05

COMPUTATIONAL DESIGN

Cheese Stairs

By Python, Rhinoceros and Grasshopper

Task:

The Lab of Constructioning Bridges and Roads of Southeast University is now waiting to be ***REFORMED.***

Condition:

1. No Electricity;

2. No Pipe;

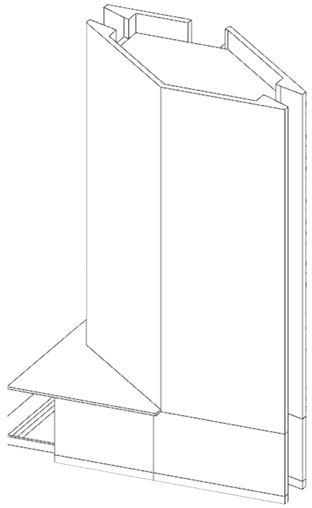
3. No Construction Machine;

How? Here's our answer.

Conditions

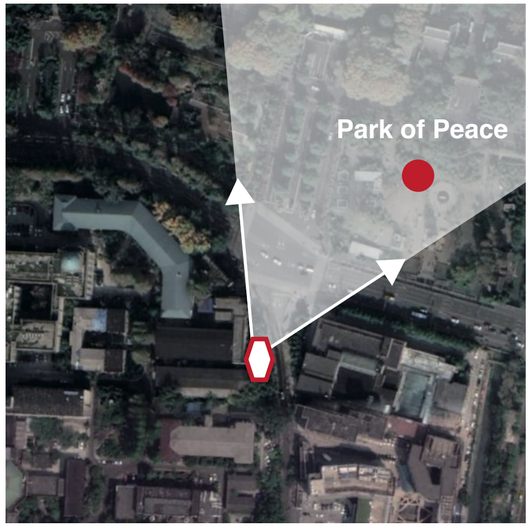
History

Lab of Constructioning Bridges and Roads, built in 1978. The original stairs is diamond-shaped to approach with the street.



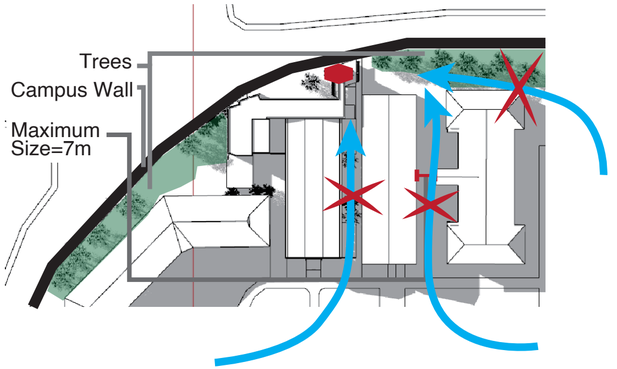
Site & Vision

It's been the focus of the crossing ever since. At the same time, It provided the best view to the Park of Peace crossroads.



Why no construction machines?

The campus building has a history for more than 50 years averagely, and the roads to the site is no wider than 7m.
For teachers and students there, save the noises for them would be a better choice.



Why no water and electricity?

Due to the situation of the building, it's hard to attach wires and electronics to the stairs.

Solution: Prefabrication

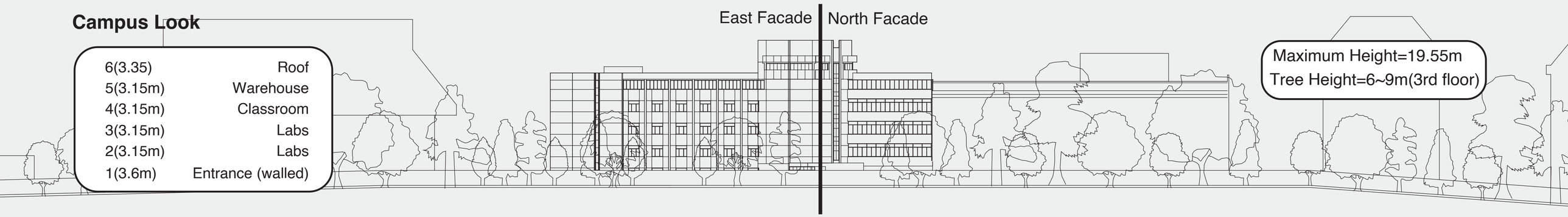
In order to build without machines, the only way to build is to "build elsewhere, import to site": Prefabrication.

