In [1]:

```
# first: find regions that are plates
```

In [2]:

```
# import settings and packages
%matplotlib inline
from skimage import io
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
from skimage.filters import threshold_otsu
from skimage.segmentation import clear_border
from skimage.measure import label, regionprops
from skimage.morphology import closing, square
from skimage.color import label2rgb
from scipy import ndimage as ndi
import glob
```

In [3]:

```
# create new class of object called cultureplate:
class cultureplate:
    def __init__ (self, name, boundaries, colonies, image, label,
pos, scanner):
        self.name = name
        self.colonies = dict()
        self.boundaries = tuple()
        self.image = ()
        self.label = ()
        self.pos = ()
        self.scanner=()
    def add_colony(self, colonies, key, value):
        self.colonies[key]=value
    def rename(self, newname):
        self.name = newname
    def add_image(self, newimage):
        self.image = newimage
```

In [5]:

```
# import final scanned image
# after having imported it into the local folder
# and separate out blue channel (could alternatively use green)
plates1 = io.imread('/home/cmarx/Documents/Jessica/images/deathat4mM_
170917/scanner2/scan_2017-09-24_12:02:02.tiff')
```

In [6]:

```
cultureplates_manual = list()
```

In [7]:

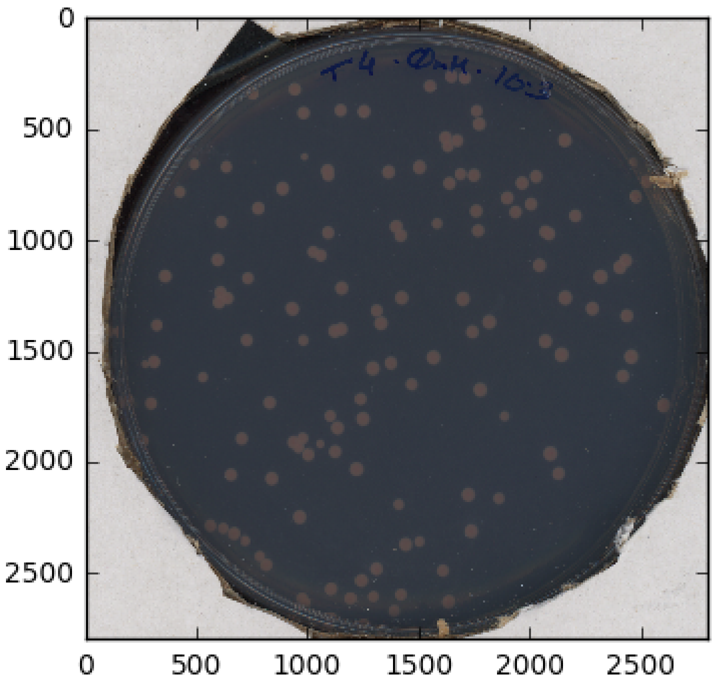
```
plateA = plates1[300:3100, 3400:6200,:]
```

In [8]:

```
plt.imshow(plateA)
```

Out[8]:

```
<matplotlib.image.AxesImage at 0x7fc56d61b470>
```



In [9]:

```
cultureplates_manual.append(cultureplate(name='T4_0mM_10-3',
                                         boundaries=(),
                                         colonies=dict(),
                                         image=(),
                                         label=(),
                                         pos=(),
                                         scanner=()))
len(cultureplates_manual)
```

Out[9]:

```
1
```

In [10]:

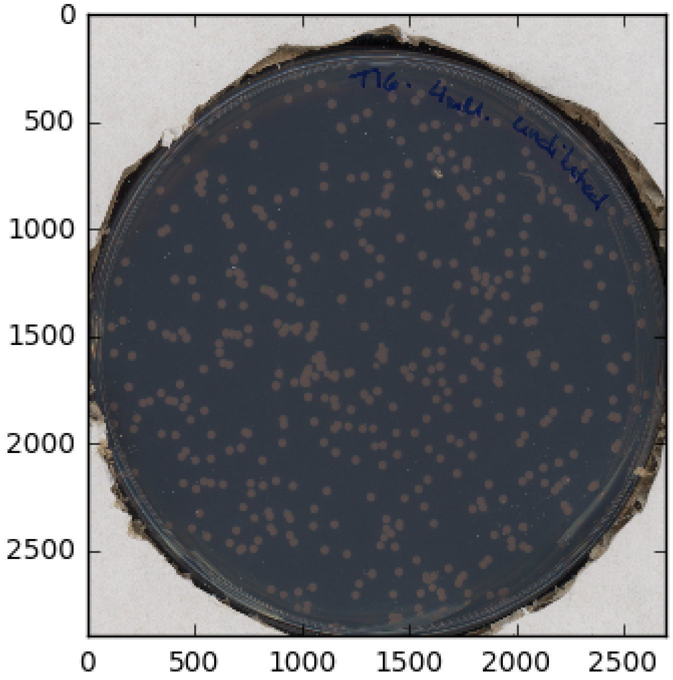
```
cultureplates_manual[0].add_image(plates1[300:3100, 3400:6200,:])
cultureplates_manual[0].boundaries=(300,3400,3100,6200)
cultureplates_manual[0].pos='topright'
cultureplates_manual[0].scanner='2'
```

In [11]:

```
plateB = plates1[3200:6100, 3600:6300,:]
plt.imshow(plateB)
```

Out[11]:

```
<matplotlib.image.AxesImage at 0x7fc5688e69b0>
```



In [12]:

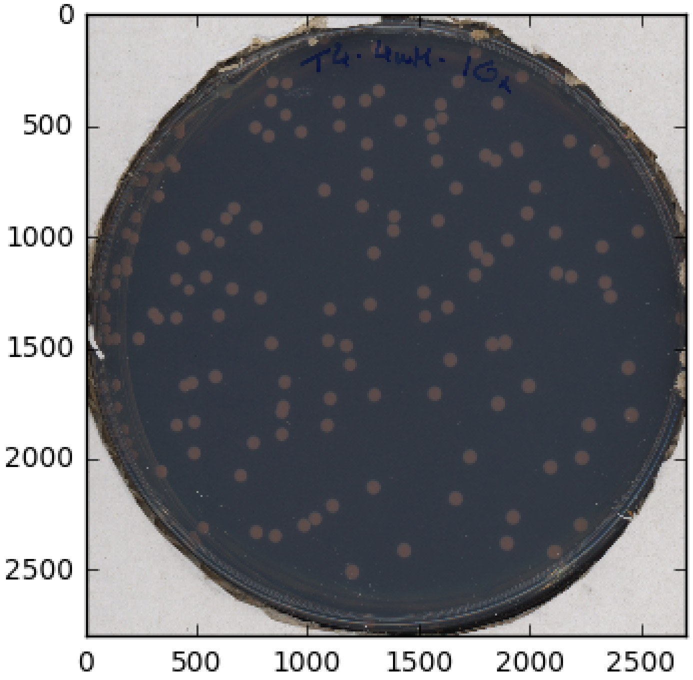
```
cultureplates_manual.append(cultureplate(name='T16_4mM_undiluted',
                                         boundaries=(),
                                         colonies=dict(),
                                         image=(),
                                         label=(),
                                         pos=(),
                                         scanner=()))
cultureplates_manual[1].add_image(plates1[3200:6100, 3600:6300,:])
cultureplates_manual[1].boundaries=(3200,3600,6100,6300)
cultureplates_manual[1].pos='middleright'
cultureplates_manual[1].scanner='2'
len(cultureplates_manual)
```

Out[12]:

```
2
```

```
In [13]: plateC= plates1[300:3100, 400:3100,:]
         plt.imshow(plateC)

Out[13]: <matplotlib.image.AxesImage at 0x7fc566a77438>
```

