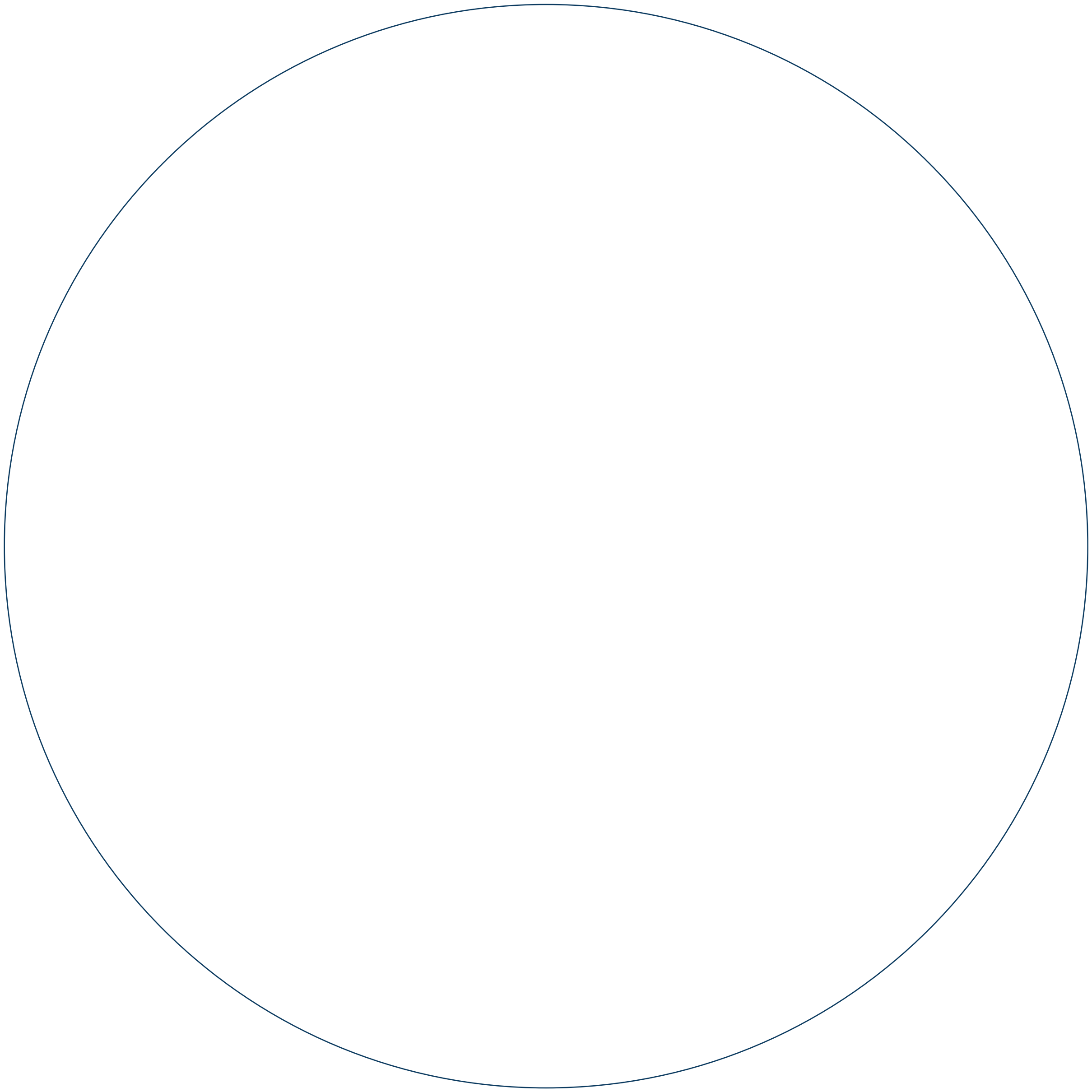


On Cirlcles

ERICSON

A circle is a plane figure contained by one line such that all straight lines falling upon it from a point among those lying with the figure are equal to one another; And the point is called the the centre of the circle.

(Euclid, Elements, Book One Definitions 16 and 17, 300 B.C.E)

ERICSON

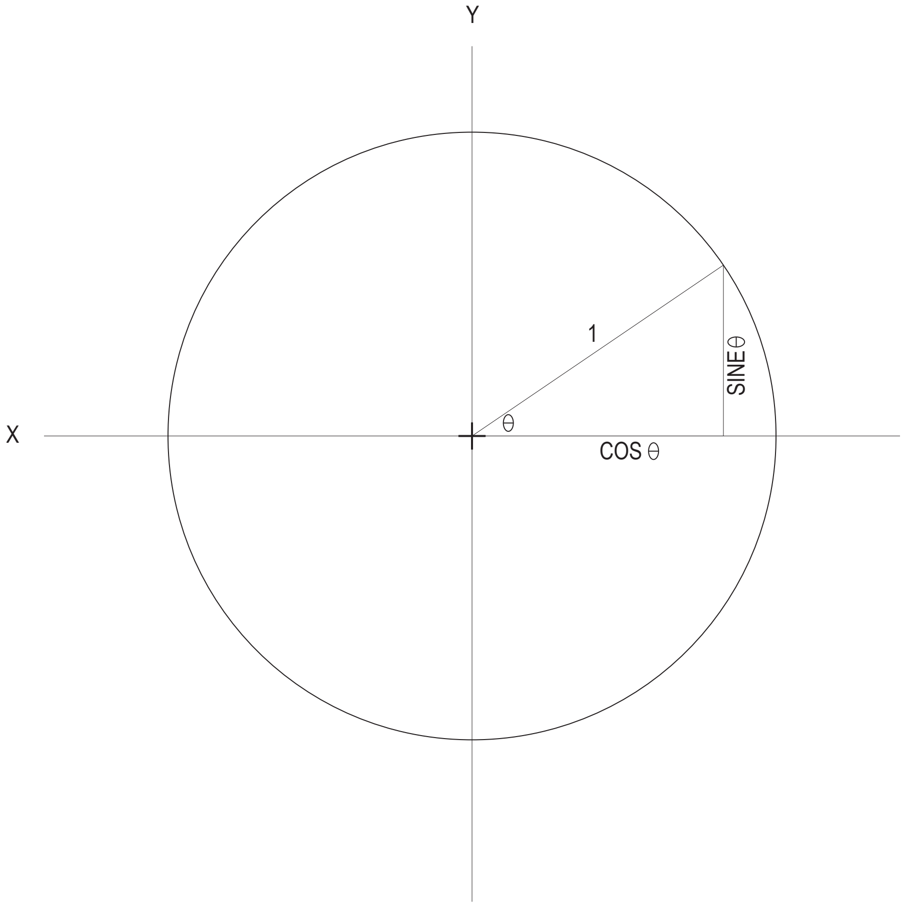
Thus, for instance, nothing is more opposed in geometry than “straightness” and “curvilinearity”; and yet in the infinitely great circle the circumfrence coincides with the tangent, and the infinitely small one, with the diameter.

(Nicholas of Cusa 1440, paraphrased by, Alexandre Koyre in From the Closed World to the Infiinite Universe, 1957)

The next simplest curve is the circle. Even so simple a figure as this has given rise to so many and such profound investigations that they could constitute a course all by themselves. We define the circle as the curve whose points are of constant distance from a given point. We generate it by means of the well-known construction using a thread or a compass. From this construction it is evident that the circle is a closed curve that is everywhere convex.

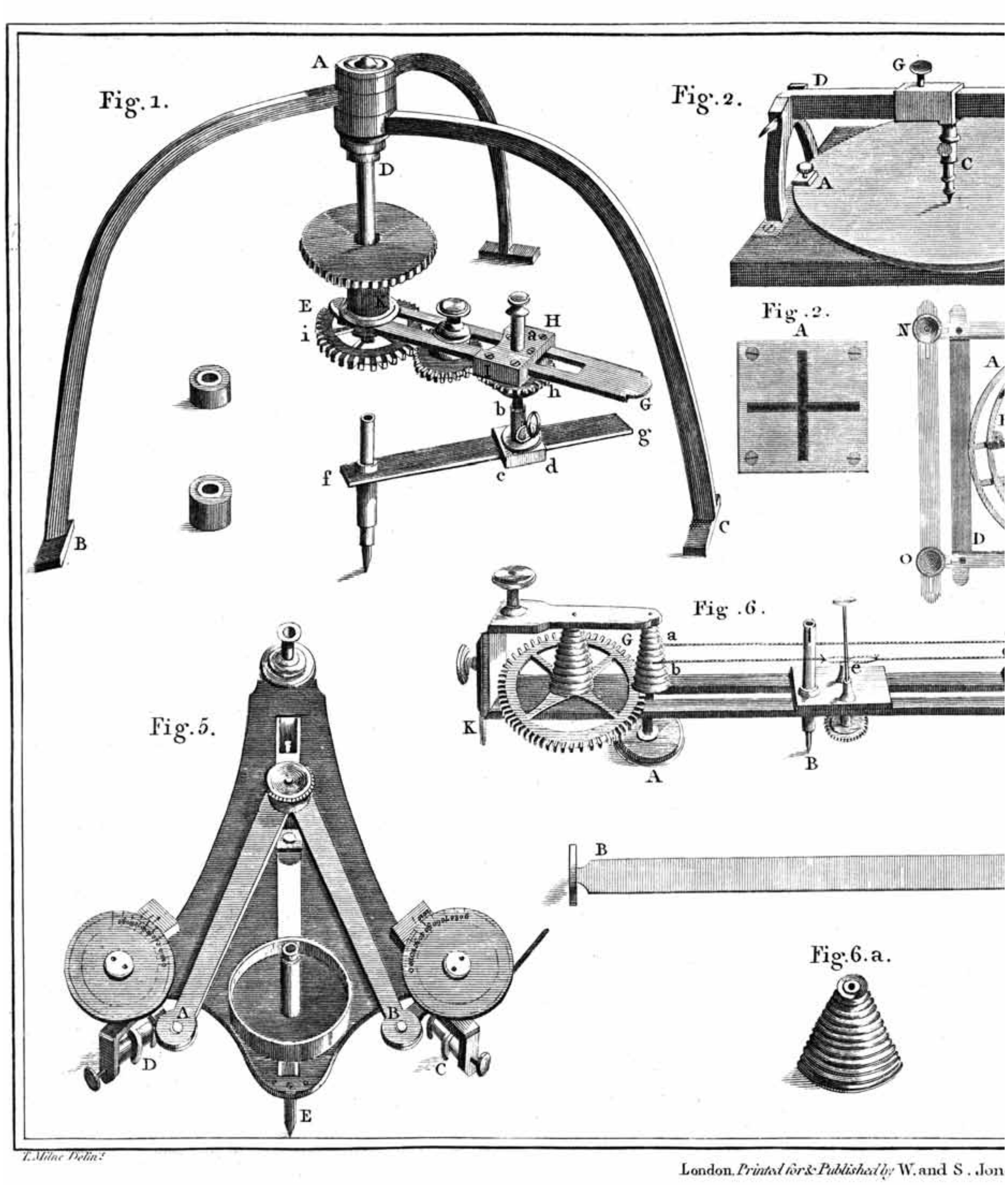
(Hilbert and S. Cohn-Vossen, Geometry and The Imagination)

ERICSON


Y
X
1
θ
SINE θ
COS θ

And nowadays, in our digital age, doe the computer compel us to think and live in a multidimensional, non-Euclidean topological space? Or should we instead consider the computer as a variable compass that will open up new potentialities of old Euclidean space? These questions require that we investigate in close detail what happened to geometry, and the answers are all the more important as much as geometry still remains the very basic tool of architecture.

