Concept and preliminary design of a hospital system *

Alberto Paoluzzi

January 14, 2015

Contents

*This document is part of the *Linear Algebraic Representation with CoChains* (LAR-CC) framework [CL13]. January 14, 2015

Abstract

In this module we develop stepwise the concept and the preliminary building program of a hospital of medium size, using as source the document [AM13] of the World Health Organisation.

1 Introduction

2 Model planning

2.1 Data sources

2.2 Reference grid

Reference grid

⟨ Reference grid 1 ⟩ ≡

```
""" Reference grid """
X = [0]+[7.5,9.5,7.5]+4*[8.4]+[7.5,9.5,7.5]+[0]
Y = [0]+14*[8.4]+[0]
xgrid = QUOTE(X[1:-1])
ygrid = QUOTE(Y[1:-1])
structuralGrid = PROD([xgrid,ygrid])
ymax = SUM(Y)
```

◇

Macro referenced in 20.

From array indices to grid coordinates

⟨ From array indices to grid coordinates 2a ⟩ ≡

```
""" From array indices to grid coordinates """
def index2coords(theArray):
    return CONS(AA(T([1,2]))(CAT((theArray).tolist())))
```

◇

Macro referenced in 20.

2.3 Architecture of modeling process

3 Building units planning

3.1 Wire-frame input

As already said, the data input for this project was made by hand. Of course, an interactive user-interface in underway. I would like to notice that to enter apart the coordinates of the vertices of cells, as two (or three) adjacent arrays, is much faster and lesser in danger of getting errors than to enter an array of points.

The several building units contained in this storey are given in the below script, each associated to a single ordered polyline, transposed on coordinates. Let us notice the used of a capitalised variable for storage, in order to distinguish from the corresponding `Struct` object with the same name.

⟨ Storey input 2b ⟩ ≡

```
""" Storey input """
⟨ Ground floor 3a ⟩
⟨ Mezanine floor 4b ⟩
⟨ First floor 6b ⟩
⟨ Second floor 8b ⟩
⟨ Third floor 10b ⟩
⟨ Fourth floor 11b ⟩
⟨ Fifth floor 12a ⟩

""" Building unit structure """
⟨ Ground floor structure 4a ⟩
⟨ Mezanine floor structure 6a ⟩
⟨ First floor structure 7b ⟩
⟨ Second floor structure 10a ⟩
⟨ Third floor structure 11a ⟩
⟨ Fourth floor structure 11d ⟩
⟨ Fifth floor structure 12c ⟩
◇
```

Macro referenced in 20.

3.1.1 Ground floor

Ground floor input

⟨ Ground floor 3a ⟩ ≡

```
""" Ground floor """
OpenCourt10 = TRANS([[3,3,4,4,6,6,6.65,6.65],[4,8,8,7.8,7.8,8,8,4]])
RadioDiagnosticImaging = TRANS([[7,7,9,10,10,8.7],[4,8,8,8,4,4]])
ServiceCore10 = TRANS([[1.15, 1.15, 1.3,2.55, 2.55,2], [2.85, 3.7,3.7,3.7,
```

```
    2.85,2.85]])
ServiceCore20 = TRANS([[7,7,8.7,8.8,8.8],[2.8,3.7,3.7,3.7,2.8]])
EmergencyDepartment = TRANS([[4.7,4.7,7,7,8.8,8.8,9.65,9.65],[0,3.7,3.7,
    2.8,2.8,3.7,3.7,0]])
Endoscopy = TRANS([[3,3,3,4.4,4.4],[0,2.5,3.7,3.7,0]])
OutPatientDepartment10 = TRANS([[4./7.5, 4./7.5,1.15,1.15,2,2,3,3],
    [0,3.7,3.7,2.85,2.85,2.5,2.5,0]])
OutPatientDepartment20 = TRANS([[0,0,2.65,2.65,1.3],[4,5.85,5.85,4,4]])
RenalDialysis = TRANS([[0,0,1,2.65,2.65],[5.85,8,8,8,5.85]])
OpenCourt20 = TRANS([[2,2,2,2,4,4,4,4],[10,11,11.35,12,12,11.35,11,10]])
ChemiotherapyUnit = TRANS([[0,0,4.5,4.5,4,4,2,2,1],
    [11.35,14,14,11.35,11.35,12,12,11.35,11.35,]])
Service = TRANS([[0,0,1,1,2,2,2,1],[8.35,10,10,9,9,8.5, 8.35,8.35]])
PhysicalMedicineDept = TRANS([[2,2,1,1,0,0, 1,2,2,4,4,4.5,4.5,4,4],
    [8.5,9,9,10,10,11,11,11,10,10,11,11,9,9,8.5]])
MainEntrance = TRANS([[4,4,4,4.5,4.75,4.75,6.65,6.65,6,6],
    [8.4,8.5,9,9,9,11,11, 9,9,8.4]])
Unknown = TRANS([[7.25,7.25, 6.65,6.65,6.65,10,10,9,8.2],
    [8.35,8.5,8.5,9,11,11,8.35,8.35,8.35]])
#Mortuary = TRANS([[],[]])
Corridor0 = [[4.4,0],[4.4,3.7],[3,3.7],[3,2.5],[2,2.5],[2,2.85],[2.55,2.85],
    [2.55,3.7],[1.3,3.7],[1.3,4],[2.65,4],[2.65,5.85],[2.65,8],[1,8],[1,8.35],
    [2,8.35],[2,8.5],[4,8.5],[4,8.4],[6,8.4],[6,9],[6.65,9],[6.65,8.5],[7.25,8.5],
    [7.25,8.35],[8.2,8.35],[9,8.35],[9,8],[7,8],[7,4],[8.7,4],[8.7,3.7],
    [7,3.7],[4.7,3.7],[4.7,0]]
Corridor0a = TRANS([[1, 1, 2, 2], [11, 11.35, 11.35, 11]])
Corridor0b = TRANS([[4.5, 4.5, 4, 4, 4.5, 4.5, 4.75,4.75, 4.75],
    [9, 11, 11, 11.35, 11.35, 14,14, 11, 9]])
◇
```

Macro referenced in 2b.

