz[key][0]
    attr['y'] = xyz[key][1]
    attr['z'] = xyz[key][2]
    attr['rx'] = r[key][0]
    attr['ry'] = r[key][1]
    attr['rz'] = r[key][2]

for index, (u, v, attr) in enumerate(mesh.edges(True)):
    attr['f'] = f[index][0]
```


Untitled — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Command
Properties: Viewport
Title: Perspective
Size: 2120 x 1274
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.61
Y location: -28.8
Z location: 7.65
Target
X target: 6.11
Y target: 4.33
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
Reconnecting to an existing server proxy.
Meters
CPlane
X: 6.206
Y: 11.104
Z: 0.000

Extended Visualisation

1d_fofin_simple.py


Untitled — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Command
Properties: Viewport
Title: Perspective
Size: 2120 x 1274
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.61
Y location: -28.8
Z location: 7.65
Target
X target: 6.11
Y target: 4.33
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
Reconnecting to an existing server proxy.
Meters
CPlane
X: 4.261
Y: 24.121
Z: 0.000

```
forces = []
for ..., ..., ... in mesh.edges(True):
    force = attr['...']
    ... = mesh.vertex_coordinates(u)
    ... = mesh.vertex_coordinates(v)
    radius = 0.01 * force
    forces.append({
        'start': start,
        'end': end,
        'radius': radius,
        'color': (..., ..., ...)})

compas_rhino.draw_cylinders(forces, layer="Mesh::Forces", clear=True)
```

```
forces = []
for u, v, attr in mesh.edges(True):
    force = attr['f']
    start = mesh.vertex_coordinates(u)
    end = mesh.vertex_coordinates(v)
    radius = 0.01 * force
    forces.append({
        'start': start,
        'end': end,
        'radius': radius,
        'color': (255, 0, 0)})

compas_rhino.draw_cylinders(forces, layer="Mesh::Forces", clear=True)
```

```
reactions = []
for key, attr in mesh.vertices_where({'...': True}, True):
    reaction = [attr['...'], attr['...'], attr['...']]
    vector = scale_vector(reaction, ...)
    start = mesh.vertex_coordinates(key)
    end = add_vectors(start, vector)
    reactions.append({
        'start': start,
        'end': end,
        'arrow': '...',
        'color': (..., ..., ...)})

compas_rhino.draw_lines(reactions, layer="Mesh::Reactions", clear=True)
```

```
reactions = []
for key, attr in mesh.vertices_where({'is_fixed': True}, True):
    reaction = [attr['rx'], attr['ry'], attr['rz']]
    vector = scale_vector(reaction, -0.1)
    start = mesh.vertex_coordinates(key)
    end = add_vectors(start, vector)
    reactions.append({
        'start': start,
        'end': end,
        'arrow': 'end',
        'color': (0, 255, 0)})

compas_rhino.draw_lines(reactions, layer="Mesh::Reactions", clear=True)
```

2_fofin_custom.py

```python
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division

import os

from fofin.shell import Shell
from fofin.shellartist import ShellArtist

# ==============================================================================
# Input file
# ==============================================================================

HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'faces.obj')

# ==============================================================================
# Shell
# ==============================================================================

shell = Shell.from_obj(FILE)
```

```
# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = list(shell.vertices_where({'vertex_degree': 2}))
high = [0, 35]

shell.set_vertices_attribute('is_fixed', True, keys=corners)
shell.set_vertices_attribute('z', 7.0, keys=high)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(shell.edges_on_boundary())

shell.set_edges_attribute('q', 5.0, keys=boundary)
```

```
# ==============================================================================
# Fofin run
# ==============================================================================

shell.fofin()

# ==============================================================================
# Visualize result
# ==============================================================================

artist = ShellArtist(shell, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
artist.draw_forces(scale=0.01)
artist.draw_reactions(scale=0.1)
```

Shell & ShellArtist

2a_fofin_custom.py

```python
# ==============================================================================
# Customizations
# ==============================================================================


class Shell(Mesh):

    def __init__(self):
        super(Shell, self).__init__()
        self.default_vertex_attributes.update({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
        self.default_edge_attributes.update({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

    def fofin(self):
        # fofin input
        ...
        # fofin run
        ...
        # fofin update
        ...
```

```python
def fofin(self):
    # fofin input
    xyz = mesh.get_vertices_attributes('xyz')
    fixed = list(mesh.vertices_where({'is_fixed': True}))
    loads = mesh.get_vertices_attributes(('px', 'py', 'pz'))
    edges = list(mesh.edges())
    q = mesh.get_edges_attribute('q')
    # fofin run
    ...
    # fofin update
    ...
```

```python
def fofin(self):
    # fofin input
    xyz = mesh.get_vertices_attributes('xyz')
    fixed = list(mesh.vertices_where({'is_fixed': True}))
    loads = mesh.get_vertices_attributes(('px', 'py', 'pz'))
    edges = list(mesh.edges())
    q = mesh.get_edges_attribute('q')
    # fofin run
    ...
    # fofin update
    ...
```

```python
def fofin(self):
    # fofin input
    xyz = self.get_vertices_attributes('xyz')
    fixed = list(self.vertices_where({'is_fixed': True}))
    loads = self.get_vertices_attributes(('px', 'py', 'pz'))
    edges = list(self.edges())
    q = self.get_edges_attribute('q')
    # fofin run
    ...
    # fofin update
    ...
```

```python
def fofin(self):
    # fofin input
    xyz = self.get_vertices_attributes('xyz')
    fixed = list(self.vertices_where({'is_fixed': True}))
    loads = self.get_vertices_attributes(('px', 'py', 'pz'))
    edges = list(self.edges())
    q = self.get_edges_attribute('q')
    # fofin run
    xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
    # fofin update
    for key, attr in self.vertices(True):
        attr['x'] = xyz[key][0]
        attr['y'] = xyz[key][1]
        attr['z'] = xyz[key][2]
        attr['rx'] = r[key][0]
        attr['ry'] = r[key][1]
        attr['rz'] = r[key][2]
    for index, (u, v, attr) in enumerate(self.edges(True)):
        attr['f'] = f[index][0]
```

```
# ==============================================================================
# Customizations
# ==============================================================================


class ShellArtist(...):

    def draw_forces(self, scale=1.0, layer="Mesh::Forces"):
        ...

    def draw_reactions(self, scale=1.0, layer="Mesh::Reactions"):
        ...
```

```python
def draw_forces(self, scale=1.0, layer="Mesh::Forces"):
    forces = []
    for u, v, attr in mesh.edges(True):
        force = attr['f']
        start = mesh.vertex_coordinates(u)
        end = mesh.vertex_coordinates(v)
        radius = 0.01 * force
        forces.append({
            'start': start,
            'end': end,
            'radius': radius,
            'color': (255, 0, 0)})
    compas_rhino.draw_cylinders(forces, layer=..., clear=True)

def draw_reactions(self, scale=1.0, layer="Mesh::Reactions"):
    ...
```

```
def draw_forces(self, scale=1.0, layer="Mesh::Forces"):
    forces = []
    for u, v, attr in mesh.edges(True):
        force = attr['f']
        start = mesh.vertex_coordinates(u)
        end = mesh.vertex_coordinates(v)
        radius = 0.01 * force
        forces.append({
            'start': start,
            'end': end,
            'radius': radius,
            'color': (255, 0, 0)})
    compas_rhino.draw_cylinders(forces, layer=..., clear=True)