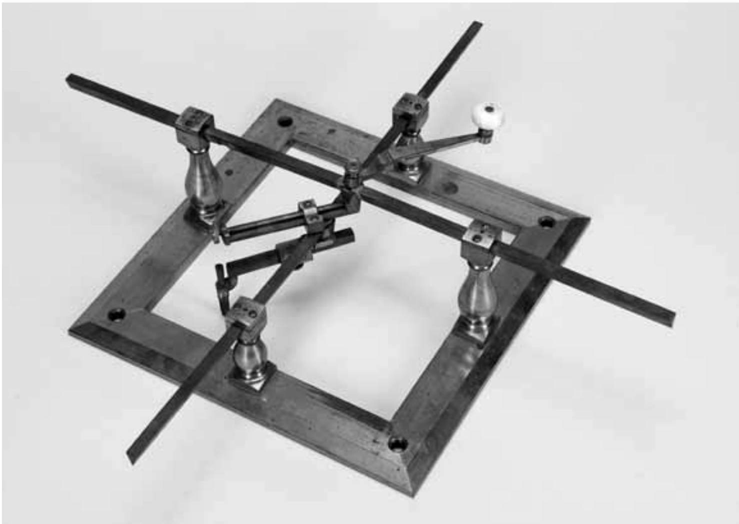
(Bernard Cache, “Plea for Euclid”, 1999)



Drawing Machines of Giambattista Surdis from Witt, Andrew. 2010. “Machine Epistemology in Architecture.” Translated by Annette Wiethuchter. Candide-Journal for Architectural Knowledge 12 (3): 37–88.

Dieser Ellipsograf, der um 1813 hergestellt wurde, mit dem sich Kurven durch mechanische Teilbewegungen darstellen lassen, hat große Ähnlichkeit mit Giambattista Suardis „geometrischem Schreiber“.

This ellipsograph, dating to circa 1813, resembles closely the design proposed by Giambattista Suardi for a geometrical pen, capable of the description of curves of compound mechanical motion.



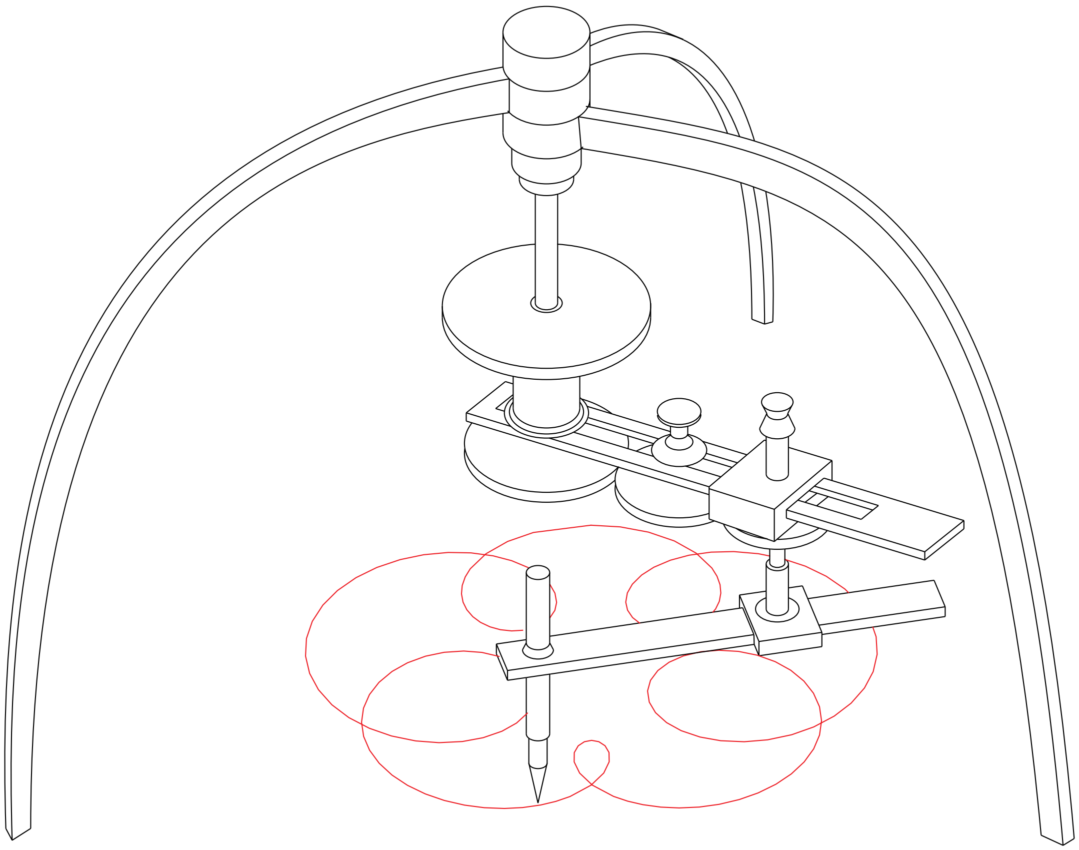
Drawing Machines of Giambattista Surdis from Witt, Andrew. 2010. “Machine Epistemology in Architecture.” Translated by Annette Wiethuchter. Candide-Journal for Architectural Knowledge 12 (3): 37–88.

Suardis „geometrischer Schreiber“

Der „geometrische Schreiber“ wurde 1750 vom Italiener Giambattista Suardi entwickelt und ist ein Vorläufer zahlreicher späterer Zeichengeräte für komplexe Kurven, insbesondere für den Einsatz mehrerer miteinander verbundener Teilbewegungen. Siehe auch Seite 79.

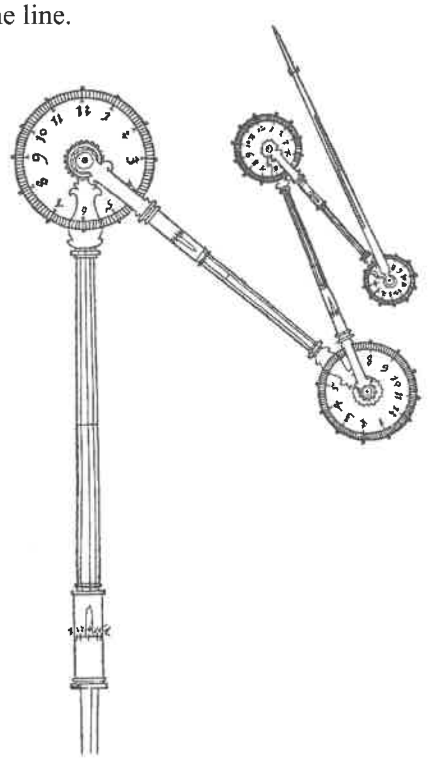
Suardi's Geometrical Pen

The geometrical pen (1750) was in instrument designed by the Italian Giambattista Suardi. This geometrical pen anticipated many of the later developments in instruments for drawing complex curves, including notably the employment of compound motion. See also page 79.



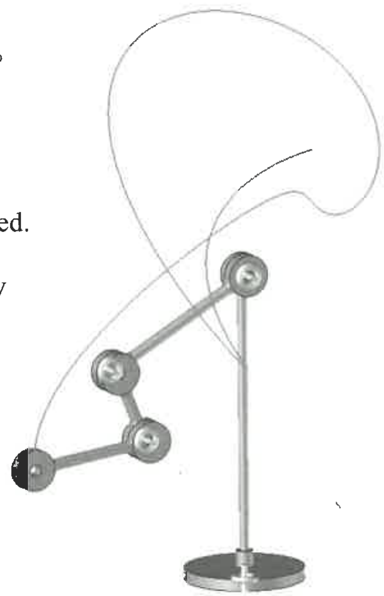
Drawing Machines of Giambattista Surdis from Witt, Andrew. 2010. “Machine Epistemology in Architecture.” Translated by Annette Wiethuchter. Candide-Journal for Architectural Knowledge 12 (3): 37–88.

In the treatise on human proportions, Dürer rejects the idea of one single perfect system of proportions. There are differences between men and women, between adults and children, and also between different temperaments. For Dürer, there is only a system of variation. Besides static topological deformations, there are deformations of the human body with motion. In order to show this, he needs an instrument that generates lines in 3D space. This is our third animal —the snake or the serpentine line.



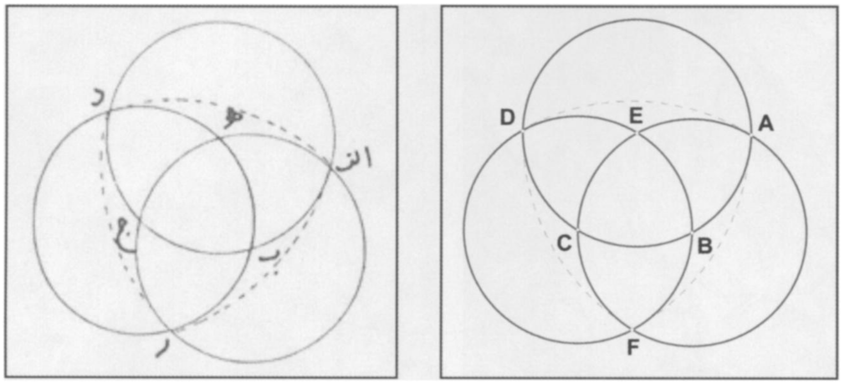
This is the only picture that Dürer provides that is related to the serpentine.[20] Strangely enough, he does not provide any drawing of the line itself. We have to use contemporary software in order to visualize it. Below are only two examples of the infinite variety of curves that can be generated with such an instrument.

This is described by Dürer in a parametric way: "The rods shall be arranged in a manner that they can be advanced by degrees and can be shortened or extended. The instrument should also be made with few or many dials or rods, according to the intended applications. The rods can be pulled apart or pushed together, also by degree, so that they become shorter or longer."[21] This is perfectly compatible with what we can do with mechanical software today.



Drawing Machines of Albrecht Durer from Cache, Bernard. 2012. "Instruments of Thought: Another Classical Tradition." Edited by Caroline O'Donnell. Cornell Journal of Architecture, no. 9.

Figure 2. Tiling of the sphere with 90° spherical equilateral triangles.

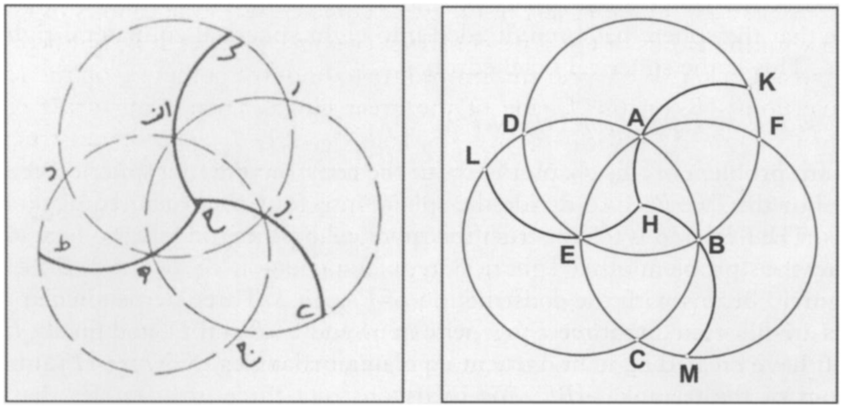


through AH, BH, and EH, respectively. Each circle passes through the centroid of a neighboring triangle, which has only a vertex in common with the original triangle. Name them K (ک), L (ط), and M (ع). Then $\Delta\ KLM$ is one face of our spherical tetrahedron. This process will divide the sphere into four congruent equilateral triangles: a spherical tetrahedron.

