

Gewölbt und Gestrickt

Die Eleganz formoptimierter
Tragstrukturen

Dr. Matthias Rippmann – ETH Zürich

Matthias Rippmann



Dr. Matthias Rippmann ist Senior Researcher in der Block Research Group an der ETH Zürich.

Er leitet und koordiniert interdisziplinäre Forschungsprojekte und Industriekooperationen im Bereich *ressourceneffizienter Tragstrukturen* und *innovativer Konstruktionstechnik*. Nach seiner Promotion auf dem Gebiet der digitalen Werkzeuge für den Tragwerksentwurf war er Postdoc und Mitglied des Lenkungsausschusses im Nationalen Forschungsschwerpunkt (NSF) Digitale Fabrikation an der ETH.

Vor seinem Doktoratstudium arbeitete Matthias Rippmann in Stuttgart bei Behnisch Architekten, LAVA, dem Institut für Leichtbau Entwerfen und Konstruieren (ILEK) und im Ingenieurbüro Werner Sobek.

Er war 2010 Mitgründer des Architektur- und Planungsbüro ROK (Rippmann Oesterle Knauss GmbH), das sich auf computerbasierte Planung und automatisierte Fertigung spezialisiert hat.

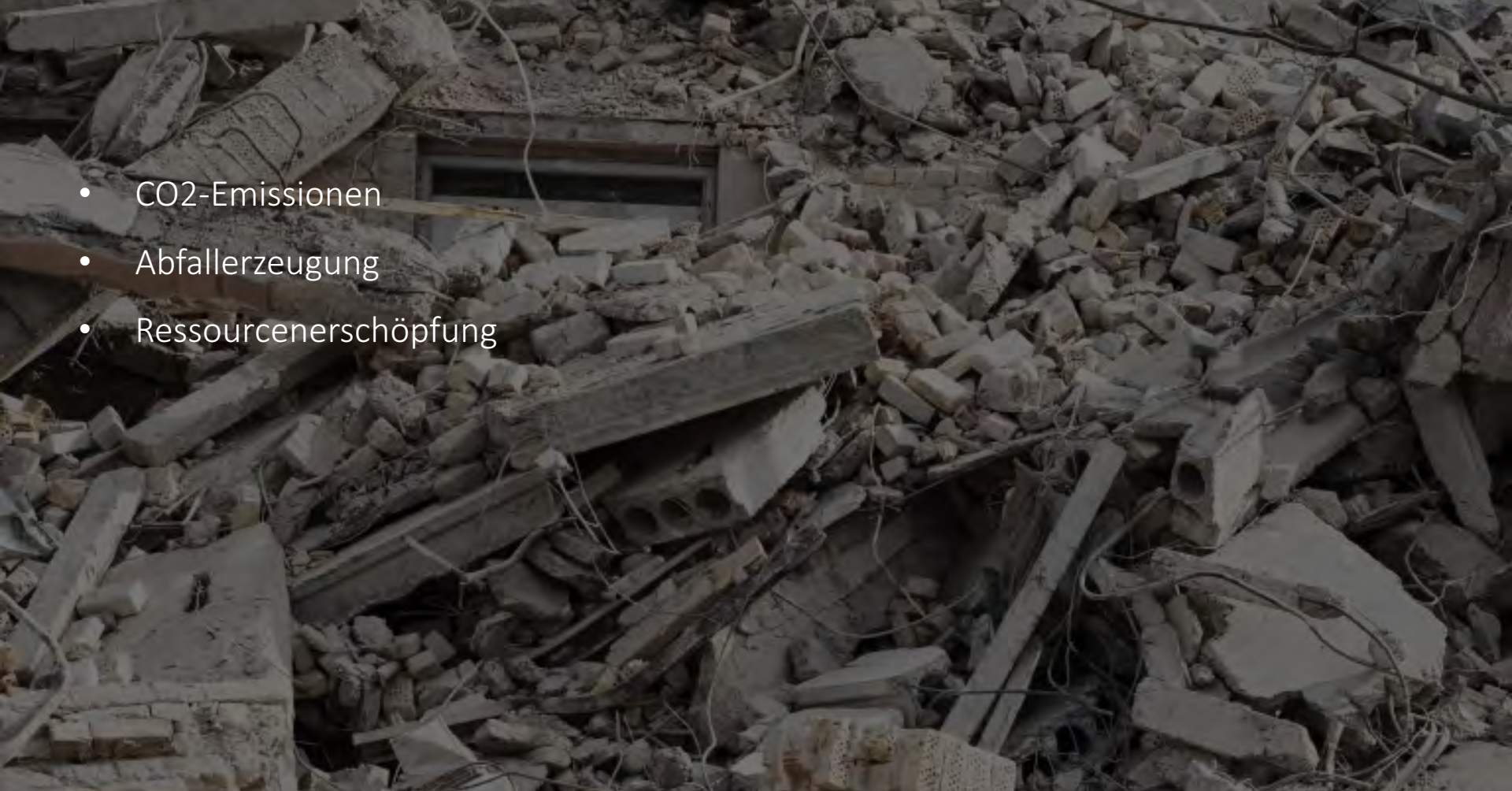



- CO2-Emissionen




- CO2-Emissionen
- Abfallerzeugung



- 
- CO2-Emissionen
 - Abfallerzeugung
 - Ressourcenerschöpfung

- 
- CO2-Emissionen ← Weniger Verbrauch
 - Abfallerzeugung ← Innovative Konstruktion
 - Ressourcenerschöpfung ← Alternative Materialien

- 
- CO2-Emissionen ← Weniger Verbrauch
 - Abfallerzeugung ← Innovative Konstruktion
 - Ressourcenerschöpfung ← Alternative Materialien
- } Formoptimierte Tragstrukturen



- CO2-Emissionen

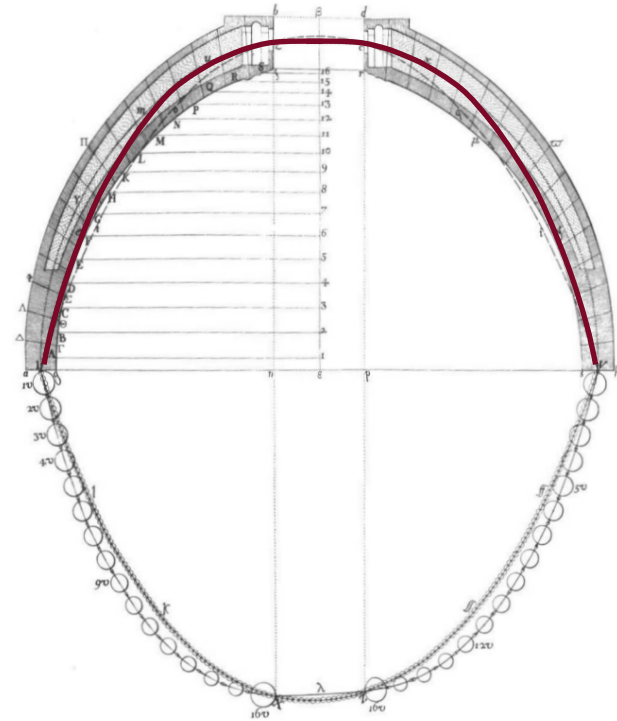
← Weniger Verbrauch





« As hangs the flexible line, so but inverted will stand the rigid arch. »

Hooke (1676)



The monitor displays a 3D visualization of a network structure, likely a mesh or a complex geometric form, rendered in blue and green. The structure is composed of interconnected vertices and edges, forming a complex, organic shape. The code editor on the right side of the screen shows Python code for network analysis and mesh generation.

```

210 if not breakpoints:
211     network.clear()
212     network.clear_halfedges()
213     network_halfedge = dict((key, []) for key in network.vertices)
214     for u, v in network.edges_iter():
215         network_halfedge[u][v] = None
216         network_halfedge[v][u] = None
217     sort_neighbours(network)
218     leaves = network.leaves()
219     if leaves:
220         u = sorted([(key, network.vertices[key]) for key in leaves], key=lambda x: (x[1][y], x[1][x]))
221         v = sorted(network.vertices_iter(True), key=lambda x: (x[1][y], x[1][x]))
222         u = find_first_neighbours(network)
223         find_edge_face(u, v, network)
224         for u, v in network.edges_iter():
225             if network_halfedge[u][v] is None:
226                 find_edge_face(u, v, network)
227             if network_halfedge[v][u] is None:
228                 find_edge_face(v, u, network)
229         break_faces(network, breakpoints)
230     return network.face

def find_first_neighbours(key, network):
231     origin = []
232     nbrs = network_halfedge[key], keys()
233     if len(nbrs) == 1:
234         return nbrs[0]
235     u = [u[1][0], u[1][1]]
236     for nbr in nbrs:
237         u = [network.vertices[nbr[1]] for nbr in '012']
238         v = [network.vertices[key][1] for nbr in '012']
239         w = [u[1][0], u[1][1], v[1][0], v[1][1]]
240         angle = math.atan2(math.sin(math.pi/2), math.cos(math.pi/2))
241     return nbr[0][1].index(min(angle))

def sort_neighbours(network, count):
242     sorted_neighbours = []
243     break
244     if not dup:
245         pts.append(s)
246     if faces:
247         faces.append(face)
248     return pts, faces

if __name__ == '__main__':
249     import math
250     import random
251     from compass.geometry import distance_point_point
252     from compass.helpers.mesh import draw_mesh
253     from compass.datastructures.mesh import Mesh

    radius = 5
    origin = (0, 0, 0)
    count = 1
    points = []
    while count < 2000:
254         x = (random.random() - 0.5) * radius * 2
255         y = (random.random() - 0.5) * radius * 2
256         z = (random.random() - 0.5) * radius * 2
257         pt = x, y, z
258         if distance_point_point(origin, pt) <= radius:
259             points.append(pt)
260             count += 1
261     faces = convex_hull(points)
262     mesh = Mesh.from_vertices_and_faces(points, faces)
263     draw_mesh(mesh,
264               show_faces = True,
265               show_vertices = False,
  
```

Armadillo Vault

Venice Architecture Biennale 2016

Block Research Group, ETH Zurich

Prof. Dr. Philippe Block
Dr. Tom Van Mele
Dr. Matthias Rippmann
Edyta Augustynowicz
Cristián Calvo Barentin
Dr. Tomás Méndez Echenagucia
Mariana Popescu
Dr. Andrew Liew
Anna Maragkoudaki
Ursula Frick
Robin Oval
Nick Krouwel
Dr. Noelle Paulson

Ochsendorf DeJong & Block

Prof. Dr. John Ochsendorf
Prof. Dr. Matthew DeJong
Prof. Dr. Philippe Block
Anjali Mehrotra

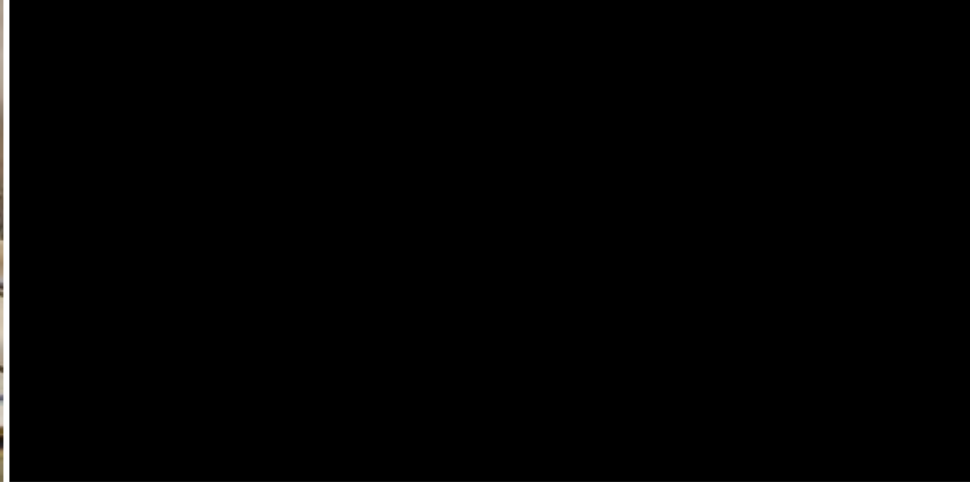
The Escobedo Group

David Escobedo
Matthew Escobedo
Salvador Crisanto
John Curry
Francisco Tovar Yebra
Joyce I-Chin Chen
Adam Bath
Hector Betancourt
Luis Rivera
Antonio Rivera
Carlos Rivera
Carlos Zuniga Rivera
Samuel Rivera
Jairo Rivera
Humberto Rivera
Jesus Rosales
Dario Rivera



A photograph of a white eggshell that has been cracked open. The top portion of the shell is missing, revealing a hollow interior. The number '1/3' is printed in large, white, sans-serif font on the front of the eggshell. Several pieces of broken eggshell are scattered on the light-colored surface in front of the main eggshell. The background is a plain, light-colored surface.