def draw_reactions(self, scale=1.0, layer="Mesh::Reactions"):
    ...
```

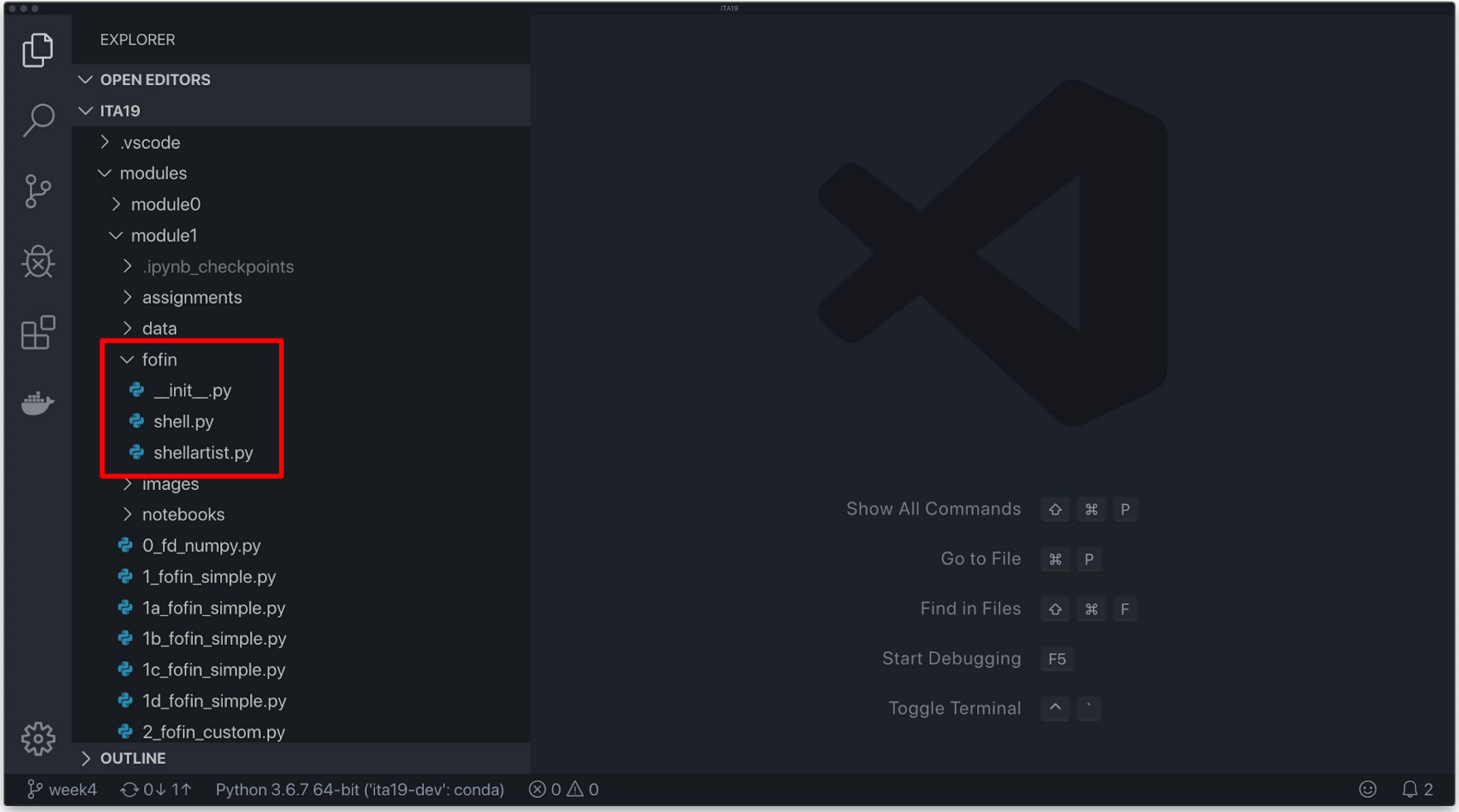
```python
def draw_forces(self, scale=1.0, layer="Mesh::Forces"):
    forces = []
    for u, v, attr in self.datastructure.edges(True):
        force = attr['f']
        start = self.datastructure.vertex_coordinates(u)
        end = self.datastructure.vertex_coordinates(v)
        radius = 0.01 * force
        forces.append({
            'start': start,
            'end': end,
            'radius': radius,
            'color': (255, 0, 0)})
    compas_rhino.draw_cylinders(forces, layer=..., clear=True)

def draw_reactions(self, scale=1.0, layer="Mesh::Reactions"):
    ...
```

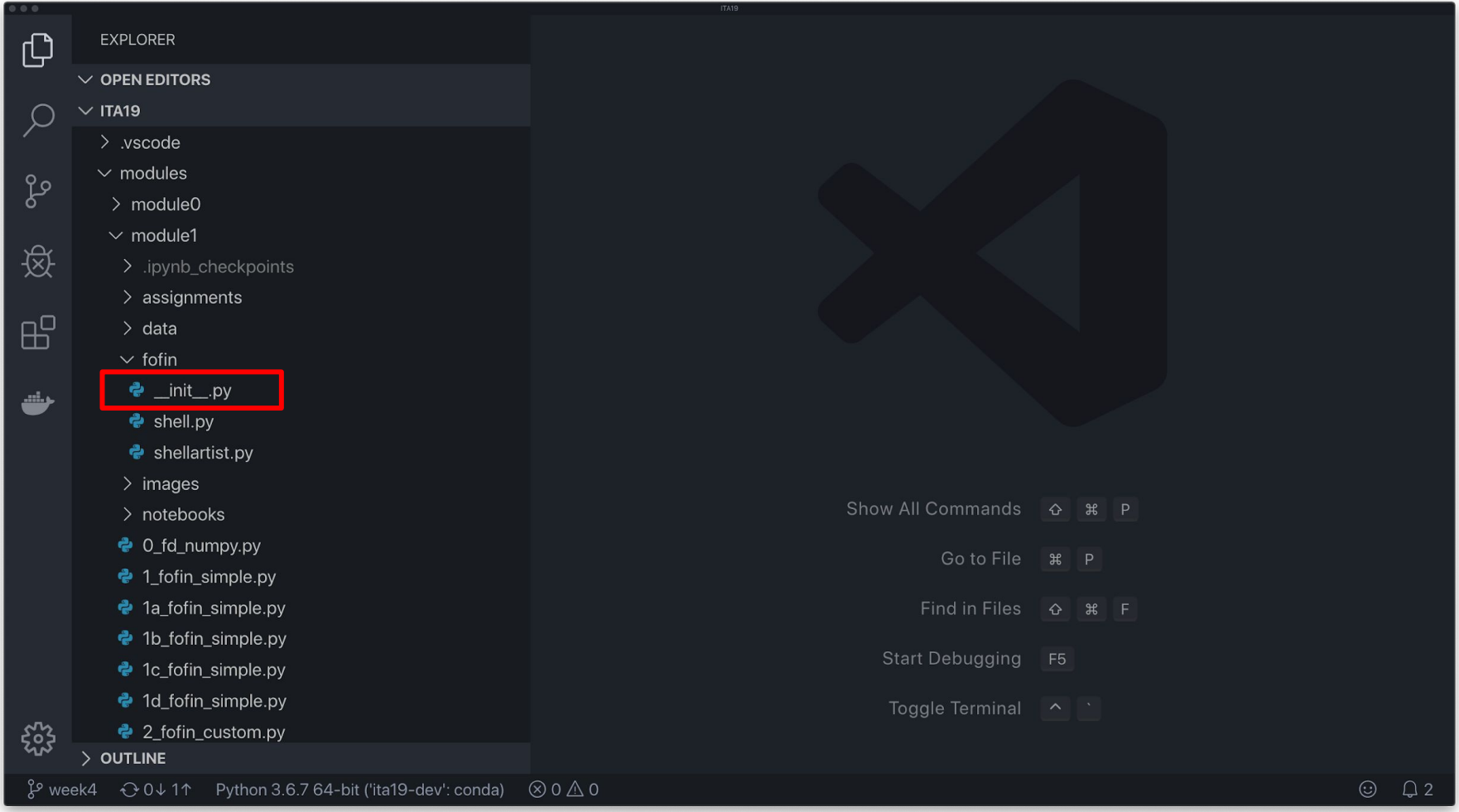
Shell

fofin.shell.py

```
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division

import os

import compas
from compas.datastructures import Mesh
from compas.geometry import add_vectors
from compas.geometry import scale_vector

import compas_rhino
from compas_rhino.artists import MeshArtist
```

```
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division

import os

from fofin.shell import Shell
from fofin.shellartist import ShellArtist
```


ITA19
EXPLORER
OPEN EDITORS
ITA19
.vscode
modules
module0
module1
.ipynb_checkpoints
assignments
data
fofin
__init__.py
shell.py
shellartist.py
images
notebooks
0_fd_numpy.py
1_fofin_simple.py
1a_fofin_simple.py
1b_fofin_simple.py
1c_fofin_simple.py
1d_fofin_simple.py
2_fofin_custom.py
OUTLINE
Show All Commands ⇧ ⌘ P
Go to File ⌘ P
Find in Files ⇧ ⌘ F
Start Debugging F5
Toggle Terminal ^ `
week4
0↓ 1↑
Python 3.6.7 64-bit ('ita19-dev': conda)
0 0
2


ITA19
EXPLORER
OPEN EDITORS
ITA19
.vscode
modules
module0
module1
.ipynb_checkpoints
assignments
data
fofin
__init__.py
shell.py
shellartist.py
images
notebooks
0_fd_numpy.py
1_fofin_simple.py
1a_fofin_simple.py
1b_fofin_simple.py
1c_fofin_simple.py
1d_fofin_simple.py
2_fofin_custom.py
OUTLINE
Show All Commands ⇧ ⌘ P
Go to File ⌘ P
Find in Files ⇧ ⌘ F
Start Debugging F5
Toggle Terminal ^ `
week4
0↓ 1↑
Python 3.6.7 64-bit ('ita19-dev': conda)
0 0
2