By 3D printing elements in the factory with its least size at no more than 2m, they can be transported by trucks.

Once moved to site, elements are put together and worked on connections.

Theoretically the assembling takes just 2 days in silent.

Campus Look

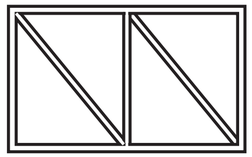


Choosing Cheese

After establishing prefabrication as the way to build, the new question is the shape to use.

However, the best option can be simple. Here's the logic chain.

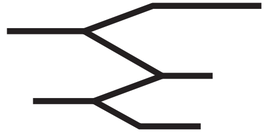
What does a stair need?

- Stable
  - Other Structures
    - Arch
  - Triangle
    - 
    - Frame(Truss)

What is prefabrication good at?

- Curve
  - Circle
    - 
    - Digging bubbles in the box, at places that are shined and blowed.

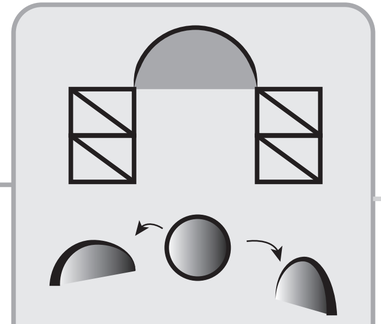
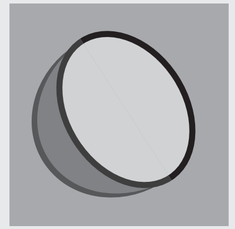
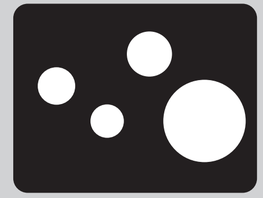
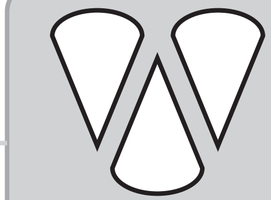
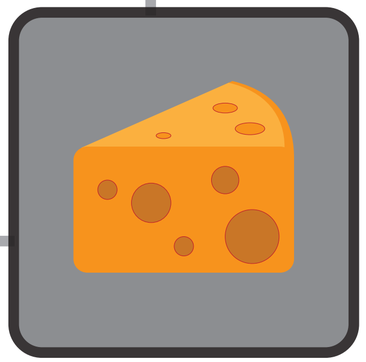
What does the site need?

- Good Vision
  - 
    - Holes
  - 
    - Platforms
    - 
    - Stair lines go through the box. They encounter bubbles.

Starting at three basic questions, we raise some targets for our project.

Associating with the properties narrows our scope of selection: Pourous, Arch, Triangle.

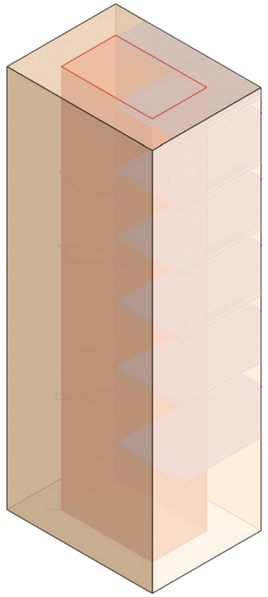
Finally, cartoon cheese gives the best imagination.

- 
  - Spheres makes arches in every section. Arches do well in structure.
- 
  - Sphere lets light in, and makes soft edges, which shows a wide angle
- 
  - Orifices give wind a path, which helps to improve ventilation.
- 
  - Pourous
- 
  - Arc & Triangle
- 

Generating Cheese

We use ***Rhinoceros*** and ***Grasshopper*** to Generate the Cheese Stair, vision and ventilation are optimized.

