Computational topology: Lecture 11

Alberto Paoluzzi

April 5, 2019

Lab work

Bug fixed

just run:

```
$ cd ~/Documents/dev/LinearAlgebraicRepresentation.jl/
$ julia11 examples/lario2obj.jl
```

Sparse matrix internals 4/4

```
function vcycle( copEV::Lar.ChainOp, copFE::Lar.ChainOp, f::Int64 )
    edges,signs = findnz(copFE[f,:])
    vpairs = [s>0 ? findnz(copEV[e,:])[1] : reverse(findnz(copEV[e,:])[1])
                for (e,s) in zip(edges,signs)]
    vdict = Dict((v1,v2) for (v1,v2) in  vpairs)

    v0 = collect(vdict)[1][1]
    chain_0 = Int64[v0]
    v = vdict[v0]
    while v !== v0
        push!(chain_0,v)
        v = vdict[v]
    end
    return chain_0
end
```

Look at sources . . .

https://github.com/cvdlab/LinearAlgebraicRepresentation.jl/blob/julia-1.0/src/utilities.jl

Refactoring job

https://github.com/cvdlab/LinearAlgebraicRepresentation.jl/blob/julia-1.0/test/test_planar_arrangement.jl

aaaaaaaaaa

Exporting to file

```
open("testfile.obj","w") do f
    write(f, Lar.lar2obj(V::Lar.Points, cc) )
end
```

For reading/writing text files in Julia, see:

Introducing Julia/Working with text files
https://juliabyexample.helpmanual.io

Reading the file

Reading the text file as a single string

```
s = open("testfile.obj") do file
    read(file, String)
end;

println(s)
```

Visualization from python

```
> from pyplasm import *
> batches=[]
> filename = "testfile.obj"
> batches+=Batch.openObj(filename)
> octree=Octree(batches)
> glcanvas=GLCanvas()
> glcanvas.setOctree(octree)
> glcanvas.runLoop()

Building octree from 1 batches....
Scene number of nodes of the octree 1
Scene max depth                     0
Scene number of batches             1
...done in 0 msec
```

Readings . . .

Not only for technical computing: changing the narrative around the usecase for Julia