

Affine Transformation 3

Computational Visual Design Laboratory
(<https://github.com/cvlab>) “Roma Tre” Univ, Italy

Computational Graphics 2013

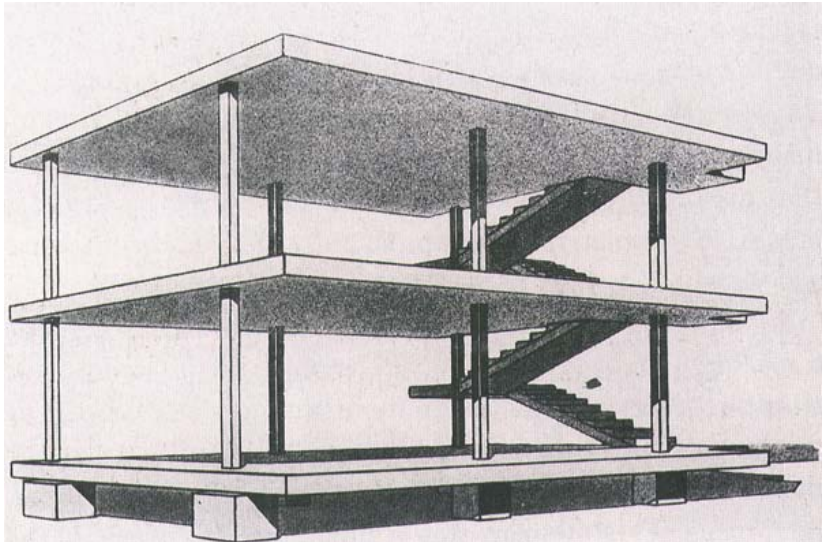


Contents

Building fabric modeling

A great example: Maison Domino

Le Corbusier in 1920's developed the Maison-Domino, a basic building prototype for mass production with free-standing pillars and rigid floors.

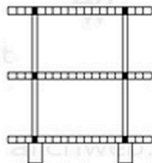


A great example: Maison Domino

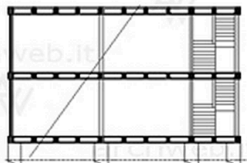
Nel 1914 Le Corbusier elabora un sistema costruttivo adatto ad affrontare i problemi della ricostruzione postbellica. Le Corbusier progetta quindi un sistema strutturale, una sorta di scheletro del tutto indipendente dalla pianta della casa. Questa ossatura, che sostiene esclusivamente i solai e le scale è costituita da elementi standard componibili e permette una grande varietà nell'aggregazione delle unità.



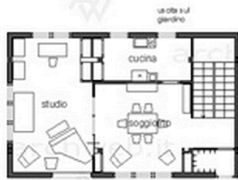
ALLOGGIO TIPO 1 - PIANO TERRA



SEZIONE



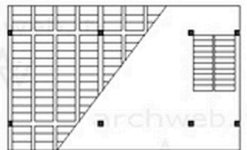
SEZIONE LONGITUDINALE



ALLOGGIO TIPO 2 - PIANO TERRA



ALLOGGIO TIPO 2 - PIANO PRIMO



SEZIONE SOLAIO



The building fabric

A functional breakdown of the construction

- ▶ foundations
- ▶ building frame
 - ▶ beams
 - ▶ pillars
- ▶ building enclosures
 - ▶ horizontal enclosures
 - ▶ vertical enclosures
- ▶ building partitions
 - ▶ horizontal partitions
 - ▶ vertical partitions
- ▶ vertical communications
 - ▶ staircases
 - ▶ elevators
- ▶ mechanical, electrical, and plumbing (MEP) plants



Utility functions

```
from pyplasm import *  
GRID = COMP ( [ INSR ( PROD ) , AA ( QUOTE ) ] )
```

