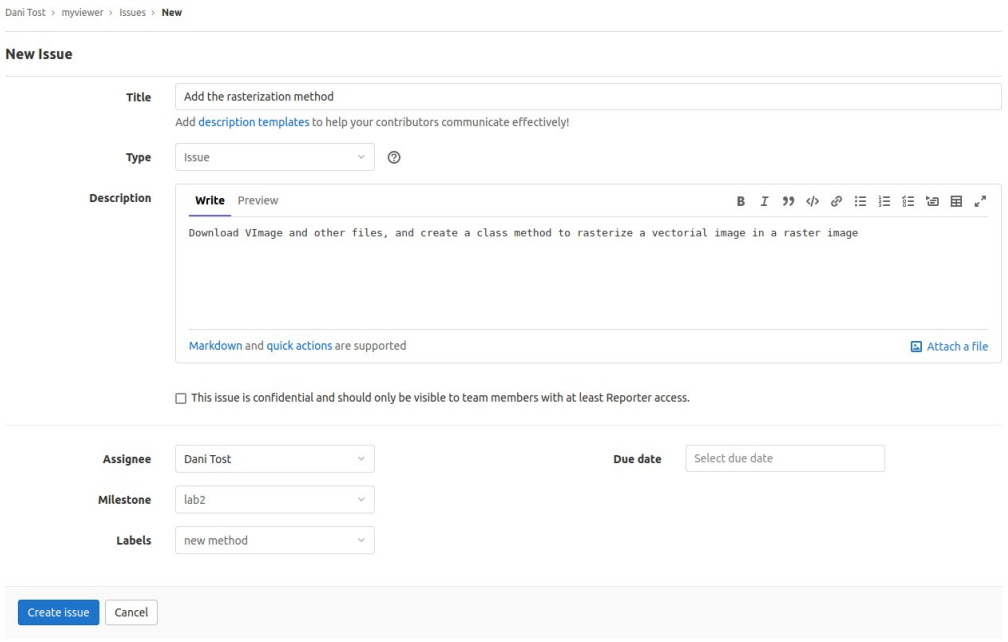
Session 6 lab
Rasterization

Dani Tost

Objective

- To add a new class method to Rimage to be able to rasterize a vectorial image
- Partial objectives
  - Add given classes to your git project
  - Test the rasterization of a pointç
  - Test the class Vimage
  - Implement the rasterization

Create an issue



And then create a merge request

In the new branch, download the new code

$ git fetch
$ git checkout newbranch

You'll find the new files in http://gie.cs.upc.edu/mi/Lab/classes/index.html

Download: point.py, circle.py, rectangle.py and polygon.py

$ git add point.py (idem for the others)
$ git commit -m « new files»
$ git push

Understand the class Point

Check the specification. What are the attributes? Try to create one from a dictionary. Apply the identity transformation matrix. Do you agree? Try to define and apply a rotation of 30º matrix.

```
>>> from point import Point
>>> p = Point(2, 3)
>>> str(p)
'Point (2, 3)'
>>> p = Point.from_dict({'x':7, 'y':9})
>>> p = Point.from_dict({'x':7, 'y':9})
>>> q = p.rasterization([[1, 0, 0], [0, 1, 0], [0, 0, 1]])
>>> p == q
True
>>> from math import cos, sin, pi
>>> a = pi/6
>>> m = [[cos(a), -sin(a), 0], [sin(a), cos(a), 0], [0, 0, 1]]
>>> str(p.TG(m))
'Point (6.123233995736766e-17, 1.0)'
```

Understand the class VImage

A customized class container of 2D figures (polygons, circles and rectangles) of different colors. You can add new primitives if you wish. For instance, how would it be to add a regular polygon of n vertices, a given center and radius?

Try to understand how we read a vectorial scene. Open the test file vim1.json. Try to modify it and test it.

Homework

In the class rimage, create a new class method called rasterization

class Rimage

class methods

rasterization(vim, width, height, back_color)

Parameters:

vim: the vectorial image Vimage to be rasterized

width, height: the size of the raster image to be created

back_color: the background color of the image to be created

Returns:

a raster image with the given size and background color with the vectorial image rasterized on top, maximizing the space occupancy, centered and without deformations

Suppose that the **window is the bounding box** of the vectorial image, readjusted to have the same aspect ratio as the raster image, so there are **no deformations**. The name of the returned raster image is the same as the name of the VImage

Step by step

- first, create a raster image with the given color, width and height.
- next, compute the VImage bounding box. It will be the first approximation to the “window»
- then, compute a new rectangle as tied as possible to the window, but having the aspect ratio of the raster image
- after, compute the window-to viewport transformation matrix (already implemented in Rectangle)
- finally, traverse the different figures of the Vimage

  for each figure, invoke its rasterization methods and assign to the corresponding RImage pixels the color of the figure

You have to implement the rasterization methods of the figures using `skimage.draw`.