```
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division

from compas.datastructures import Mesh
from compas.rpc import Proxy

numerical = Proxy('compas.numerical')
fd_numpy = numerical.fd_numpy
```

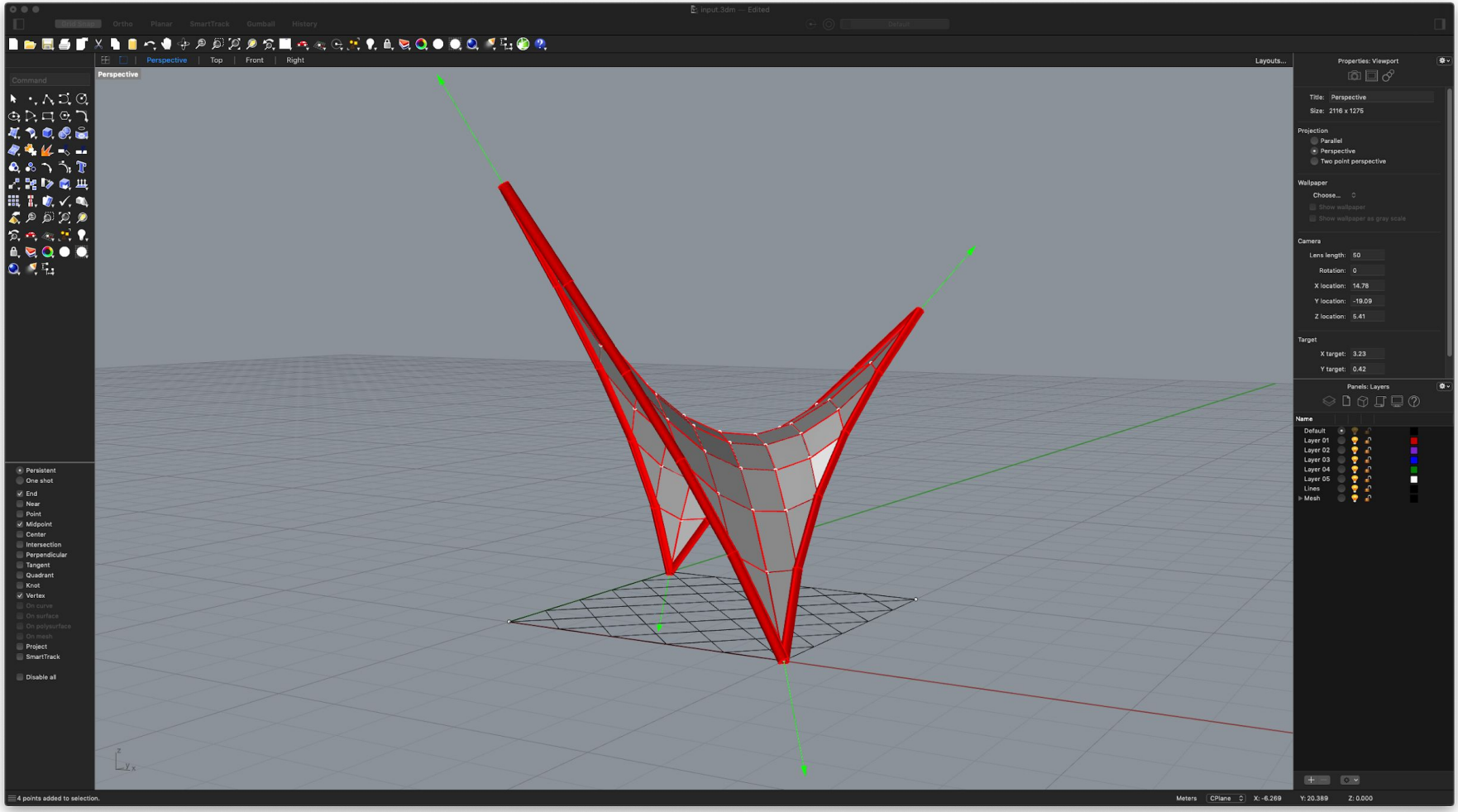
```python
class Shell(Mesh):

    def __init__(self):
        super(Shell, self).__init__()
        self.default_vertex_attributes.update({'is_fixed': False, 'px': 0.0, 'py': 0.0, 'pz': 0.0})
        self.default_edge_attributes.update({'q': 1.0, 'f': 0.0, 'rx': 0.0, 'ry': 0.0, 'rz': 0.0})

    def fofin(self):
        xyz = self.get_vertices_attributes('xyz')
        fixed = list(self.vertices_where({'is_fixed': True}))
        loads = self.get_vertices_attributes(('px', 'py', 'pz'))
        edges = list(self.edges())
        q = self.get_edges_attribute('q')
        xyz, q, f, l, r = fd_numpy(xyz, edges, fixed, q, loads)
        for key, attr in self.vertices(True):
            attr['x'] = xyz[key][0]
            attr['y'] = xyz[key][1]
            attr['z'] = xyz[key][2]
            attr['rx'] = r[key][0]
            attr['ry'] = r[key][1]
            attr['rz'] = r[key][2]
        for index, (u, v, attr) in enumerate(self.edges(True)):
            attr['f'] = f[index][0]
```

```python
# ================================================================================
# Main
# ================================================================================

if __name__ == '__main__':
    pass
```

ShellArtist

fofin.shellartist.py

3_fofin_input.py


input.3dm — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.78
Y location: -19.09
Z location: 5.41
Target
X target: 3.23
Y target: 0.42
Panels: Layers
Name
Default
Layer 01
Layer 02
Layer 03
Layer 04
Layer 05
Lines
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
4 points added to selection.
Meters
CPlane
X: -6.269
Y: 20.389
Z: 0.000

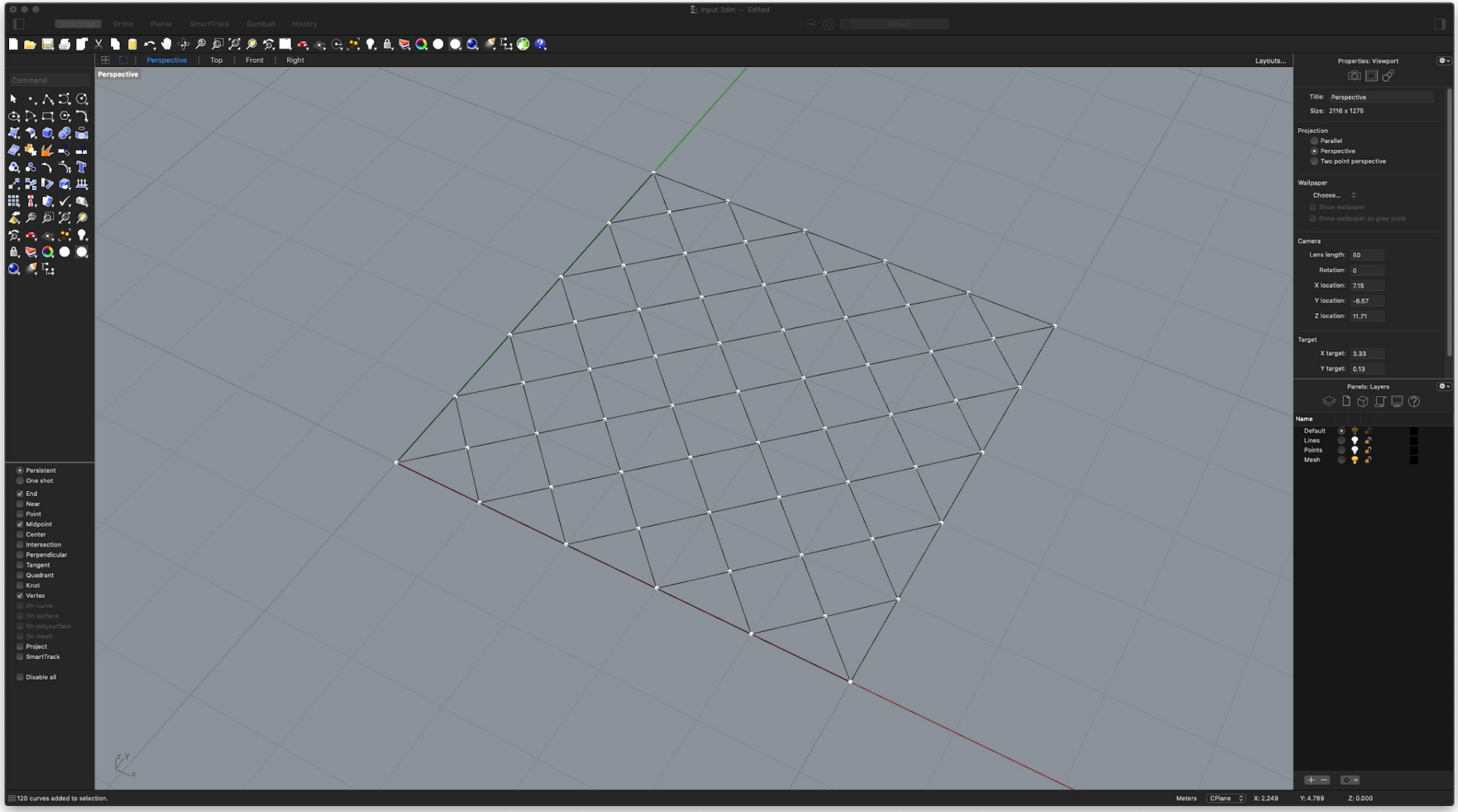
```
# ==============================================================================
# Input
# ==============================================================================

guids = compas_rhino.select_lines()
lines = compas_rhino.get_line_coordinates(guids)

guids = compas_rhino.select_points()
points = compas_rhino.get_point_coordinates(guids)
names = compas_rhino.get_object_names(guids)

# ==============================================================================
# Shell
# ==============================================================================

shell = Shell.from_lines(lines, delete_boundary_face=True)
```

```
# ==============================================================================
# Geometric key map
# ==============================================================================

gkey_key = shell.gkey_key()

# ==============================================================================
# Vertex attributes
# ==============================================================================

for name, point in zip(names, points):
    gkey = geometric_key(point)
    if gkey in gkey_key:
        key = gkey_key[gkey]
        shell.set_vertex_attribute(key, 'is_fixed', True)
        if name:
            parts = name.split('=')
            if len(parts) == 2:
                z = float(parts[1])
                shell.set_vertex_attribute(key, 'z', z)
```

Shell.from_lines(...)

3a_fofin_input.py

```
# ==============================================================================
# Input
# ==============================================================================

guids = compas_rhino.select_lines()
lines = compas_rhino.get_line_coordinates(guids)

# ==============================================================================
# Shell
# ==============================================================================

shell = Shell.from_lines(lines, delete_boundary_face=True)