Input a list of lists of numbers

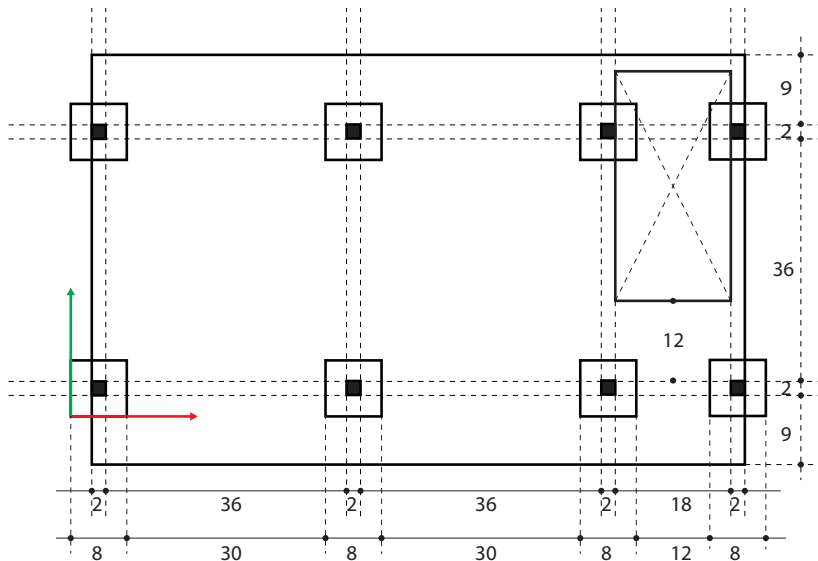
Output a geometric value (HPC – Hierarchical Polyhedral Complex)

Useful to generate a grid of convex cells from either 2 or 3 lists of positive or negative numbers, corresponding to full or empty cells, respectively



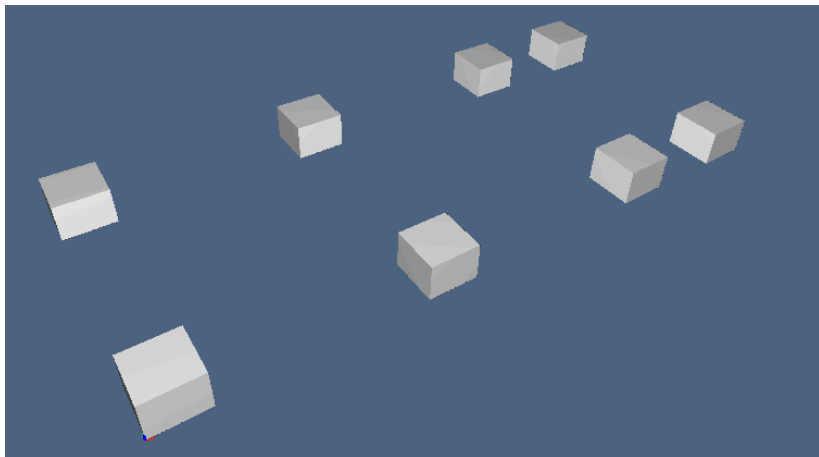
Unit side and Fundamental measures

we assume all measures given in multiples of $M = 10 \text{ cm}$



Foundations

notice the position of origin of the coordinate frame



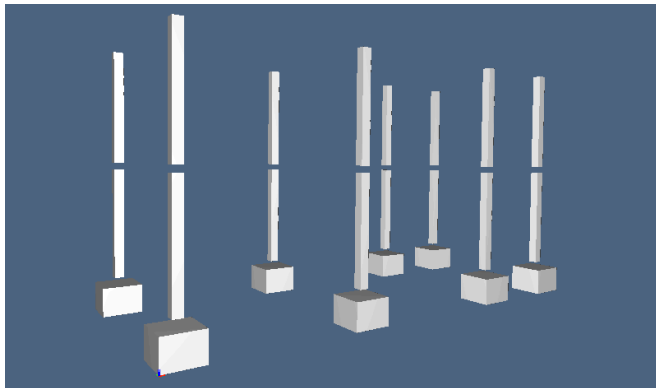
```
foundations = GRID([[8,-30,8,-30,8,-12,8],[8,-30,8],[6]])  
VIEW(foundations)
```



