Image Fusion For Situational Awareness

...

Team Lead: Kevin Shea
Team Members: Timothy Baker, Yusef Cardona, Jeremy Choyce, Eve Ciancia, John Madigan III

Rowan University

Principles of Software Engineering
Instructor: Dr. Adrian Rusu
Spring 2016

Presentation Outline

- Project Introduction
- Overview of Objectives
- Handling of Each Objective
- Software Demonstration

Project Introduction

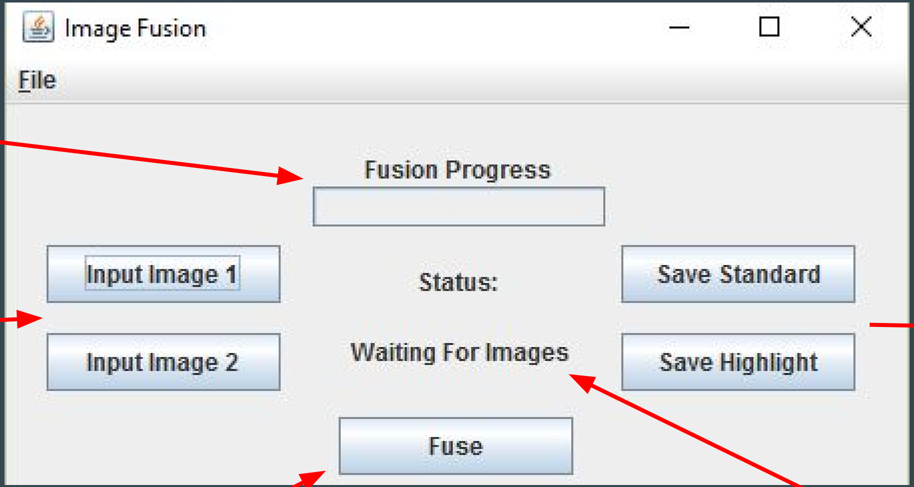
With the increasing availability and ease to captures images from satellites, UAVs, drones, and even mobile devices, image fusion is required to combine relevant information into a single image to increase situational awareness.

To achieve this we met with ASRC representatives Mr. Michael Peacock and Ms. Kimberly Davis and agreed on specific objectives for this project.

Objectives

- Simple and User Friendly GUI
- Image Saving Functionality
- Fusion of 2 Images
- Highlighting of Differences