Ground floor's building units

⟨ Ground floor's building units 3b ⟩ ≡

```
""" Ground floor's building units """
openCourt10 = buildingUnit(OpenCourt10,"OpenCourt10")
radioDiagnosticImaging = buildingUnit(RadioDiagnosticImaging,"RadioDiagnosticImaging")
serviceCore10 = buildingUnit(ServiceCore10,"ServiceCore10")
serviceCore20 = buildingUnit(ServiceCore20,"ServiceCore20")
emergencyDepartment = buildingUnit(EmergencyDepartment,"EmergencyDepartment")
endoscopy = buildingUnit(Endoscopy,"Endoscopy")
outPatientDepartment10 = buildingUnit(OutPatientDepartment10,"OutPatientDepartment10")
outPatientDepartment20 = buildingUnit(OutPatientDepartment20,"OutPatientDepartment20")
renalDialysis = buildingUnit(RenalDialysis,"RenalDialysis")
openCourt20 = buildingUnit(OpenCourt20,"OpenCourt20")
chemiotherapyUnit = buildingUnit(ChemiotherapyUnit,"ChemiotherapyUnit")
```

```
service = buildingUnit(Service,"Service")
physicalMedicineDept = buildingUnit(PhysicalMedicineDept,"PhysicalMedicineDept")
mainEntrance = buildingUnit(MainEntrance,"MainEntrance")
unknown = buildingUnit(Unknown,"Unknown")
#mortuary = buildingUnit(Mortuary,"Mortuary")
corridor0 = buildingUnit(Corridor0,"Corridor0")
corridor0a = buildingUnit(Corridor0a,"Corridor0a")
corridor0b = buildingUnit(Corridor0b,"Corridor0b")
◇
```

Macro referenced in 4a.

⟨ Ground floor structure 4a ⟩ ≡

```
""" Ground floor structure """

⟨ Ground floor's building units 3b ⟩

buildingUnits0 = [openCourt10,radioDiagnosticImaging,serviceCore10,serviceCore20,
    emergencyDepartment,endoscopy,outPatientDepartment10,outPatientDepartment20,
    renalDialysis,openCourt20,chemiotherapyUnit,service,physicalMedicineDept,
    mainEntrance,unknown,corridor0,corridor0a,corridor0b]

groundFloor = Struct(buildingUnits0, "groundFloor")
◇
```

Macro referenced in 2b.

3.1.2 Mezanine floor

Mezanine floor input

⟨ Mezanine floor 4b ⟩ ≡

```
""" Mezanine floor """
MedicalWaste = TRANS([[4./7.5,4./7.5,.8,1.25,1.25],[0,1.5,1.5,1.5,0]])
CentralStores = TRANS([[1.25,1.25,.8,.8,3.7,3.7,2.55,2.55,2.2,2.2],[0,1.5,1.5,
    2.65,2.65,.35,.35,.65,.65,0]])
StaffDining = TRANS([[3.95,3.95,6.7,6.7,6.95,6.95],[0,3.7,3.7,2,2,0]])
CSSD = TRANS([[6.95,6.95,6.95,8.8,8.8,9.65,9.65],[0,2,2.65,2.65,2,2,0]])
HouseKeeping = TRANS([[8.8,8.8,8.8,8.8,9.65,9.65],[2,2.65,2.8,3.7,3.7,2]])
CentralStaffChanging11 = TRANS([[4./7.5,4./7.5,1.15,1.15],[2.85,3.7,3.7,2.85]])
CentralStaffChanging21 = TRANS([[2.55,2.55,3.7,3.7],[2.85,3.7,3.7,2.85]])
OpenCourt11 = TRANS([[3,3,7,7,7],[4,8,8,6,4]])
Pharmacy = TRANS([[0,0,2.65,2.65,1.3],[4,6.45,6.45,4,4]])
CentralWorkshop = TRANS([[0,0,1,2.65,2.65],[6.45,8,8,8,6.45]])
Laundry = TRANS([[7,7,10,10,8.7],[4,6,6,4,4]])
AdministrationSuite11 = TRANS([[7,7,9,10,10],[6,8,8,8,6]])
MainLaboratories = TRANS([[1,1,0,0,2,2,5,5,4,4,4],[8.3,8.4,8.4,11,11,10,10,9,
    9,8.4,8.3]])
```

```
MedicalLibrary = TRANS([[6.7,6.7,8,8,7.75],[9.7,11,11,9.7,9.7]])
MedicalRecords = TRANS([[8,8,8,8.85,8.85,8.85],[8.3,9.7,11,11,9.75,8.3]])
AdministrationSuite21 = TRANS([[8.85,8.85,10,10,9,9],[8.3,9.75,9.75,8.4,8.4,8.3]])
MeetingRooms = TRANS([[6,6,6,6.7,6.7,7.75,7.75,7.45,7,7],[8.3,8.4,9,9,9.7,9.7,
    8.7,8.7,8.7,8.3]])
DataCenter = TRANS([[7,7,7.45,7.45],[8.3,8.7,8.7,8.3]])
ServerRoom = TRANS([[7.45,7.45,7.75,7.75],[8.3,8.7,8.7,8.3]])
PublicCore = TRANS([[4,4,5,6,6],[8.4,9,9,9,8.4]])
ServiceCore11 = TRANS([[1.15,1.15,1.3,2.55,2.55],[2.85,3.7,3.7,3.7,2.85]])
ServiceCore21 = TRANS([[7,7,8.7,8.8,8.8],[2.8,3.7,3.7,3.7,2.8]])
Corridor1 = [[2.2,0],[2.2,0.65],[2.55,0.65],[2.55,0.35],[3.7,0.35],[3.7,2.65],
    [0.8,2.65],[0.8,1.5],[0.5333,1.5],[0.5333,2.85],[1.15,2.85],[2.55,2.85],[3.7,
    2.85],[3.7,3.7],[2.55,3.7],[1.3,3.7],[1.3,4],[2.65,4],[2.65,6.45],[2.65,
    8],[1,8],[1,8.3],[4,8.3],[4,8.4],[6,8.4],[6,8.3],[7,8.3],[7.45,8.3],
    [7.75,8.3],[7.75,8.7],[7.75,9.7],[8,9.7],[8,8.3],[8.85,8.3],[9,8.3],[9,8],
    [7,8],[3,8],[3,4],[7,4],[8.7,4],[8.7,3.7],[7,3.7],[7,2.8],[8.8,2.8],
    [8.8,2.65],[6.95,2.65],[6.95,2],[6.7,2],[6.7,3.7],[3.95,3.7],[3.95,0]]
GroundRoof = TRANS([[4,4,2,2,1,1,0,0,4.75,4.75],[10,12,12,11,11,11.35,11.35,14,
    14,10]])
◇
```

Macro referenced in 2b.