Pillars

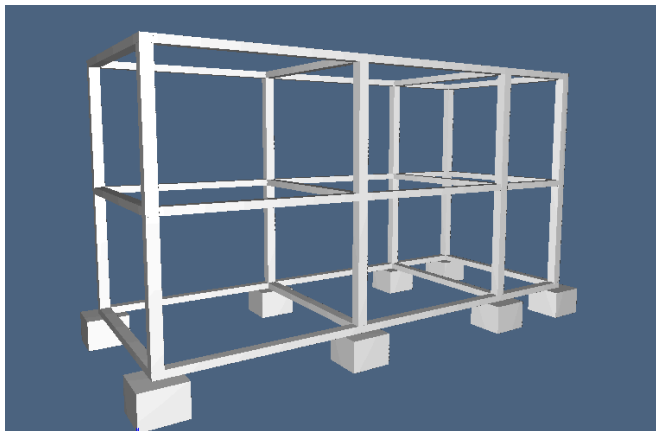
notice the initial (implicit) translation

```
pillars = GRID([[-3,2,-36,2,-36,2,-18,2], [-3,2,-36,2],  
               [-7.4,23.6,-1.4,23.6,-1.4]])  
building = STRUCT([foundations,pillars])  
VIEW(building)
```

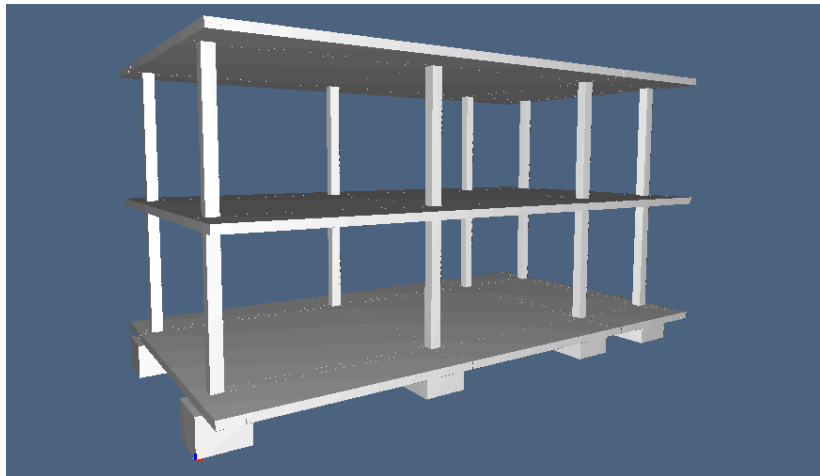


Building frame

```
beams_x = GRID([[-3,2,36,2,36,2,18,2],[ -3,2,-36,2],  
               [-6,1.4,-23.6,1.4,-23.6,1.4]])  
beams_y = GRID([[-3,2,-36,2,-36,2,-18,2],[ -3,2,36,2],  
               [-6,1.4,-23.6,1.4,-23.6,1.4]])  
beams = STRUCT([beams_x,beams_y])  
frame = STRUCT([foundations,pillars,beams])  
VIEW(frame)
```



Horizontal enclosures and partitions



```
horiz_partitions = T(2) (-6) (GRID ([[ -1,2,2,36,2,36,2,18,2,2],  
    [9,2,36,2,9], [-6,1.4,-23.6,1.4,-23.6,1.4]]))  
building = STRUCT ([frame, horiz_partitions])  
VIEW (building)
```



Horizontal enclosures and partitions

When Boolean operations are available, to make some hollows is much easier :o)