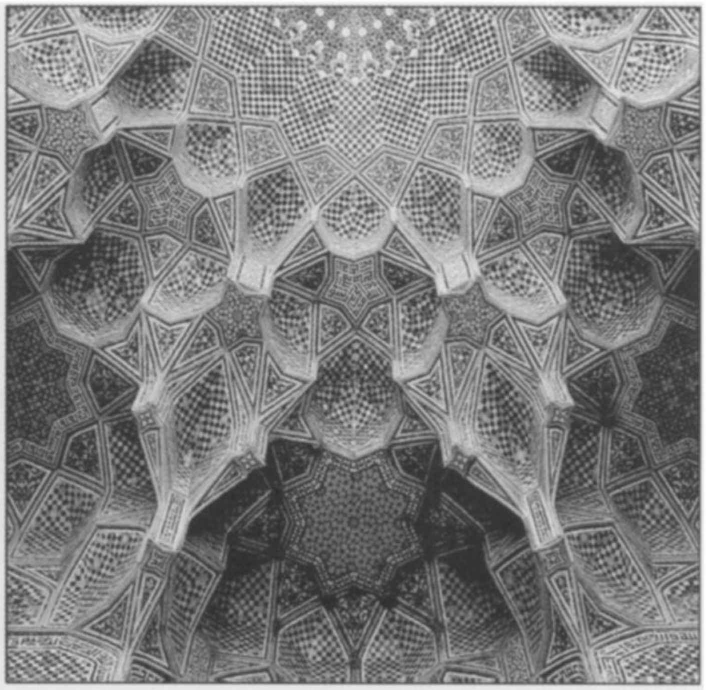
Tiling with Six Squares

To construct a spherical hexahedron (cube), we first construct the spherical octahedron. Then we connect the centroid of each pair of neighboring triangles that share a common side by an arc from a great circle. This will divide the sphere into six congruent spherical squares (Figure 4).

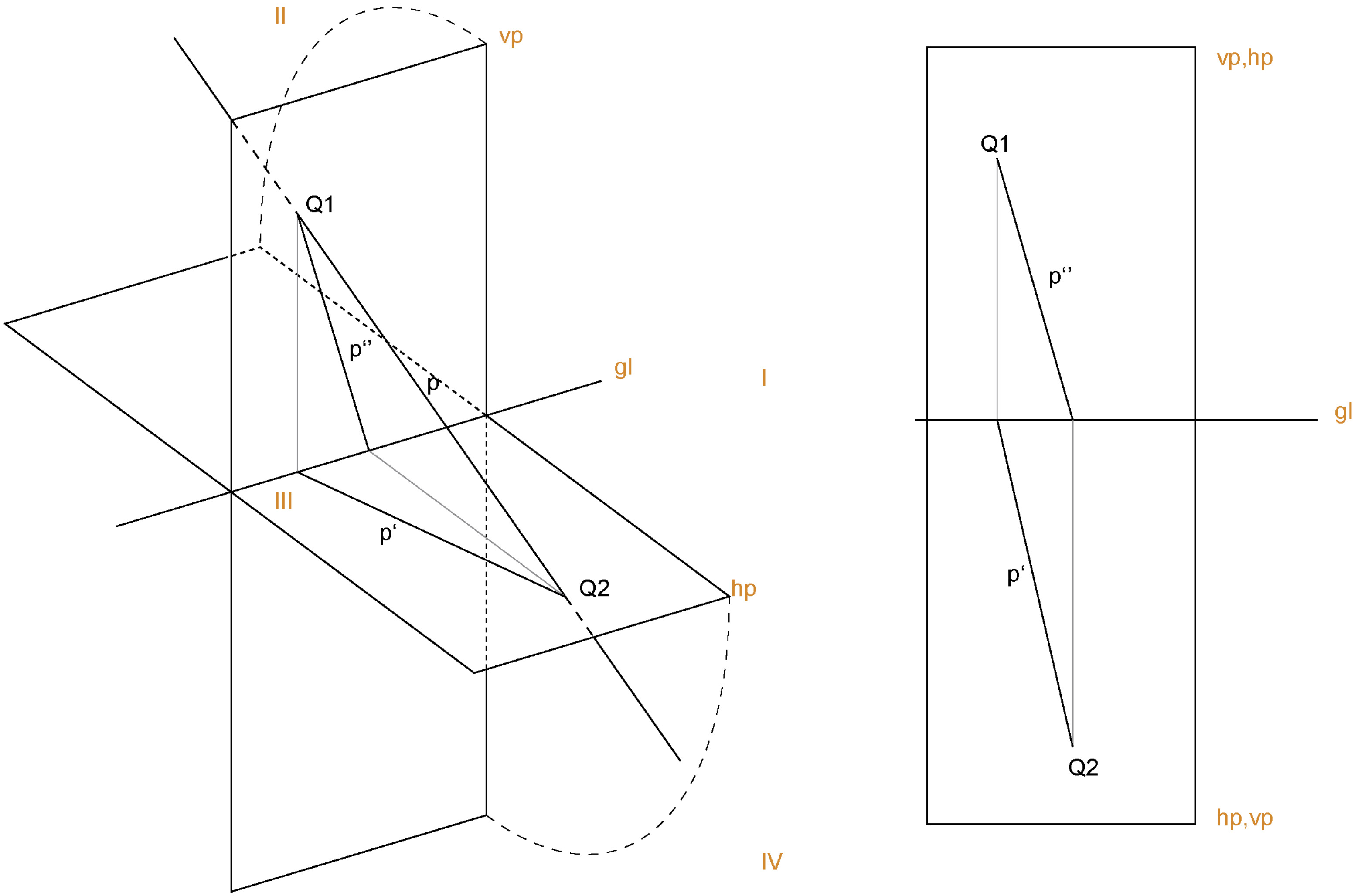
Figure 3. Tiling of the sphere with 120° spherical equilateral triangles.



Sarhangi, Reza. 2008. "Illustrating Abu Al-Wafa Buzjani: Flat Images, Spherical Constructions." Iranian Studies 14 (4): 511–23.

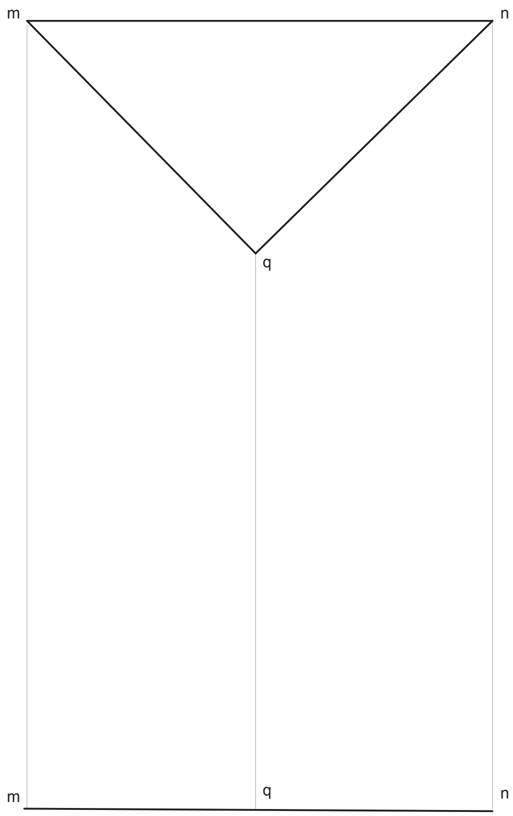


Sarhangi, Reza. 2008. "Illustrating Abu Al-Wafa Buzjani: Flat Images, Spherical Constructions." Iranian Studies 14 (4): 511–23.