Mezanine floor's building units

⟨ Mezanine floor's building units 5 ⟩ ≡

```
""" Mezanine floor's building units """
medicalWaste = buildingUnit(MedicalWaste,"MedicalWaste")
centralStores = buildingUnit(CentralStores,"CentralStores")
staffDining = buildingUnit(StaffDining,"StaffDining")
cSSD = buildingUnit(CSSD,"CSSD")
houseKeeping = buildingUnit(HouseKeeping,"HouseKeeping")
centralStaffChanging11 = buildingUnit(CentralStaffChanging11,"CentralStaffChanging1")
centralStaffChanging21 = buildingUnit(CentralStaffChanging21,"CentralStaffChanging2")
openCourt11 = buildingUnit(OpenCourt11,"OpenCourt11")
pharmacy = buildingUnit(Pharmacy,"Pharmacy")
centralWorkshop = buildingUnit(CentralWorkshop,"CentralWorkshop")
laundry = buildingUnit(Laundry,"Laundry")
administrationSuite11 = buildingUnit(AdministrationSuite11,"AdministrationSuite11")
mainLaboratories = buildingUnit(MainLaboratories,"MainLaboratories")
medicalLibrary = buildingUnit(MedicalLibrary,"MedicalLibrary")
medicalRecords = buildingUnit(MedicalRecords,"MedicalRecords")
administrationSuite21 = buildingUnit(AdministrationSuite21,"AdministrationSuite21")
meetingRooms = buildingUnit(MeetingRooms,"MeetingRooms")
dataCenter = buildingUnit(DataCenter,"DataCenter")
serverRoom = buildingUnit(ServerRoom,"ServerRoom")
publicCore = buildingUnit(PublicCore,"PublicCore")
```

```
serviceCore11 = buildingUnit(ServiceCore11,"ServiceCore11")
serviceCore21 = buildingUnit(ServiceCore21,"ServiceCore21")
corridor1 = buildingUnit(Corridor1,"Corridor1")
groundRoof = buildingUnit(GroundRoof,"GroundRoof")
◇
```

Macro referenced in 6a.

⟨ Mezanine floor structure 6a ⟩ ≡

```
""" Mezanine floor structure """

⟨ Mezanine floor's building units 5 ⟩

buildingUnits1 = [medicalWaste, centralStores, staffDining, cSSD, houseKeeping,
    centralStaffChanging11, centralStaffChanging21, pharmacy, centralWorkshop, laundry,
    administrationSuite11, mainLaboratories, medicalLibrary, medicalRecords,
    administrationSuite21, meetingRooms, dataCenter, serverRoom, publicCore,
    serviceCore11, serviceCore21, corridor1, groundRoof]

mezanineFloor = Struct(buildingUnits1, "mezanineFloor")
◇
```

Macro referenced in 2b.

3.1.3 First floor

First floor

⟨ First floor 6b ⟩ ≡

```
""" First floor """
OpenCourt3 = TRANS([[3.,3.,7.,7.],[4.,8.,8.,4.]])
Surgery = TRANS([[4.15,4.15,7.,7.,8.8,8.8,9.65,9.65],[0,3.7,3.7, 2.8,2.8, 3.7,3.7,0]])
CatheterizationLab = TRANS([[3,3,4.15,4.15],[0,3.7,3.7,0]])
ServiceCore32 = TRANS([[7.,7.,8.7,8.8,8.8],[2.8,3.7,3.7,3.7,2.8]])
CoronaryCareUnit = TRANS([[7.,7.,8.3,9.,10.,10.,8.7],[4.,8.,8.,8.,8.,4.,4.]])
DeliveryAndNicu = TRANS([[0,0, 1.7,2.65,2.65,1.3],[4.,8.,8.,8.,4.,4.]])
ServiceCore31 = TRANS([[1.15, 1.15, 1.3,2.65, 2.65], [2.85, 3.7,3.7, 3.7, 2.85]])
IntensiveCareUnit = TRANS([[4./7.5, 4./7.5,1.15,1.15,2.65, 2.65,1.95,1.95],
    [0.,3.7,3.7,2.85,2.85,.6,.6,0.]])
ServiceCore33 = TRANS([[1.95,1.95,2.65, 2.65],[0,.6,.6,0]])
PublicCore3 = TRANS([[1.7,1.7,4.,4.,6.,6.,8.3,8.3,7,3,2.65],
    [8,8.4,8.4,9,9,8.4,8.4,8,8,8,8]])
Corridor3 = TRANS([[2.65,2.65,2.65,2.65,1.3,1.3,2.65,2.65,3.0,3.0,7.0,8.7,8.7,
    7.0,4.15,3.0,3.0],[0.0,0.6,2.85,3.7,3.7,4.0,4.0,8.0,8.0,4.0,4.0,4.0,3.7,
    3.7,3.7,3.7,0.0]])
MezanineRoof = TRANS([[1,1,0,0,2,2,4.75,4.75,10,10,9,9,8.3,8.3, 6,6,4,4 ,1.7,1.7],
    [8,8.4,8.4,11,11,10,10,11,11,8.4,8.4,8,8,8.4,8.4,9,9,8.4,8.4,8]])
◇
```

Macro referenced in 2b.

First floor's building units

⟨ First floor's building units 7a ⟩ ≡

```
""" First floor's building units """
openCourt3 = buildingUnit(OpenCourt3,"OpenCourt3")
surgery = buildingUnit(Surgery,"Surgery")
catheterizationLab = buildingUnit(CatheterizationLab,"CatheterizationLab")
serviceCore32 = buildingUnit(ServiceCore32,"ServiceCore32")
coronaryCareUnit = buildingUnit(CoronaryCareUnit,"CoronaryCareUnit")
deliveryAndNicu = buildingUnit(DeliveryAndNicu,"DeliveryAndNicu")
serviceCore31 = buildingUnit(ServiceCore31,"ServiceCore31")
intensiveCareUnit = buildingUnit(IntensiveCareUnit,"IntensiveCareUnit")
serviceCore33 = buildingUnit(ServiceCore33,"ServiceCore33")
publicCore3 = buildingUnit(PublicCore3,"PublicCore3")
corridor3 = buildingUnit(Corridor3,"Corridor3")
mezanineRoof = buildingUnit(MezanineRoof,"MezanineRoof")
◇
```

Macro referenced in 7b.