1/3


















Abfallerzeugung ← Innovative Konstruktion

NEST HiLo – Roof construction prototype

ETH Zürich, Switzerland, 2016 - 2017

Design and Engineering

Block Research Group, ETH Zürich

Supermanoeuvre

Bollinger+Grohmann

Mathematical and Physical Geodesy, ETH Zürich

Automatic Control Laboratory, ETH Zürich

Construction

Marti [general contractor]

Bürgin Creations [concrete works]

Doka [scaffolding]

Documentation

Naida Iljazovic

Mike Lyrenmann

Fabrication

Jakob [cables]

Bruno Lehmann [rods + cable net components]

Blumer Lehmann [timber]

Dafotech [steel supports + plates]

Bieri [fabric cutting + sewing]

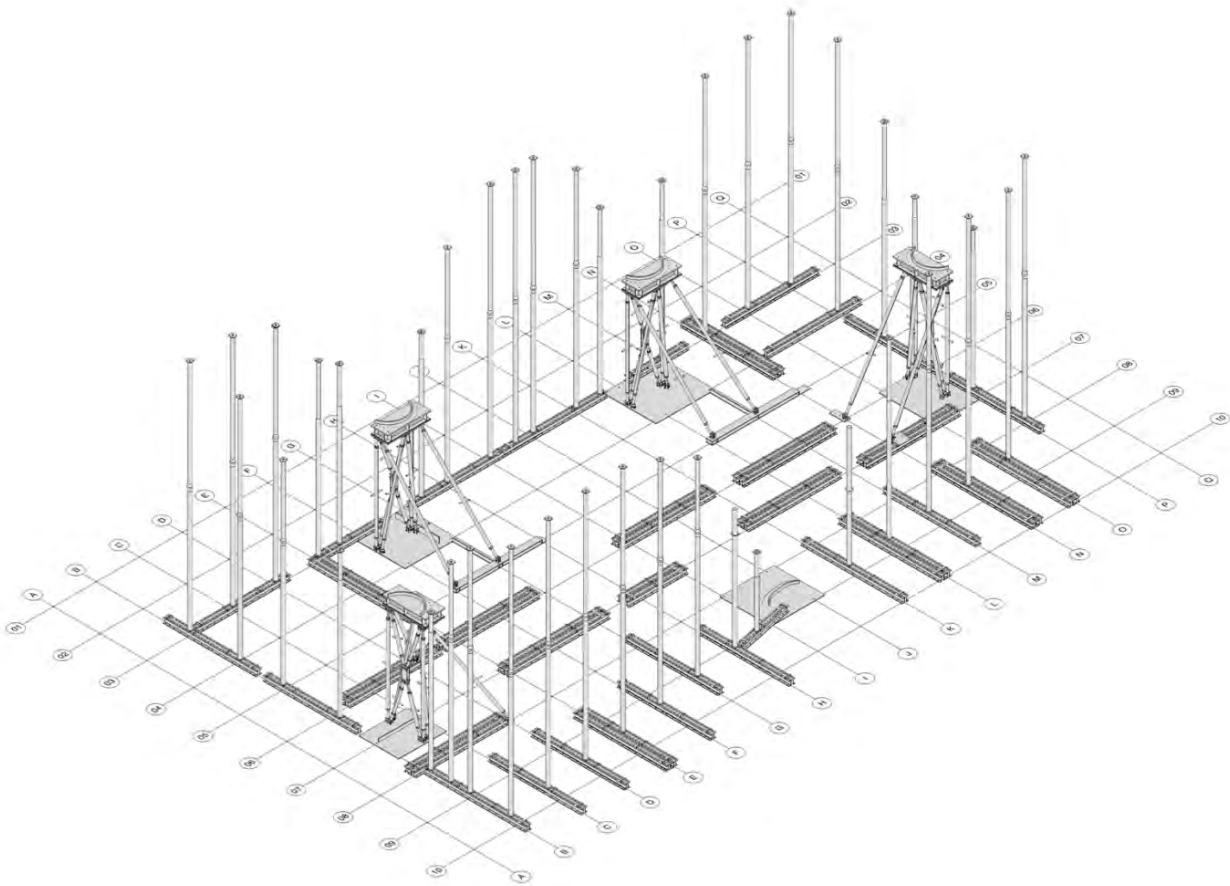
Sponsors

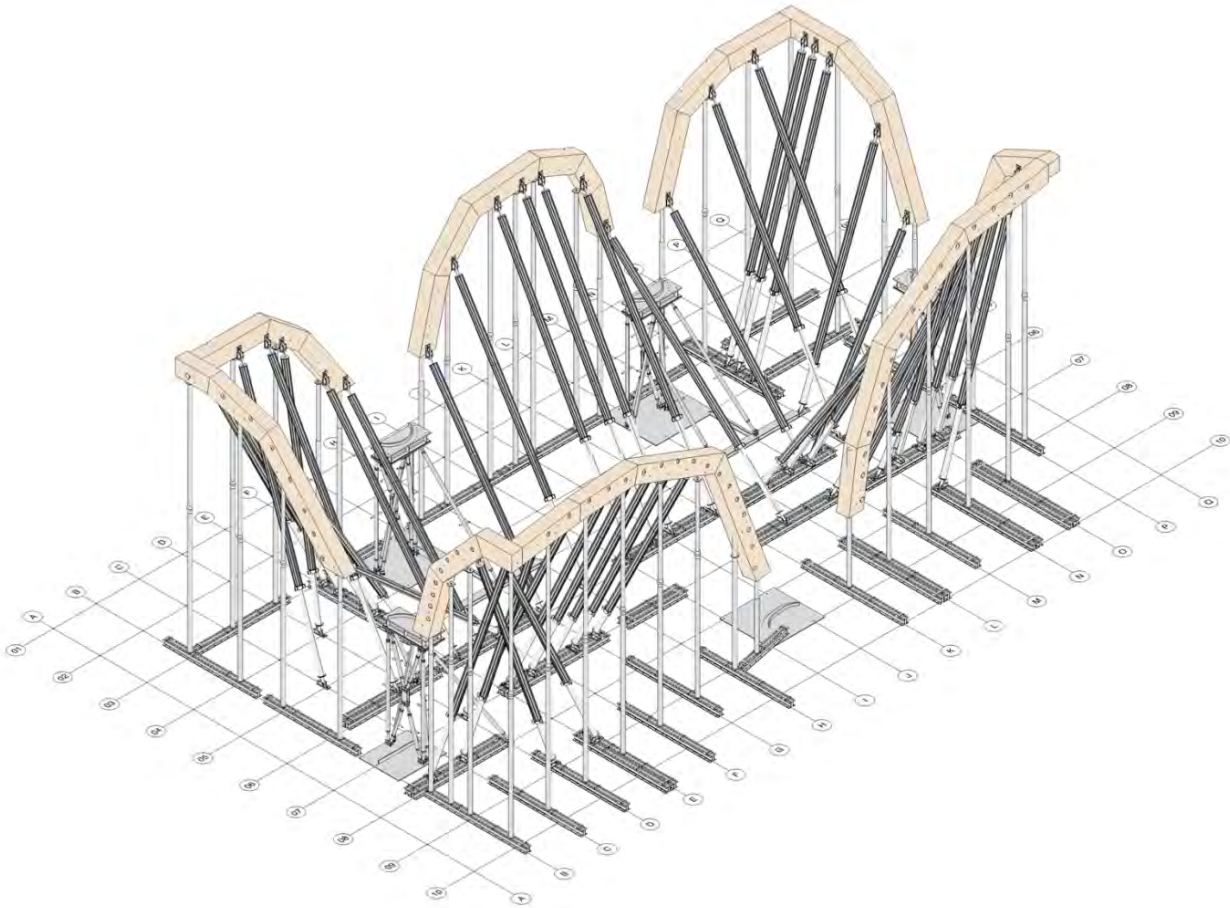
ETH Zürich

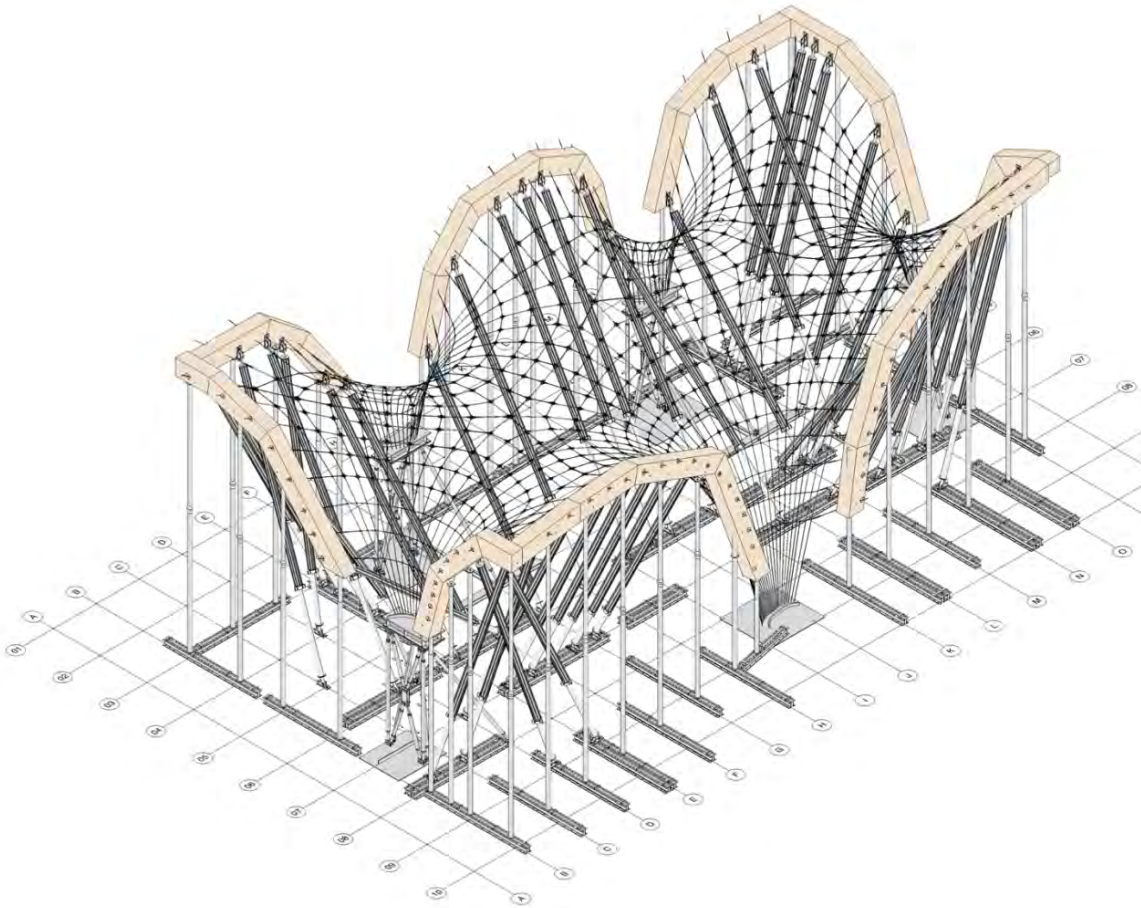
NCCR Digital Fabrication

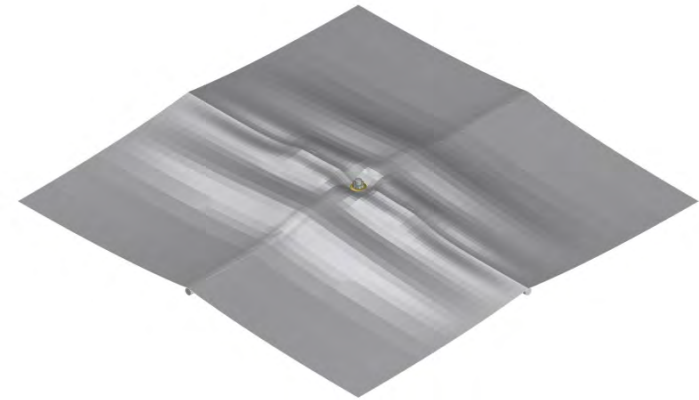
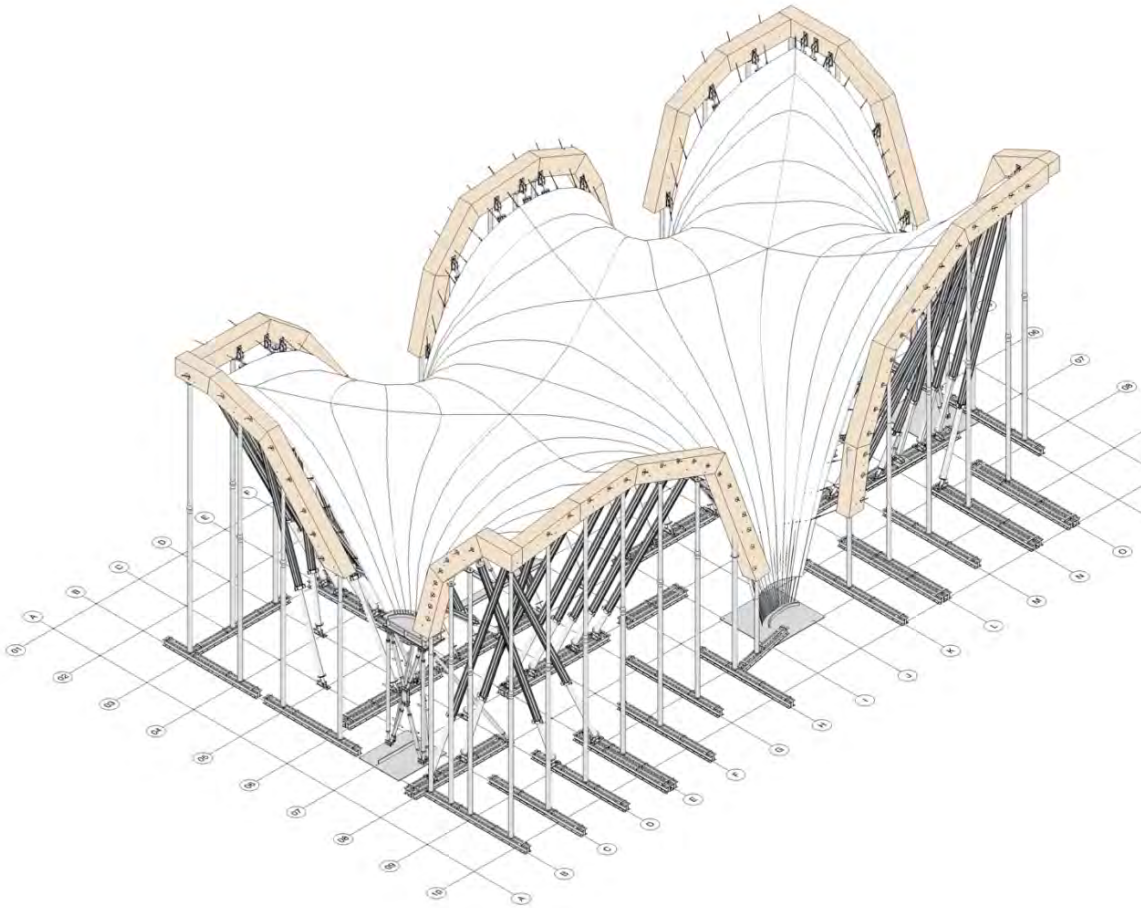
Holcim Schweiz

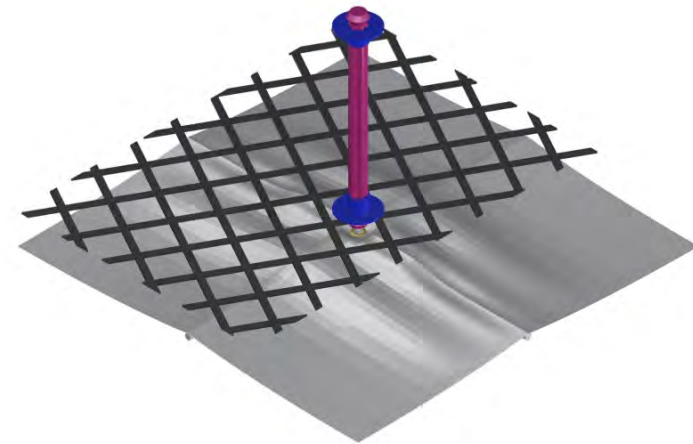
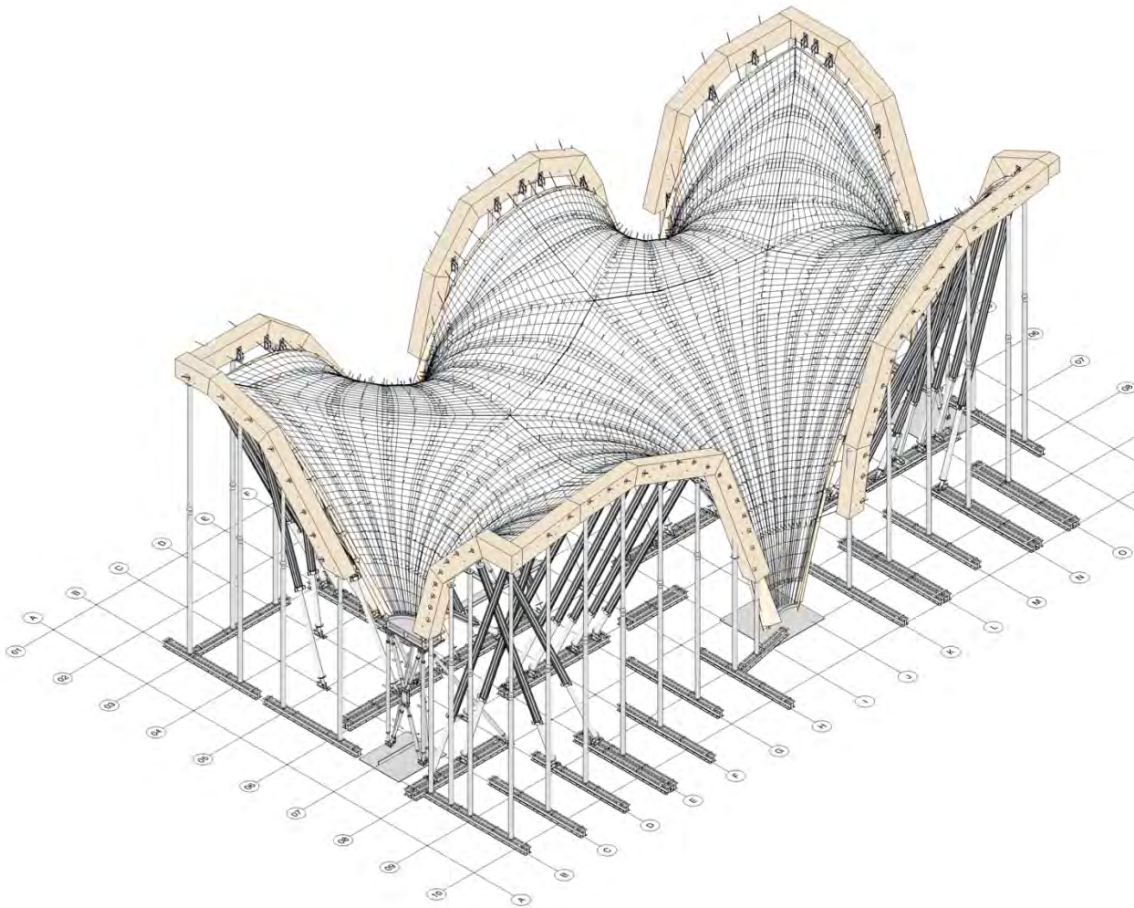


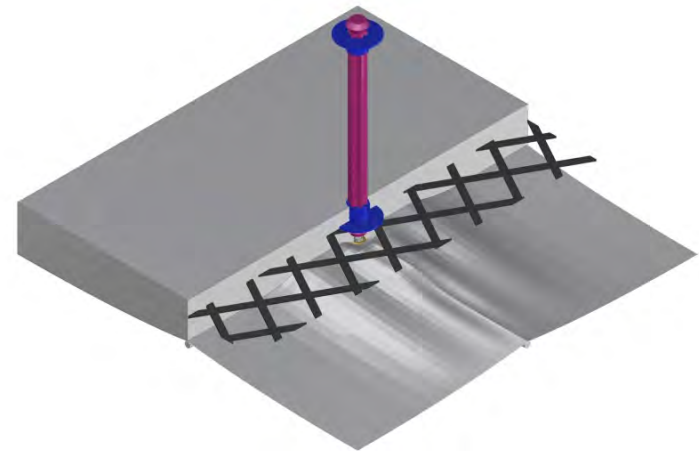
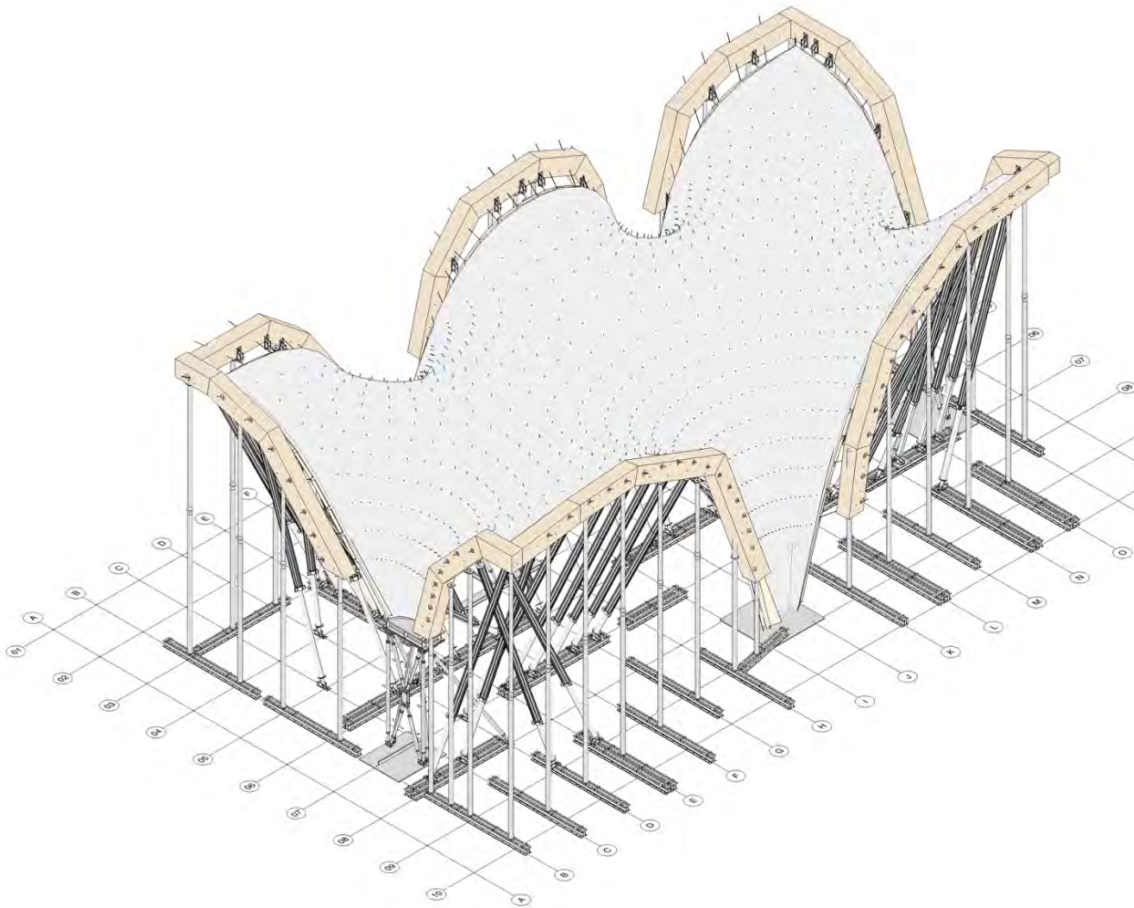