# ==============================================================================
# Visualize result
# ==============================================================================

artist = ShellArtist(shell, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
```

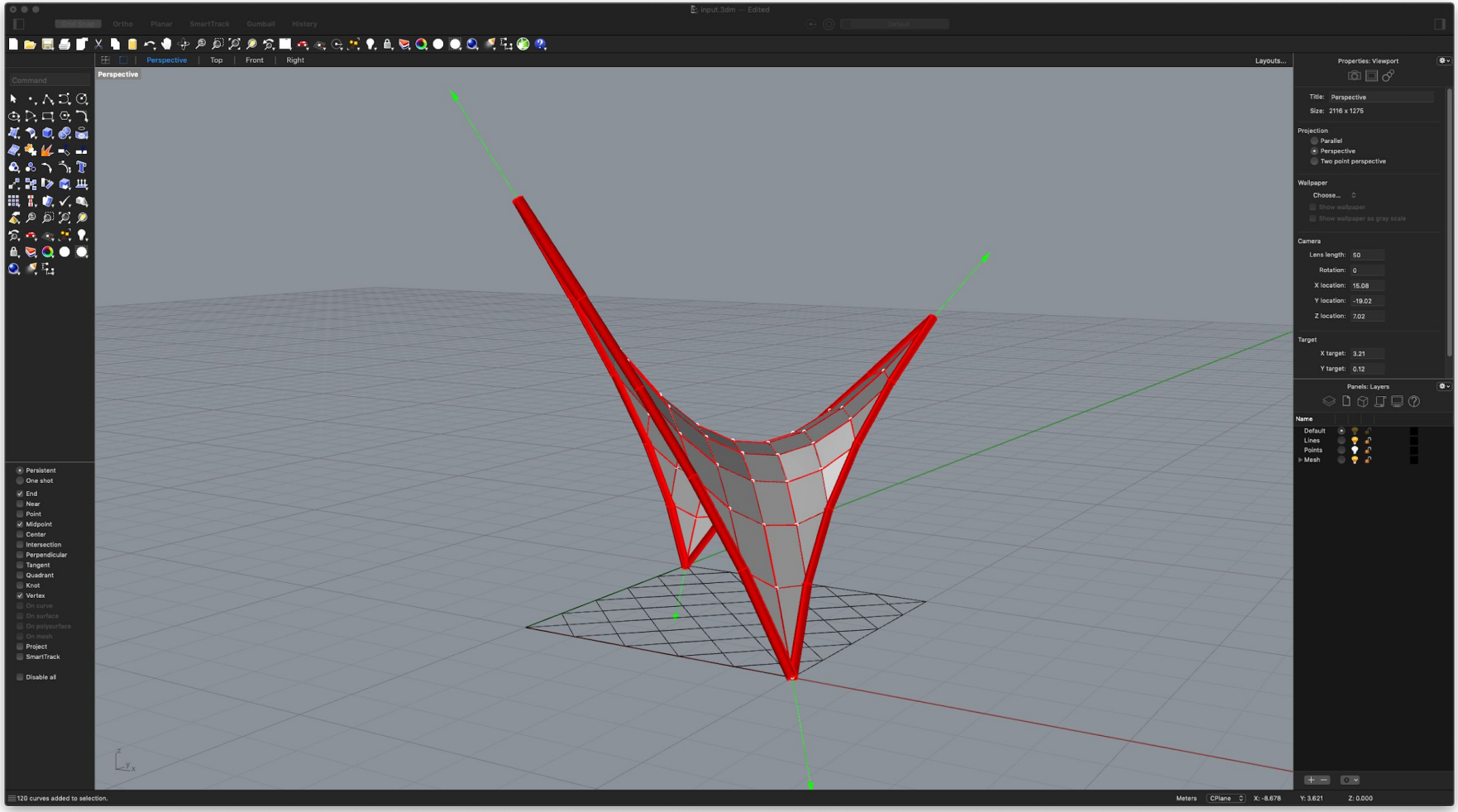

input.3dm — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 7.15
Y location: -6.57
Z location: 11.71
Target
X target: 3.33
Y target: 0.13
Panels: Layers
Name
Default
Lines
Points
Mesh
120 curves added to selection.
Meters
CPlane
X: 2.249
Y: 4.789
Z: 0.000

Set Attributes & Fofin

3b_fofin_input.py

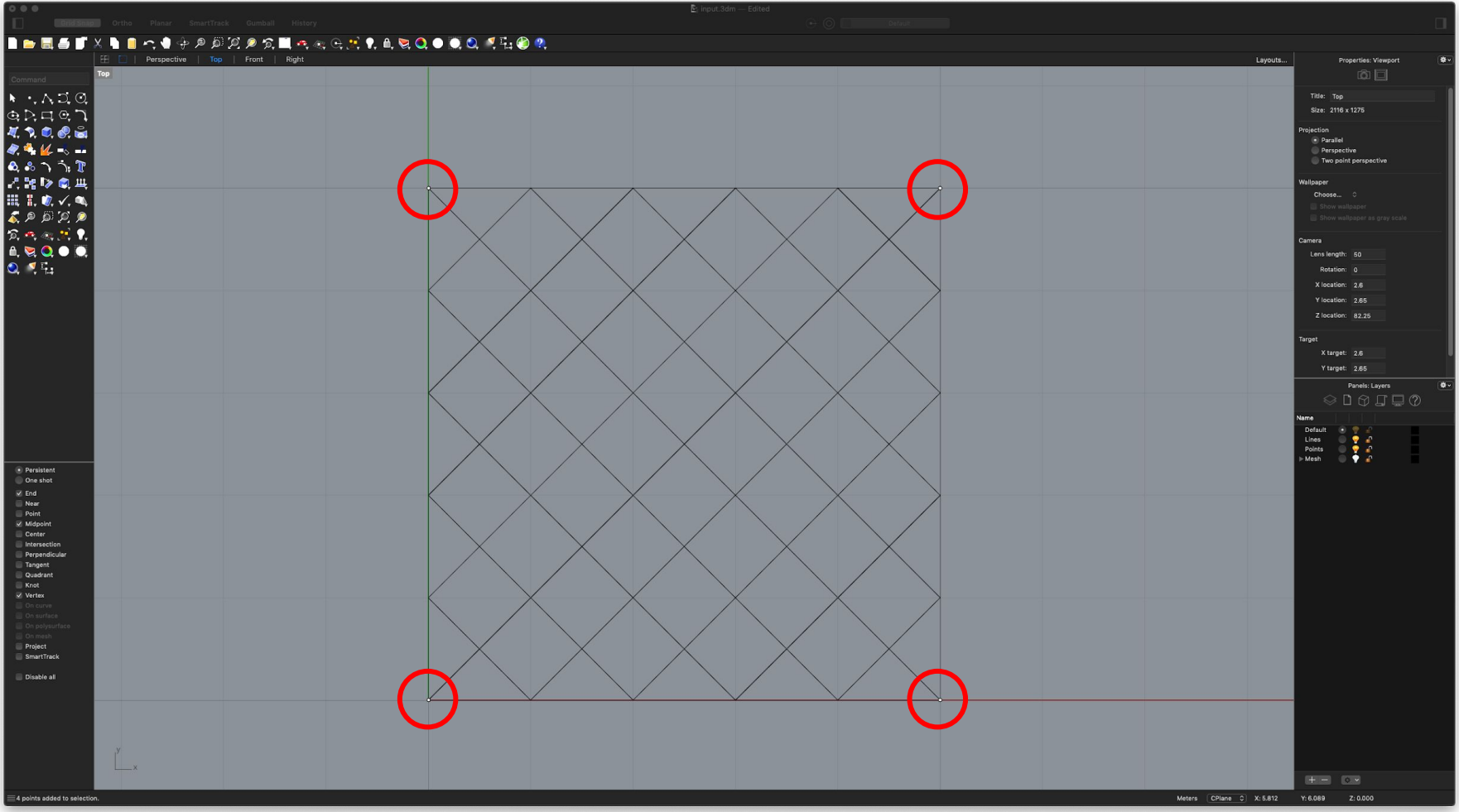
```
# ==============================================================================
# Vertex attributes
# ==============================================================================

corners = list(shell.vertices_where({'vertex_degree': 3}))
high = 16
higher = 1

shell.set_vertices_attribute('is_fixed', True, keys=corners)
shell.set_vertex_attribute(higher, 'z', 7.0)
shell.set_vertex_attribute(high, 'z', 5.0)

# ==============================================================================
# Edge attributes
# ==============================================================================

boundary = list(shell.edges_on_boundary())

shell.set_edges_attribute('q', 5.0, keys=boundary)
```

```
# ==============================================================================
# Fofin run
# ==============================================================================

shell.fofin()

