QGIS Before and After Comparisons Concept Code

1. First, we need to import the libraries that we need in to the QGIS console:

**from PyQt5.QtGui import \***  
**from PyQt5.QtCore import \***  
**from qgis.analysis import \***

1. Now, we’ll set up the path names and raster names for our images:

**before = "/Users/qgis\_data/rasters/change-detection/before.tif"**  
**|after = "/Users/qgis\_data/rasters/change-detection/after.tif"**  
**beforeName = "Before"**  
**afterName = "After"**

1. Next, we’ll establish our images as raster layers:

**beforeRaster = QgsRasterLayer(before, beforeName)**  
**afterRaster = QgsRasterLayer(after, afterName)**

1. Then, we can build the calculator entries:

**beforeEntry = QgsRasterCalculatorEntry()**  
**afterEntry = QgsRasterCalculatorEntry()**  
**beforeEntry.raster = beforeRaster**  
**afterEntry.raster = afterRaster**  
**beforeEntry.bandNumber = 1**  
**afterEntry.bandNumber = 2**  
**beforeEntry.ref = beforeName + "@1"**  
**afterEntry.ref = afterName + "@2"**  
**entries = [afterEntry, beforeEntry]**

1. Now, we’ll set up the simple expression that does the math for remote sensing:

**exp = "%s - %s" % (afterEntry.ref, beforeEntry.ref)**

1. Then, we can set up the output file path, the raster extent, and pixel width and height:

**output = "/Users/joellawhead/qgis\_data/rasters/change-detection/change.tif"**  
**e = beforeRaster.extent()**  
**w = beforeRaster.width()**  
**h = beforeRaster.height()**

1. Now, we perform the calculation:

**change = QgsRasterCalculator(exp, output, "GTiff", e, w, h, entries)**  
**change.processCalculation()**

1. Finally, we’ll load the output as a layer, create the color ramp shader, apply it to the layer, and add it to the map, as shown here:

**lyr = QgsRasterLayer(output, "Change")**  
**algorithm = QgsContrastEnhancement.StretchToMinimumMaximum**  
**limits = QgsRaster.ContrastEnhancementMinMax**  
**lyr.setContrastEnhancement(algorithm, limits)**  
**s = QgsRasterShader()**  
**c = QgsColorRampShader()**  
**c.setColorRampType(QgsColorRampShader.INTERPOLATED)**  
**i = []**  
**qri = QgsColorRampShader.ColorRampItem**  
**i.append(qri(0, QColor(0,0,0,0), 'NODATA'))**  
**i.append(qri(-101, QColor(123,50,148,255), 'Significant Itensity Decrease'))**  
**i.append(qri(-42.2395, QColor(194,165,207,255), 'Minor Itensity Decrease'))**  
**i.append(qri(16.649, QColor(247,247,247,0), 'No Change'))**  
**i.append(qri(75.5375, QColor(166,219,160,255), 'Minor Itensity Increase'))**  
**i.append(qri(135, QColor(0,136,55,255), 'Significant Itensity Increase'))**  
**c.setColorRampItemList(i)**  
**s.setRasterShaderFunction(c)**  
**ps = QgsSingleBandPseudoColorRenderer(lyr.dataProvider(), 1, s)**  
**lyr.setRenderer(ps)**  
**QgsMapLayerRegistry.instance().addMapLayer(lyr)**

How it works…

If a building is added in the new image, it will be brighter than its surroundings. If a building is removed, the new image will be darker in that area. The same holds true for vegetation, to some extent.

Summary

The concept is simple. We subtract the older image data from the new image data. Concentrating on urban areas tends to be highly reflective and results in higher image pixel values.