⟨ First floor structure 7b ⟩ ≡

```
""" First floor structure """

⟨ First floor's building units 7a ⟩

buildingUnits2 = [surgery,catheterizationLab,serviceCore32,coronaryCareUnit,
    deliveryAndNicu,serviceCore31,intensiveCareUnit,serviceCore33,publicCore3,
    corridor3,mezanineRoof]

firstFloor = Struct(buildingUnits2, "firstFloor")
◇
```

Macro referenced in 2b.

3.1.4 Ward sections

Ward sections Here input by polylines and structure modeling are freely mixed. Just notice that the affine maps included in structures are given in grid coordinates. This fact does not permit an immediate transformation in Cartesian coordinates using the `metric` function.

⟨ Ward sections 8a ⟩ ≡

```
""" Ward sections """
Room = polyline2lar([TRANS([[0,0,1,1,2./3,2./3],[0,0.5,0.5,0.25,0.25,0]])])
RestRoom = polyline2lar([TRANS([[2./3,2./3,1,1],[0,0.25,0.25,0]])])
Nursing1 = polyline2lar([TRANS([[0,0,.2,.2],[0,.4,.4,.0]])])
Nursing2 = polyline2lar([TRANS([[.2,.2,.4,.4],[0,.4,.4,.0]])])
```

```
Nursing3 = polyline2lar([TRANS([[0,0,.4,.4,.2],[.4,.8,.8,.4,.4]])])
Nursing4 = polyline2lar([TRANS([[0,0,.4,.4],[.8,1.1,1.1,.8]])])
Nursing5 = polyline2lar([TRANS([[0,0,.4,.4],[1.1,1.4,1.4,1.1]])])

room = Struct([Room],"Room")
restRoom = Struct([RestRoom],"RestRoom")
nursing1 = Struct([Nursing1],"Nursing1")
nursing2 = Struct([Nursing2],"Nursing2")
nursing3 = Struct([Nursing3],"Nursing3")
nursing4 = Struct([Nursing4],"Nursing4")
nursing5 = Struct([Nursing5],"Nursing5")

service2 = Struct([nursing1,nursing2,nursing3,nursing4,nursing5],"Service2")
service1 = Struct([t(0,1.4),s(1,-1),service2],"Service1")
wardServices = Struct([t(1.3,.3),service1,t(0,2),service2],"WardServices")
hospitalRoom = Struct([room,restRoom],"HospitalRoom")
doubleRoom =  Struct([hospitalRoom,t(0,1),s(1,-1),hospitalRoom],"DoubleRoom")
halfWard = Struct(4*[doubleRoom,t(0,1)],"HalfWard")
ward = Struct([halfWard, wardServices, t(3,0),s(-1,1), halfWard],"Ward")
V,FV,EV = struct2lar(ward)
theWard = lar2lines((V,FV))
◇
```

Macro referenced in 8b.

3.1.5 Second floor

Second floor

⟨ Second floor 8b ⟩ ≡

```
⟨ Ward sections 8a ⟩

""" Second floor """
PublicCore4 = TRANS([[1.7,1.7,4,4,6,6,8.3,8.3, 8,7+2./3, 7, 3, 2+1./3,2],
    [8,8.4,8.4,9,9,8.4,8.4,8,8,8,8,8,8,8]])
Filter1 = TRANS([[1,1,1.35,1.35,1.15],[3.7,4,4,3.7,3.7]])
Filter2 = TRANS([[8.65,8.65,9,9,8.8],[3.7,4,4,3.7,3.7]])
ServiceCore14 = TRANS([[1.15, 1.15, 1.35,2.55, 2.55], [2.8, 3.7,3.7, 3.7, 2.8]])
ServiceCore24 = TRANS([[7,7,8.65,8.8,8.8],[2.8,3.7,3.7,3.7,2.8]])
FirstRoof = TRANS([[4./7.5, 4./7.5,1.15,1.15,2.55,2.55,7,7,8.8,8.8,9.65,9.65],
    [0,3.7,3.7,2.8,2.8,3.7,3.7,2.8,2.8,3.7,3.7,0]])
Corridor4a = [[1.35,3.7],[1.35,4],[2,4],[2.3333,4],[3,4],[7,4],[7.6667,4],[8,4],
    [8.65,4],[8.65,3.7],[7,3.7],[2.55,3.7]]
Corridor4b = [[1,4.0],[1,4.25],[1,4.5],[1,4.75],[1,5.0],[1,5.25],[1,5.5],
    [1,5.75],[1,6.0],[1,6.25],[1,6.5],[1,6.75],[1,7.0],[1,7.25],[1,7.5],
    [1,7.75],[1,8.0],[2,8.0],[2,7.75],[2,7.5],[2,7.25],[2,7.0],[2,6.75],
```

```
    [2,6.5],[2,6.25],[2,6.0],[2,5.75],[2,5.5],[2,5.25],[2,5.0],[2,4.75],
    [2,4.5],[2,4.25],[2,4.0],[1.35,4.0]]
Corridor4b1 = [[1.3,4.3],[1.3,4.6],[1.3,4.9],[1.3,5.3],[1.3,5.7],[1.5,5.7],[1.7,5.7],
    [1.7,5.3],[1.7,4.9],[1.7,4.6],[1.7,4.3]]
Corridor4b2 = [[1.3,6.3],[1.3,6.7],[1.3,7.1],[1.3,7.4],[1.3,7.7],[1.7,7.7],[1.7,7.4],
    [1.7,7.1],[1.7,6.7],[1.7,6.3],[1.5,6.3]]
Corridor4c = [[8,4.0],[8,4.25],[8,4.5],[8,4.75],[8,5.0],[8,5.25],[8,5.5],
    [8,5.75],[8,6.0],[8,6.25],[8,6.5],[8,6.75],[8,7.0],[8,7.25],[8,7.5],
    [8,7.75],[8,8.0],[8.3,8.0],[9,8.0],[9,7.75],[9,7.5],[9,7.25],[9,7.0],
    [9,6.75],[9,6.5],[9,6.25],[9,6.0],[9,5.75],[9,5.5],[9,5.25],[9,5.0],
    [9,4.75],[9,4.5],[9,4.25],[9,4.0],[8.65,4.0]]
Corridor4c1 = [[8.3,4.3],[8.3,4.6],[8.3,4.9],[8.3,5.3],[8.3,5.7],[8.5,5.7],[8.7,5.7],
    [8.7,5.3],[8.7,4.9],[8.7,4.6],[8.7,4.3]]
Corridor4c2 = [[8.3,6.3],[8.3,6.7],[8.3,7.1],[8.3,7.4],[8.3,7.7],[8.7,7.7],[8.7,7.4],
    [8.7,7.1],[8.7,6.7],[8.7,6.3],[8.5,6.3]]
◇
```

Macro referenced in 2b.