# ==============================================================================
# Visualize result
# ==============================================================================

artist = ShellArtist(shell, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
artist.draw_faces()
artist.draw_forces(scale=0.01)
artist.draw_reactions(scale=0.1)
```


input.3dm — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 15.08
Y location: -19.02
Z location: 7.02
Target
X target: 3.21
Y target: 0.12
Panels: Layers
Name
Default
Lines
Points
Mesh
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
120 curves added to selection.
Meters
CPlane
X: -8.678
Y: 3.621
Z: 0.000

Identify fixed points

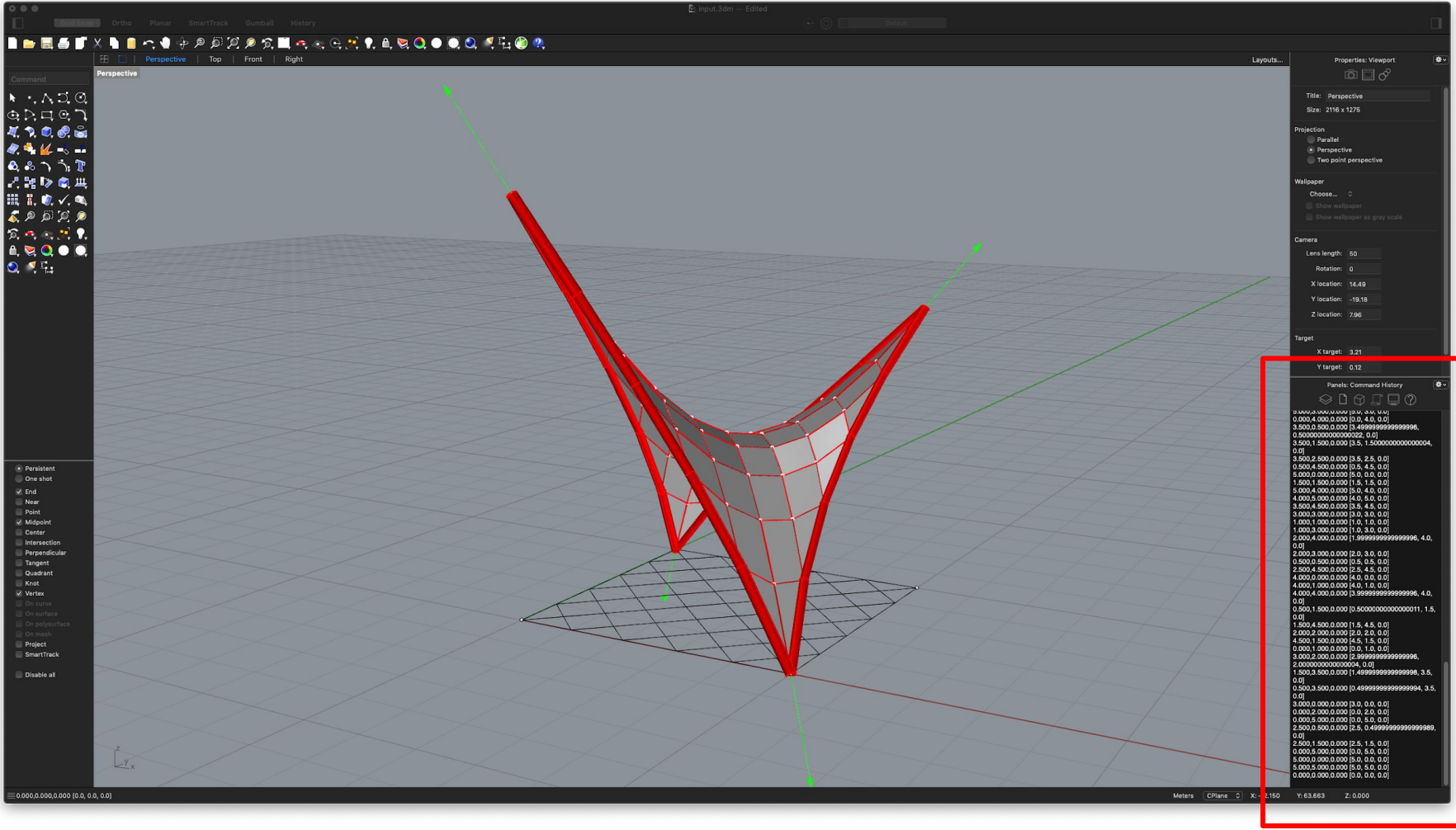
3c_fofin_input.py


input.3dm — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Top
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Camera
Lens length: 50
Rotation: 0
X location: 2.6
Y location: 2.65
Z location: 82.25
Target
X target: 2.6
Y target: 2.65
Panels: Layers
Name
Default
Lines
Points
Mesh
4 points added to selection.
Meters
CPlane
X: 5.812
Y: 6.089
Z: 0.000

```
# ==============================================================================
# Input
# ==============================================================================

guids = compas_rhino.select_lines()
lines = compas_rhino.get_line_coordinates(guids)

guids = compas_rhino.select_points()
points = compas_rhino.get_point_coordinates(guids)

# ==============================================================================
# Shell
# ==============================================================================

shell = Shell.from_lines(lines, delete_boundary_face=True)
```

```
# ==============================================================================
# Shell
# ==============================================================================

shell = Shell.from_lines(lines, delete_boundary_face=True)

# ==============================================================================
# Geometric key map
# ==============================================================================

gkey_key = shell.gkey_key()

# ==============================================================================
# Vertex attributes
# ==============================================================================

for point in points:
    gkey = geometric_key(point)
    if gkey in gkey_key:
        key = gkey_key[gkey]
        shell.set_vertex_attribute(key, 'is_fixed', True)
```

```python
# ==============================================================================
# Geometric key map
# ==============================================================================

gkey_key = {}
for key in shell.vertices():
    xyz = shell.vertex_coordinates(key)
    gkey = geometric_key(xyz)
    gkey_key[gkey] = key
    print(gkey, xyz)

# ==============================================================================
# Vertex attributes
# ==============================================================================