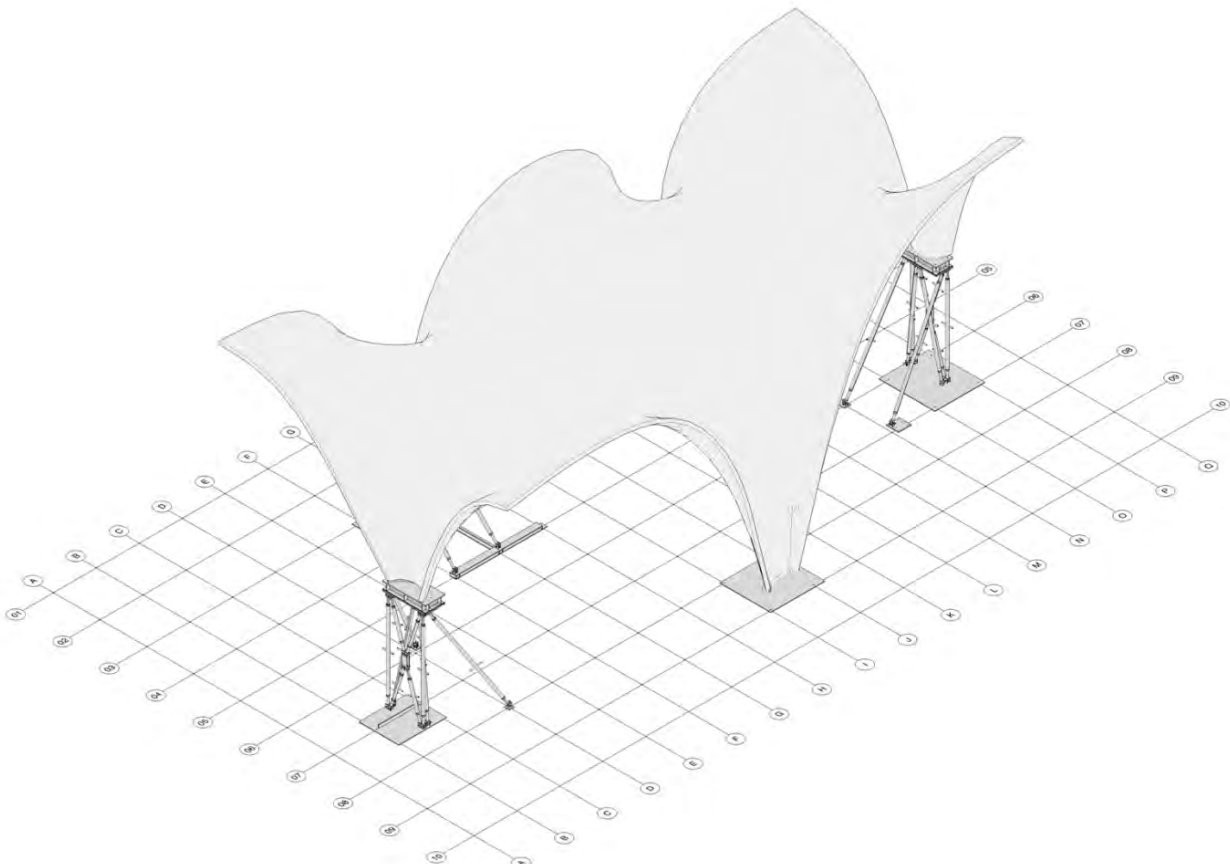










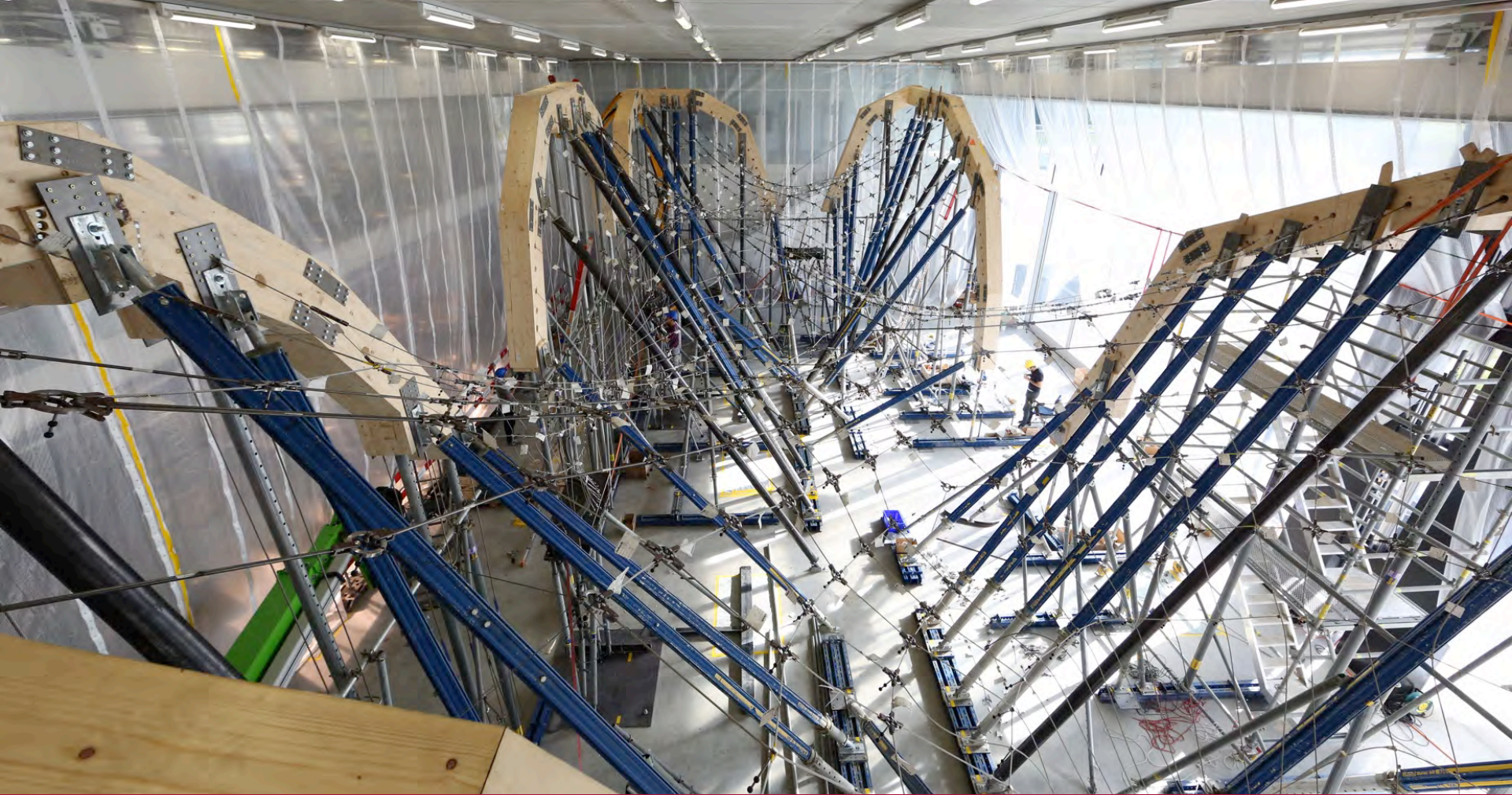


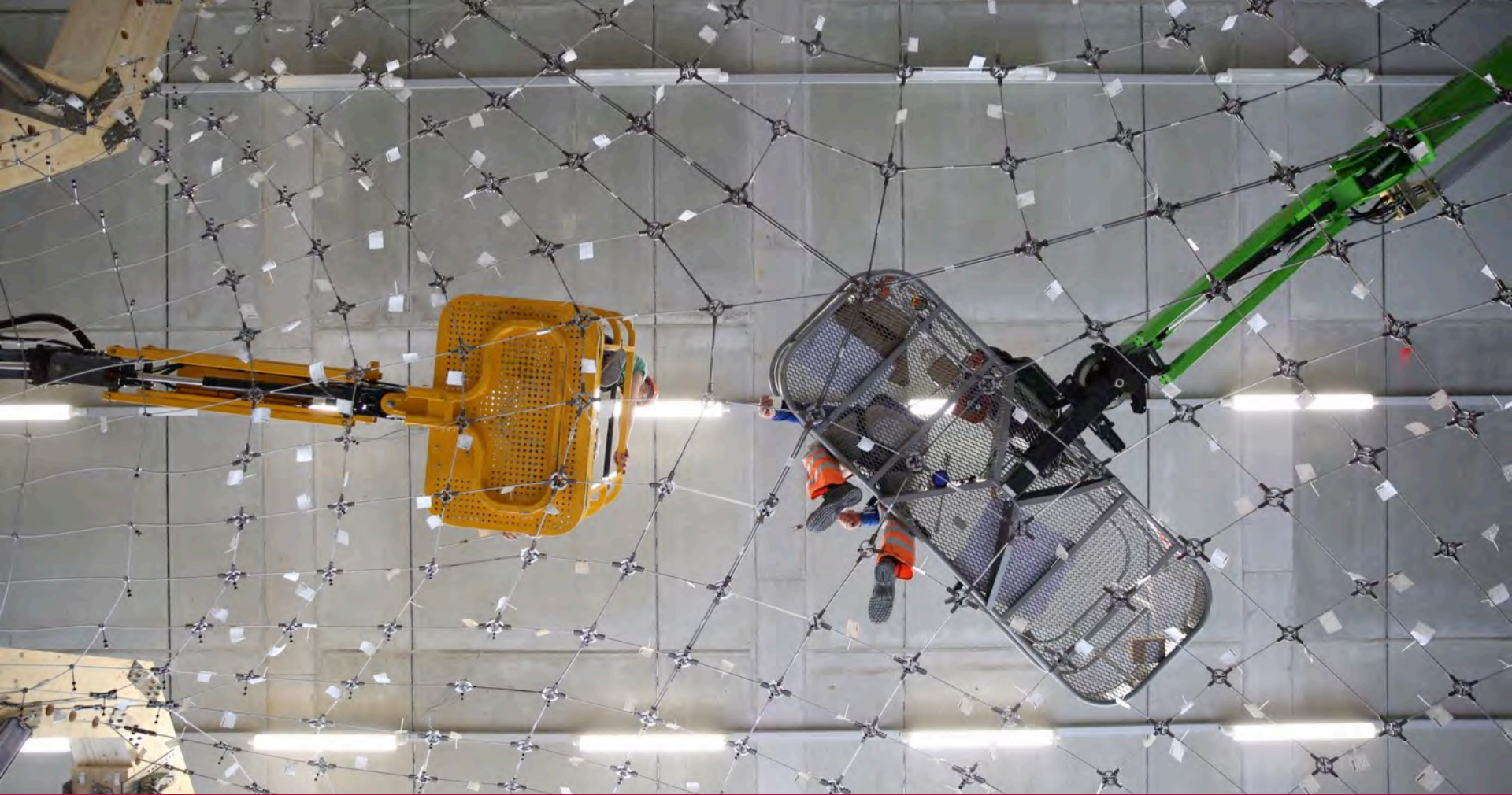


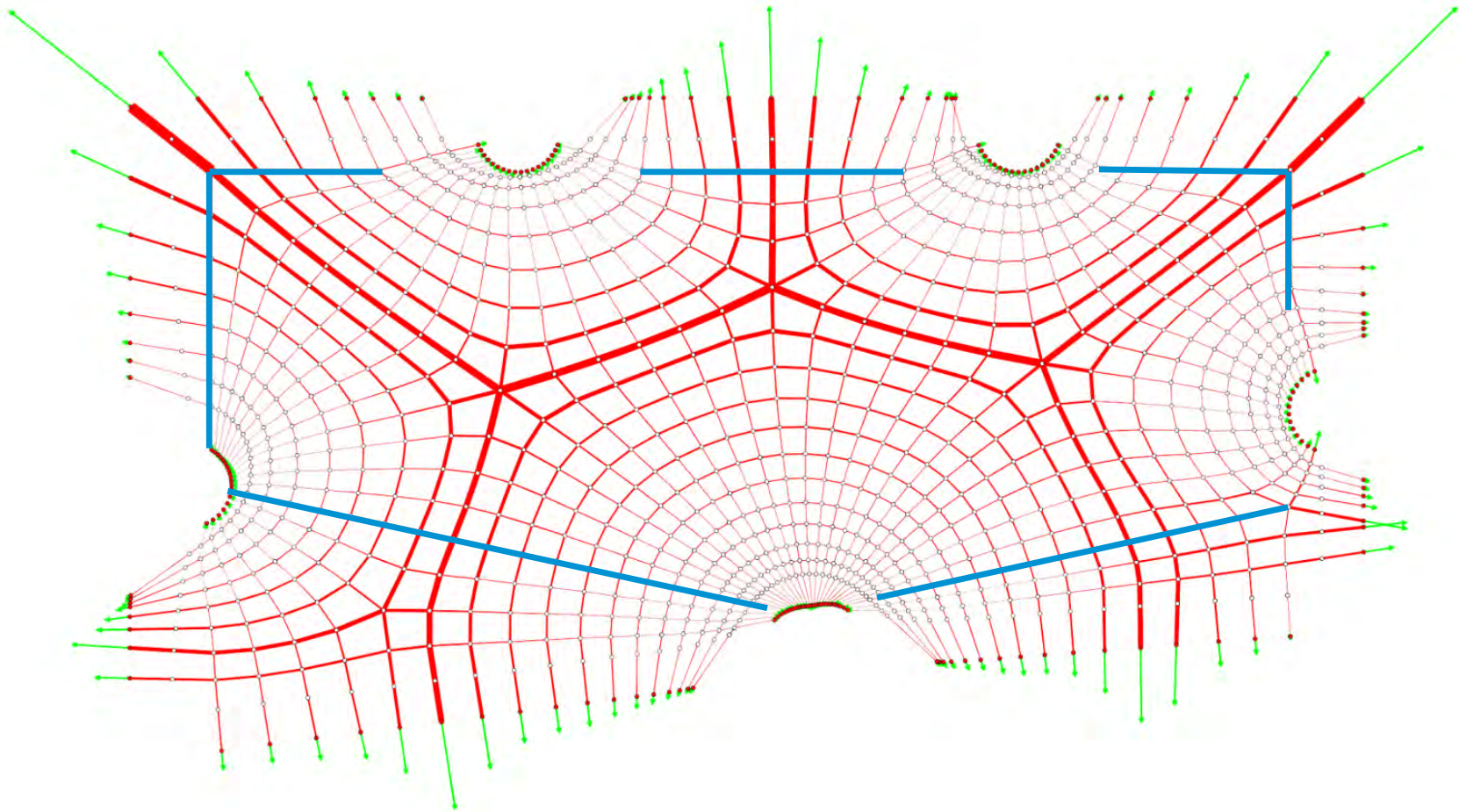
















```
loc = [0, 0]
format_time = lambda x: '%02d:%02d' % (x // 60, x % 60)

print('---')
print('Result:')
print('Time taken: (%d) s' % format_time(loc))
print('Starting node: (%d, %d) on format_time(loc))')
print('Final node: (%d, %d) on format_time(loc))')
print('---')

# plotting

# plot net w/ plot results
delete(objects_all)

# Plot Network

# plot net
network.set_edges_attributes(edges) for i in loc:
    color = 'red'
network.set_edges_attributes(edges) for i in loc:
    color = 'green'
network.set_edges_attributes(edges) for i in loc:
    color = 'blue'
network.set_edges_attributes(edges) for i in loc:
    color = 'yellow'
draw_network(network, type='line', edge_name='type')

# Plot Results

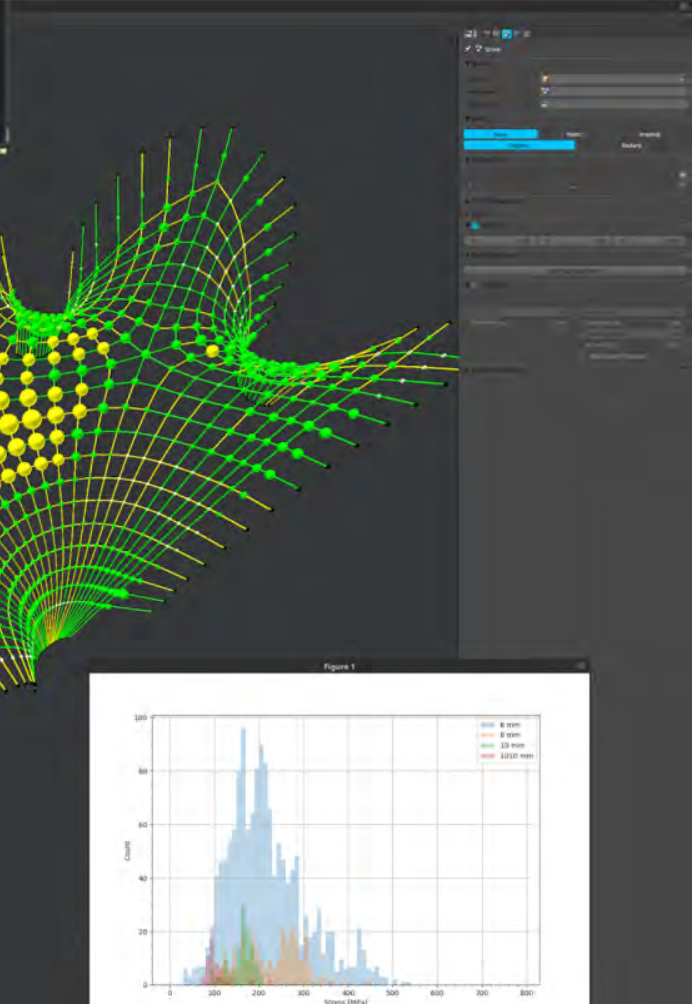
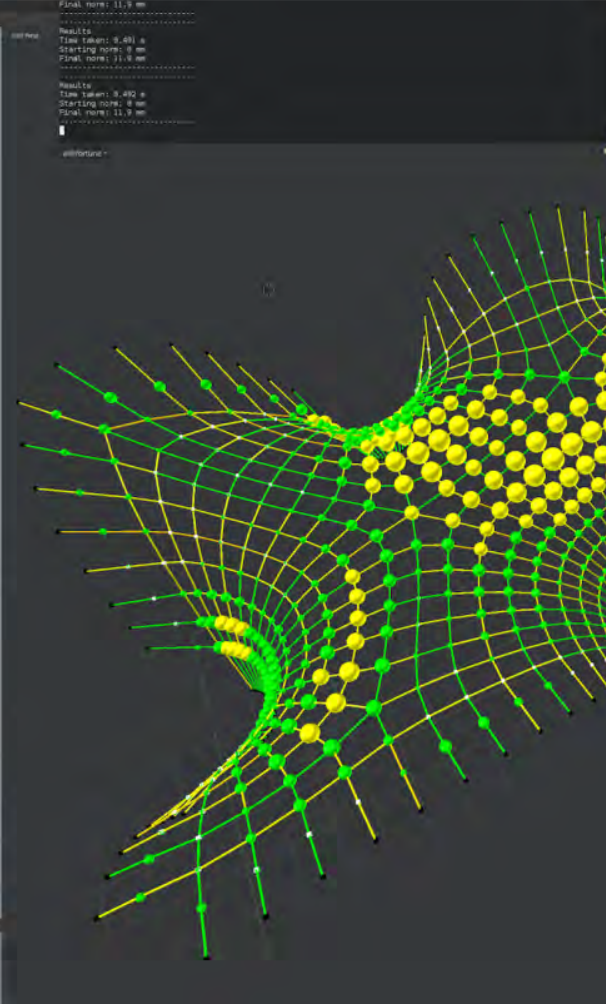
network.set_edges_attributes(edges) for i in range(0, 1000):
    color = 'red'
network.set_edges_attributes(edges) for i in range(0, 1000):
    color = 'green'
network.set_edges_attributes(edges) for i in range(0, 1000):