II
vp
Q1
p‘‘
p
gl
I
III
p‘
Q2
hp
IV
vp,hp
Q1
p‘‘
gl
p‘
Q2
hp,vp

Line a,b

Plane m,n,q



ERICSON

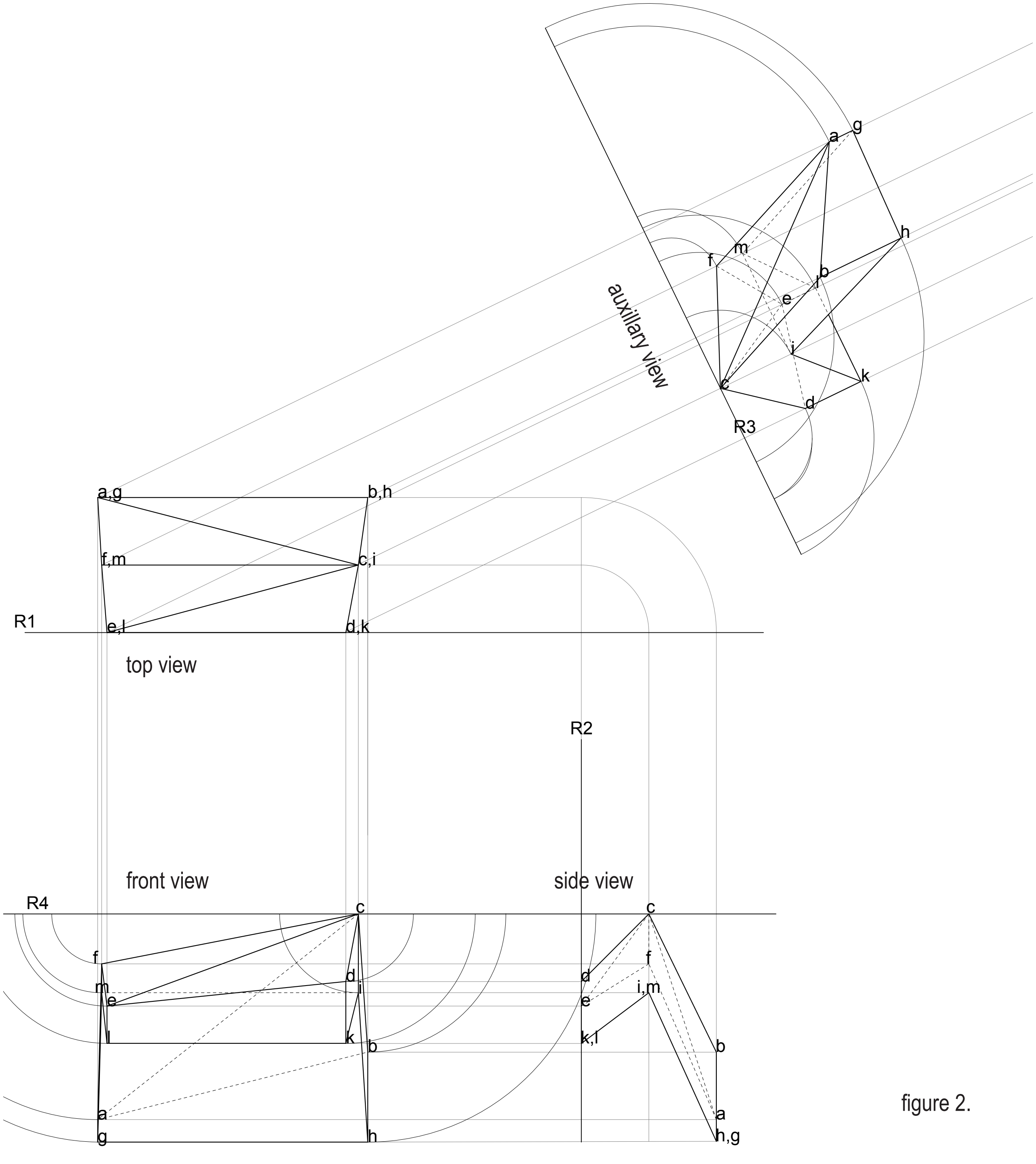
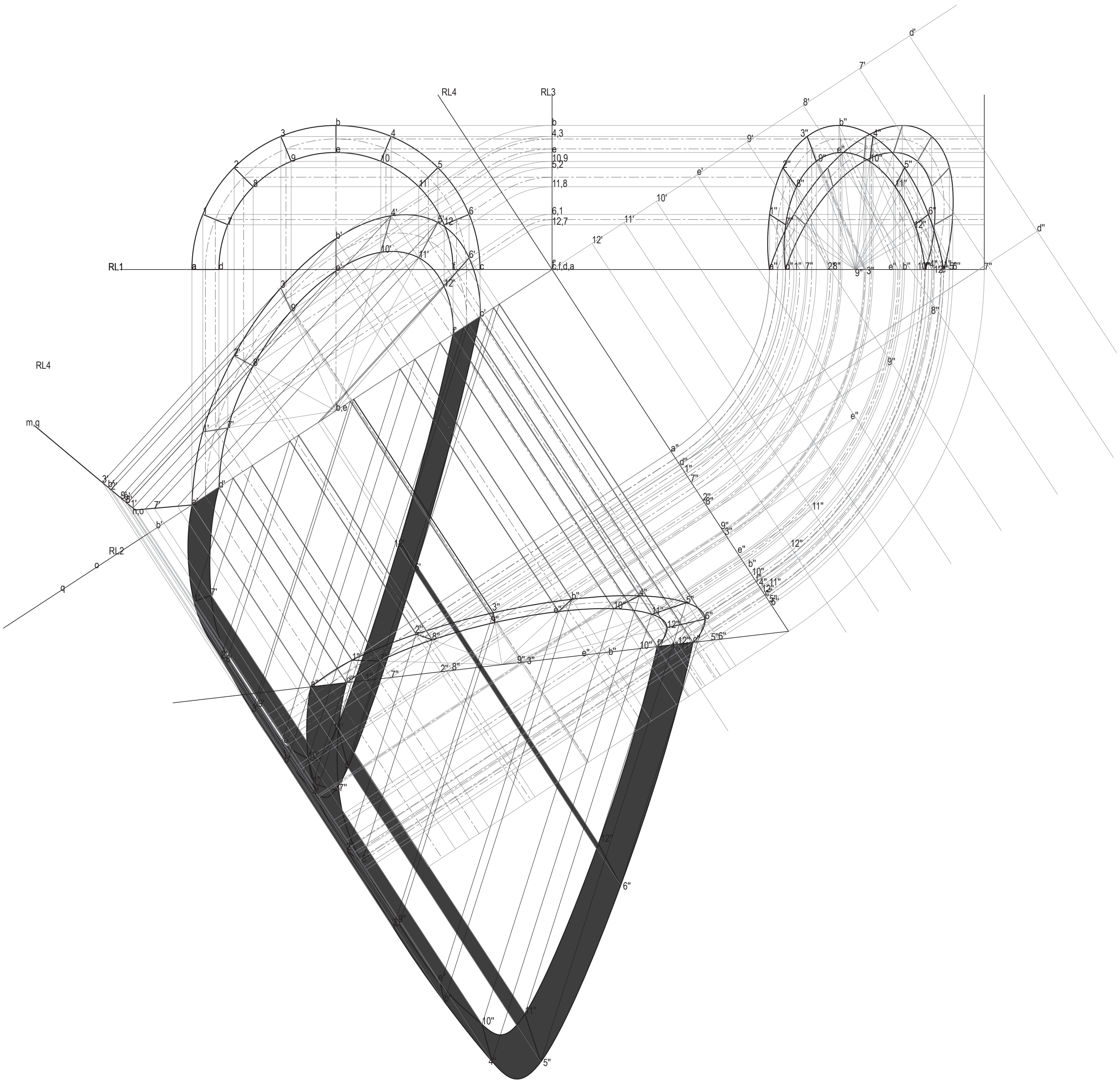
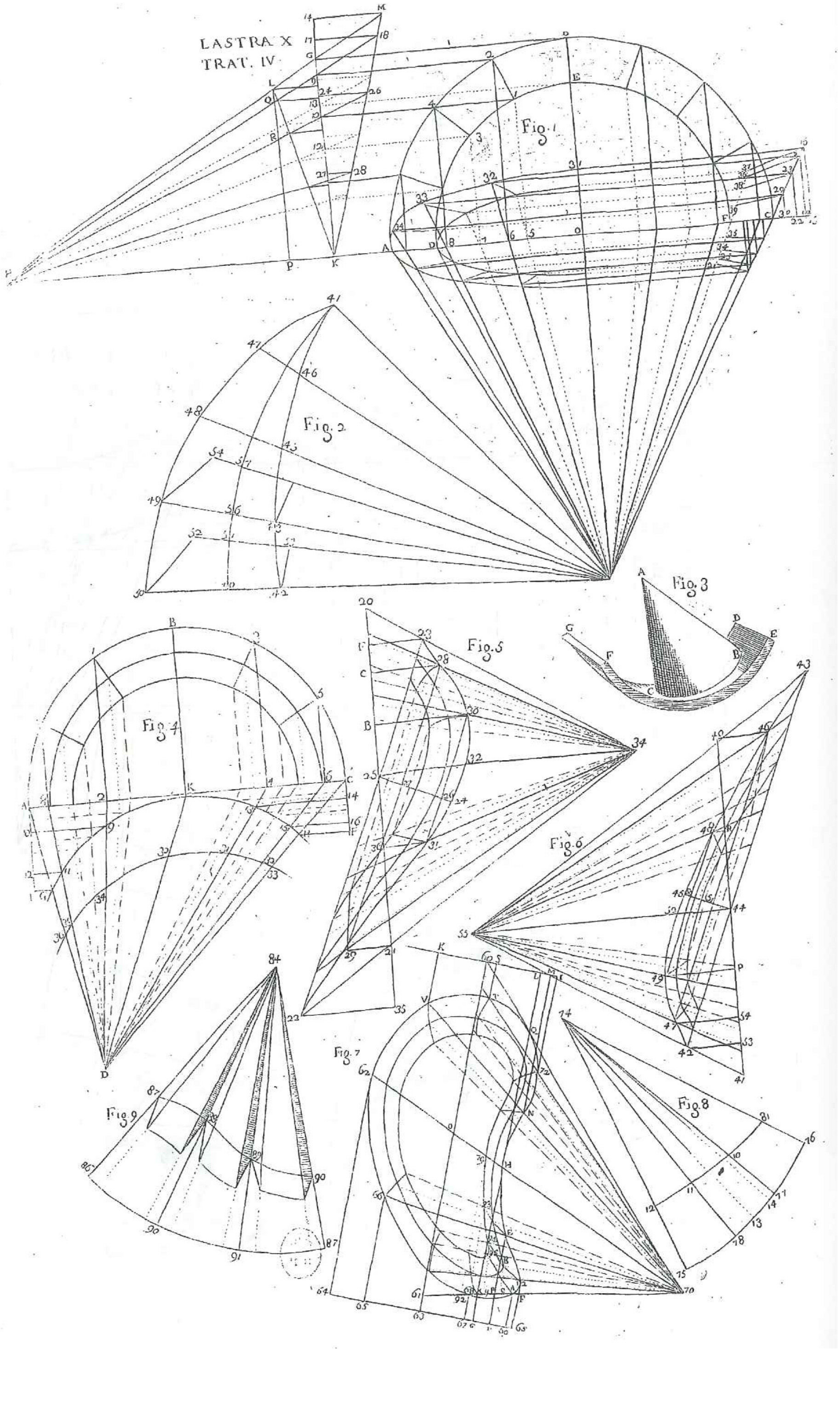

auxiliary view
top view
front view
side view
R1
R2
R3
R4
a,g
b,h
f,m
c,i
e,l
d,k

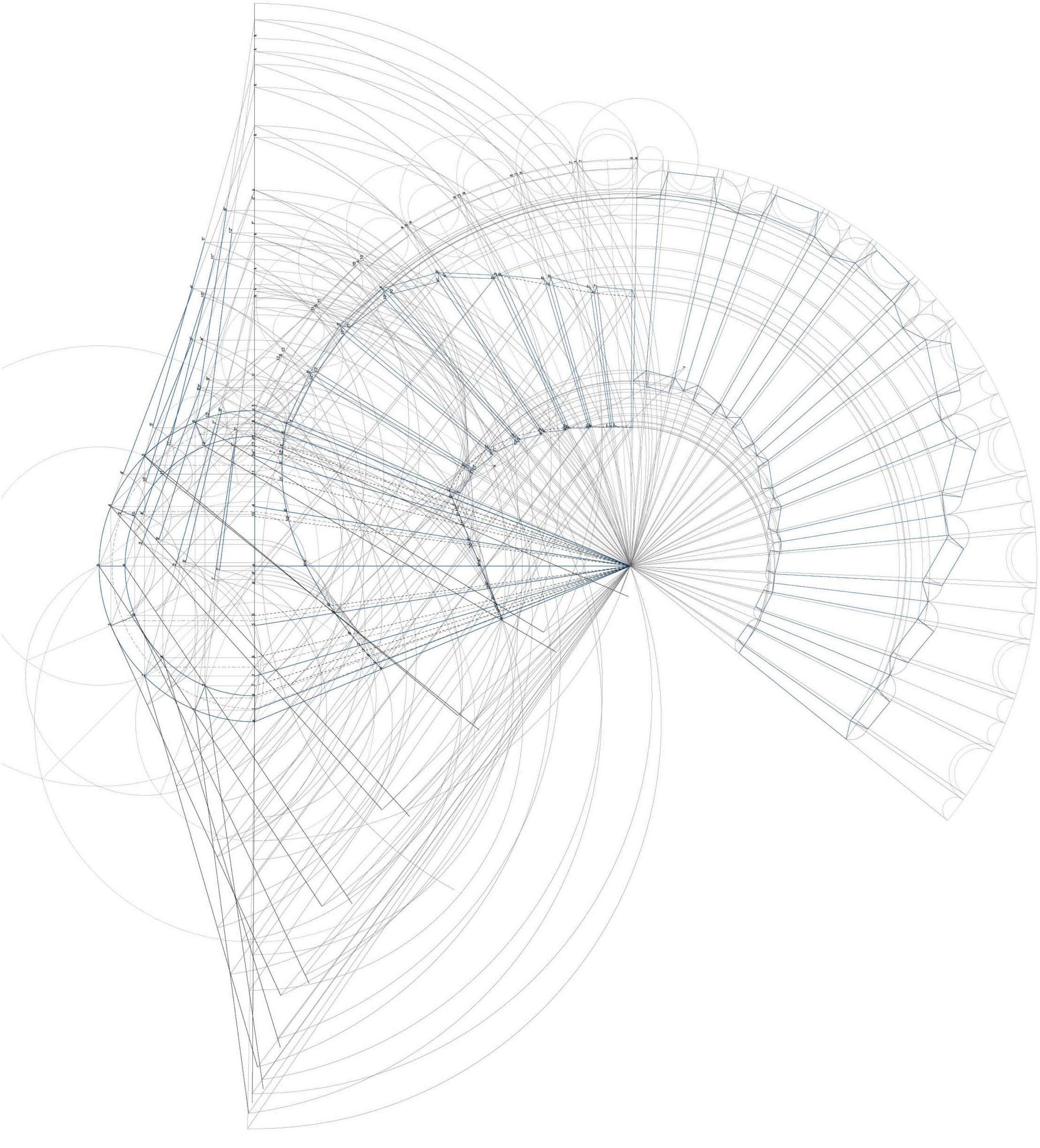
figure 2.