for point in points:
    gkey = geometric_key(point)
    print(gkey, point)
    if gkey in gkey_key:
        key = gkey_key[gkey]
        shell.set_vertex_attribute(key, 'is_fixed', True)
```


input.3dm — Edited
Perspective
Top
Front
Right
Layouts...
Properties: Viewport
Title: Perspective
Size: 2116 x 1275
Projection
Parallel
Perspective
Two point perspective
Wallpaper
Choose...
Show wallpaper
Show wallpaper as gray scale
Camera
Lens length: 50
Rotation: 0
X location: 14.49
Y location: -19.18
Z location: 7.96
Target
X target: 3.21
Y target: 0.12
Panels: Command History
0.000,4.000,0.000 [0.0, 4.0, 0.0]
3.500,0.500,0.000 [3.4999999999999996, 0.5000000000000022, 0.0]
3.500,1.500,0.000 [3.5, 1.5000000000000004, 0.0]
3.500,2.500,0.000 [3.5, 2.5, 0.0]
0.500,4.500,0.000 [0.5, 4.5, 0.0]
5.000,0.000,0.000 [5.0, 0.0, 0.0]
1.500,1.500,0.000 [1.5, 1.5, 0.0]
5.000,4.000,0.000 [5.0, 4.0, 0.0]
4.000,5.000,0.000 [4.0, 5.0, 0.0]
3.500,4.500,0.000 [3.5, 4.5, 0.0]
3.000,3.000,0.000 [3.0, 3.0, 0.0]
1.000,1.000,0.000 [1.0, 1.0, 0.0]
1.000,3.000,0.000 [1.0, 3.0, 0.0]
2.000,4.000,0.000 [1.9999999999999996, 4.0, 0.0]
2.000,3.000,0.000 [2.0, 3.0, 0.0]
0.500,0.500,0.000 [0.5, 0.5, 0.0]
2.500,4.500,0.000 [2.5, 4.5, 0.0]
4.000,0.000,0.000 [4.0, 0.0, 0.0]
4.000,1.000,0.000 [4.0, 1.0, 0.0]
4.000,4.000,0.000 [3.9999999999999996, 4.0, 0.0]
0.500,1.500,0.000 [0.5000000000000011, 1.5, 0.0]
1.500,4.500,0.000 [1.5, 4.5, 0.0]
2.000,2.000,0.000 [2.0, 2.0, 0.0]
4.500,1.500,0.000 [4.5, 1.5, 0.0]
0.000,1.000,0.000 [0.0, 1.0, 0.0]
3.000,2.000,0.000 [2.9999999999999996, 2.0000000000000004, 0.0]
1.500,3.500,0.000 [1.4999999999999998, 3.5, 0.0]
0.500,3.500,0.000 [0.4999999999999994, 3.5, 0.0]
3.000,0.000,0.000 [3.0, 0.0, 0.0]
0.000,2.000,0.000 [0.0, 2.0, 0.0]
0.000,5.000,0.000 [0.0, 5.0, 0.0]
2.500,0.500,0.000 [2.5, 0.4999999999999989, 0.0]
2.500,1.500,0.000 [2.5, 1.5, 0.0]
0.000,5.000,0.000 [0.0, 5.0, 0.0]
5.000,0.000,0.000 [5.0, 0.0, 0.0]
5.000,5.000,0.000 [5.0, 5.0, 0.0]
0.000,0.000,0.000 [0.0, 0.0, 0.0]
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
0.000,0.000,0.000 [0.0, 0.0, 0.0]
Meters
CPlane
Y: 63.663
Z: 0.000

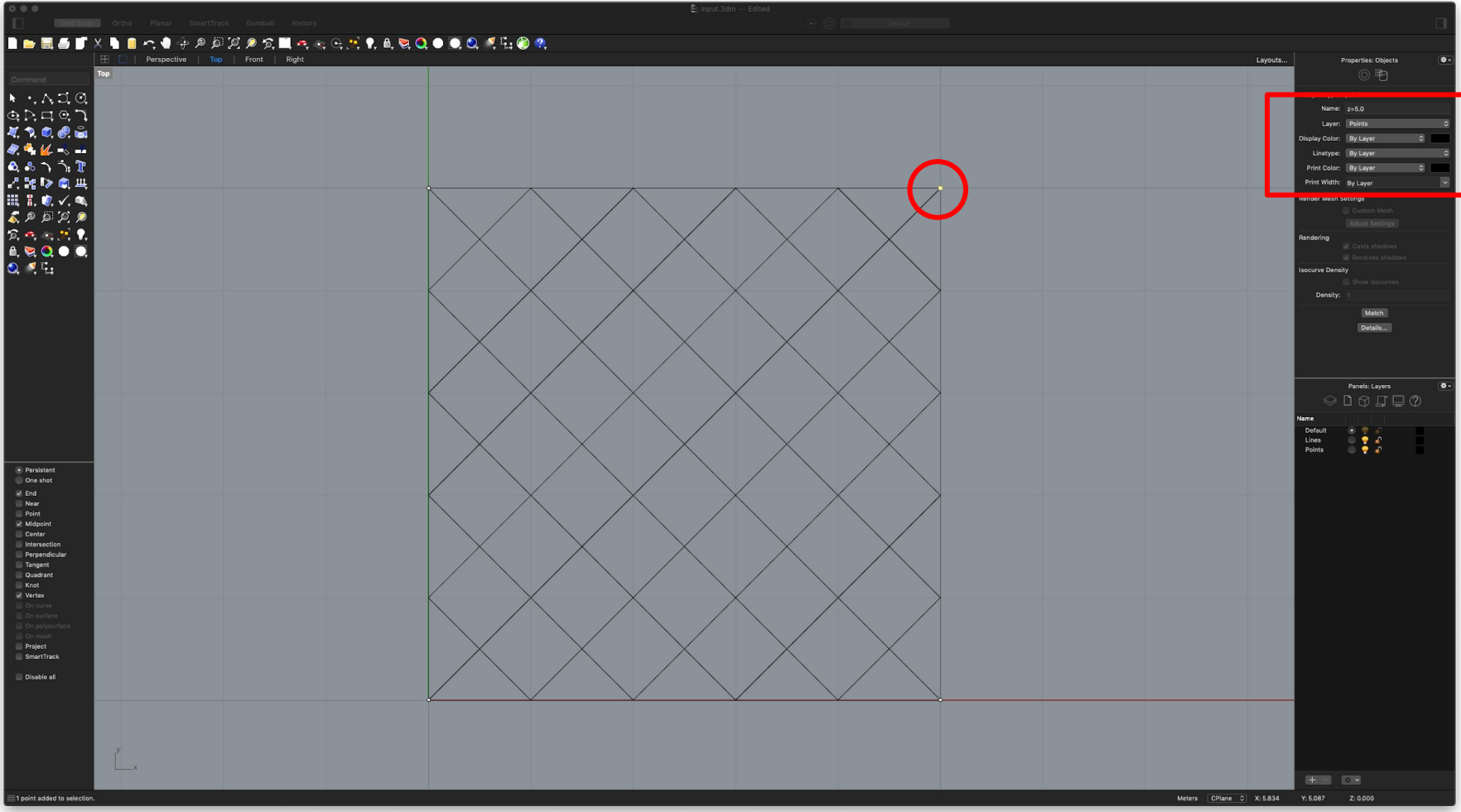
```
0.500,2.500,0.000 [0.50000000000000011, 2.5, 0.0]
0.000,0.000,0.000 [0.0, 0.0, 0.0]
2.500,2.500,0.000 [2.5, 2.5, 0.0]
4.500,4.500,0.000 [4.5, 4.5, 0.0]
5.000,2.000,0.000 [5.0, 2.0, 0.0]
3.000,5.000,0.000 [3.0, 5.0, 0.0]
1.500,2.500,0.000 [1.5, 2.5, 0.0]
1.000,0.000,0.000 [1.0, 0.0, 0.0]
1.000,5.000,0.000 [1.0, 5.0, 0.0]
4.000,2.000,0.000 [4.0, 1.9999999999999998, 0.0]
3.000,4.000,0.000 [3.0, 4.0, 0.0]
3.000,1.000,0.000 [3.0, 0.99999999999999978, 0.0]
2.000,0.000,0.000 [2.0, 0.0, 0.0]
5.000,1.000,0.000 [5.0, 1.0, 0.0]
2.000,1.000,0.000 [1.9999999999999998, 1.0, 0.0]
4.500,2.500,0.000 [4.5, 2.5, 0.0]
5.000,5.000,0.000 [5.0, 5.0, 0.0]
0.000,3.000,0.000 [0.0, 3.0, 0.0]
4.000,3.000,0.000 [4.0, 3.0, 0.0]
4.500,3.500,0.000 [4.5, 3.5, 0.0]
2.000,5.000,0.000 [2.0, 5.0, 0.0]
3.500,3.500,0.000 [3.5, 3.5, 0.0]
2.500,3.500,0.000 [2.5, 3.4999999999999996, 0.0]
```

```
0.000,5.000,0.000 [0.0, 5.0, 0.0]
5.000,0.000,0.000 [5.0, 0.0, 0.0]
5.000,5.000,0.000 [5.0, 5.0, 0.0]
0.000,0.000,0.000 [0.0, 0.0, 0.0]
```

```
0.500,2.500,0.000 [0.50000000000000011, 2.5, 0.0]
0.000,0.000,0.000 [0.0, 0.0, 0.0]
2.500,2.500,0.000 [2.5, 2.5, 0.0]
...
2.000,1.000,0.000 [1.9999999999999998, 1.0, 0.0]
4.500,2.500,0.000 [4.5, 2.5, 0.0]
5.000,5.000,0.000 [5.0, 5.0, 0.0]
...
0.500,4.500,0.000 [0.5, 4.5, 0.0]
5.000,0.000,0.000 [5.0, 0.0, 0.0]
1.500,1.500,0.000 [1.5, 1.5, 0.0]
5.000,4.000,0.000 [5.0, 4.0, 0.0]
3.000,0.000,0.000 [3.0, 0.0, 0.0]
...
0.000,2.000,0.000 [0.0, 2.0, 0.0]
0.000,5.000,0.000 [0.0, 5.0, 0.0]
2.500,0.500,0.000 [2.5, 0.49999999999999989, 0.0]
2.500,1.500,0.000 [2.5, 1.5, 0.0]
```

Object names => Attributes

3d_fofin_input.py


input.3dm — Edited
Grid Snap
Ortho
Planar
SmartTrack
Gumball
History
Perspective
Top
Front
Right
Layouts...
Top
Command
Properties: Objects
Name: z=5.0
Layer: Points
Display Color: By Layer
Linetype: By Layer
Print Color: By Layer
Print Width: By Layer
Render Mesh Settings
Custom Mesh
Adjust Settings
Rendering
Casts shadows
Receives shadows
Isocurve Density
Show isocurves
Density:
Match
Details...
Panels: Layers
Name
Default
Lines
Points
Persistent
One shot
End
Near
Point
Midpoint
Center
Intersection
Perpendicular
Tangent
Quadrant
Knot
Vertex
On curve
On surface
On polysurface
On mesh
Project
SmartTrack
Disable all
1 point added to selection.
Meters
CPlane
X: 5.834
Y: 5.087
Z: 0.000

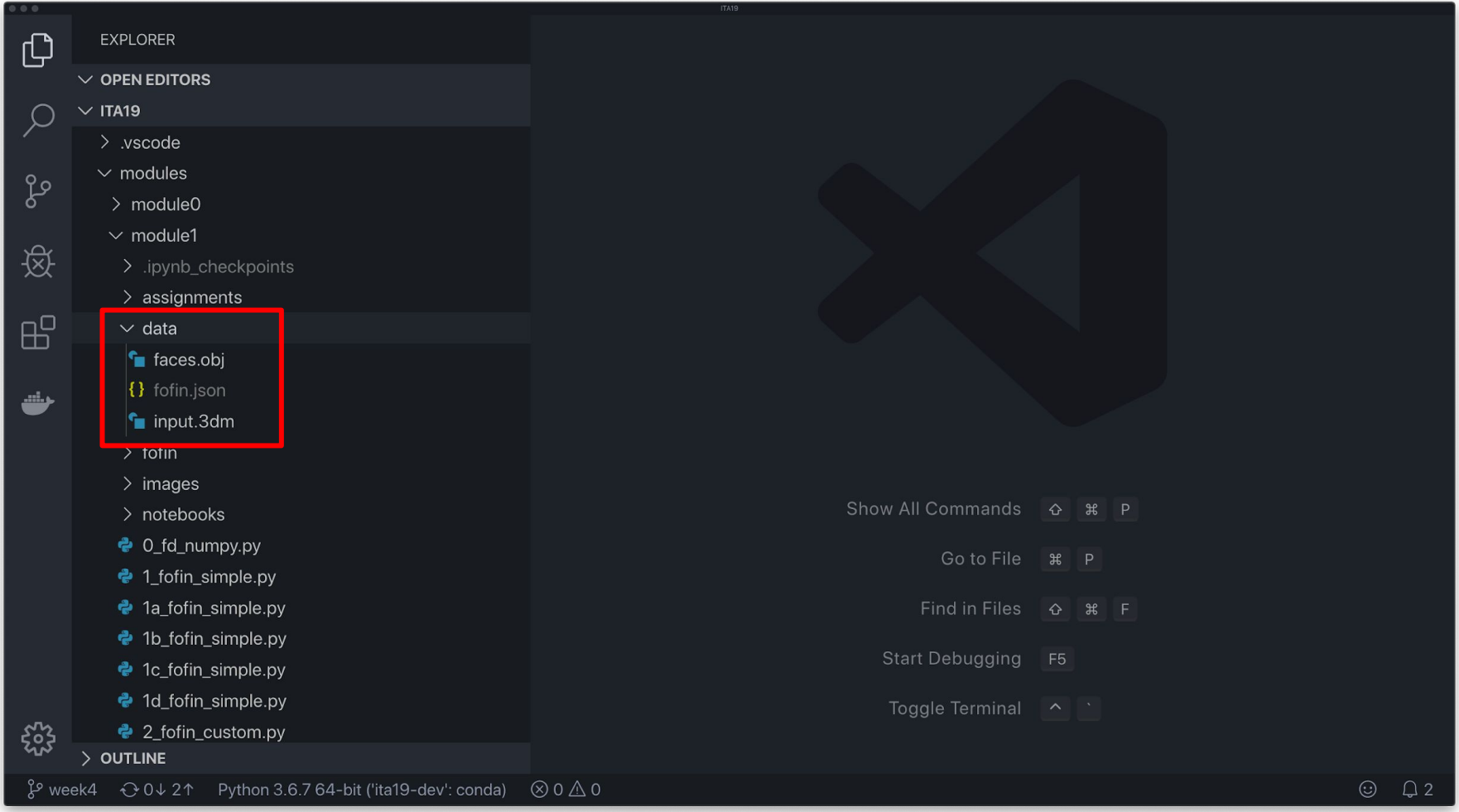
```
# ==============================================================================
# Input
# ==============================================================================