GUI



Visual Update of Progress

Choose Two Images to Fuse

Save Standard or Highlighted Fused Image to Desired FIle Path

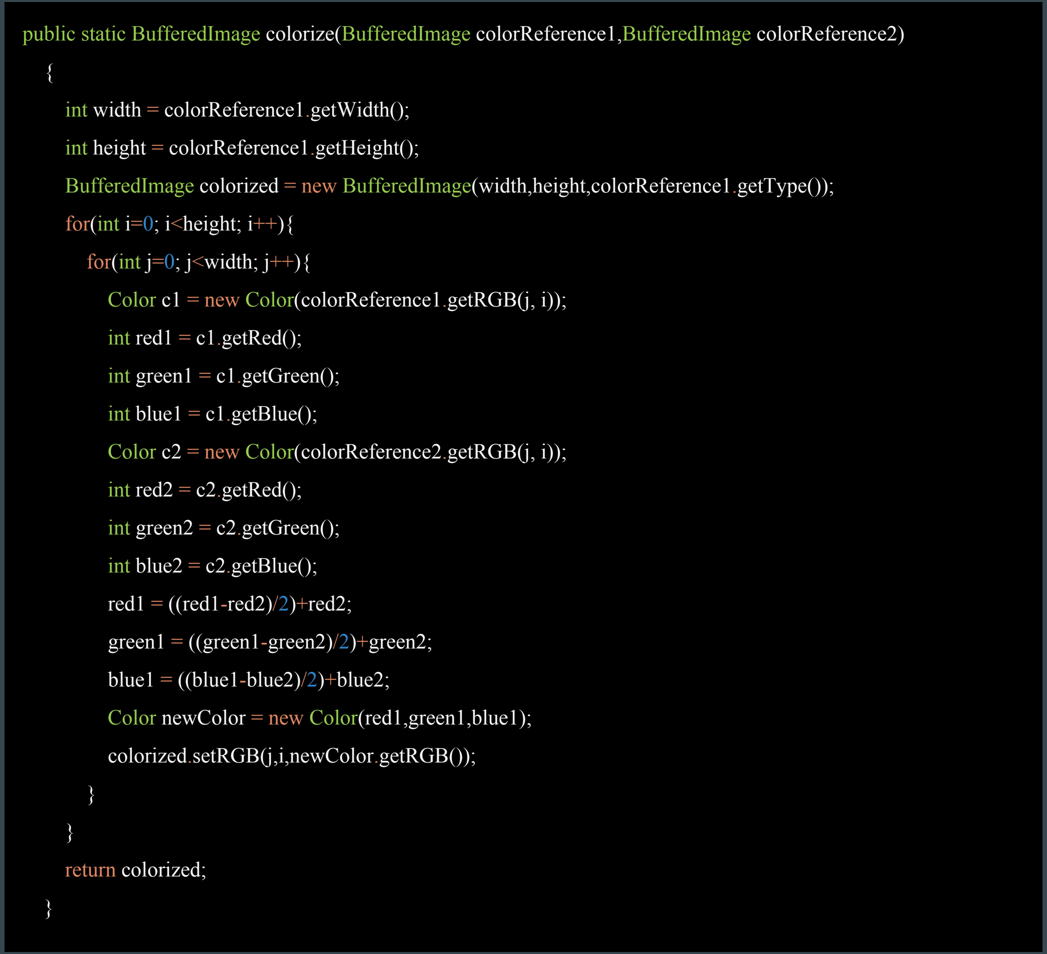
Execute Fusion Algorithm

Textual Update of Current Status

Image Fusion Algorithm

How it works

- Corresponding pixels from the same location of both images are broken down into their RGB components
- The median value between each Red, Blue, and Green of both images is calculated.
- These median values are then used to create a new color which is then set in the same pixel location on the final image



```
public static BufferedImage colorize(BufferedImage colorReference1,BufferedImage colorReference2)
  {
      int width = colorReference1.getWidth();
      int height = colorReference1.getHeight();
      BufferedImage colorized = new BufferedImage(width,height,colorReference1.getType());
      for(int i=0; i<height; i++){
        for(int j=0; j<width; j++){
          Color c1 = new Color(colorReference1.getRGB(j, i));
          int red1 = c1.getRed();
          int green1 = c1.getGreen();
          int blue1 = c1.getBlue();
          Color c2 = new Color(colorReference2.getRGB(j, i));
          int red2 = c2.getRed();
          int green2 = c2.getGreen();
          int blue2 = c2.getBlue();
          red1 = ((red1-red2)/2)+red2;
          green1 = ((green1-green2)/2)+green2;
          blue1 = ((blue1-blue2)/2)+blue2;
          Color newColor = new Color(red1,green1,blue1);
          colorized.setRGB(j,i,newColor.getRGB());
        }
      }
      return colorized;
  }
```

Image Fusion Results

Image 1

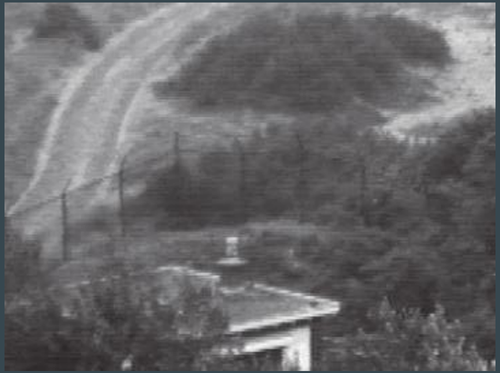
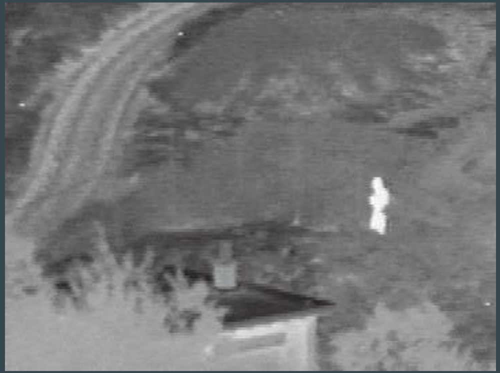
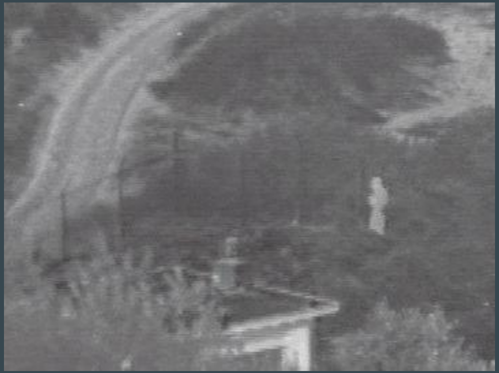