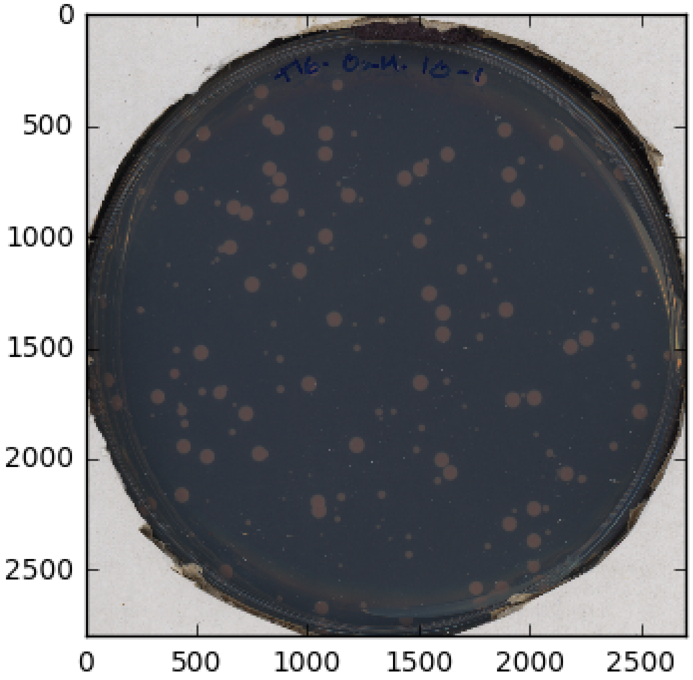


```
In [14]: cultureplates_manual.append(cultureplate(name='T4_4mM_10x',
                                                  boundaries=(),
                                                  colonies=dict(),
                                                  image=(),
                                                  label=(),
                                                  pos=(),
                                                  scanner=()))
         cultureplates_manual[2].add_image(plates1[300:3100, 400:3100,:])
         cultureplates_manual[2].boundaries=(300,400,3100,3100)
         cultureplates_manual[2].pos='topleft'
         cultureplates_manual[2].scanner='2'
         len(cultureplates_manual)

Out[14]: 3
```

```
In [15]: plateD= plates1[3400:6200, 500:3200,:]
         plt.imshow(plateD)

Out[15]: <matplotlib.image.AxesImage at 0x7fc566a5dd30>
```

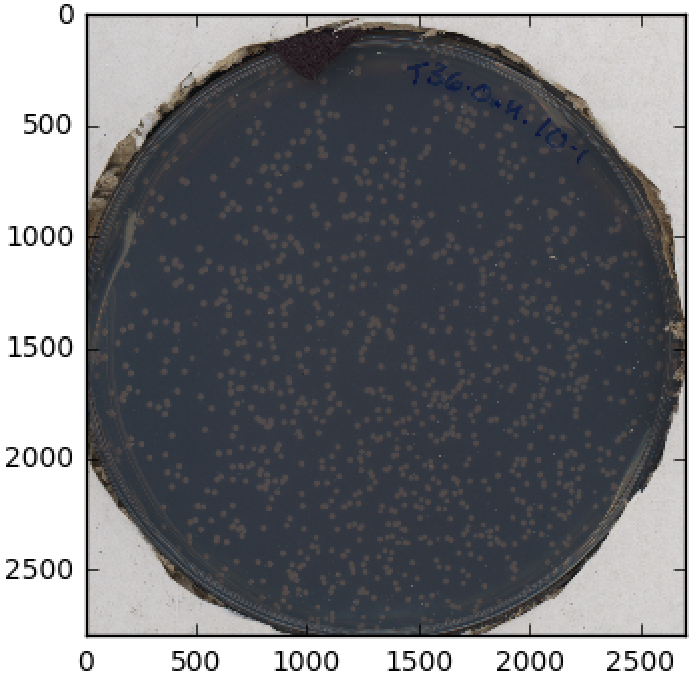


```
In [16]: cultureplates_manual.append(cultureplate(name='T16_0mM_10-1',
                                                  boundaries=(),
                                                  colonies=dict(),
                                                  image=(),
                                                  label=(),
                                                  pos=(),
                                                  scanner=()))
         cultureplates_manual[3].add_image(plates1[3400:6200, 500:3200,:])
         cultureplates_manual[3].boundaries=(3400,500,6200,3200)
         cultureplates_manual[3].pos='middleleft'
         cultureplates_manual[3].scanner='2'
         len(cultureplates_manual)

Out[16]: 4
```