Second floor's building units

⟨ Second floor's building units 9 ⟩ ≡

```
""" Second floor's building units """
publicCore4 = buildingUnit(PublicCore4,'PublicCore4')
ward21 = deepcopy(ward)
ward22 = deepcopy(ward)
obstetricGinecologicWard = Struct([t(0,4),ward21],'ObstetricGinecologicWard')
surgicalWard1 = Struct([t(7,4),ward22],'SurgicalWard1')
filter1 = buildingUnit(Filter1,'Filter1')
filter2 = buildingUnit(Filter2,'Filter2')
serviceCore14 = buildingUnit(ServiceCore14,'ServiceCore14')
serviceCore24 = buildingUnit(ServiceCore24,'ServiceCore24')
firstRoof = buildingUnit(FirstRoof,'FirstRoof')
serviceCore11 = buildingUnit(ServiceCore11,'ServiceCore11')
serviceCore21 = buildingUnit(ServiceCore21,'ServiceCore21')
corridor4a = buildingUnit(Corridor4a,'Corridor4a')
corridor4b = buildingUnit(Corridor4b,'Corridor4b')
corridor4b1 = buildingUnit(Corridor4b1,'Corridor4b1')
corridor4b2 = buildingUnit(Corridor4b2,'Corridor4b2')
corridor4c = buildingUnit(Corridor4c,'Corridor4c')
corridor4c1 = buildingUnit(Corridor4c1,'Corridor4c1')
corridor4c2 = buildingUnit(Corridor4c2,'Corridor4c2')
◇
```

Macro referenced in 10a.

⟨ Second floor structure 10a ⟩ ≡

```
""" Second floor structure """

⟨ Second floor's building units 9 ⟩

buildingUnits3 = [publicCore4,obstetricGinecologicWard,surgicalWard1,filter1,filter2,
serviceCore14,serviceCore24,firstRoof,corridor4a,
corridor4b,corridor4b1,corridor4b2,corridor4c,corridor4c1,corridor4c2]

secondFloor = Struct(buildingUnits3, "secondFloor")
◇
```

Macro referenced in 2b.

3.1.6 Third floor

Third floor

⟨ Third floor 10b ⟩ ≡

```
""" Third floor floor
GeneralWard1 = AA(metric)(AA(larTranslate([0,4]))(theWard))
SurgicalWard2 = AA(metric)(AA(larTranslate([7,4]))(theWard)) """
◇
```

Macro referenced in 2b.

Third floor's building units

⟨ Third floor's building units 10c ⟩ ≡

```
""" Third floor's building units """
ward31 = deepcopy(ward)
ward32 = deepcopy(ward)
generalWard1 = Struct([t(0,4),ward31],'GeneralWard1')
surgicalWard2 = Struct([t(7,4),ward32],'SurgicalWard2')
◇
```

Macro referenced in 11a.

⟨ Third floor structure 11a ⟩ ≡

```
""" Third floor structure """

⟨ Third floor's building units 10c ⟩

buildingUnits4 = [generalWard1,surgicalWard2,publicCore4,serviceCore14,serviceCore24,
                  filter1,filter2,corridor4a,corridor4b,corridor4b1,corridor4b2,corridor4c,
                  corridor4c1,corridor4c2]

thirdFloor = Struct(buildingUnits4, "thirdFloor")
◇
```

Macro referenced in 2b.

3.1.7 Fourth floor

Fourth floor

⟨ Fourth floor 11b ⟩ ≡

```
""" Fourth floor floor
PediatricWard1 = AA(metric)(AA(larTranslate([0,4]))(theWard))
PediatricWard2 = AA(metric)(AA(larTranslate([7,4]))(theWard)) """
```

◇

Macro referenced in 2b.

Fourth floor's building units

⟨ Fourth floor's building units 11c ⟩ ≡

```
""" Fourth floor's building units """
ward41 = deepcopy(ward)
ward42 = deepcopy(ward)
pediatricWard1 = Struct([t(0,4),ward41],'PediatricWard1')
pediatricWard2 = Struct([t(7,4),ward42],'PediatricWard2')
```

◇

Macro referenced in 11d.

⟨ Fourth floor structure 11d ⟩ ≡

```
""" Fourth floor structure """

⟨ Fourth floor's building units 11c ⟩

buildingUnits5 = [pediatricWard1,pediatricWard2,publicCore4,serviceCore14,serviceCore24,
                  filter1,filter2,corridor4a,corridor4b,corridor4b1,corridor4b2,corridor4c,
                  corridor4c1,corridor4c2]

fourthFloor = Struct(buildingUnits5, "fourthFloor")
```

◇

Macro referenced in 2b.

3.1.8 Fifth floor

Fifth floor

⟨ Fifth floor 12a ⟩ ≡

```
""" Fifth floor floor
GeneralWard2 = AA(metric)(AA(larTranslate([0,4]))(theWard))
GeneralWard3 = AA(metric)(AA(larTranslate([7,4]))(theWard)) """
```

◇

Macro referenced in 2b.

Fifth floor's building units

⟨ Fifth floor's building units 12b ⟩ ≡

```
""" Fifth floor's building units """
ward51 = deepcopy(ward)
ward52 = deepcopy(ward)
generalWard2 = Struct([t(0,4),ward51],'GeneralWard2')
generalWard3 = Struct([t(7,4),ward52],'GeneralWard3')
```

◇

Macro referenced in 12c.

⟨ Fifth floor structure 12c ⟩ ≡

```
""" Fifth floor structure """

⟨ Fifth floor's building units 12b ⟩

buildingUnits6 = [generalWard2,generalWard3,publicCore4,serviceCore14,serviceCore24,
                  filter1,filter2,corridor4a,corridor4b,corridor4b1,corridor4b2,corridor4c,
                  corridor4c1,corridor4c2]

fifthFloor = Struct(buildingUnits6, "fifthFloor")
```

◇

Macro referenced in 2b.