    color = 'blue'
network.set_edges_attributes(edges) for i in range(0, 1000):
    color = 'yellow'

for i in range(0, 1000):
    x1 = (0, 50)
    x2 = (0, 50)
    network.set_edges_attributes(edges) for i in range(0, 1000):
        color = 'red'
        color = 'green'
        color = 'blue'
        color = 'yellow'
        network.set_edges_attributes(edges) for i in range(0, 1000):
            color = 'red'
            color = 'green'
            color = 'blue'
            color = 'yellow'

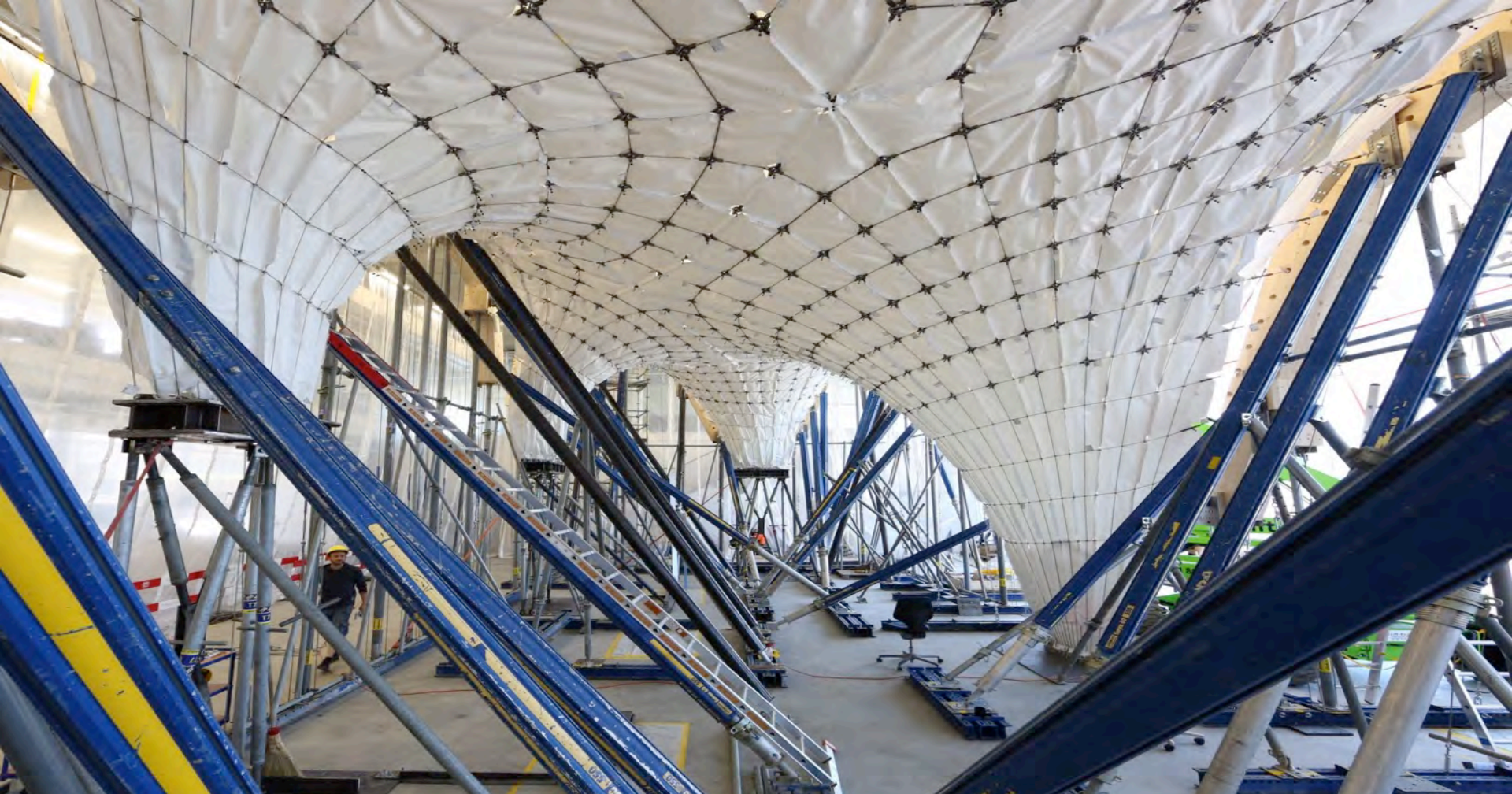
# Plot Histograms

fig = plt.figure(figsize=(10, 10))
plt.hist(hist, bins=100, label='0 mm')
plt.hist(hist, bins=100, label='0 mm')
plt.hist(hist, bins=100, label='0 mm')
plt.hist(hist, bins=100, label='0 mm')
plt.legend(loc='upper right')
plt.xlabel('Stress [MPa]')
plt.ylabel('Count')
plt.grid()
plt.show()

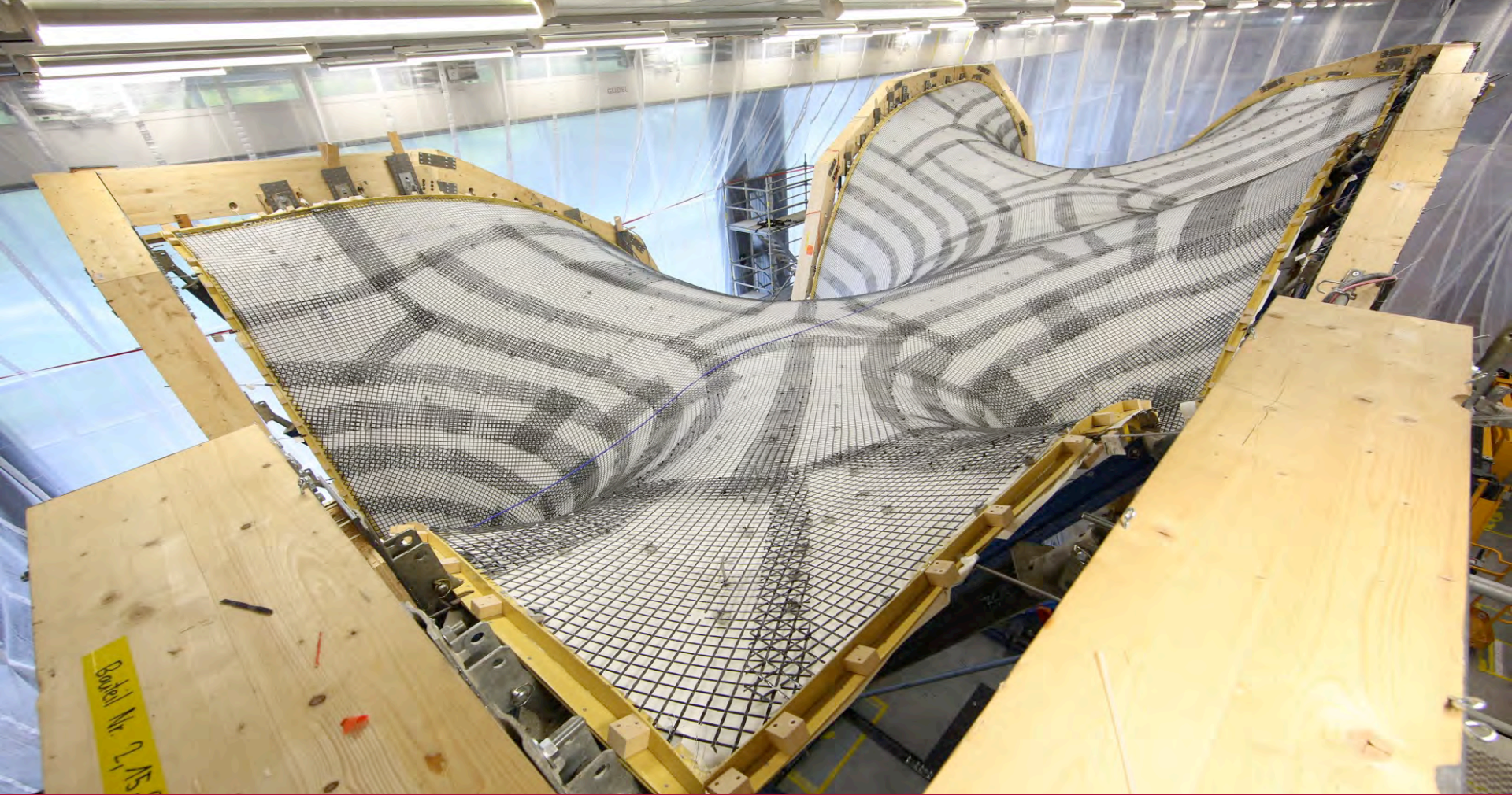
fig = plt.figure(figsize=(10, 10))
plt.hist(hist, bins=100, label='0 mm')
plt.hist(hist, bins=100, label='0 mm')
plt.hist(hist, bins=100, label='0 mm')
plt.legend(loc='upper right')
plt.xlabel('Stress [MPa]')
plt.ylabel('Count')
plt.grid()
plt.show()
```





