guids = compas_rhino.select_lines()
lines = compas_rhino.get_line_coordinates(guids)

guids = compas_rhino.select_points()
points = compas_rhino.get_point_coordinates(guids)
names = compas_rhino.get_object_names(guids)
```

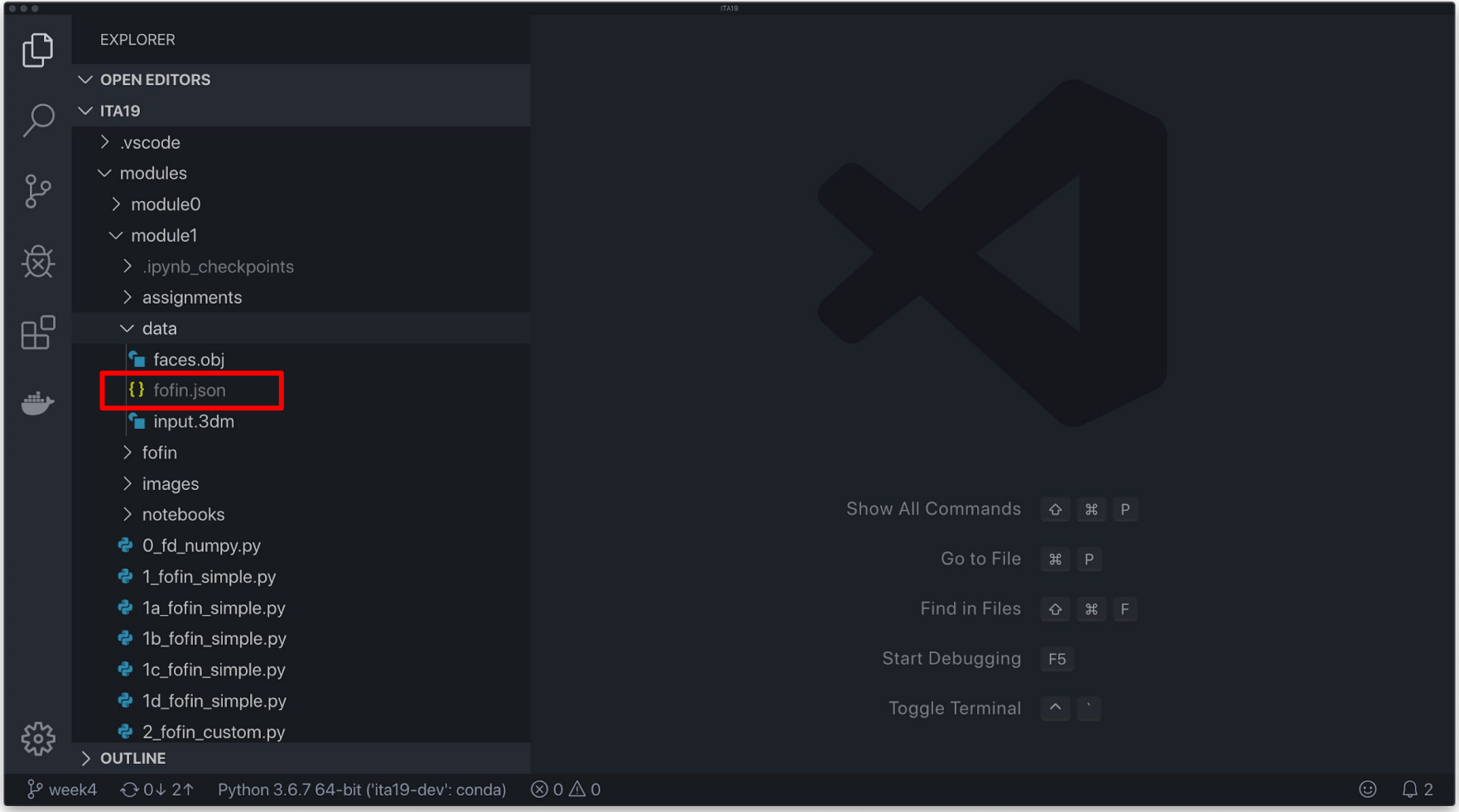
```python
# ==============================================================================
# Vertex attributes
# ==============================================================================

for name, point in zip(names, points):
    gkey = geometric_key(point)
    if gkey in gkey_key:
        key = gkey_key[gkey]
        shell.set_vertex_attribute(key, 'is_fixed', True)
        if name:
            parts = name.split('=')
            if len(parts) == 2:
                z = float(parts[1])
                shell.set_vertex_attribute(key, 'z', z)
```

4_fofin_output.py

```
# ==============================================================================
# Fofin run
# ==============================================================================

shell.fofin()

# ==============================================================================
# Export result
# ==============================================================================

shell.to_json(FILE)

# ==============================================================================
# Visualize result
# ==============================================================================

artist = ShellArtist(shell, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
artist.draw_forces(scale=0.01)
artist.draw_reactions(scale=0.1)
```

```
# ================================================================================
# Output
# ================================================================================

HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'fofin.json')
```


ITA19
EXPLORER
OPEN EDITORS
ITA19
.vscode
modules
module0
module1
.ipynb_checkpoints
assignments
data
faces.obj
fofin.json
input.3dm
fofin
images
notebooks
0_fd_numpy.py
1_fofin_simple.py
1a_fofin_simple.py
1b_fofin_simple.py
1c_fofin_simple.py
1d_fofin_simple.py
2_fofin_custom.py
OUTLINE
Show All Commands ⇧ ⌘ P
Go to File ⌘ P
Find in Files ⇧ ⌘ F
Start Debugging F5
Toggle Terminal ^ `
week4
0↓ 2↑
Python 3.6.7 64-bit ('ita19-dev': conda)
⊗ 0 ⚠ 0
2


ITA19
EXPLORER
OPEN EDITORS
ITA19
.vscode
modules
module0
module1
.ipynb_checkpoints
assignments
data
faces.obj
fofin.json
input.3dm
fofin
images
notebooks
0_fd_numpy.py
1_fofin_simple.py
1a_fofin_simple.py
1b_fofin_simple.py
1c_fofin_simple.py
1d_fofin_simple.py
2_fofin_custom.py
OUTLINE
Show All Commands ⇧ ⌘ P
Go to File ⌘ P
Find in Files ⇧ ⌘ F
Start Debugging F5
Toggle Terminal ^ `
week4
0↓ 2↑
Python 3.6.7 64-bit ('ita19-dev': conda)
0 0
2

{"dfa": {}, "vertex": {"60": {"z": 0.0, "rx": 7.2023225957126566, "px": 0.0, "is_fixed": true, "rz": 13.992044047290246, "x": 0.0, "py": 0.0, "y": 5.0, "sorted_neighbors": [28, 32, 8], "pz": 0.0, "ry": -7.2023225957126655}, "21": {"z": 3.1057262455824226, "rx": 0.0, "px": 0.0, "is_fixed": false, "rz": -1.3322676295501878e-15, "x": 3.3857783785423603, "py": 0.0, "y": 3.3857783785423612, "sorted_neighbors": [18, 49, 10, 38], "pz": 0.0, "ry": -1.3322676295501878e-15}, "36": {"z": 3.9776614498683327, "rx": 7.1054273576010019e-15, "px": 0.0, "is_fixed": false, "rz": 7.1054273576010019e-15, "x": 3.9828746462927458, "py": 0.0, "y": 4.6994639352859977, "sorted_neighbors": [5, 35, 3, 16], "pz": 0.0, "ry": 0.0}, "52": {"z": 5.4505968824998314, "rx": 0.0, "px": 0.0, "is_fixed": false, "rz": -1.4210854715202004e-14, "x": 0.30053606471400141, "py": 0.0, "y": 1.0171253537072533, "sorted_neighbors": [2, 43, 47, 57], "pz": 0.0, "ry": 0.0}, "10": {"z": 2.9944291337009705, "rx": -2.2204460492503131e-15, "px": 0.0, "is_fixed": false, "rz": 4.4408920985006262e-16, "x": 2.9586357906322949, "py": 0.0, "y": 3.7860739349631496, "sorted_neighbors": [21, 35, 44, 22], "pz": 0.0, "ry": -8.8817841970012523e-16}, "14": {"z": 3.6923130239391471, "rx": -4.4408920985006262e-16, "px": 0.0, "is_fixed": false, "rz": 3.1086244689504383e-15, "x": 2.0413642093677042, "py": 0.0, "y": 1.2139260650368497, "sorted_neighbors": [59, 58, 34, 25], "pz": 0.0, "ry": -9.9920072216264089e-16}, "13": {"z": 1.1425529760172668, "rx": 4.4408920985006262e-15, "px": 0.0, "is_fixed": false, "rz": 6.6613381477509392e-16, "x": 4.6994639352859977, "py": 0.0, "y": 1.0171253537072540, "sorted_neighbors": [4, 51, 26, 33], "pz": 0.0, "ry": 0.0}, "57": {"z": 4.0675461668162685, "rx": 8.8817841970012523e-16, "px": 0.0, "is_fixed": false, "rz": 0.0, "x": 0.44365744600435369, "py": 0.0, "y": 2.0080507049734959, "sorted_neighbors": [52, 47, 0, 17], "pz": 0.0, "ry": 7.1054273576010019e-15}, "31": {"z": 3.0482294272074109, "rx": -7.1054273576010019e-15, "px": 0.0, "is_fixed": false, "rz": -3.5527136788005009e-15, "x": 4.5563425539956457, "py": 0.0, "y":