Here's how.

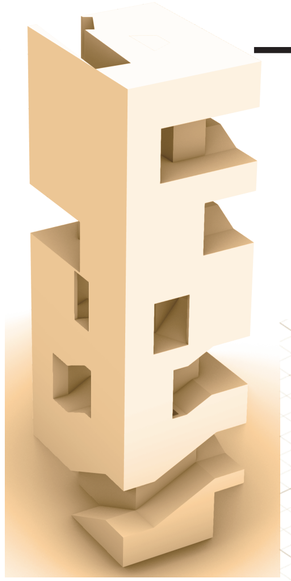


A. Boundingbox
Original defined space.

MINUS

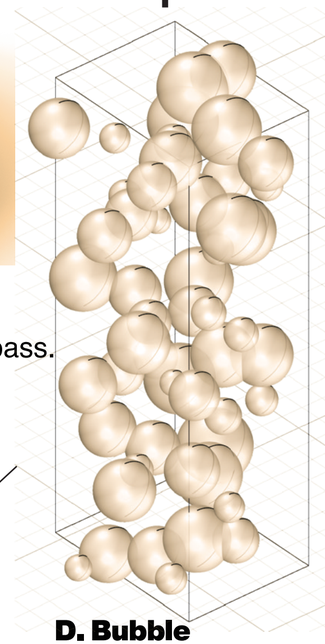
B. Space for stairs
Random path with no offense to structure and remain balance.

```
def spc_nts(pointn,pointm,m):
    pntmnh=rs.PointCoordinates(pointm)
    pntnnh=rs.PointCoordinates(pointn)
    pointm_newheight=rs.AddPoint(pntmnh.X,pntmnh.Y,pntnnh.Z)
    length=rs.Distance(pointn,pointm_newheight)
    stairs=length//q
    if m<5:
        height_of_next_layer=ha*(m+1)
    else:
        height_of_next_layer=ha*m+hb*1
    ntg=(height_of_next_layer-pntnnh.Z)/p
    if stairs>ms:
        return False,ntg
    if pntnnh.Z+p*stairs>height_of_next_layer:
        return False,ntg
    if pntnnh.Z+p*stairs<=height_of_next_layer:
        return True,ntg
def spc_ubl(pointn,pointm,m):
    pntmnh=rs.PointCoordinates(pointm)
    pntnnh=rs.PointCoordinates(pointn)
    pointm_newheight=rs.AddPoint(pntmnh.X,pntmnh.Y,pntnnh.Z)
    length=rs.Distance(pointn,pointm_newheight)
    stairs=length//q
    if m<5:
        height_of_next_layer=ha*(m+1)
    else:
        height_of_next_layer=ha*m+hb*1
    ntg=(height_of_next_layer-pntnnh.Z)/p
    if pntnnh.Z+p*stairs<height_of_next_layer:
        return False,ntg
    if pntnnh.Z+p*stairs>=height_of_next_layer:
        return True,ntg
    if pntnnh.Z+p*stairs>=height_of_next_layer:
        return True,ntg
def ghtp(lastpoint,start,end,floor,iflast,ifflat):
    newpoint=ppfl(start,end)
    lstpnt=rs.PointCoordinates(lastpoint)
    newpoint=ghtpnt(newpoint,lstpnt.Z)
    spc_judge,ntg=spc_nts(lastpoint,newpoint,floor)
    count=0
    while(spc_judge==False):
        count+=1
        newpoint=ppfl(start,end)
        newpoint=ghtpnt(newpoint,lstpnt.Z)
        if count>10:
            newpoint=ghtpnt(end,lstpnt.Z)
            spc_judge=True
    if ifflat or ntg==0:
        linen=rs.AddLine(lastpoint,newpoint)
        midpoint=ppfl(lastpoint,newpoint,0.5)
        linen=rs.AddLine(lastpoint,midpoint)
        linem=rs.AddLine(midpoint,newpoint)
    else:
        spc_judge,ntg=spc_ubl(lastpoint,newpoint,floor)
        if iflast or spc_judge:
            ratio=ntg*q/rs.Distance(lastpoint,newpoint)
            spc_judge,ntg=spc_tm(lastpoint,newpoint,floor)
            count=0
            while(spc_judge==False):
                count+=1
                newpoint=ppfl(start,end)
                spc_judge,ntg=spc_tm(lastpoint,newpoint,floor)
                if count>10:
                    newpoint=ghtpnt(end,lstpnt.Z)
                    spc_judge=True
            linen,linem,ending=iafd_al(lastpoint,newpoint,ntg)
        else:
            linen,linem,ending=iafd(lastpoint,newpoint)
        newpoint=ending
    return linen,linem,newpoint
```