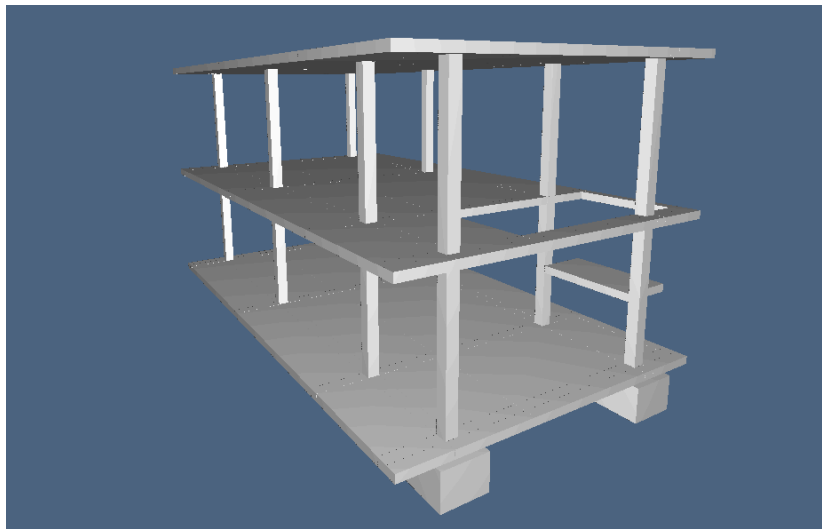
```
hollow = T([1,2,3])([5+38*2, 5+12, 6+25])(CUBOID([18,33,1.4]))
horiz_partitions = DIFFERENCE([horiz_partitions, hollow])
building = STRUCT([frame,horiz_partitions])
VIEW(building)
```

otherways:

```
hpartition1 = GRID([[-1,2,2,36,2,36,2,-18,2,2],[9,2,12,24,2,9],
[-6,1.4,-23.6,1.4,-23.6,1.4]])
hpartition2 = GRID([[-1-2-2-36-2-36-2,18],[9,2,12,-24,-9,2],
[-6,1.4,-23.6,1.4,-23.6,1.4]])
hpartition3 = GRID([[-1-2-2-36-2-36-2,18],[9,2,12,24,2,9],
[-6,1.4,-23.6,-1.4,-23.6,1.4]])
hpartition4 = GRID([[-1-2-2-36-2-36-2,18],[-9,-2,-12,-24,9],
[-6,1.4,-23.6/2 +1.25,1.4]])
horiz_partitions = STRUCT([hpartition1, hpartition2,
hpartition3, T(2)(-6)(hpartition4)])
building = STRUCT([frame, horiz_partitions])
VIEW(building)
```



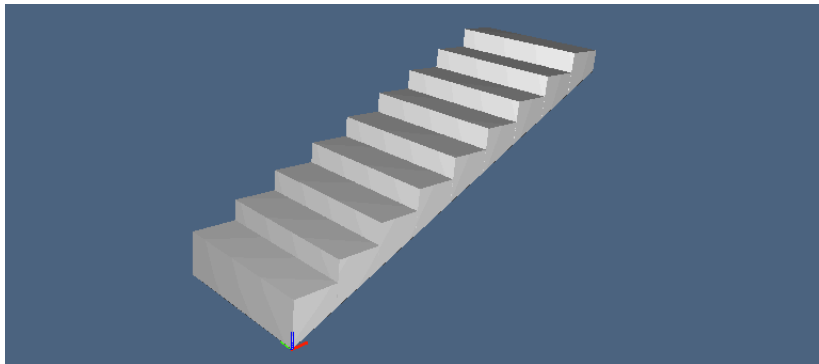
Horizontal enclosures and partitions



Staircases

Use `MAP` for permutations of coordinates

```
step2D = MKPOL ([[0,0], [0,2.65], [2.66,2.5/2], [2.66,2.65]],  
                [[1,2,3,4]],None))  
step3D = MAP ([S1,S3,S2]) (PROD ([step2D,Q(9)]))  
ramp = STRUCT (NN(9) ([step3D,T([1,3]) ([2.66,2.5/2])]))  
VIEW(ramp)
```



