

Hello Tilman,

I modified the shading package, and the code are available at <https://bitbucket.org/xinshu/pdfbox.git>, midway done, may have bugs.

However, It's better to summarize the progress made in this week and talk to you to get some feedback. Any suggestion is appreciated, take you time.

I encountered some questions which I put in this power point and highlighted by blue color. If these questions are too vague, you don't need to give an explicit answer.

Thanks!

Best,  
Shaola

# Shading type 6

Questions are highlighted by blue color

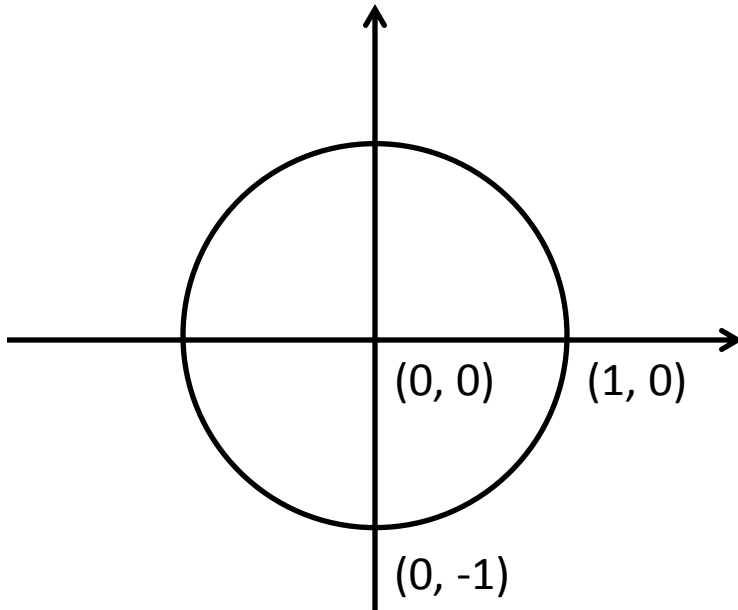
Shaola Ren

5.23.2014

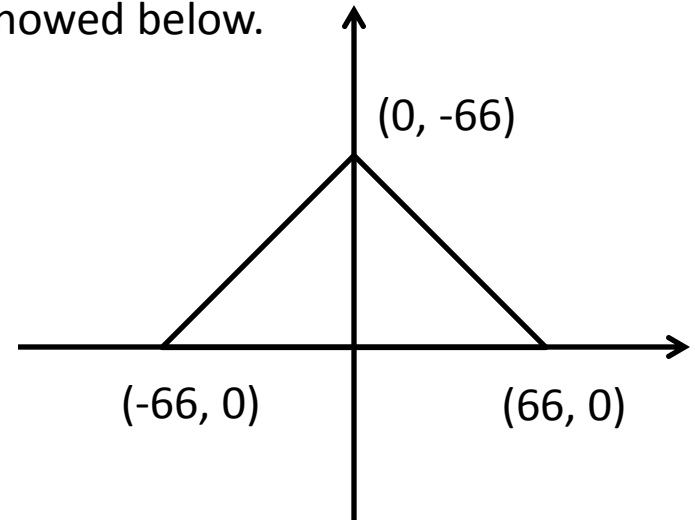
# test result for CONICAL.pdf

- 4 Coons patches  
this part should be right according to my tests
- 12 control points of each Coons patch  
this part should be right as well according to my tests
- Got device coordinates from source coordinates  
using the value I got from my code, I can obtain a unit circle, has radius 1
- Coordinates and Color map are reasonable  
Cubic Bezier curve and bilinear interpolation

# device coordinates



This is the shape I got from conical .pdf  
the radius is only 1.  
I test Gouraud.pdf, the reading is  
showed below.



The questions are,

- The unit of those calculated values should be **pixel**, right?
- Those values indicate how large the image displayed in a device screen, right?  
Or I missed some parameters?
- Is this radius 1 reasonable? Is there other parameters that I missed to indicate the size of a graph?