Image 2



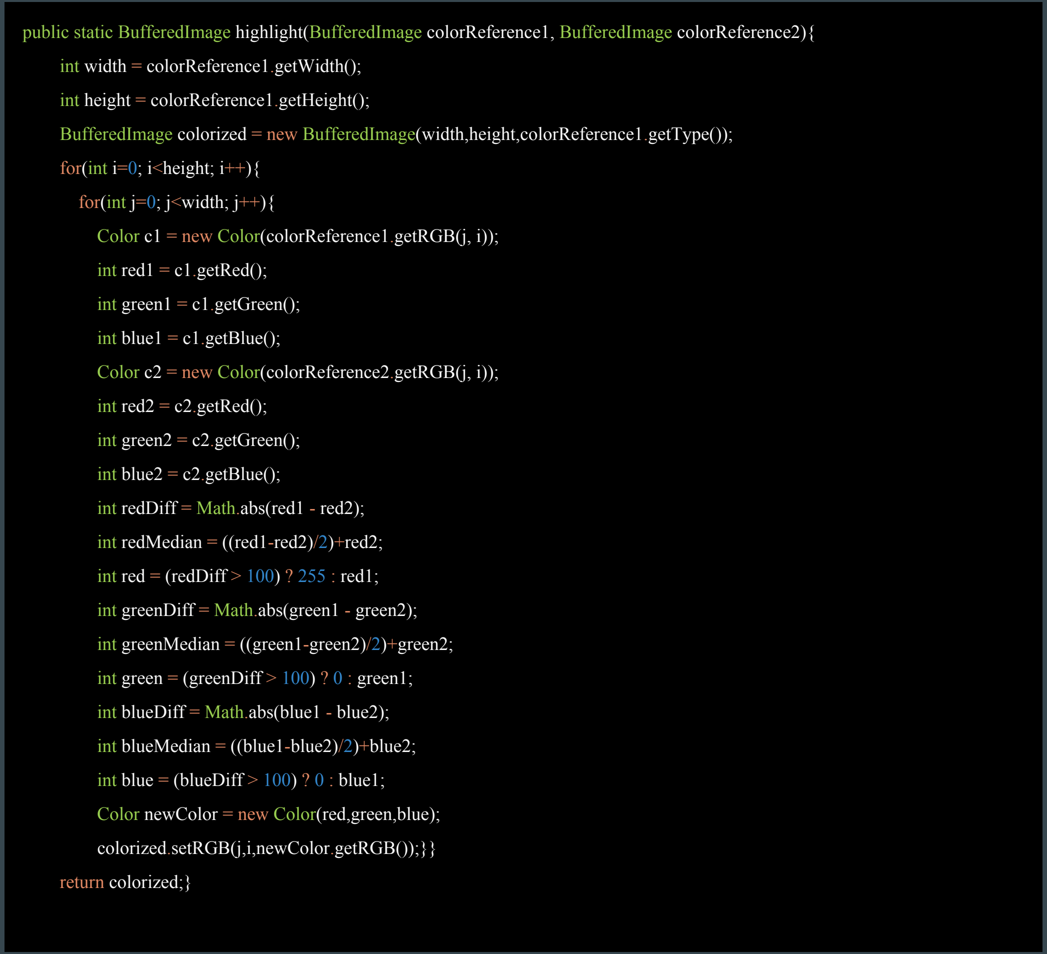
Fused Image



Highlighting Differences in Images

How it works

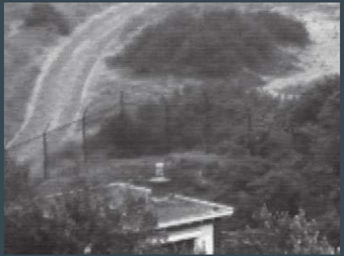
- The highlighting algorithm works by breaking down both images at the same corresponding pixel location into RGB components
- The difference is then calculated between each Red, Green, and Blue value of both images
- If the absolute value of this difference is above a certain threshold, the color is set to pure red on the final image to highlight the major differences

