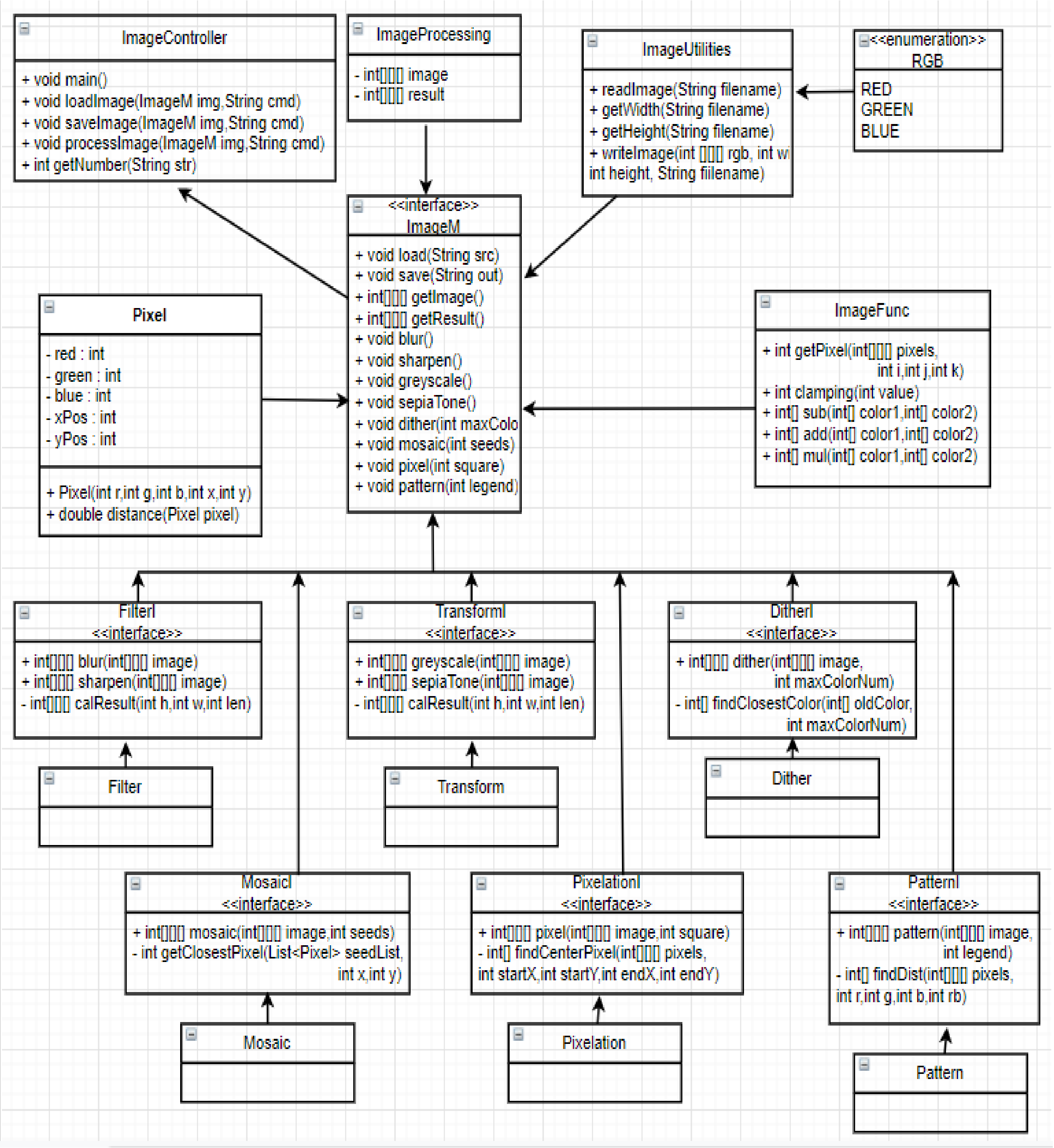
Project 4 -- Cross-Stitch Controller Preliminary Design

This project follows the MVC design. Here, ImageM represents the **Model** and ImageController represents the **Controller** of the design. The Controller will be tested by creating a mock model, which will check the input to the model from the controller and the output to the controller from the model.


ImageController
+ void main()
+ void loadImage(ImageM img,String cmd)
+ void saveImage(ImageM img,String cmd)
+ void processImage(ImageM img,String cmd)
+ int getNumber(String str)
ImageProcessing
- int[][][] image
- int[][][] result
ImageUtilities
+ readImage(String filename)
+ getWidth(String filename)
+ getHeight(String filename)
+ writeImage(int [][][] rgb, int wi
int height, String fiilename)
<<enumeration>>
RGB
RED
GREEN
BLUE
<<interface>>
ImageM
+ void load(String src)
+ void save(String out)
+ int[][][] getImage()
+ int[][][] getResult()
+ void blur()
+ void sharpen()
+ void greyscale()
+ void sepiaTone()
+ void dither(int maxColo
+ void mosaic(int seeds)
+ void pixel(int square)
+ void pattern(int legend)
Pixel
- red : int
- green : int
- blue : int
- xPos : int
- yPos : int
+ Pixel(int r,int g,int b,int x,int y)
+ double distance(Pixel pixel)
ImageFunc
+ int getPixel(int[][][] pixels,
int i,int j,int k)
+ int clamping(int value)
+ int[] sub(int[] color1,int[] color2)
+ int[] add(int[] color1,int[] color2)
+ int[] mul(int[] color1,int[] color2)
FilterI
<<interface>>
+ int[][][] blur(int[][][] image)
+ int[][][] sharpen(int[][][] image)
- int[][][] calResult(int h,int w,int len)
Filter
TransformI
<<interface>>
+ int[][][] greyscale(int[][][] image)
+ int[][][] sepiaTone(int[][][] image)
- int[][][] calResult(int h,int w,int len)
Transform
DitherI
<<interface>>
+ int[][][] dither(int[][][] image,
int maxColorNum)
- int[] findClosestColor(int[] oldColor,
int maxColorNum)
Dither
MosaicI
<<interface>>
+ int[][][] mosaic(int[][][] image,int seeds)
- int getClosestPixel(List<Pixel> seedList,
int x,int y)
Mosaic
PixelationI
<<interface>>
+ int[][][] pixel(int[][][] image,int square)
- int[] findCenterPixel(int[][][] pixels,
int startX,int startY,int endX,int endY)
Pixelation
PatternI
<<interface>>
+ int[][][] pattern(int[][][] image,
int legend)
- int[] findDist(int[][][] pixels,
int r,int g,int b,int rb)
Pattern

Testing plan

A Mock Model will be created to test the Controller.

1. Test image being loaded into the model on input.
   assertEquals(“image loaded”,log.toString()) // input reaches model correctly

2. Test image processing – blur
   assertEquals(“image blurred”,out.toString()) // output from model received correctly

3. Test image processing – sharpen
   assertEquals(“image sharpened”,out.toString())

4. Test image processing – greyscale
   assertEquals(“image greyscaled”,out.toString())

5. Test image processing – sepiaTone
   assertEquals(“image sepiatoned”,out.toString())

6. Test image processing – dither
   assertEquals(“image dithered”,out.toString())

7. Test image processing – mosaic
   assertEquals(“image mosaic”,out.toString())

8. Test image processing – pixelation
   assertEquals(“image pixeled”,out.toString())

9. Test image processing – pattern
   assertEquals(“image pattern generated”,out.toString())

10. Test image being saved after processing.
    assertEquals(“image saved”,out.toString())