# device coordinates

The following two slides show the reading screen snapshot of the coordinates

# reading screen snapshot of Gourad

```
System.out.println("x: " + x + " " + maxSrcCoord + " " + rangeX.getMin() + " " + rangeX.getMax());  
System.out.println("y: " + y + " " + maxSrcCoord + " " + rangeY.getMin() + " " + rangeY.getMax());  
System.out.println("interpolate: " + dstX + " " + dstY);
```

```
x: 8354712 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: -66.2021484375 0.001953125  
x: 8388608 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: 0.001953125 0.001953125  
x: 8388608 16777215 -16384.0 16384.0  
y: 8422504 16777215 -16384.0 16384.0  
interpolate: 0.001953125 66.205078125  
x: 8422504 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: 66.205078125 0.001953125
```

# reading screen snapshot of CONICAL

```
System.out.println("x: " + x + " " + maxSrcCoord + " " + rangeX.getMin() + " " + rangeX.getMax());  
System.out.println("y: " + y + " " + maxSrcCoord + " " + rangeY.getMin() + " " + rangeY.getMax());  
System.out.println("interpolate: " + px + " " + py);
```

```
x: 8388608 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: 0.001953125 0.001953125  
x: 8388608 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: 0.001953125 0.001953125  
x: 8388948 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: 0.666015625 0.001953125  
x: 8389120 16777215 -16384.0 16384.0  
y: 8388608 16777215 -16384.0 16384.0  
interpolate: 1.001953125 0.001953125  
x: 8389120 16777215 -16384.0 16384.0  
y: 8388326 16777215 -16384.0 16384.0  
interpolate: 1.001953125 -0.5498046875  
x: 8388890 16777215 -16384.0 16384.0  
y: 8388096 16777215 -16384.0 16384.0  
interpolate: 0.552734375 -0.9990234375  
x: 8388608 16777215 -16384.0 16384.0  
y: 8388096 16777215 -16384.0 16384.0  
interpolate: 0.001953125 -0.9990234375  
x: 8388608 16777215 -16384.0 16384.0  
y: 8388268 16777215 -16384.0 16384.0  
interpolate: 0.001953125 -0.6630859375  
x: 8388608 16777215 -16384.0 16384.0  
y: 8388438 16777215 -16384.0 16384.0
```

Drag to move and enlarge

# debug

- Every time I changed the code, even a single line, I need to build the whole project in order to test the result, is it possible to generate the changed .class file separately and then update this new .class file to the old pdfbox-app-2.0.0-SNAPSHOT.jar file?
- LOG.debug(), I don't know how to use this method, I usually use System.out.println() to print out anything I need to Command prompt window to see a middle way result, could you simply guide me how to use this LOG.debug()? If it's too complicated, just ignore it.



# time and space complexity of getRaster() method

I looked this method in “GouraudShadingContext.java”,  $t \sim O(w * h * triangleList.size())$ , when  $triangleList.size()$  is small, it's efficient and space complexity is  $O(w * h)$  which is small.

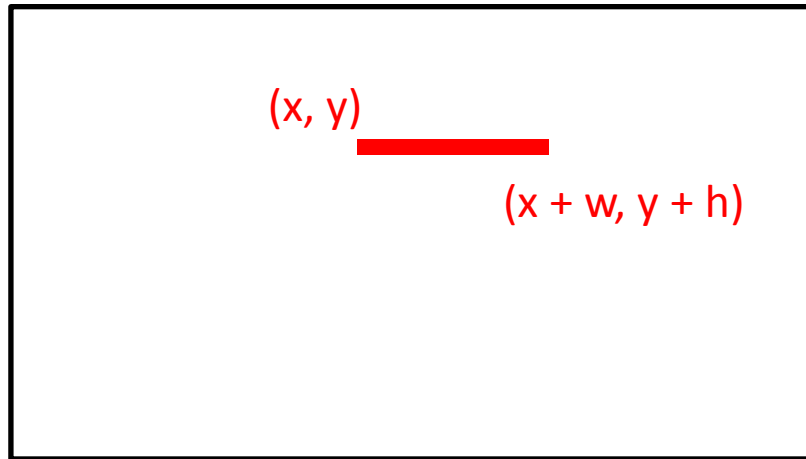
For shading type 6, I thought to generate all the points inside an image once and part of the code was wrote in this way now, then  $t \sim O(w * h)$ , but this way is very space consuming as all the image points need to be stored in a table, generally about tens Megabytes for a 1000 pixels\* 1000 pixels image. The question is, what is the acceptable memory usage for converting an image?