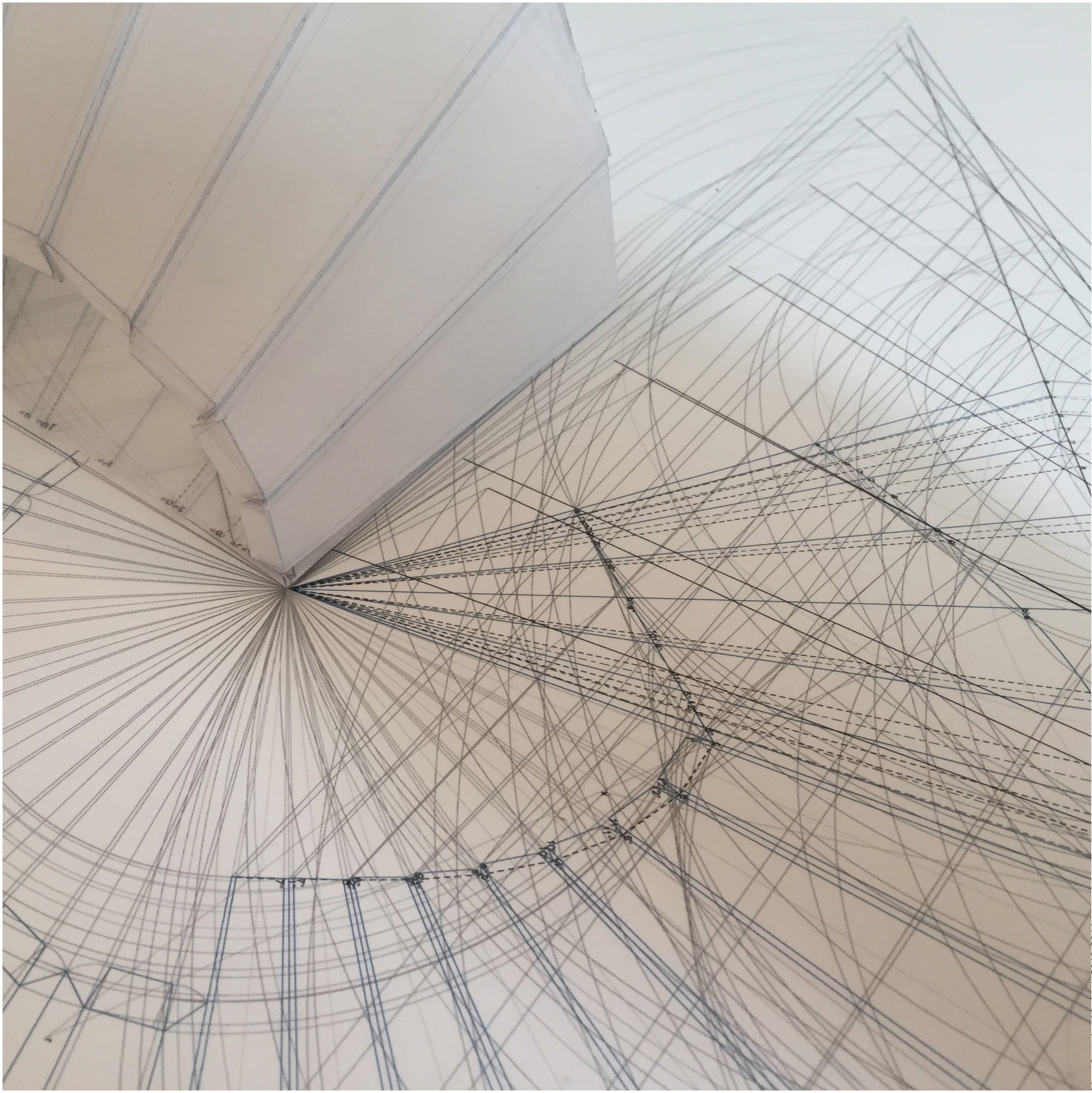
The Stereotomic Primer, 2012 -2015
ERICSON



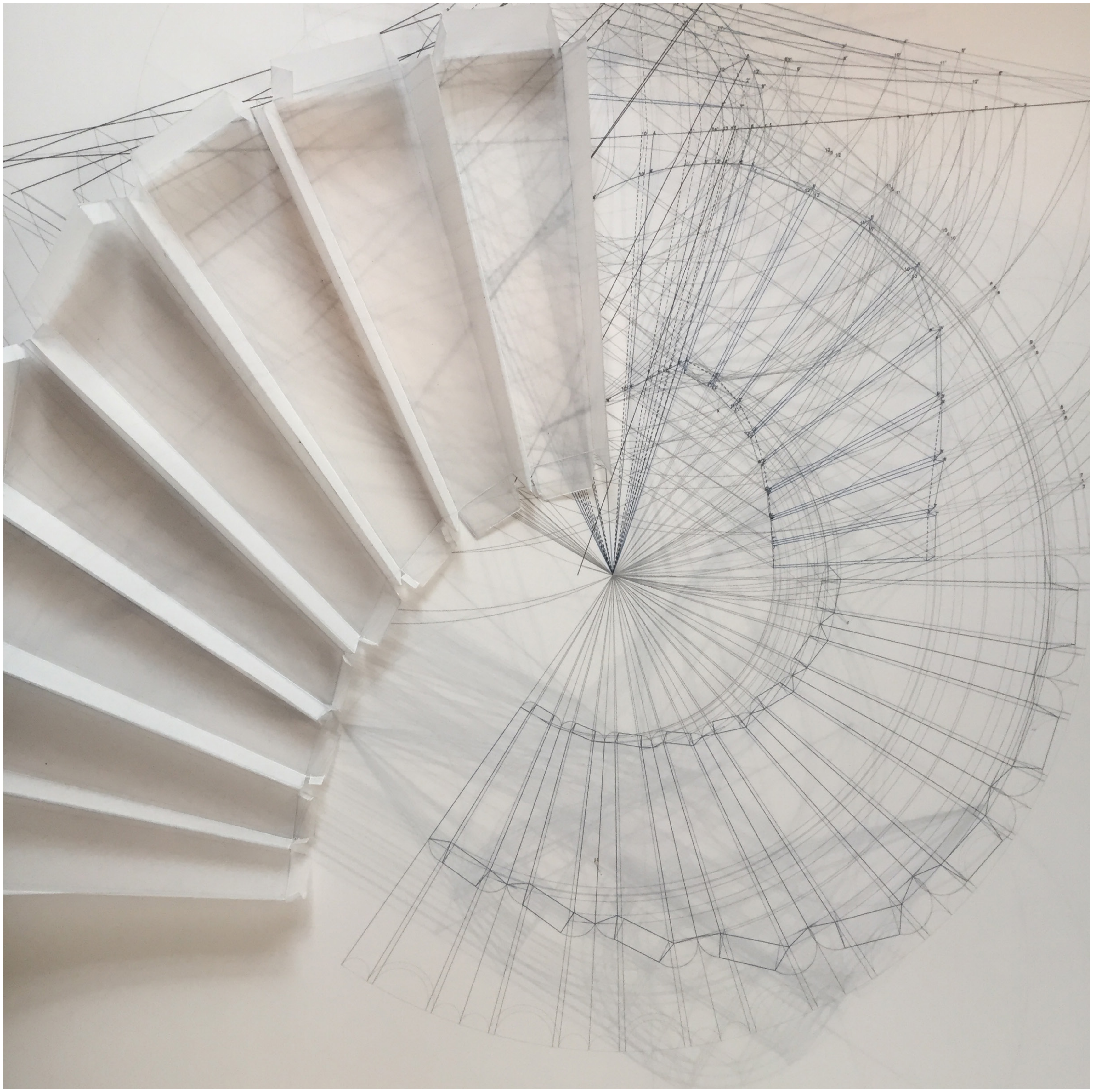
Guarino Guarini, Architettura civile, 1735
ERICSON