C. Basic Entity
Volume of common stairs and space to pass.

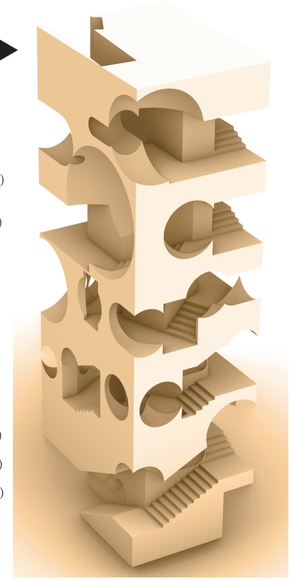
MINUS



D. Bubble
Aimed spheres for optimizing physical factors for stairs

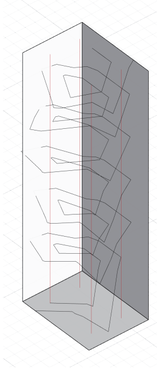
```
for j in range(M):
    height_of_next_layer=ha*(j+1)
    if j==0:
        for i in range(3):
            if i==0:
                linen,linem,pointm=ghtp(zsi,sppua,zxi,j,False,False)
            if i==1:
                linen,linem,pointm=ghtp(pointm,zxi,sppub,j,False,True)
            if i==2:
                linen,linem,pointm=ghtp(pointm,ec,sppuc,j,True,False)
                new_sppuc=rs.AddPoint(a-d,c+1,height_of_next_layer)
                if not rs.PointCompare(pointm,new_sppuc):
                    llhl.append(rs.AddLine(pointm,new_sppuc))
                pointm=rs.AddPoint(a-d,e,height_of_next_layer)
            llhl.append(linen)
            llhl.append(linem)
        llhl.append(rs.AddLine((a-d,c+1,height_of_next_layer),
        (a-d,e,height_of_next_layer)))
    elif j in [1,2,3,4,5]:
        if j==M-1:
            height_of_next_layer=ha*j+hb
        for i in range(5):
            if i==0:
                linen,linem,pointm=ghtp(pointm,ee,ysi,j,False,True)
            if i==1:
                linen,linem,pointm=ghtp(pointm,sppda,zsi,j,False,False)
            if i==2:
                linen,linem,pointm=ghtp(pointm,zsi,sppdb,j,False,True)
            if i==3:
                linen,linem,pointm=ghtp(pointm,zxi,sppdc,j,False,False)
            if i==4:
                linen,linem,pointm=ghtp(pointm,yxi,ec,j,True,False)
                new_ec=rs.AddPoint(a-d,c,height_of_next_layer)
                if not rs.PointCompare(pointm,new_ec):
                    llhl.append(rs.AddLine(pointm,new_ec))
                pointm=rs.AddPoint(a-d,e,height_of_next_layer)
            llhl.append(linen)
            llhl.append(linem)
        llhl.append(rs.AddLine((a-d,c,height_of_next_layer),
        (a-d,e,height_of_next_layer)))
lhl=rs.JoinCurves(llhl,False)
newcurve=rs.OffsetCurve(lhl[0],zx,d,hv,1)

pts=[]
for i in range(1):
    if Line1:
        points=rs.DivideCurve(Line1,25)
        for pt in points:
            rs.AddPoint(pt)
            pts.append(pt)
    if Line2:
        points=rs.DivideCurve(Line2,25)
        for pt in points:
            rs.AddPoint(pt)
            pts.append(pt)
balls=[]
for i in range(1):
    for pt in pts:
        radius=random.random()+0.5
        center=rs.AddPoint(pt)
        ball=rs.AddSphere(center,radius)
        balls.append(ball)
z0=0.5*random.random()+1
for i in range(1):
    for balls in balls:
            rs.MoveObjects(balls,(0,0,z0))
```