NEST HiLo Roof Prototype

Zurich, Switzerland

2016-2017

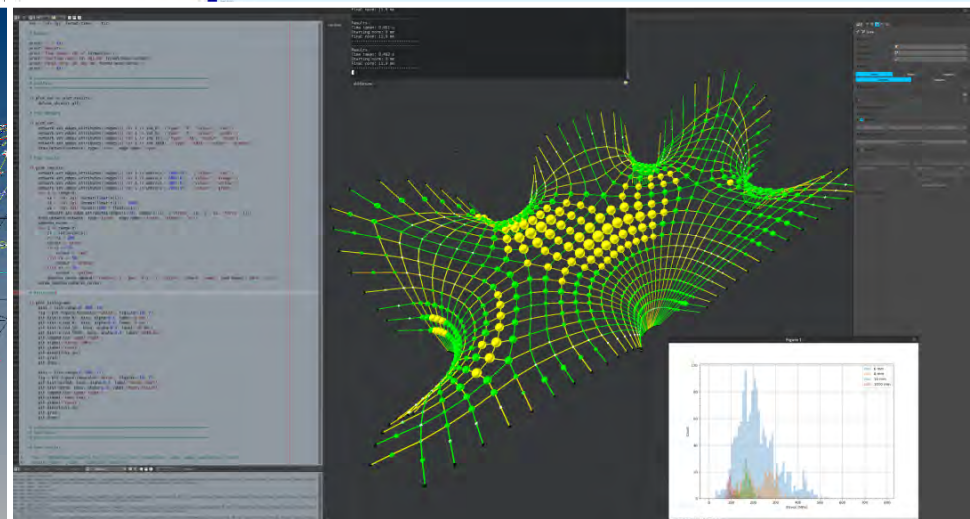
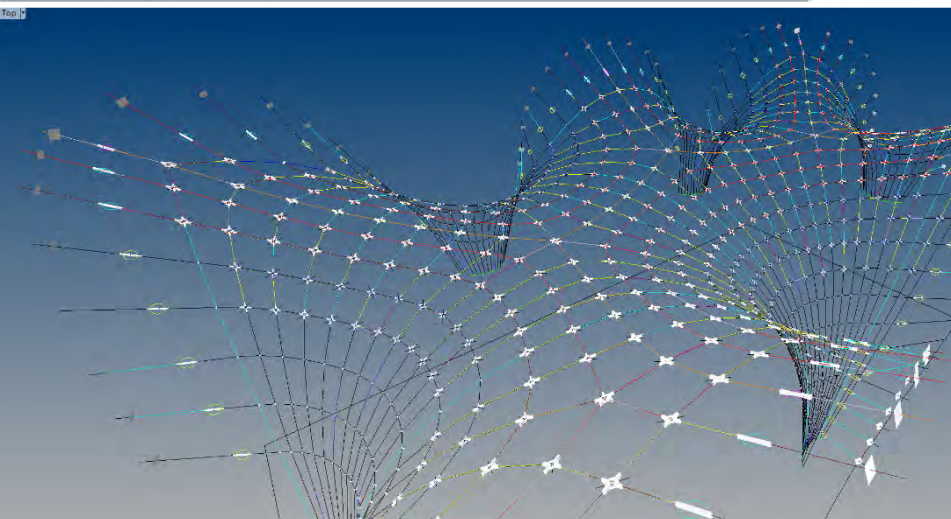
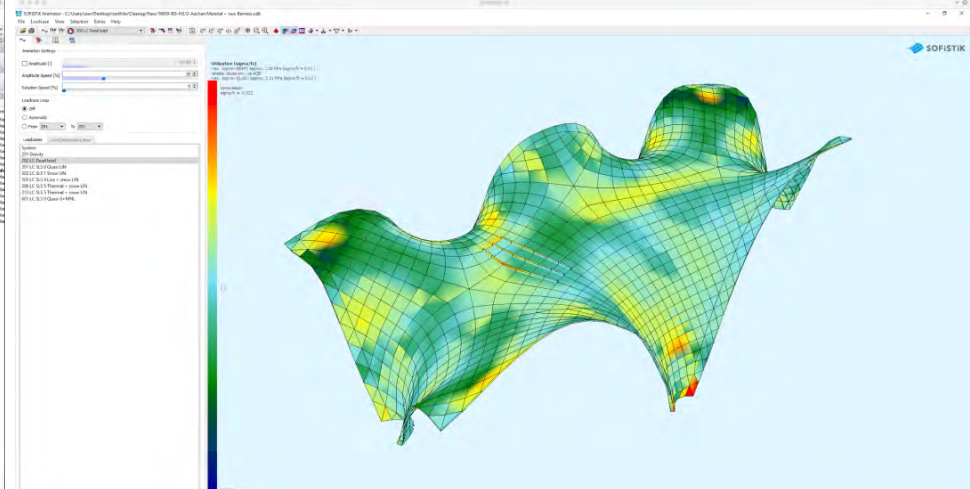
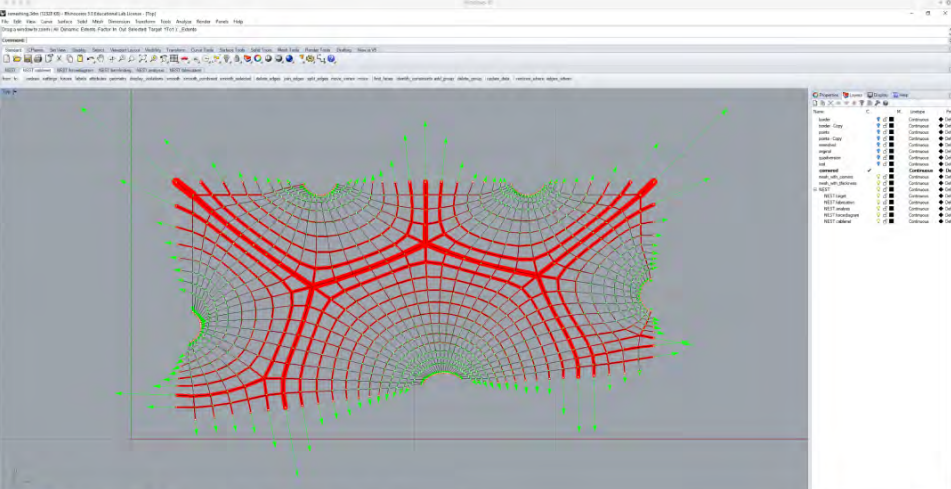
Block Research Group, ETH Zürich
with supermanoeuvre