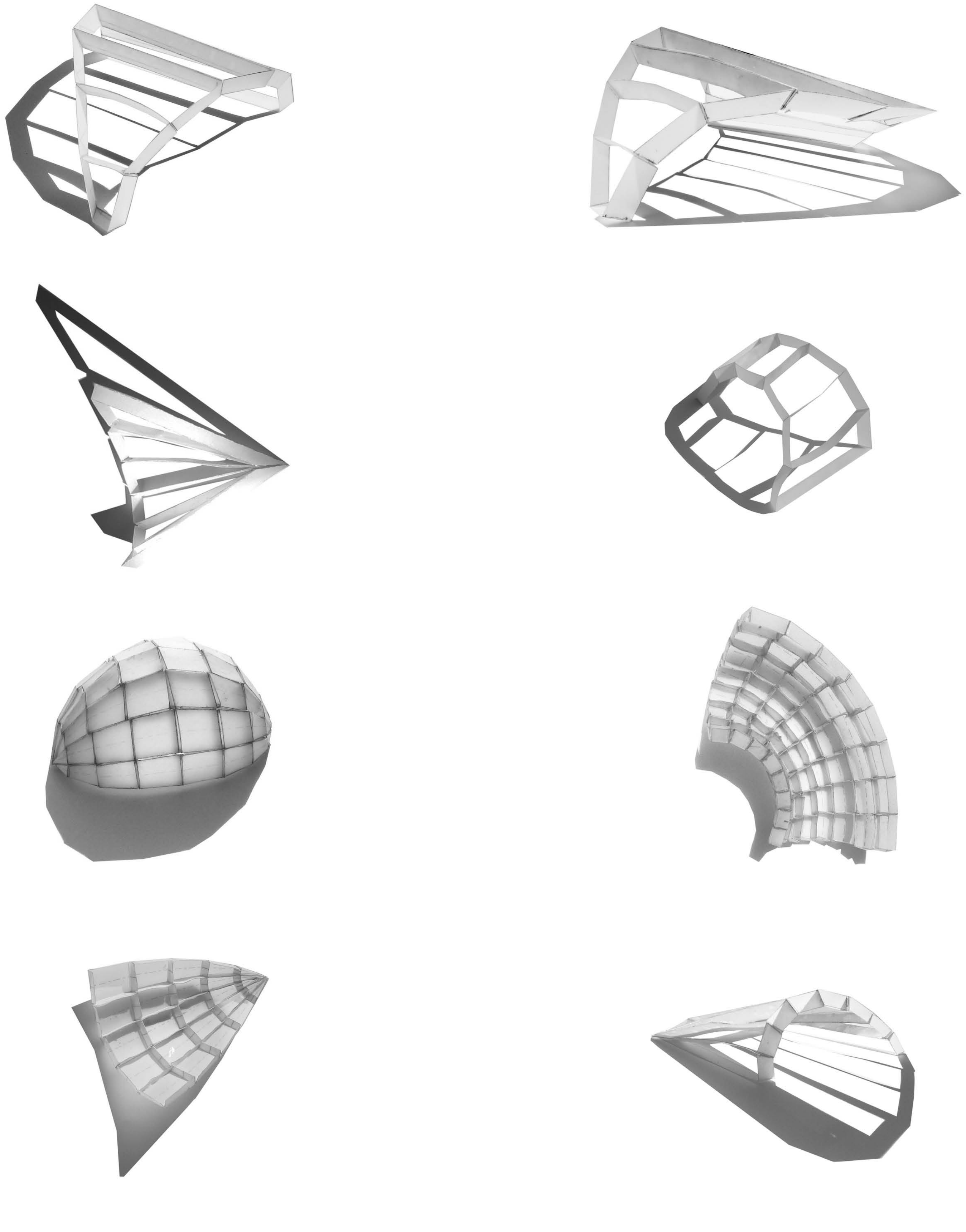
Manufacturing Method, 2013-2014
ERICSON



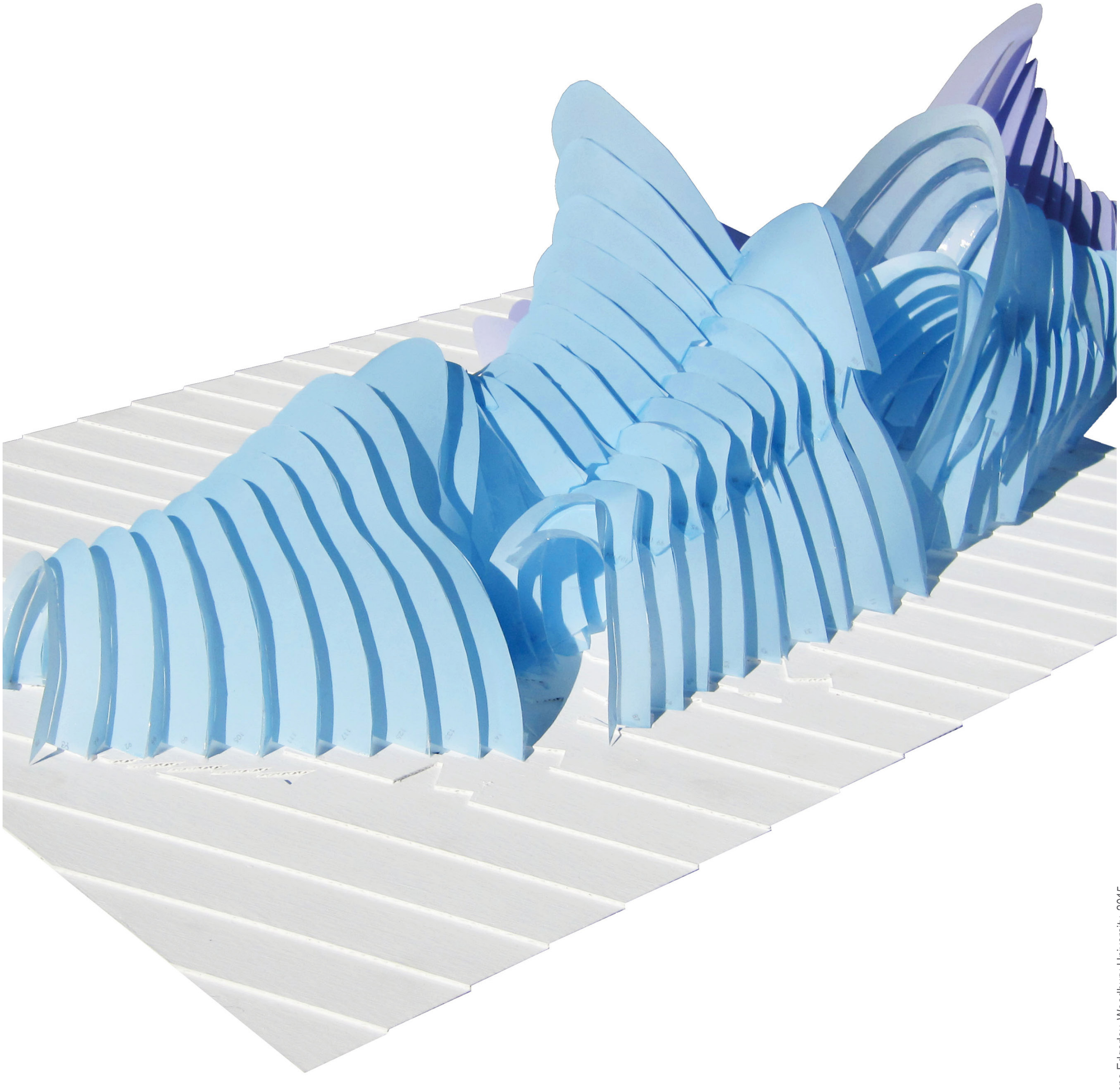
Manufacturing Method, 2013-2014
ERICSON



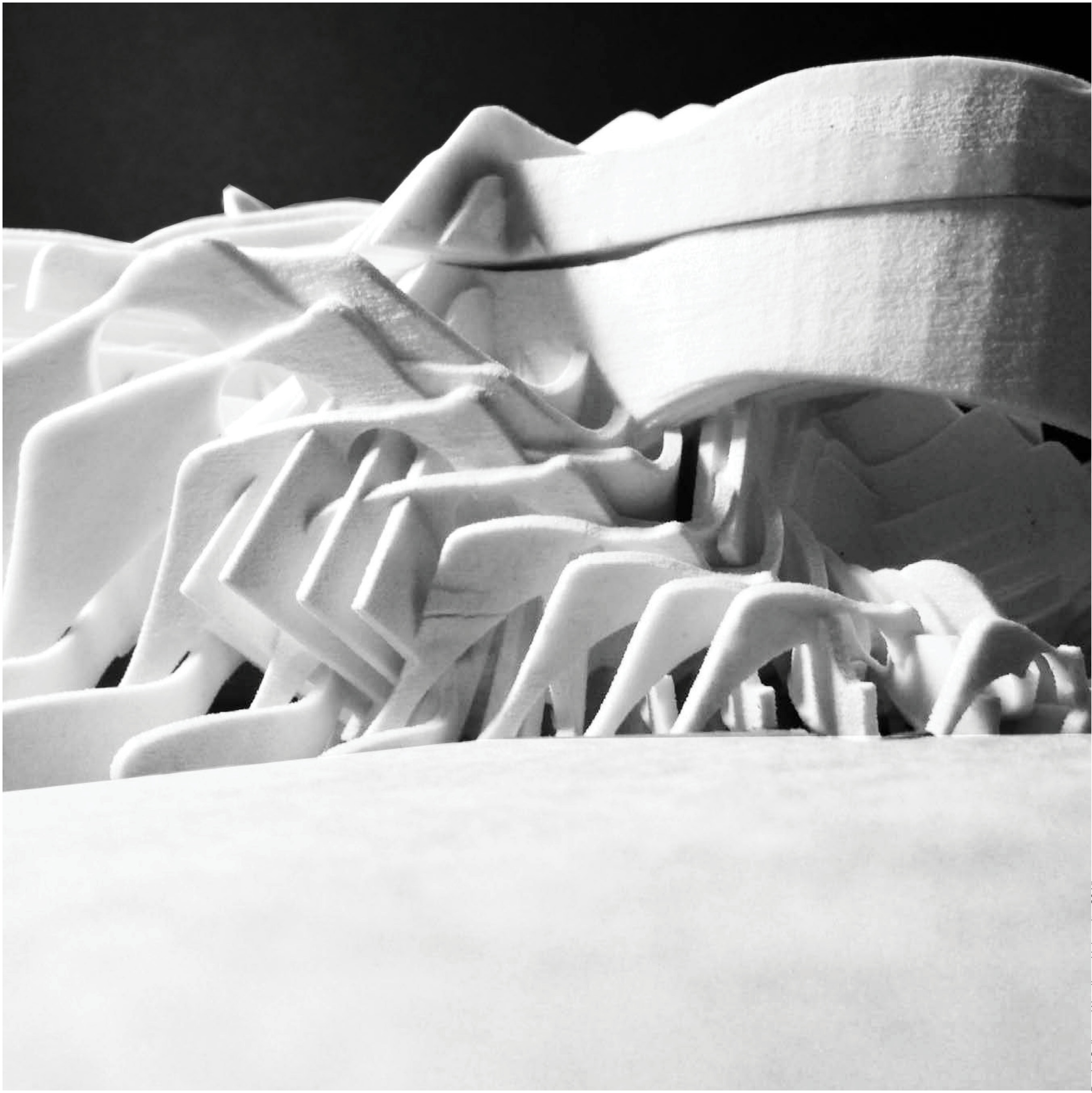
Manufacturing Method, 2013-2014
ERICSON



Manufacturing Method, 2013-2014
ERICSON