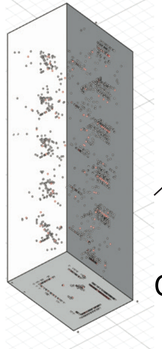


E. Outcome

```
def aptfm(list,floor):
    m=floor
    if m<5:
        height_of_next_layer=ha*(m+1)
    else:
        height_of_next_layer=ha*m+hb*1
    ree=rs.AddPoint(a-d,e,height_of_next_layer)
    rec=rs.AddPoint(a-d,c,height_of_next_layer)
    rpt=rs.AddLine(rec,ree)
    return list.append(rpt)
def iafd(pointn,pointm):
    pntmnh=rs.PointCoordinates(pointm)
    pntnnh=rs.PointCoordinates(pointn)
    pointm_newheight=rs.AddPoint(pntmnh.X,
    pntmnh.Y,pntnnh.Z)
    length=rs.Distance(pointn,pointm_newheight)
    stairs=length//q
    inclined_length=stairs*q
    ratio=inclined_length/length
    pointq=rs.AddPoint(pntnnh.X+(pntmnh.X-pntnnh.X)
    *ratio,pntnnh.Y+(pntmnh.Y-pntnnh.Y)
    *ratio,pntnnh.Z+stairs*p)
    pointm=rs.AddPoint(pntmnh.X,pntmnh.Y,
    pntnnh.Z+stairs*p)
    linen=rs.AddLine(pointn,pointq)
    linem=rs.AddLine(pointq,pointm)
    ending=pointm
    return linen,linem,ending
```



Root Path



Center

Making Cheese

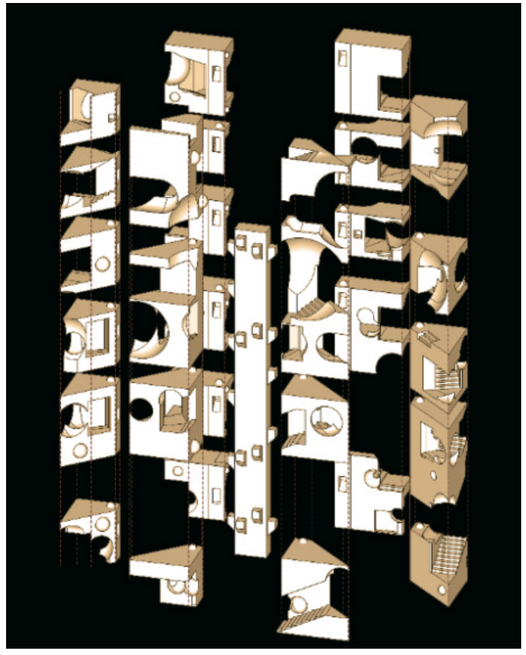
After shapes are done, it's time to dive into the details. The questions are:

- ***How to construct?***
- ***How to attach?***
- ***How to assemble?***

A. Assembling

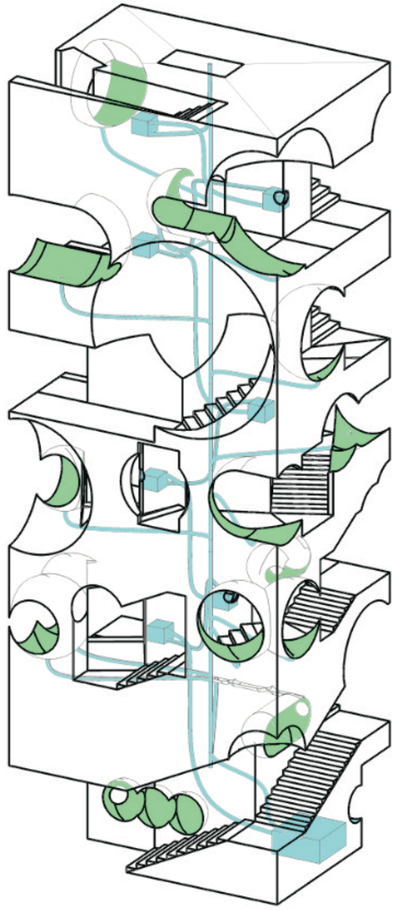
By dividing the shape into 30 parts along the arches, stucture stays strong and portable.

With a small iron frame stem, the elements can be attached.



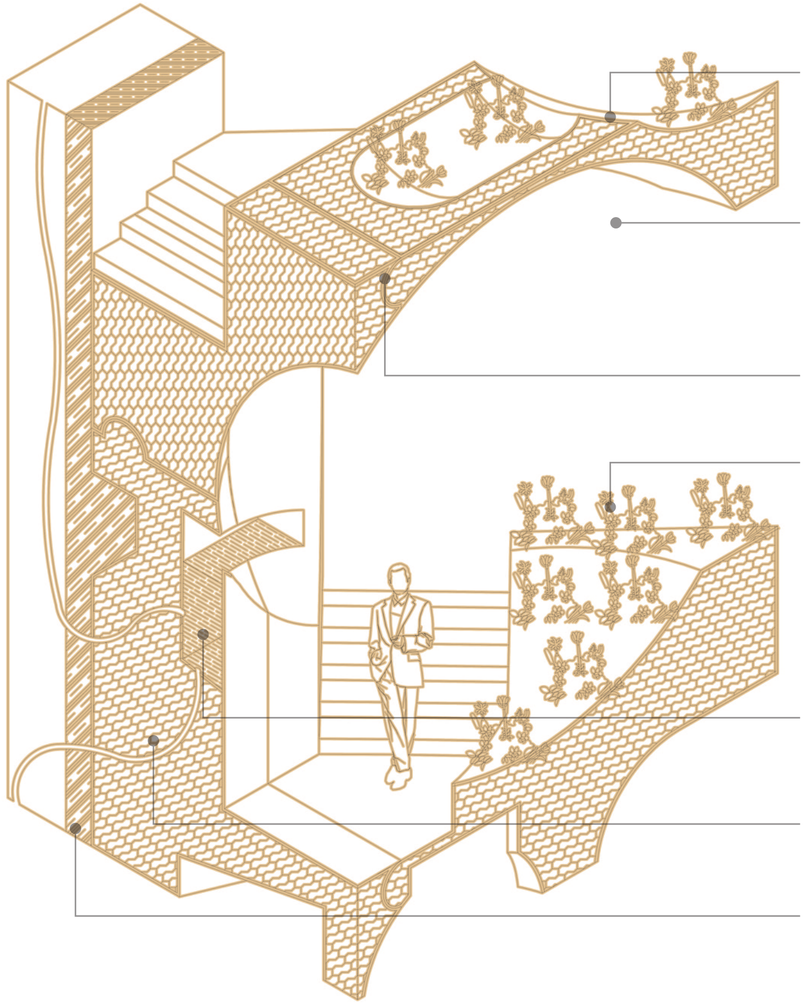
B. Draining & Irrigating

The lower parts of the spheres naturally collect water. Through the tubes, irragating water balances with the height.



C. Tectonics

Elements each have its own bulges and pits to connect, and space for a concrete wall as to strengthen the stairs.