```
# ==============================================================================
# Fofin run
# ==============================================================================

shell.fofin()

# ==============================================================================
# Export result
# ==============================================================================

shell.to_json(FILE, pretty=True)

# ==============================================================================
# Visualize result
# ==============================================================================

artist = ShellArtist(shell, layer="Mesh")
artist.clear_layer()
artist.draw_vertices()
artist.draw_edges()
artist.draw_forces(scale=0.01)
artist.draw_reactions(scale=0.1)
```

```
{
    "attributes": {
        "name": "Mesh"
    },
    "dea": {
        "f": 0.0,
        "q": 1.0,
        "rx": 0.0,
        "ry": 0.0,
        "rz": 0.0
    },
    "dfa": {},
    "dva": {
        "is_fixed": false,
        "px": 0.0,
        "py": 0.0,
        "pz": 0.0,
        "x": 0.0,
        "y": 0.0,
        "z": 0.0
    },
    "edgedata": {
        "(0, 17)": {
```

5_fofin_interactive.py

```
# ============================================================================
# I/O
# ============================================================================

HERE = os.path.dirname(__file__)
DATA = os.path.join(HERE, 'data')
FILE = os.path.join(DATA, 'fofin.json')

# ============================================================================
# Shell
# ============================================================================

shell = Shell.from_json(FILE)

...

# ============================================================================
# Export result
# ============================================================================

shell.to_json(FILE)
```

Assignments

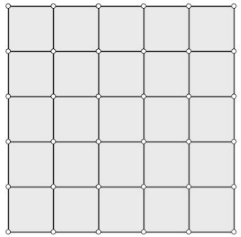
Traverse Quad Mesh in a Straight Line

In [139]:
```
import compas
from compas.datastructures import Mesh
from compas_plotters import MeshPlotter
```

Create Mesh from `faces.obj`

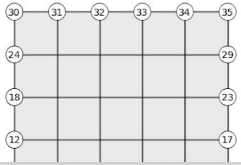
In [140]:
```
mesh = Mesh.from_obj(compas.get('faces.obj'))
```

In [141]:
```
plotter = MeshPlotter(mesh, figsize=(4, 4))
plotter.draw_vertices()
plotter.draw_edges()
plotter.draw_faces()
plotter.show()
```



Visualize the vertices on the boundary and label with their identifier.

In [142]:
```
plotter = MeshPlotter(mesh, figsize=(4, 4))
plotter.draw_vertices(radius=0.4, text='key', keys=list(mesh.vertices_on_boundary()))
plotter.draw_edges()
plotter.draw_faces()
plotter.show()
```



Update Selfweight

1. Apply selfweight as load
    a. Compute selfweight per node (tributary area)
    b. Apply selfweight (attribute ‘pz’)
    c. Compute equilibrium
    d. Visualize residual forces (attributes ‘rx’, ‘ry’, ‘rz’)

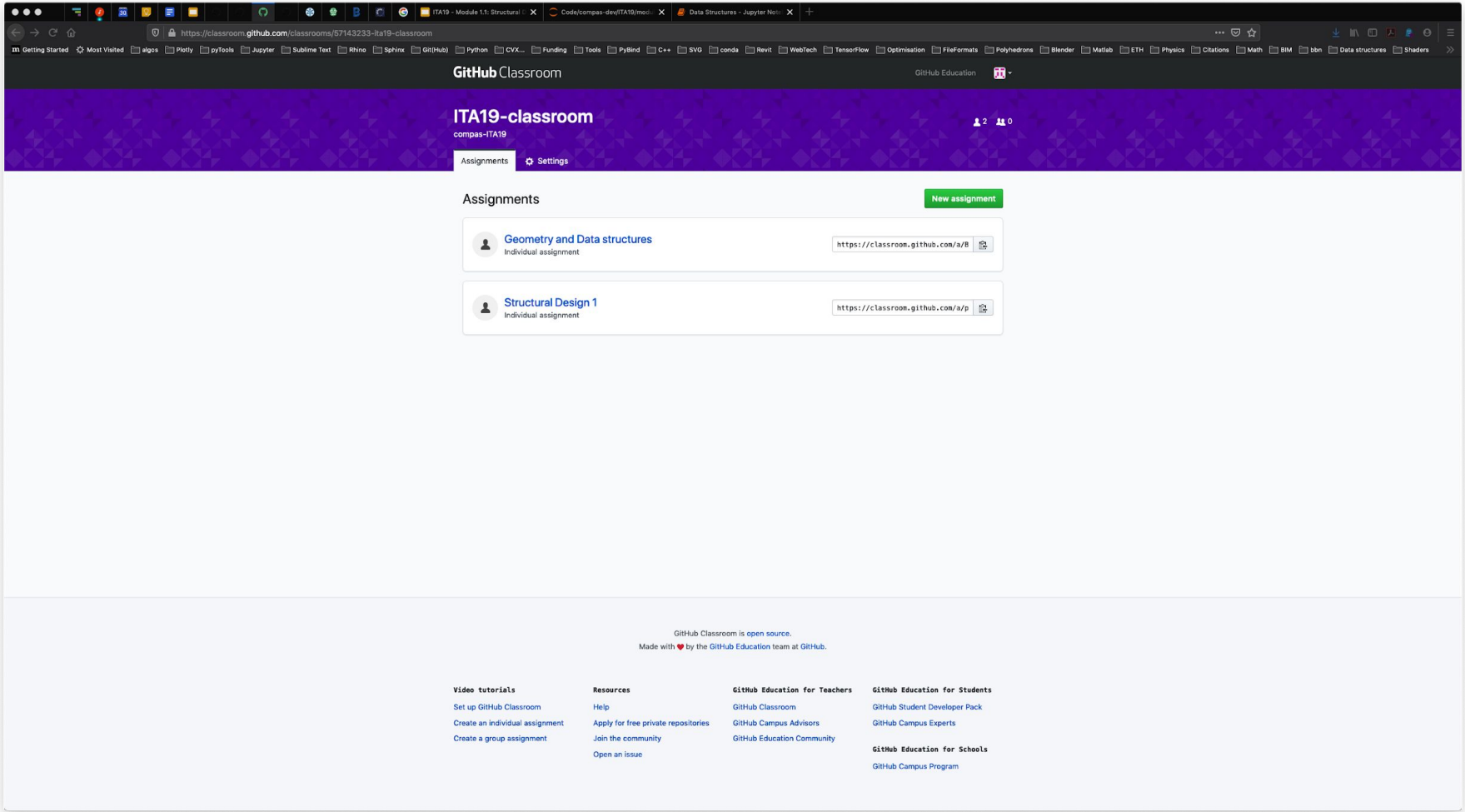
2. Find shape in equilibrium with selfweight
    a. Recompute equilibrium for new selfweight
    b. Recompute selfweight
    c. Repeat until convergence

Update Force Density of Cables

1. Select cable
    a. Select edge
    b. Find rest of cable

2. Update force density of all edges in cable

3. Update equilibrium


https://classroom.github.com/classrooms/57143233-ita19-classroom
GitHub Classroom
GitHub Education
ITA19-classroom
compas-ITA19
2
0
Assignments
Settings
Assignments
New assignment
Geometry and Data structures
Individual assignment
https://classroom.github.com/a/B
Structural Design 1
Individual assignment
https://classroom.github.com/a/p
GitHub Classroom is open source.
Made with by the GitHub Education team at GitHub.
Video tutorials
Set up GitHub Classroom
Create an individual assignment
Create a group assignment
Resources
Help
Apply for free private repositories
Join the community
Open an issue
GitHub Education for Teachers
GitHub Classroom
GitHub Campus Advisors
GitHub Education Community
GitHub Education for Students
GitHub Student Developer Pack
GitHub Campus Experts
GitHub Education for Schools
GitHub Campus Program