3.2 Preliminary 2.5D mock-up

3.2.1 Building structure

Column locations on grid

⟨ Column locations on grid 12d ⟩ ≡

```
""" Column locations on grid """
SecondPillars = [((4,5),(1,10)),((3,4),(1,4)),((3,4),(7,10)),((4,8),(0,4)),((4,8),(7,11)),((8,9
FirstPillars = [((0,5),(1,10)),((4,8),(0,4)),((4,8),(7,11)),((8,9),(0,11)),((9,10),(4,7))]
FrontPillars = [((8,10),(0,11)),((10,11),(0,6)),((10,11),(7,11)),((11,12),(0,3)),((11,12),(4,11
MezaninePillars = FirstPillars + FrontPillars
BottomPillars = [((12,15),(0,5))]
```

◇

Macro referenced in 20.

Generation of beams and structural chains

⟨ Generation of beams and structural chains 13a ⟩ ≡

```
""" Generation of beams and structural chains """
def ManhattanTest(nodes,nDict,i,j):
    hi,ki = nodes[i]
    hj,kj = nodes[j]
    return nDict.setdefault((hi,kj),-1)!=-1 and nDict.setdefault((hj,ki),-1)!=-1

def structureGrid(loci):
    nodes = AA(tuple)(CAT([CART([range(*I), range(*J)]) for (I,J) in loci]))
    nDict = dict([(node,k) for k,node in enumerate(nodes)])
    arcs = CAT([[(nDict[(i,j)], nDict.setdefault((i,j+1),-1)),
             (nDict[(i,j)], nDict.setdefault((i,j-1),-1)),
             (nDict[(i,j)], nDict.setdefault((i+1,j),-1)),
             (nDict[(i,j)], nDict.setdefault((i-1,j),-1))] for (i,j) in nodes])
    arcs1 = list(set(AA(tuple)([sorted(arc) for arc in arcs if arc[1]!=-1])))
    arcs = CAT([[(nDict[(i,j)], nDict.setdefault((i+1,j+1),-1)),
             (nDict[(i,j)], nDict.setdefault((i+1,j-1),-1)),
             (nDict[(i,j)], nDict.setdefault((i-1,j-1),-1)),
             (nDict[(i,j)], nDict.setdefault((i-1,j+1),-1))] for (i,j) in nodes])
    arcs2 = list(set(AA(tuple)([sorted(arc) for arc in arcs if arc[1]!=-1])))
    arcs2 = [[i,j] for i,j in arcs2 if ManhattanTest(nodes,nDict,i,j)]
    nodes = metric([[j,i] for i,j in nodes])
    return nodes, arcs1,arcs2
◇
```

Macro referenced in 20.

Instancing of 3D structure frame

⟨ Instancing of 3D structure frame 13b ⟩ ≡

```
""" Instancing of 3D structure frame """
nodes0, arcs10,arcs20 = structureGrid(MezaninePillars+BottomPillars)
nodes1, arcs11,arcs21 = structureGrid(MezaninePillars)
nodes2, arcs12,arcs22 = structureGrid(FirstPillars)
nodes3, arcs13,arcs23 = structureGrid(SecondPillars)
nodes4, arcs14,arcs24 = structureGrid(SecondPillars)
nodes5, arcs15,arcs25 = structureGrid(SecondPillars)
nodes6, arcs16,arcs26 = structureGrid(SecondPillars)
VIEW(STRUCT(MKPOLS((nodes0, arcs10+arcs20)) ))
◇
```

Macro referenced in 20.

Assembling 3D structure frame

⟨ Assembling 3D structure frame 14a ⟩ ≡

```
""" Assembling 3D structure frame """
Nodes0 = AA(lambda v: list(v)+[4-.3])(nodes0)
```

```
Nodes1 = AA(lambda v: list(v)+[8-.3])(nodes1)
Nodes2 = AA(lambda v: list(v)+[12-.3])(nodes2)
Nodes3 = AA(lambda v: list(v)+[16-.3])(nodes3)
Nodes4 = AA(lambda v: list(v)+[20-.3])(nodes4)
Nodes5 = AA(lambda v: list(v)+[24-.3])(nodes5)
Nodes6 = AA(lambda v: list(v)+[28-.3])(nodes6)

Frame0 = STRUCT(MKPOLS((Nodes0, arcs10))+MKPOLS((Nodes1, arcs11))+
    MKPOLS((Nodes2, arcs12))+MKPOLS((Nodes3, arcs13))+
    MKPOLS((Nodes4, arcs14))+MKPOLS((Nodes5, arcs15))+
    MKPOLS((Nodes6, arcs16)) + \
    CONS(AA(T([1,2,3]))(Nodes0+Nodes1+Nodes2+Nodes3+Nodes4+Nodes5+Nodes6))(
    POLYLINE([[0,0,0],[0,0,-4]])  ))

Frame1 = STRUCT(MKPOLS((Nodes0, arcs20))+MKPOLS((Nodes1, arcs21))+
    MKPOLS((Nodes2, arcs22))+MKPOLS((Nodes3, arcs23))+
    MKPOLS((Nodes4, arcs24))+MKPOLS((Nodes5, arcs25))+
    MKPOLS((Nodes6, arcs26)) )
SteelFrame = OFFSET([.2,.2,.3])(STRUCT([Frame0,Frame1]))
"""
ConcreteFrame = OFFSET([.4,.4,.8])(Frame0)
"""
VIEW(Frame0)
VIEW(STRUCT([Frame0,Frame1]))
◇
```

Macro referenced in 20.

3.3 Structure embedding

Structure embedding

⟨ Structure embedding 14b ⟩ ≡

```
""" Structure embedding """
def structEmbed(struct):
    new = deepcopy(struct)
    embedTraversal(new)
    return new

def embedTraversal(obj):
    for i in range(len(obj)):
        if (isinstance(obj[i],tuple) or isinstance(obj[i],list)):
            [vert.append(0.0) for vert in obj[i][0]]
        elif isinstance(obj[i],Struct):
            embedTraversal(obj[i])
        elif isinstance(obj[i],Mat):
            """
```

```
                d = obj[i].shape[0]
                print "d =",d
                mat = scipy.identity(d+1)
                print "mat =",mat
                for h in range(d-1):
                    mat[h,d] = obj[i][h,d-1]
                    for k in range(d-1):
                        mat[h,k] = obj[i][h,k]
                obj[i] = mat.view(Mat)
                """
                pass
        return None
```
◇

Macro referenced in 20.