Spherical Stucture
Makes arches in every section, stand stable

Light & Ventilation
Provided by bubble spaces

Telescopic Seam

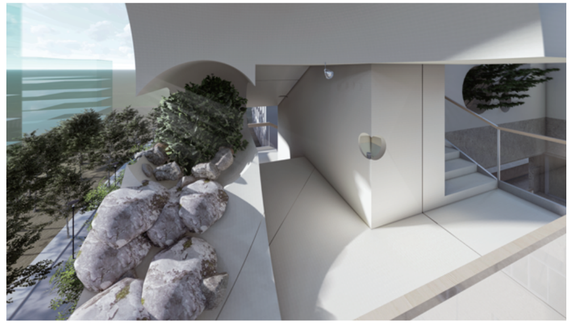
Plant Area
Collects precipitation, provides landscape

Rabbet

Drainage Tube

Loading Wall
Concrete. Concealed. Coherent.

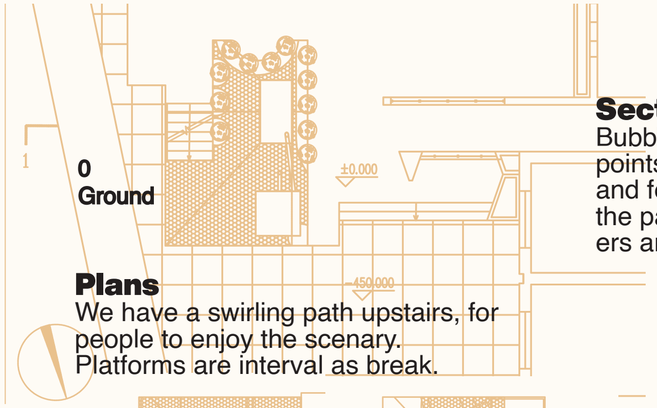
Outcomes of Cheese



▲ **Platform**
This pic shows a good vision from the platform between 3th and 4th floor.
It is bright enough to close lights.

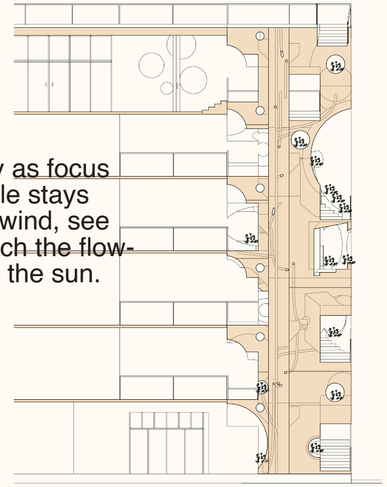
▲ **Entrance**
It shows the entrance as a good site for wind blowing.

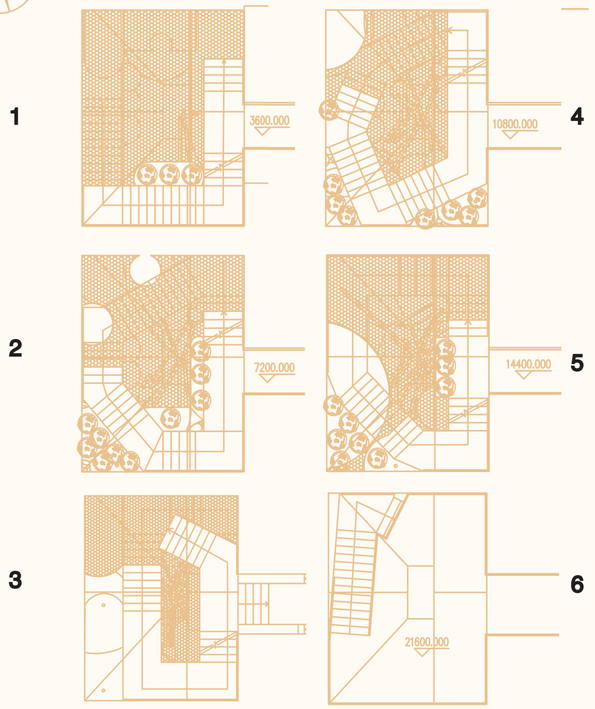
North Facade ▶
The bubbles shows a distinguished face toward vihecles passing by and the park .



Plans
We have a swirling path upstairs, for people to enjoy the scenary.
Platforms are interval as break.

Section
Bubbles play as focus points. People stays and feel the wind, see the park, touch the flowers and bath the sun.





1:100 Model