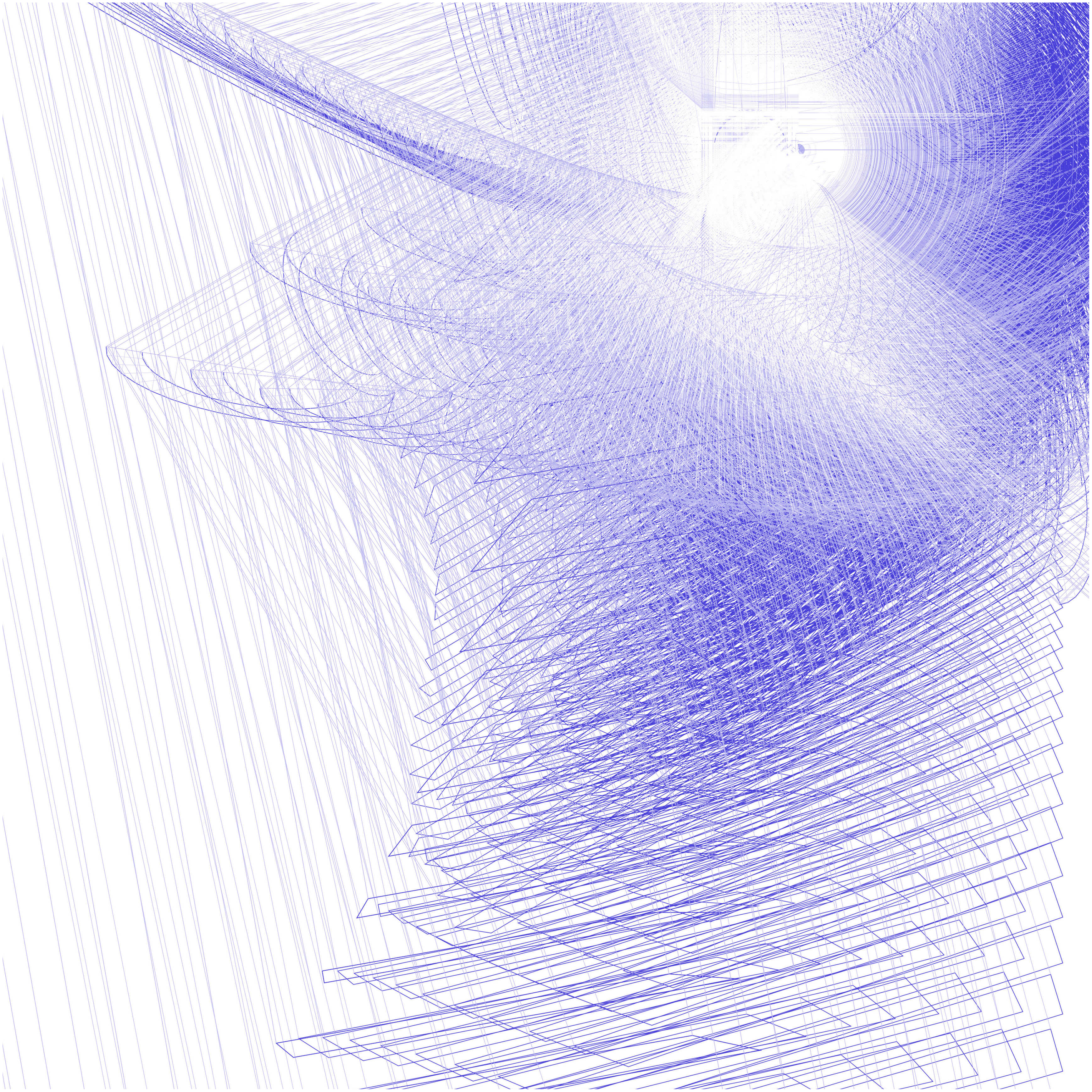


Eryanne Edgerley, Woodbury University, 2015
ERICSON

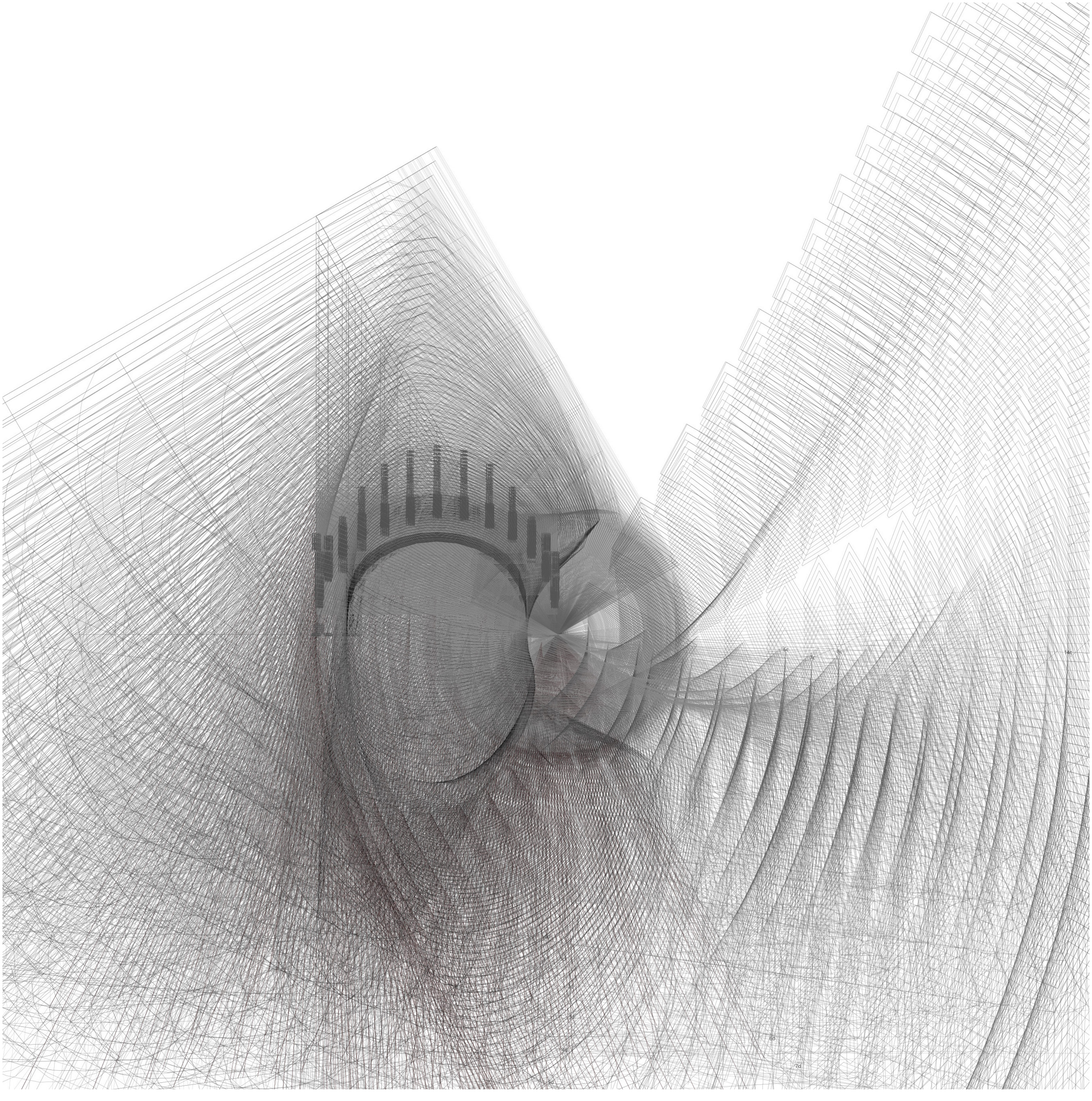


Daniel Spilman, Woodbury University, 2015
ERICSON

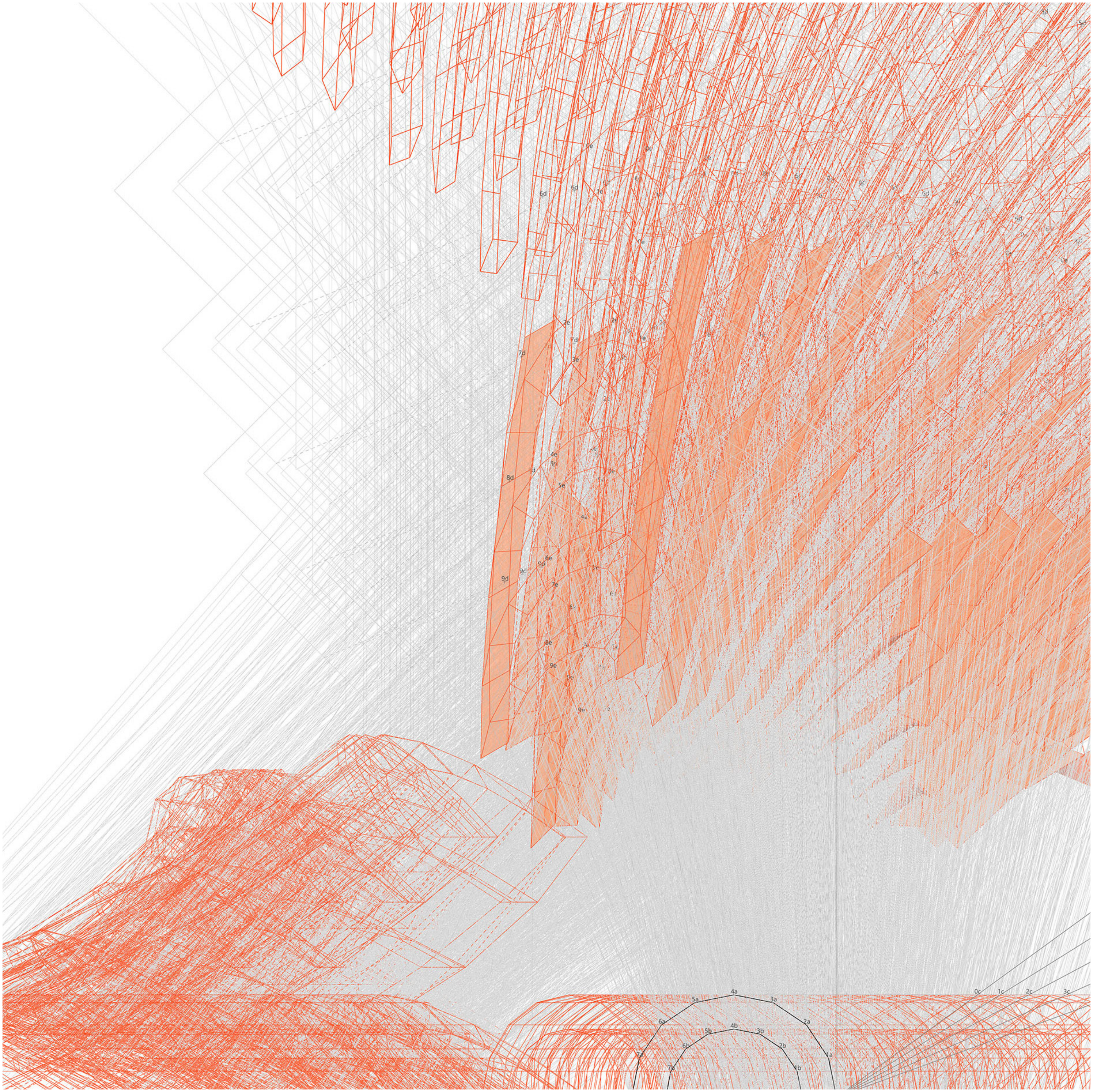
Wijdan Aldowsary, Woodbury University, 2015
ERICSON