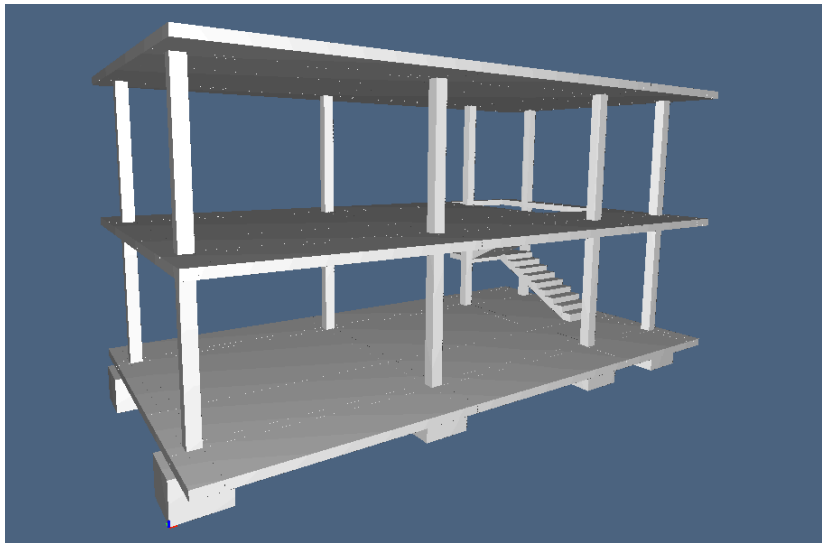
Staircases

```
ramp1 = T([1,2,3])([3+2+36+2+36+2+18,3+2+12,6])(R([1,2])(PI/2)(ramp))  
ramp2 = T([1,2,3])([3+2+36+2+36+2,3+2+12+24,6+25/2])(  
    R([1,2])(-PI/2)(ramp))
```

```
building = STRUCT([foundations,pillars,T(2)(-6)(horiz_partitions),  
    S(3)(1.05),ramp1,ramp2])  
VIEW(building)
```



Staircases



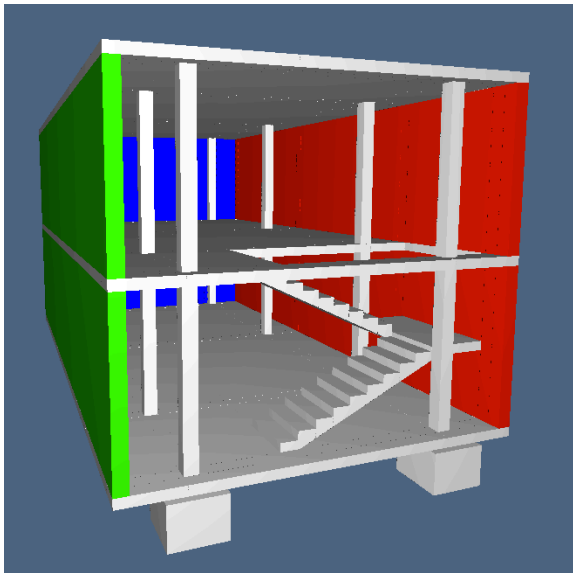
Vertical enclosures

Use colors to visually differentiate the external walls

```
enclosure_north = COLOR(RED) (GRID ([ [-1,2,2,36,2,36,2,18,2,2],  
    [-5,-2,-12,-24,-7,2], [-6,-1.4,23.6,-1.4,23.6,-1.4] ]))  
  
enclosure_south = COLOR(GREEN) (GRID ([ [-1,2+2+36+2+36+2+18+2+2],  
    [2], [-6,-1.4,23.6,-1.4,23.6,-1.4] ]))  
  
enclosure_west = COLOR(BLUE) (GRID ([ [-1,2], [-2,4,2,36,2,4,6],  
    [-6,-1.4,23.6,-1.4,23.6,-1.4] ]))  
  
building = STRUCT([foundations,pillars,T(2) (-6) (horiz_partitions),  
    ramp1,ramp2,enclosure_north,  
    T(2) (-6) (enclosure_south),T(2) (-6) (enclosure_west)])  
  
VIEW(building)
```



Vertical enclosures



Internal partitions

```
wall01 = COLOR(YELLOW) (T([1,2]) ([3+2+36+2+36+1, 3+2+12]) (  
    GRID([ [1], [24,-2,7], [-6,-1.4,23.6,-1.4,23.6] ])) )  
building = STRUCT([foundations,pillars,T(2)(-6)(horiz_partitions),  
    ramp1,ramp2, enclosure_north,T(2)(-6)(enclosure_south),  
    T(2)(-6)(enclosure_west), wall01])  
VIEW(building)
```



Internal partitions