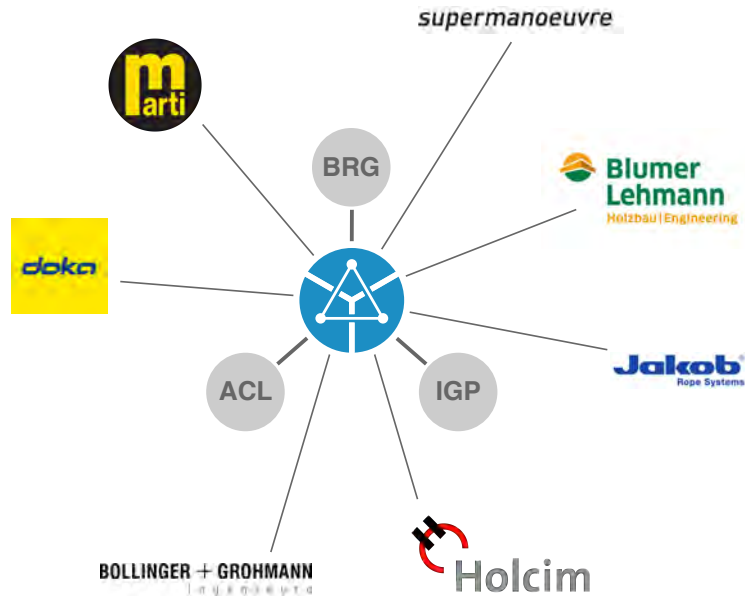














?!





KnitCandela

MUAC, Mexico City, Mexico 2018

Design

Zaha Hadid Architects CODE:

Filippo Nassetti, David Reeves, Marko Margeta,
Shajay Bhooshan, Patrik Schumacher

Block Research Group

Mariana Popescu, Matthias Rippmann, Tom Van
Mele, Philippe Block

Structural engineering

Block Research Group

Andrew Liew, Tom Van Mele

Fabrication

Block Research Group

R-Ex

KnitCrete Technology

Block Research Group

Mariana Popescu, Tom Van Mele, Philippe Block

Physical Chemistry of Building Materials

Lex Reiter, Robert Flatt

Concrete development

Holcim Mexico

Site Coordination

R-Ex

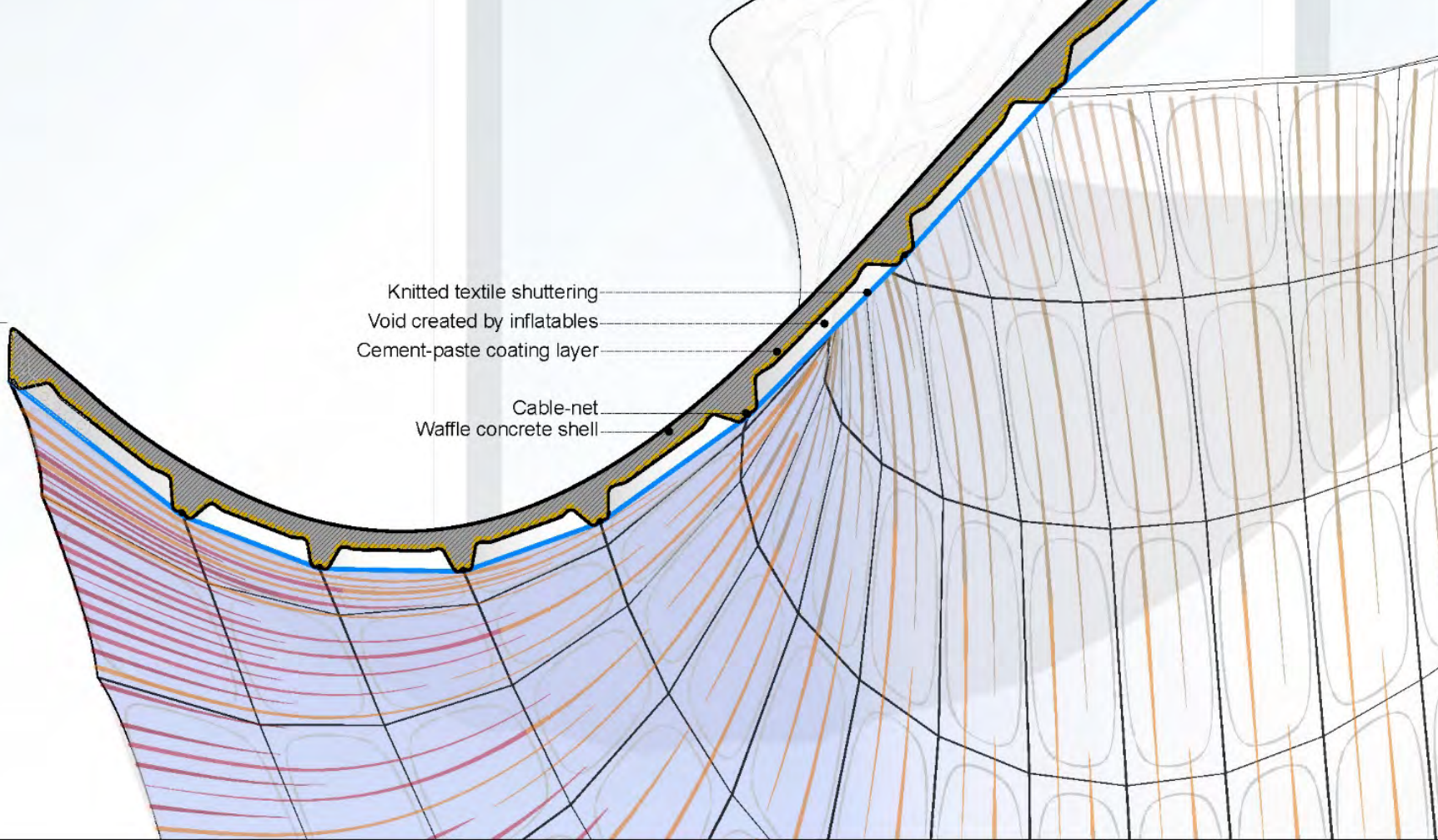
Alicia Nahmad Vasquez



+3.30 m



- Knitted textile shuttering
- Void created by inflatables
- Cement-paste coating layer
- Cable-net
- Waffle concrete shell



















KnitCandela

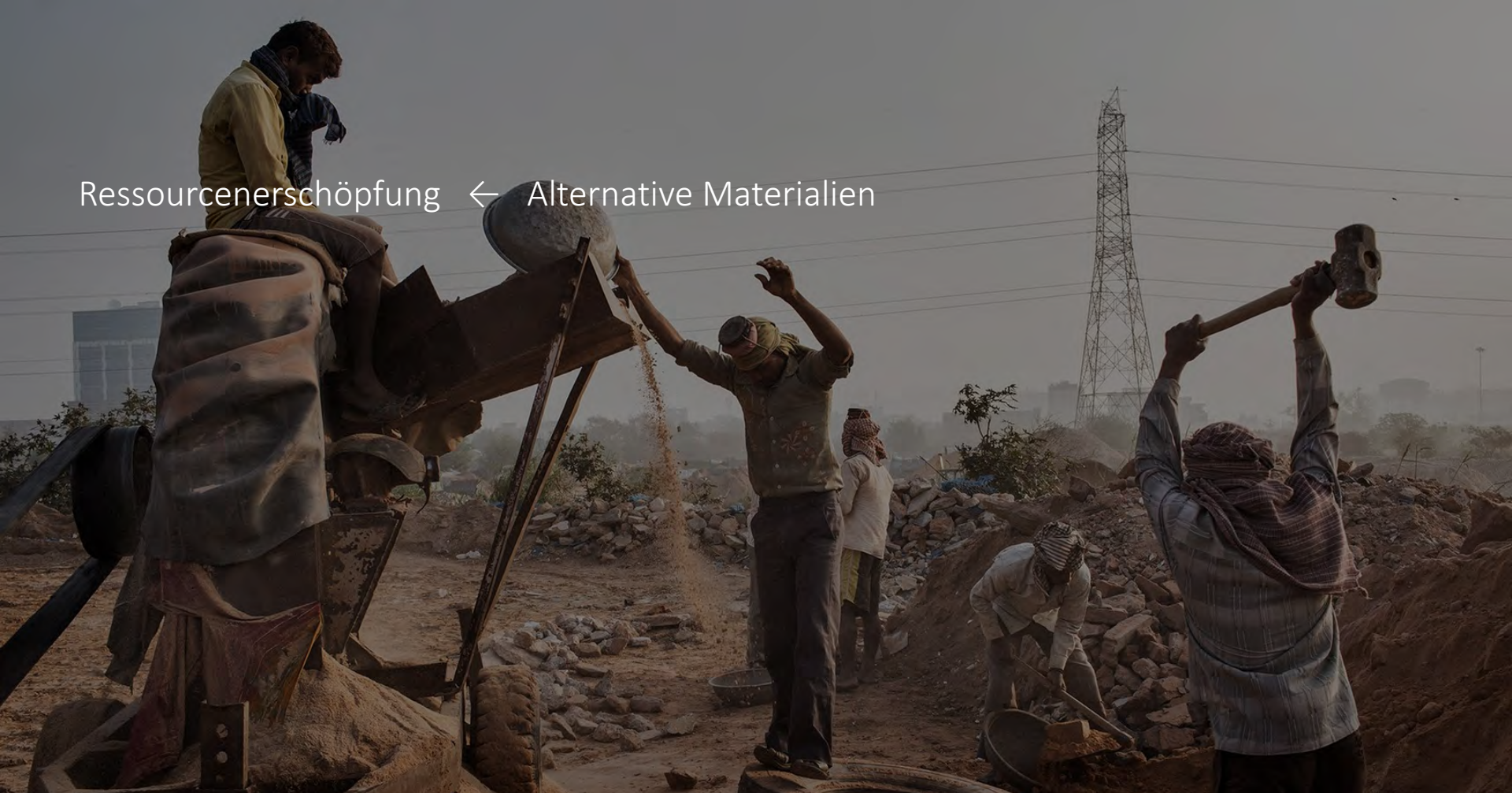
MUAC, Mexico City, 2018







Ressourcenerschöpfung ← Alternative Materialien





MycoTree - Seoul Biennale by Dirk Hebel & BRG, Seoul, Korea (2017)