2.5D building assembly

⟨ 2.5D building assembly 15 ⟩ ≡
```
""" 2.5D building assembly """
def embedBuildingUnitsIn3D(floors):
    for floor in floors:
        for buildingUnit in floor.body:
            buildingUnit = larEmbed(1)(buildingUnit)
    return floors

floors = Struct([groundFloor,mezanineFloor,firstFloor,
                secondFloor,thirdFloor,fourthFloor,fifthFloor],"Floors")

#floors3D = structEmbed(floors)

#building = Struct(CAT(DISTR([floors3D,t(0,0,4)])))



#storeys = STRUCT(CAT(DISTR([[ground,mezanine,first,second,third,fourth,fifth],T(3)(4)])))

#VIEW(STRUCT([storeys,SteelFrame] + CAT(AA(MKPOLS)(AA(CONS([S1,S3]))(building))) ))
```
◇

Macro referenced in 20.

⟨ test 16a ⟩ ≡
```
""" 2.5D building assembly """
%floors = Struct([groundFloor,mezanineFloor,firstFloor,
%                secondFloor,thirdFloor,fourthFloor,fifthFloor,fifthFloor],"building")
%
```

```
%floors3D = embedStruct(1)(floors)
%building = Struct(CAT(DISTR([floors3D.body,t(0,0,4)])))
%models = AA(CONS([S1,S3]))(building)
%VIEW(STRUCT(CAT(AA(MKPOLS)(models))))
◇
```

Macro never referenced.

3.3.1 Storey viewing

Storey viewing

⟨ Storey generation 16b ⟩ ≡

```
""" Storey generation """
def structDraw(color,scaling):
    def structDraw0(obj): return obj.draw(color,scaling)
    return structDraw0

ground,W,EV = floor(X,Y)(groundFloor)
ground2D = STRUCT([ground, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits0))
mezanine,W,EV = floor(X,Y)(mezanineFloor)
mezanine2D = STRUCT([mezanine, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits1))
first,W,EV = floor(X,Y)(firstFloor)
first2D = STRUCT([first, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits2))
second,W,EV = floor(X,Y)(secondFloor)
second2D = STRUCT([second, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits3))
third,W,EV = floor(X,Y)(thirdFloor)
third2D = STRUCT([third, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits4))
fourth,W,EV = floor(X,Y)(fourthFloor)
fourth2D = STRUCT([fourth, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits5))
fifth,W,EV = floor(X,Y)(fifthFloor)
fifth2D = STRUCT([fifth, COLOR(RED)(STRUCT(MKPOLS((W,EV))))] + \
            AA(structDraw(RED,10))(buildingUnits6))
◇
```

Macro referenced in 20.

⟨ Storey viewing 17a ⟩ ≡

```
""" Storey viewing """
VIEW(ground2D)
VIEW(mezanine2D)
VIEW(first2D)
```

```
VIEW(second2D)
VIEW(third2D)
VIEW(fourth2D)
VIEW(fifth2D)
◇
```

Macro referenced in 20.

aaaa

⟨ aaaa 17b ⟩ ≡

```
""" aaaa """

◇
```

Macro defined by 17bc, 18a.
Macro never referenced.

3.4 Structural frame

Complex of columns, and beams, girders, spandrels, and trusses connected to one another and to the columns anchored in a foundation, as well as other components or members necessary for the stability of a structure. Floors and roof panels, not connected to the columns (and called secondary members) are not considered part of the structural frame.

aaaa

⟨ aaaa 17c ⟩ ≡

```
""" aaaa """

◇
```

Macro defined by 17bc, 18a.
Macro never referenced.

3.5 Vertical communications

aaaa

⟨ aaaa 18a ⟩ ≡

```
""" aaaa """

◇
```

Macro defined by 17bc, 18a.
Macro never referenced.

4 Design review

4.1 Integration and cochains computation

Surface integration

⟨ Surface integration 18b ⟩ ≡

```
from integr import *
""" Surface integration """
def surfIntegration(model):
    V,FV,EV = model
    V = [v+[0.0] for v in V]
    cochain = []
    for face in FV:
        triangles = AA(C(AL)(face[0]))(TRANS([face[1:-1],face[2:]]))
        P = V,triangles
        area = Surface(P,signed=True)
        cochain += [abs(area)]
    return cochain
```

◇

Macro referenced in 20.

Surface cochain computation

⟨ Surface cochain computation 18c ⟩ ≡

```
""" Surface cochain computation """
def surfaceCochain(buildingUnit):
    print "\nbuildingUnit.name =",buildingUnit.name
    cochain = AA(int)(surfIntegration(struct2lar(buildingUnit)))
    names = [struct.name for struct in buildingUnit.body]
    return zip(names,cochain)

for floor in floors:
    areas = surfaceCochain(floor)
    print floor.name + " =", areas
    print floor.name + " m^2 =", sum(TRANS(areas)[1])
    print ""
```

◇

Macro referenced in 20.

Computing of a surface cochain via Struct instance traversal

⟨ Computing of a surface cochain via Struct instance traversal 19 ⟩ ≡

```
""" Computing of a surface cochain via Struct instance traversal """
def structCochain(struct,levels=1):
    cochain = defaultdict(int)
    dim = checkStruct(struct.body)
    CTM, stack = scipy.identity(dim+1), []
    nameStack,cochainMap = structCochainTraversal(CTM, stack, struct, [], [], [])
    for cell,cochainValue in cochainMap:
        nameArray = cell.split(".")
        cochain[".".join(nameArray[:levels])] += cochainValue[0]
    return cochain

def structCochainTraversal(CTM, stack, obj, scene=[], names=[], nameStack=[]):
    repeatedNames = defaultdict(int)

    def map(model):
        V,FV,EV = larApply(CTM)(model)
        return AA(int)(surfIntegration((metric(V),FV,EV)))

    for i in range(len(obj)):
        if isinstance(obj[i],Struct):
            repeatedNames[obj[i].name] += 1
            if repeatedNames[obj[i].name]==1: theName = obj[i].name
            else: theName = obj[i].name + str(repeatedNames[obj[i].name]-1)
            names.append(theName)
            nameStack = nameStack+[names]
            stack.append(CTM)

            structCochainTraversal(CTM, stack, obj[i], scene, names, nameStack)

            CTM = stack.pop()
            theName = names.pop()
        elif isinstance(obj[i],Model):
            scene += [( ".".join(names), map(obj[i]) )]
        elif (isinstance(obj[i],tuple) or isinstance(obj[i],list)) and (
              len(obj[i])==2 or len(obj[i])==3):
            scene += [( ".".join(names), map(obj[i]) )]
        elif isinstance(obj[i],Mat):
            CTM = scipy.dot(CTM, obj[i])
    return nameStack,scene

if __name__ == "__main__":
    print "\nsurface cochain(ward) =", structCochain(ward,2)
    print "\nsurface cochain(doubleRoom) =", structCochain(doubleRoom,4)
◇
```