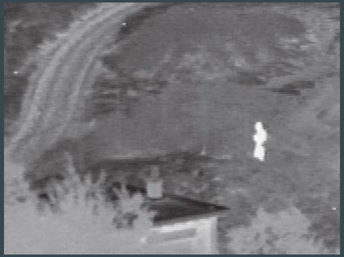


```
public static BufferedImage highlight(BufferedImage colorReference1, BufferedImage colorReference2){
    int width = colorReference1.getWidth();
    int height = colorReference1.getHeight();
    BufferedImage colorized = new BufferedImage(width,height,colorReference1.getType());
    for(int i=0; i<height; i++){
      for(int j=0; j<width; j++){
        Color c1 = new Color(colorReference1.getRGB(j, i));
        int red1 = c1.getRed();
        int green1 = c1.getGreen();
        int blue1 = c1.getBlue();
        Color c2 = new Color(colorReference2.getRGB(j, i));
        int red2 = c2.getRed();
        int green2 = c2.getGreen();
        int blue2 = c2.getBlue();
        int redDiff = Math.abs(red1 - red2);
        int redMedian = ((red1-red2)/2)+red2;
        int red = (redDiff > 100) ? 255 : red1;
        int greenDiff = Math.abs(green1 - green2);
        int greenMedian = ((green1-green2)/2)+green2;
        int green = (greenDiff > 100) ? 0 : green1;
        int blueDiff = Math.abs(blue1 - blue2);
        int blueMedian = ((blue1-blue2)/2)+blue2;
        int blue = (blueDiff > 100) ? 0 : blue1;
        Color newColor = new Color(red,green,blue);
        colorized.setRGB(j,i,newColor.getRGB());}}
    return colorized;}
```

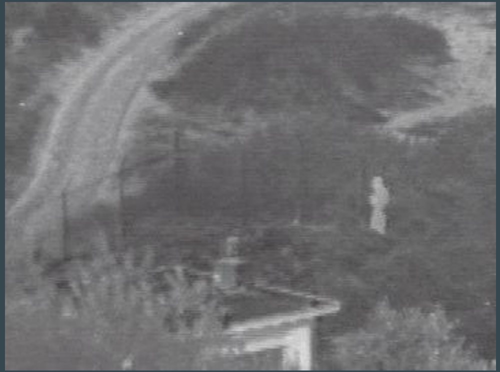
Highlight Results

Original Images

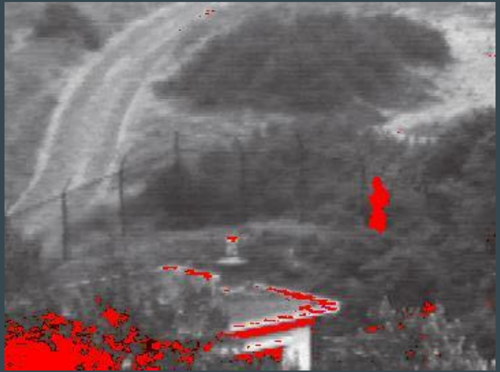




Fused Image



Fused Image Highlighted



Conclusion

Within the given deadlines for this project we have completed

- Full Documentation of Development
  - Software Requirements Specification
  - Design
  - Implementation
  - Testing
- Initial Prototype
- Functional Fusion Image Software
  - Met all agreed requirements

Potential Future Aspects

- GUI
  - Interface for Multiple Images(>2)
  - Drag and Drop Images
- Fusion of Images from Multiple Perspectives
- Connection to Central Database

Software Demonstration

Please wait while we prepare for the demonstration. Thank You.

Questions?

Thank You!