ETH Zurich Pavilion by Dirk Hebel & BRG, New York City, NY, USA (2015)

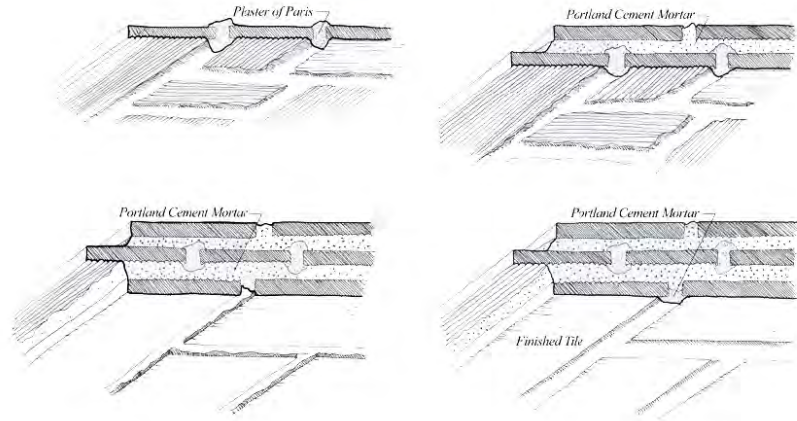


Self-supporting Formworks by BRG & DBT, Zürich, Switzerland (2018)



Droneport Prototype by Foster Foundation & BRG, Venice, Italy (2016)















C O M P A S



rhinoVAULT

```
for k in range(1, len(mesh.vertex_coordinates)):
    mesh.vertices[key] = mesh.vertex_coordinates[key]

for key in mesh.vertices():
    if key in fixed:
        continue

    p = key_xyz[key]

    nbirs = mesh.vertex_neighbours(key, order=True)
    c = center_of_mass_polygon([key_xyz[key] for key in nbirs])

    # update
    attr = mesh.vertex[key]
    attr['x'] += d * (c[0] - p[0])
    attr['y'] += d * (c[1] - p[1])
    attr['z'] += d * (c[2] - p[2])

if callback:
    callback(mesh, k, callback_args)

def smooth_mesh_length(mesh, lmin, lmax, fixed=None):
    """
    """
    if callback:
        if not callable(callback):
            raise Exception('Callback is not callable')

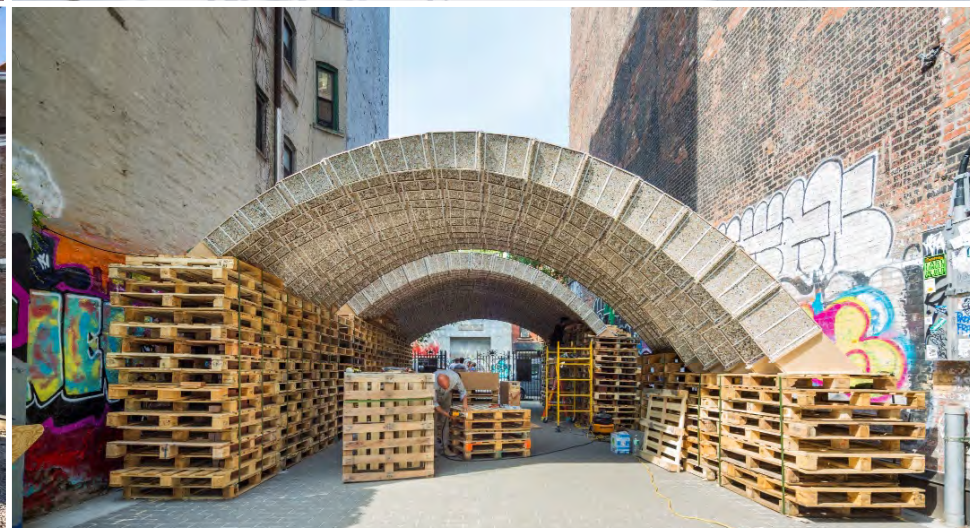
    fixed = fixed or []
    fixed = set(fixed)

    for k in range(1, len(mesh.vertex_coordinates)):
```

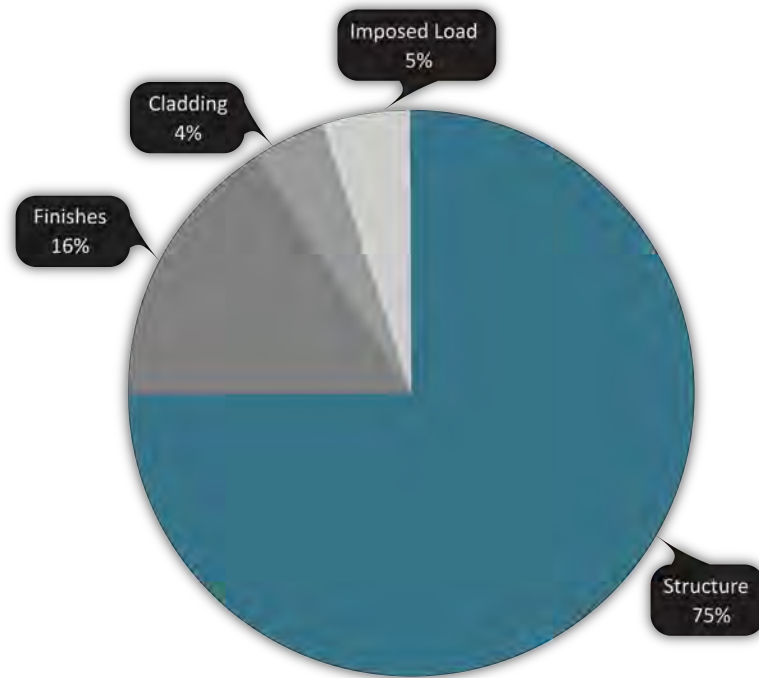




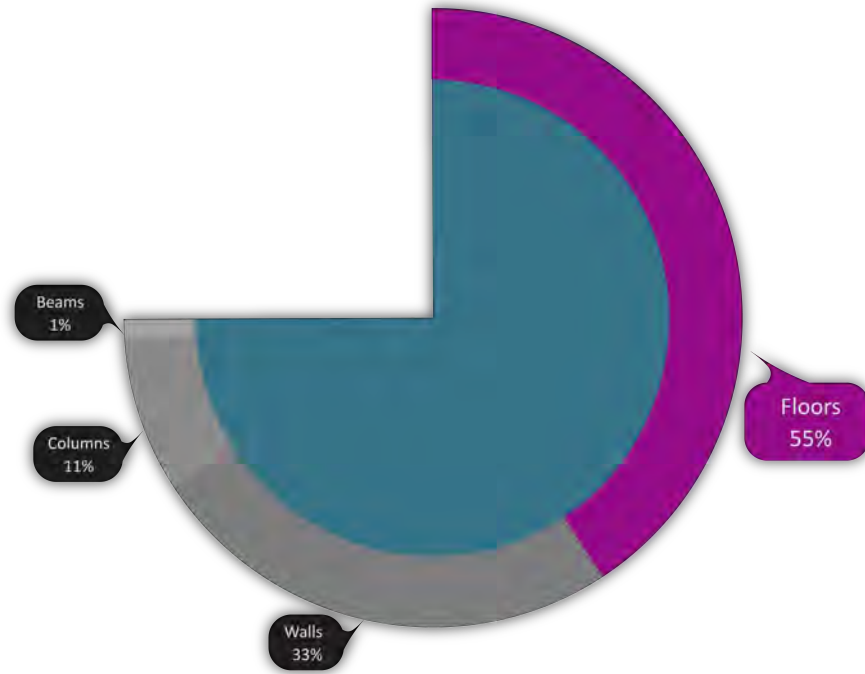






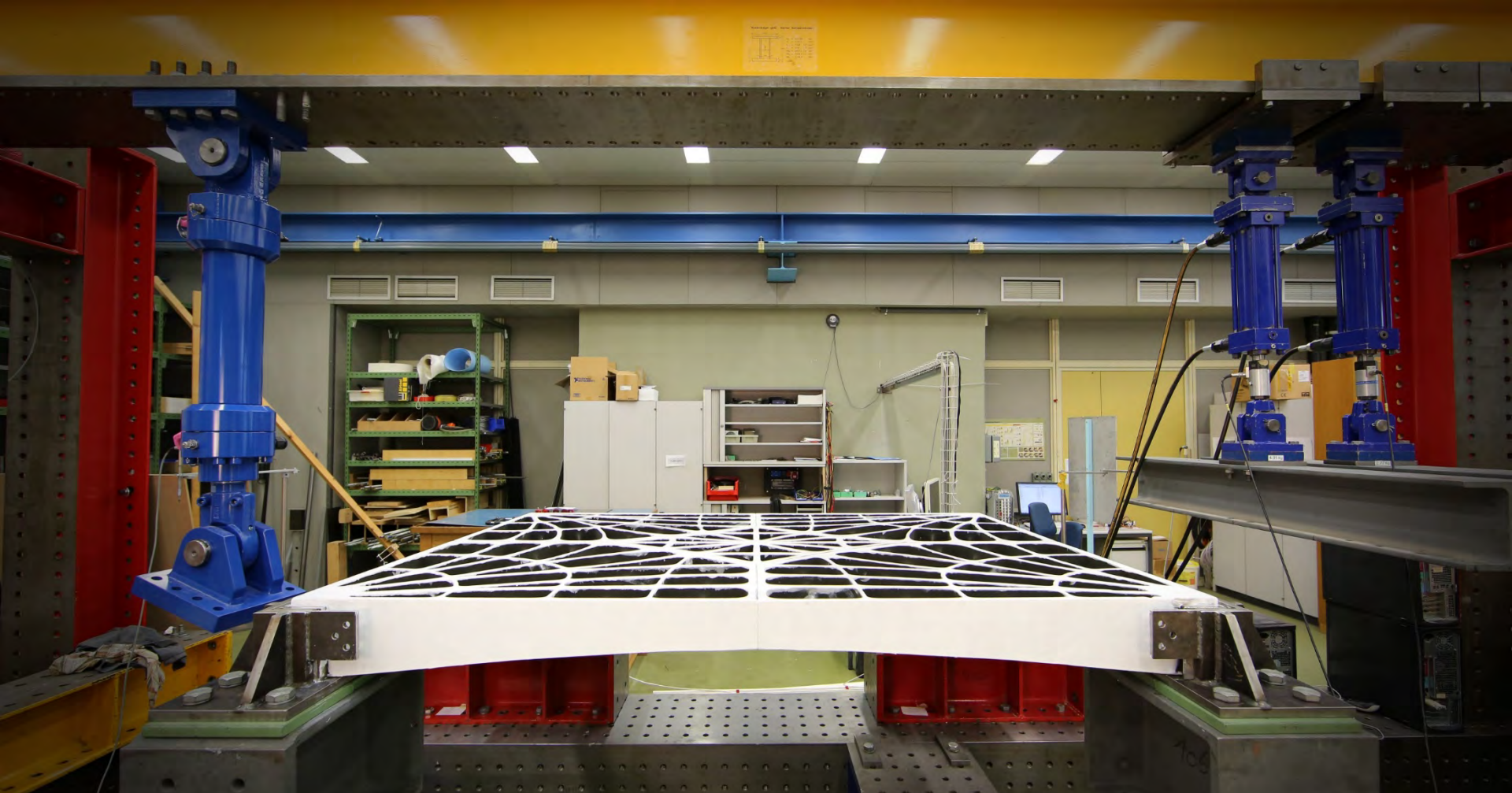


Gewichtsanteile je Bauteilkategorie



Gewichtsanteile je Tragstrukturelement





Bis zu 70% weniger Material









CNC DYNAMIX
TECHNOLOGIE DEANLEISTUNG





15th International Architecture Exhibition
La Biennale di Venezia

Funicular Shell Floors Beyond Bending

Block Research Group
ETH Zurich