ERICSON



ERICSON



ERICSON

Sphere_Generator | Processing 3.5.4

ile Edit Sketch Tools Help

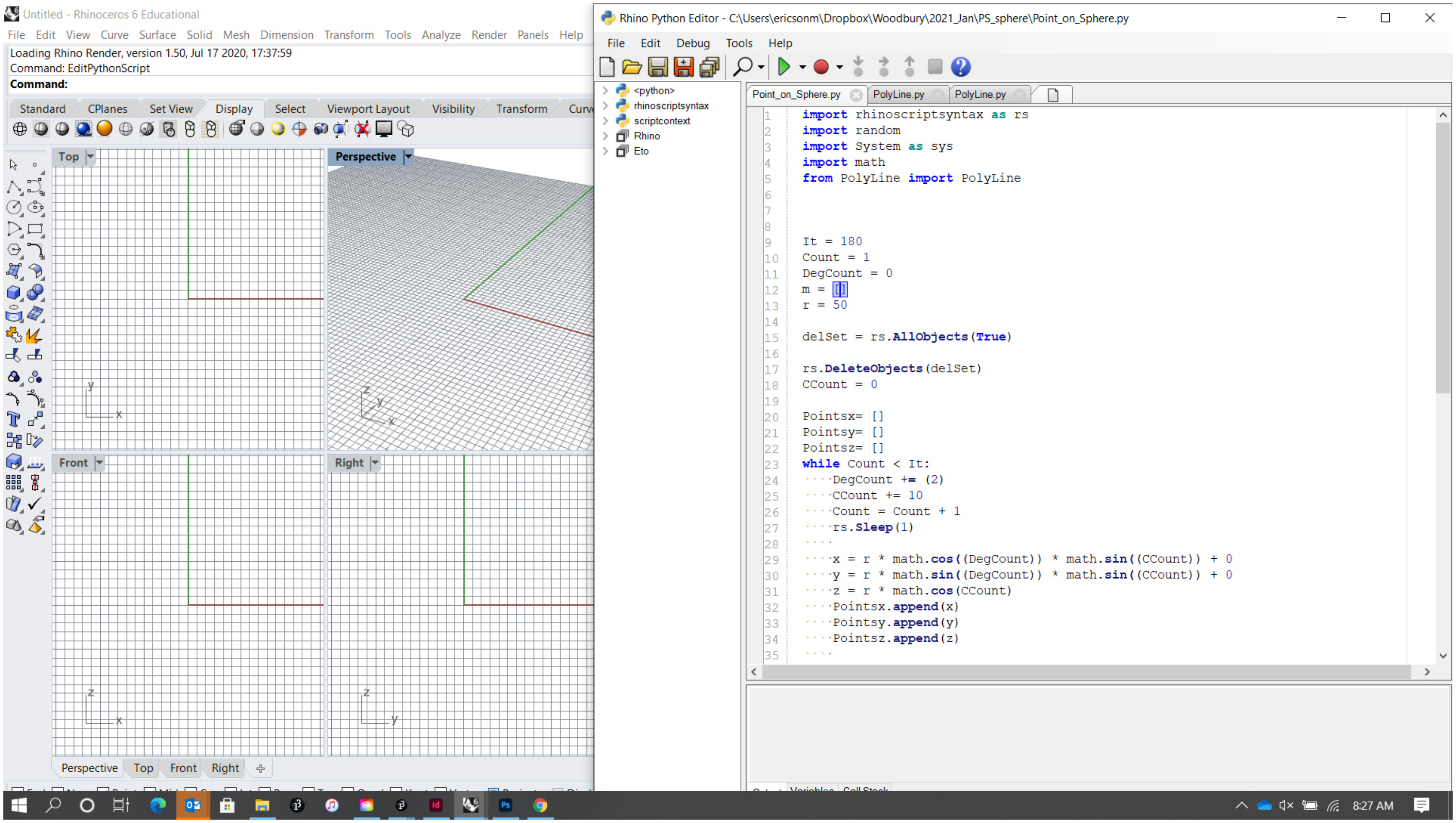
Python ▾

Sphere_Generator | PolyLine.py | SpherePoints.py

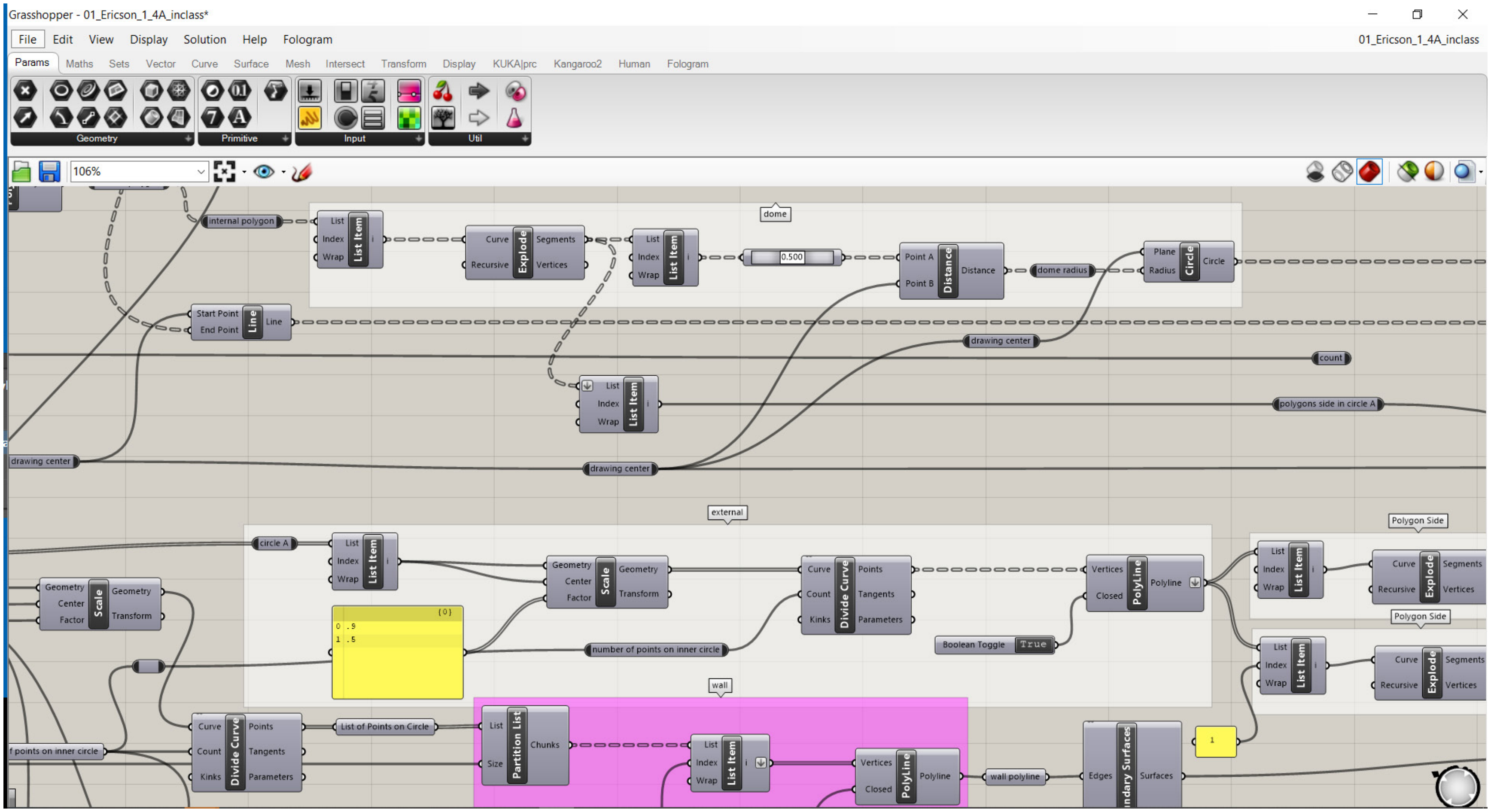
```python
from SpherePoints import SpherePoints
from PolyLine import PolyLine
tx = []
ty = []
tz = []

tx1 = []
ty1 = []
tz1 = []

rot = 0

def setup():
    background(255)
    size(1000,1000,P3D)
    frameRate(60)
    smooth(100)



def draw():
    global rot, b,c,a, tx, ty, tz
    C = color(12,59,101)
    Point1 = (0,0,0)
    a = SpherePoints()
    c = SpherePoints()
    b = SpherePoints()





    background(255)

    translate(500,500,0)
    rot += .1
```

```
145.15
145.2
145.25
145.3
```

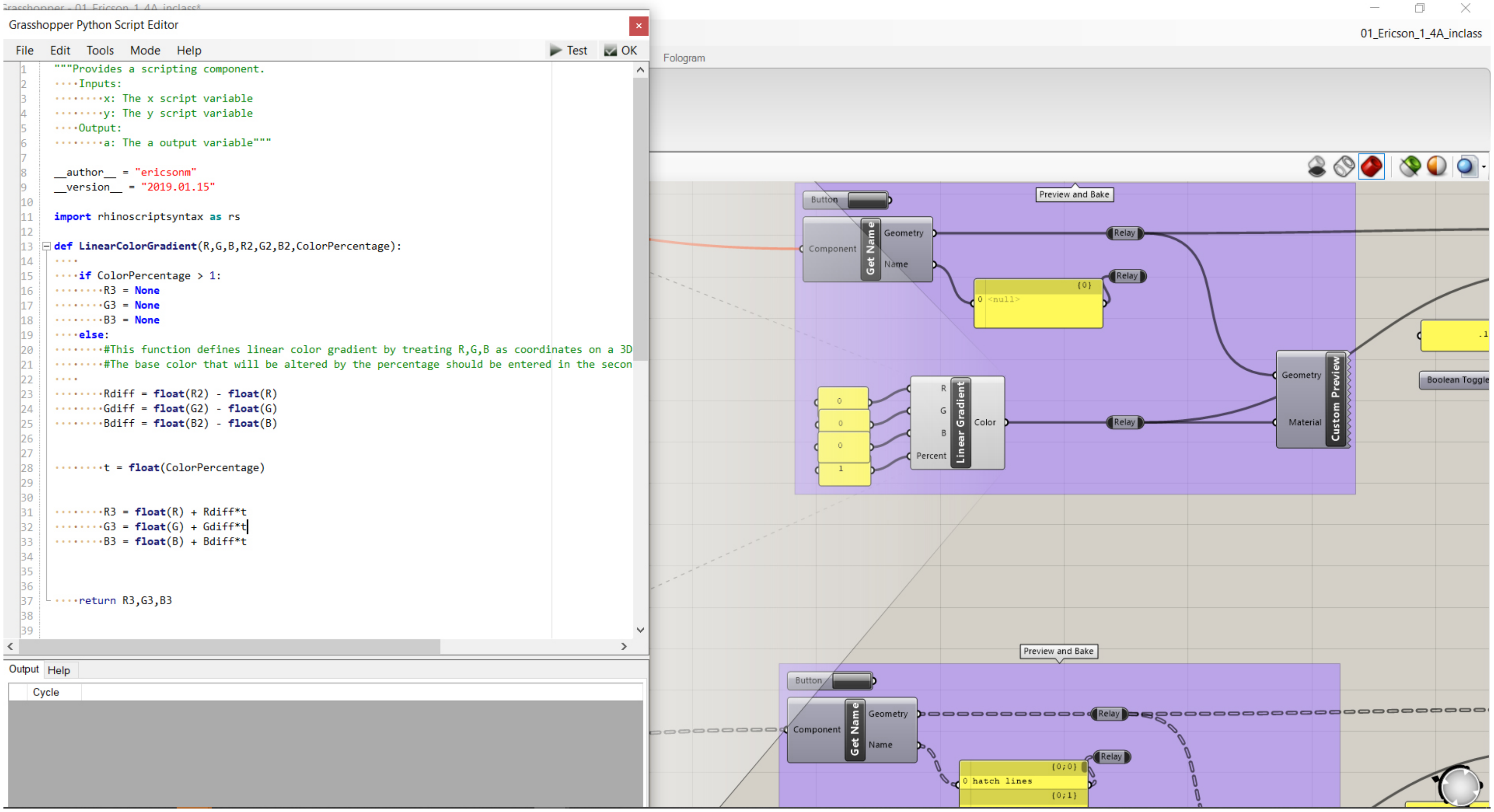
ERICSON


Untitled - Rhinoceros 6 Educational
File Edit View Curve Surface Solid Mesh Dimension Transform Tools Analyze Render Panels Help
Loading Rhino Render, version 1.50, Jul 17 2020, 17:37:59
Command: EditPythonScript
Command:
Standard CPlanes Set View Display Select Viewport Layout Visibility Transform
Top
Perspective
Front
Right
Perspective Top Front Right
Rhino Python Editor - C:\Users\ericsonm\Dropbox\Woodbury\2021_Jan\PS_sphere\Point_on_Sphere.py
File Edit Debug Tools Help
<python>
rhinoscriptsyntax
scriptcontext
Rhino
Eto
Point_on_Sphere.py
PolyLine.py
PolyLine.py
import rhinoscriptsyntax as rs
import random
import System as sys
import math
from PolyLine import PolyLine
It = 180
Count = 1
DegCount = 0
m = []
r = 50
delSet = rs.AllObjects(True)
rs.DeleteObjects(delSet)
CCount = 0
Pointsx= []
Pointsy= []
Pointsz= []
while Count < It:
DegCount += (2)
CCount += 10
Count = Count + 1
rs.Sleep(1)
x = r * math.cos((DegCount)) * math.sin((CCount)) + 0
y = r * math.sin((DegCount)) * math.sin((CCount)) + 0
z = r * math.cos(CCount)
Pointsx.append(x)
Pointsy.append(y)
Pointsz.append(z)
8:27 AM

ERICSON


Grasshopper - 01_Ericson_1_4A_inclass*
File
Edit
View
Display
Solution
Help
Fologram
01_Ericson_1_4A_inclass
Params
Maths
Sets
Vector
Curve
Surface
Mesh
Intersect
Transform
Display
KUKA|prc
Kangaroo2
Human
Fologram
Geometry
Primitive
Input
Util
106%
dome
internal polygon
dome radius
drawing center
count
polygons side in circle A
external
circle A
number of points on inner circle
Boolean Toggle
True
Polygon Side
wall
List of Points on Circle
wall polyline
f points on inner circle

ERICSON


Grasshopper Python Script Editor
File
Edit
Tools
Mode
Help
Test
OK
"""Provides a scripting component.
    Inputs:
        x: The x script variable
        y: The y script variable
    Output:
        a: The a output variable"""
__author__ = "ericsonm"
__version__ = "2019.01.15"
import rhinoscriptsyntax as rs
def LinearColorGradient(R,G,B,R2,G2,B2,ColorPercentage):
    if ColorPercentage > 1:
        R3 = None
        G3 = None
        B3 = None
    else:
        #This function defines linear color gradient by treating R,G,B as coordinates on a 3D
        #The base color that will be altered by the percentage should be entered in the secon
        Rdiff = float(R2) - float(R)
        Gdiff = float(G2) - float(G)
        Bdiff = float(B2) - float(B)
        t = float(ColorPercentage)
        R3 = float(R) + Rdiff*t
        G3 = float(G) + Gdiff*t
        B3 = float(B) + Bdiff*t
    return R3,G3,B3
Output
Help
Cycle
01_Ericson_1_4A_inclass
Fologram
Button
Preview and Bake
Component
Get Name
Geometry
Name
Relay
{0}
0 <null>
0
0
0
1
R
G
B
Percent
Linear Gradient
Color
Geometry
Material
Custom Preview
Boolean Toggle
Button
Preview and Bake
Component
Get Name
Geometry
Name
Relay
{0;0}
0 hatch lines
{0;1}

ERICSON

ERICSON