Macro referenced in 20.

5 System semantics

5.1 Topological requirements

5.2 Geometrical requirements

6 Code exporting

The `Hospital.py` module

"lib/py/hospital.py" 20 ≡

```
""" The 'Hospital' module """

from pyplasm import *

""" import modules from larcc/lib """
sys.path.insert(0, 'lib/py/')
from architectural import *
from iot3d import *
DEBUG = True
```

⟨ Reference grid 1 ⟩
⟨ Coding utilities 21a ⟩
⟨ From array indices to grid coordinates 2a ⟩
⟨ Storey input 2b ⟩
⟨ Storey generation 16b ⟩
⟨ Storey viewing 17a ⟩
⟨ Column locations on grid 12d ⟩
⟨ Generation of beams and structural chains 13a ⟩
⟨ Instancing of 3D structure frame 13b ⟩
⟨ Assembling 3D structure frame 14a ⟩
⟨ Structure embedding 14b ⟩
⟨ 2.5D building assembly 15 ⟩
⟨ Surface integration 18b ⟩
⟨ Surface cochain computation 18c ⟩
⟨ Computing of a surface cochain via Struct instance traversal 19 ⟩
◇

A Code utilities

Coding utilities

⟨ Coding utilities 21a ⟩ ≡

```
""" Coding utilities """
```

⟨ From grid to metric coordinates 21b ⟩
⟨ Mapping a grid frame to a Cartesian one 21c ⟩

⟨ Solidify the boundary of polyline-like building units 22a ⟩
⟨ Make a struct object from a 2D polyline 22b, ... ⟩
◇

Macro referenced in 20.

From grid to metric coordinates

⟨ From grid to metric coordinates 21b ⟩ ≡

```
""" From grid to metric coordinates """
def grid2coords(X,Y):
    xMeasures = list(cumsum(X))
    yMeasures = list(cumsum(Y))
    def grid2coords0(point):
        x,y = point[0:2]
        xint,yint = int(x), int(y)
        xdec,ydec = float(x-xint), float(y-yint)
        xcoord = xMeasures[xint] + xdec*X[xint+1]
        ycoord = yMeasures[yint] + ydec*Y[yint+1]
        if len(point)==2: return [xcoord, ycoord]
        else: return [xcoord, ycoord, point[2]]
    return grid2coords0

def coordMaps(ymax):
    def coordMaps0(polyline):
        polyline = AA(grid2coords(X,Y))(polyline)
        polyline = vmap(ymax)(polyline)
        return [eval(vcode(point)) for point in polyline]
    return coordMaps0

metric = coordMaps(ymax)
```

◇

Macro referenced in 21a.

Mapping the grid frame to a Cartesian right-hand frame

⟨ Mapping a grid frame to a Cartesian one 21c ⟩ ≡

```
""" Mapping the grid frame to a Cartesian right-hand frame """
def vmap(ymax):
    def vmap0(V):
        if len(V[0])==3: W = [[x,ymax-y,z] for x,y,z in V]
        else: W = [[x,ymax-y] for x,y in V]
        return W
    return vmap0

def embed(z):
```

```
        def embed0(p):
            return p+[z]
        return embed0
```
◇

Macro referenced in 21a.

Solidify the boundary of polyline-like building units

⟨ Solidify the boundary of polyline-like building units 22a ⟩ ≡

```
    """ Solidify the boundary of polyline-like building units """
    def floor(X,Y):
        def floor0(structure2D):
            V,FV,EV = struct2lar(structure2D)
            BE = [EV[e] for e in boundaryCells(FV,EV)]
            theFloor = SOLIDIFY(STRUCT([POLYLINE([V[v],V[w]]) for v,w in BE]))
            return theFloor,V,EV
        return floor0
```
◇

Macro referenced in 21a.

Make a struct object from a 2D polyline

⟨ Make a struct object from a 2D polyline 22b ⟩ ≡

```
    """ Make a struct object from a 2D polyline """
    isPolyline = ISSEQOF(ISSEQOF(ISNUM))
    isPolylineSet = ISSEQOF(ISSEQOF(ISSEQOF(ISNUM)))

    def buildingUnit(polyline,string):
        if ISSEQOF(ISSEQOF(ISNUM))(polyline): model = polyline2lar([polyline])
        else: model = polyline2lar(polyline)
        return Struct([model],str(string))
```
◇

Macro defined by 22b, 23a.
Macro referenced in 21a.

Extract 1-cells from the lar of a polylineSet

⟨ Make a struct object from a 2D polyline 23a ⟩ ≡

```
    """ Make a struct object from a 2D polyline """
    def lineSet(polylineSet):
        EV = []
        for polyline in polylineSet:
            EV += [(v,w) if v<w else (w,v) for v,w in zip(polyline,polyline[1:]+[polyline[0]])]
        return AA(list)(EV)
```
◇

Macro defined by 22b, 23a.
Macro referenced in 21a.

The 2.5D mock-up

"test/py/hospital/mock-up.py" 23b ≡

```
""" The 2.5D mock-up of an hospital building """
```

◇

References

[AM13] Adham R. Ismail Abdel-Moneim, *Hospital planning and medical equipment design*, Future Healthcare – The opportunities of new technology (Oslo, Norway), 38th World Hospital Congress, 18–20 June 2013.

[CL13] CVD-Lab, *Linear algebraic representation*, Tech. Report 13-00, Roma Tre University, October 2013.