Maybe I need to rethink about the strategy to accomplish the getRaster() method.

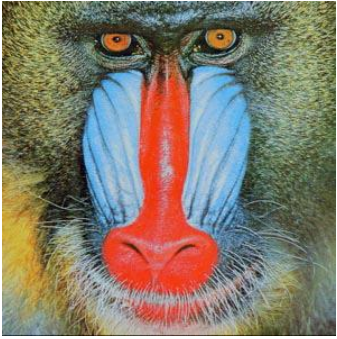
# CONICAL.pdf

- `getRaster(int x, int y, int w, int h)`

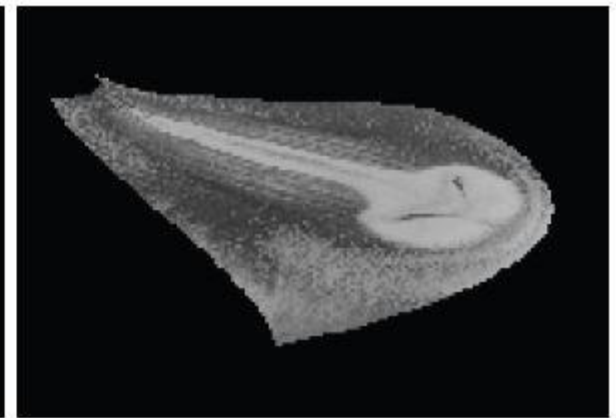
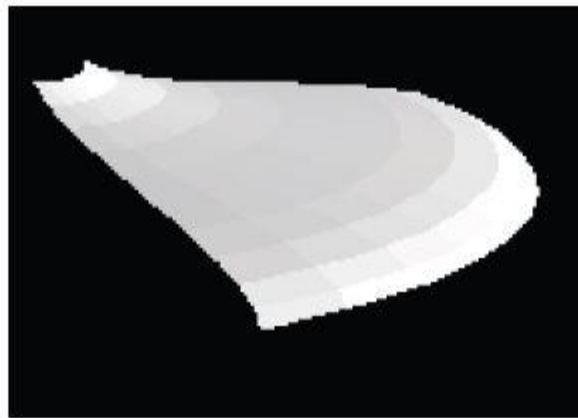
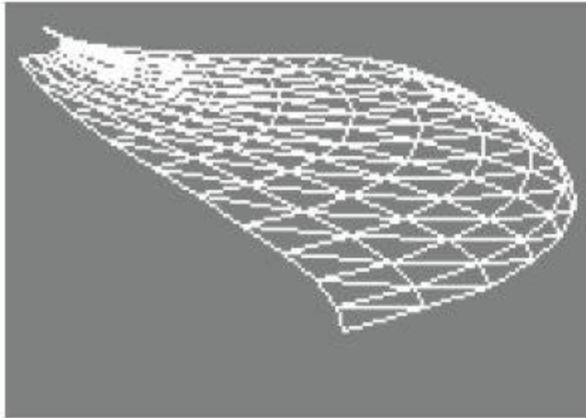
I misunderstood the parameters in this method before, I thought it were the whole rectangle (the black one) that the image would be displayed. It turned out it is just a small part (**the red one**) of the whole image. Wasted some time here, need to rewrite that part.  $(x, y)$



$(x + w, y + h)$



Using assistant parameter  $(u, v)$ , it's easy to decide the Bezier surface in this direction, as it's a multiple points to one point mapping, using the priority rule given in the pdf doc, point's color can be decided uniquely. This is the way what I thought and worked before.



Now, I think I need to accomplish the `getRaster()` in an opposite direction mapping, then to decide a point's color becomes difficult, the target point may be mapped from multiple points. Given  $t$  to calculate the value of  $f(t)$  = combination of 4 cubic Bezier  $B(t)$  is direct, however, given a value to get the corresponding  $t$  of  $f(t)$  is another story.

There should be a solution, I am working on this issue now.