In [17]:
```
plateE= plates1[6500:9300, 400:3100,:]
plt.imshow(plateE)
```

Out[17]:
```
<matplotlib.image.AxesImage at 0x7fc5669b9cf8>
```



In [18]:
```
cultureplates_manual.append(cultureplate(name='T36_0mM_10-1',
                                         boundaries=(),
                                         colonies=dict(),
                                         image=(),
                                         label=(),
                                         pos=(),
                                         scanner=()))
cultureplates_manual[4].add_image(plates1[6500:9300, 400:3100,:])
cultureplates_manual[4].boundaries=(6500,400,9300,3100)
cultureplates_manual[4].pos='bottomleft'
cultureplates_manual[4].scanner='2'
len(cultureplates_manual)
```

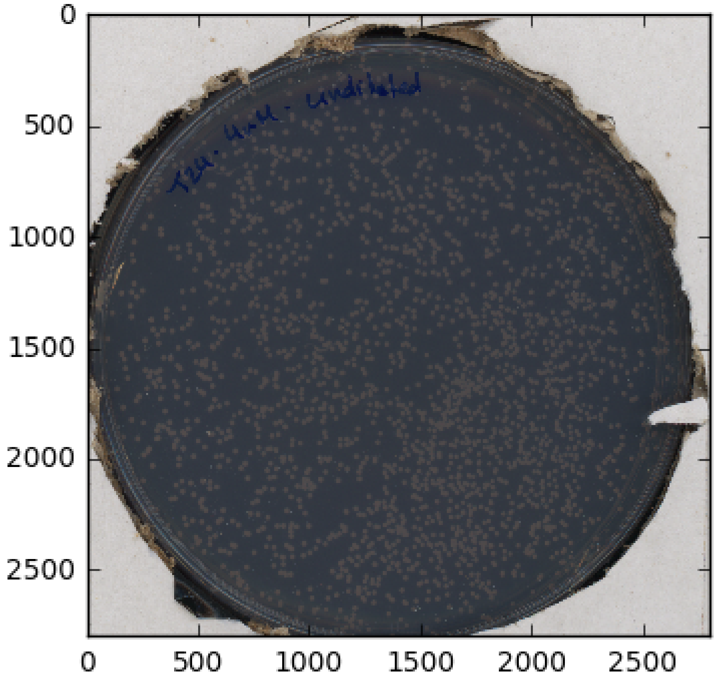
Out[18]:
```
5
```

In [19]:

```
plateF= plates1[6400:9200, 3500:6300,:]
plt.imshow(plateF)
```

Out[19]:

```
<matplotlib.image.AxesImage at 0x7fc5669a3518>
```



In [20]:

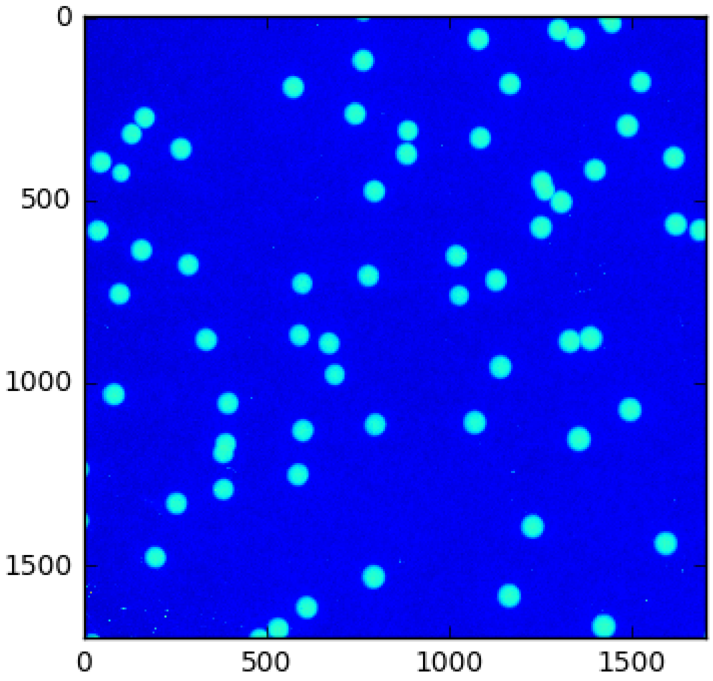
```
cultureplates_manual.append(cultureplate(name='T24_4mM_undiluted',
                                          boundaries=(),
                                          colonies=dict(),
                                          image=(),
                                          label=(),
                                          pos=(),
                                          scanner=()))
cultureplates_manual[5].add_image(plates1[6400:9200, 3500:6300,:])
cultureplates_manual[5].pos='bottomright'
cultureplates_manual[5].scanner='2'
cultureplates_manual[5].boundaries=(6400,3500,9200,6300)
len(cultureplates_manual)
```

Out[20]:

```
6
```

```
In [23]: # now find thresholds
         testplate = plateC[600:2300, 500:2200, 0]
         plt.imshow(testplate)
```

```
Out[23]: <matplotlib.image.AxesImage at 0x7fc564a3db70>
```



```
In [24]: thresh = threshold_otsu(testplate)
```

```
In [25]: thresh
```

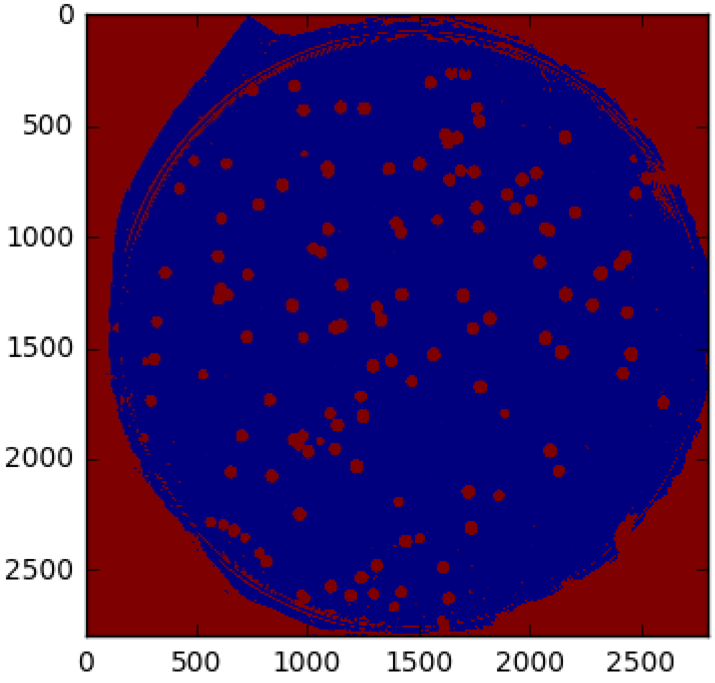
```
Out[25]: 68
```

In [26]:

```
bw_plate1 = closing(plateA[:,:,0] > thresh, square(3))
plt.imshow(bw_plate1)
```

Out[26]:

```
<matplotlib.image.AxesImage at 0x7fc564a18e10>
```



In [27]:

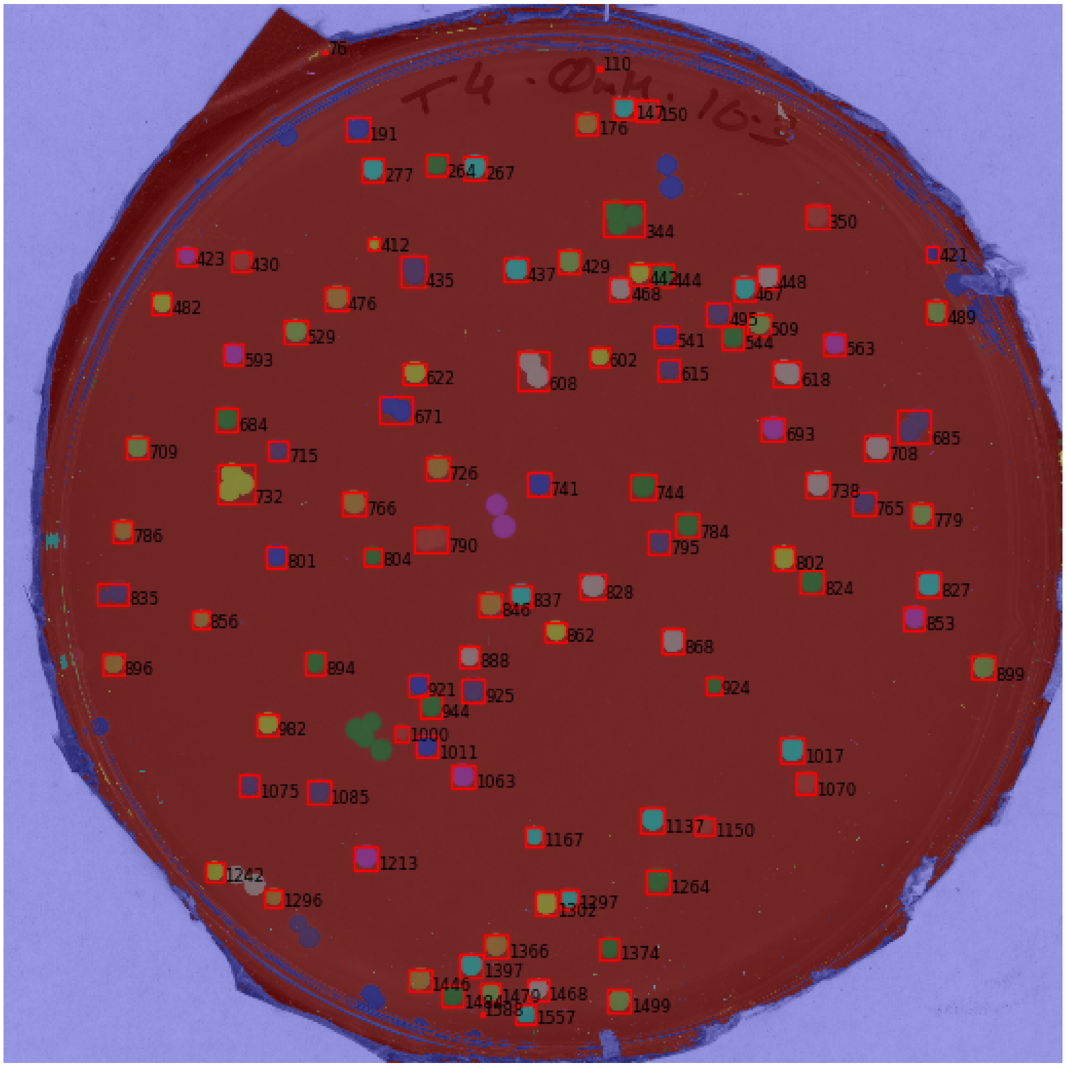
```
del plateA
del plateB
del plateC
del plateD
del plateE
del plateF
```

In [28]:

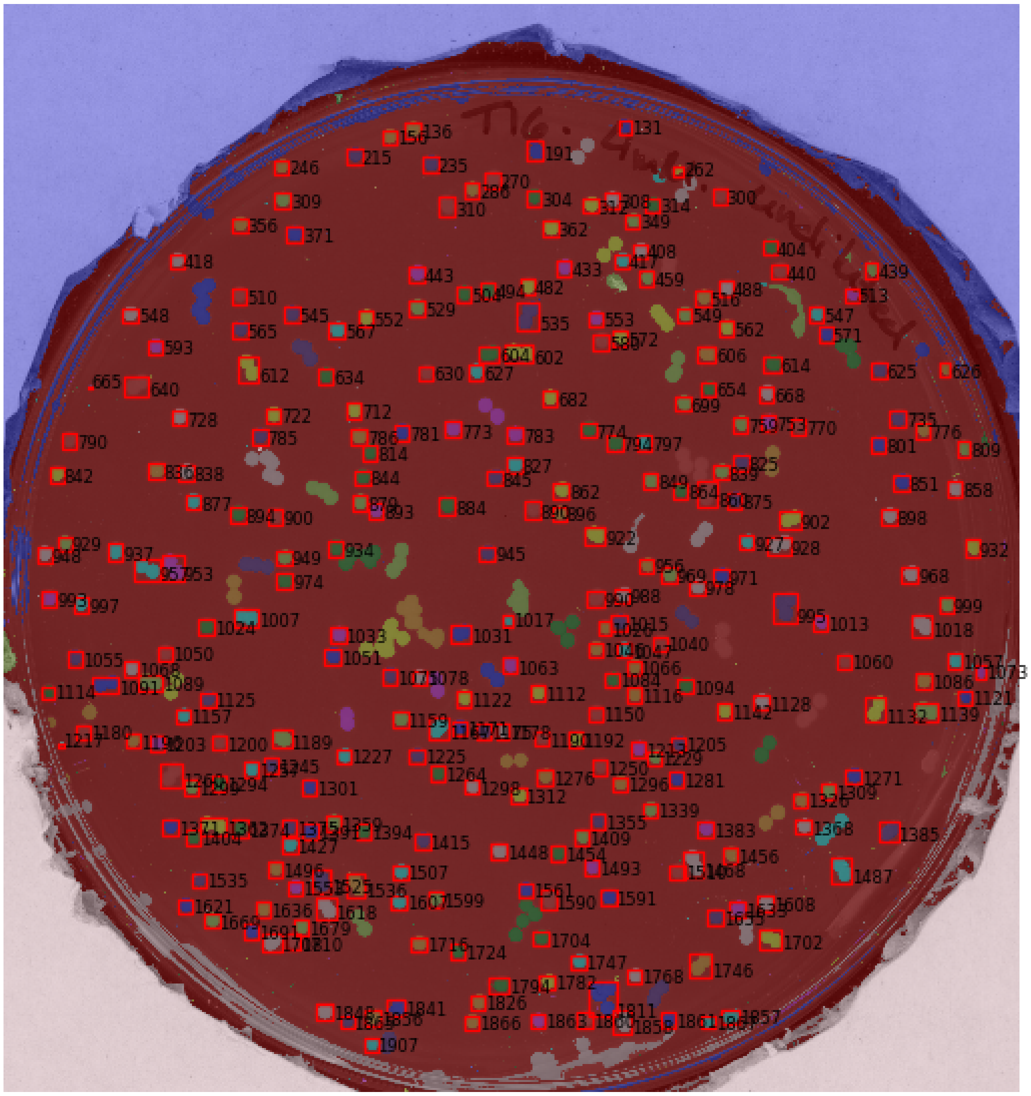
```
for platenum in range(0,6):
    bw_plate = closing(cultureplates_manual[platenum].image[:,:,0] >
thresh, square(3))
    labeled_plate = label(bw_plate)
    overlaid_plate = label2rgb(labeled_plate, image=cultureplates_man
ual[platenum].image)
    cultureplates_manual[platenum].label = labeled_plate # add the in
fo about labeling
    cultureplates_manual[platenum].add_image(overlaid_plate) # add th
is overlay image to the object
    for region in regionprops(labeled_plate):
        if 40 < region.area < 10000:
            minr, minc, maxr, maxc = region.bbox
            if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                if region.area>0.6*((maxr-minr)*(maxc-minc)):
                    cultureplates_manual[platenum].colonies[region.la
bel] = region.coords # add only the good colonies to the dictionary
```

In [30]:

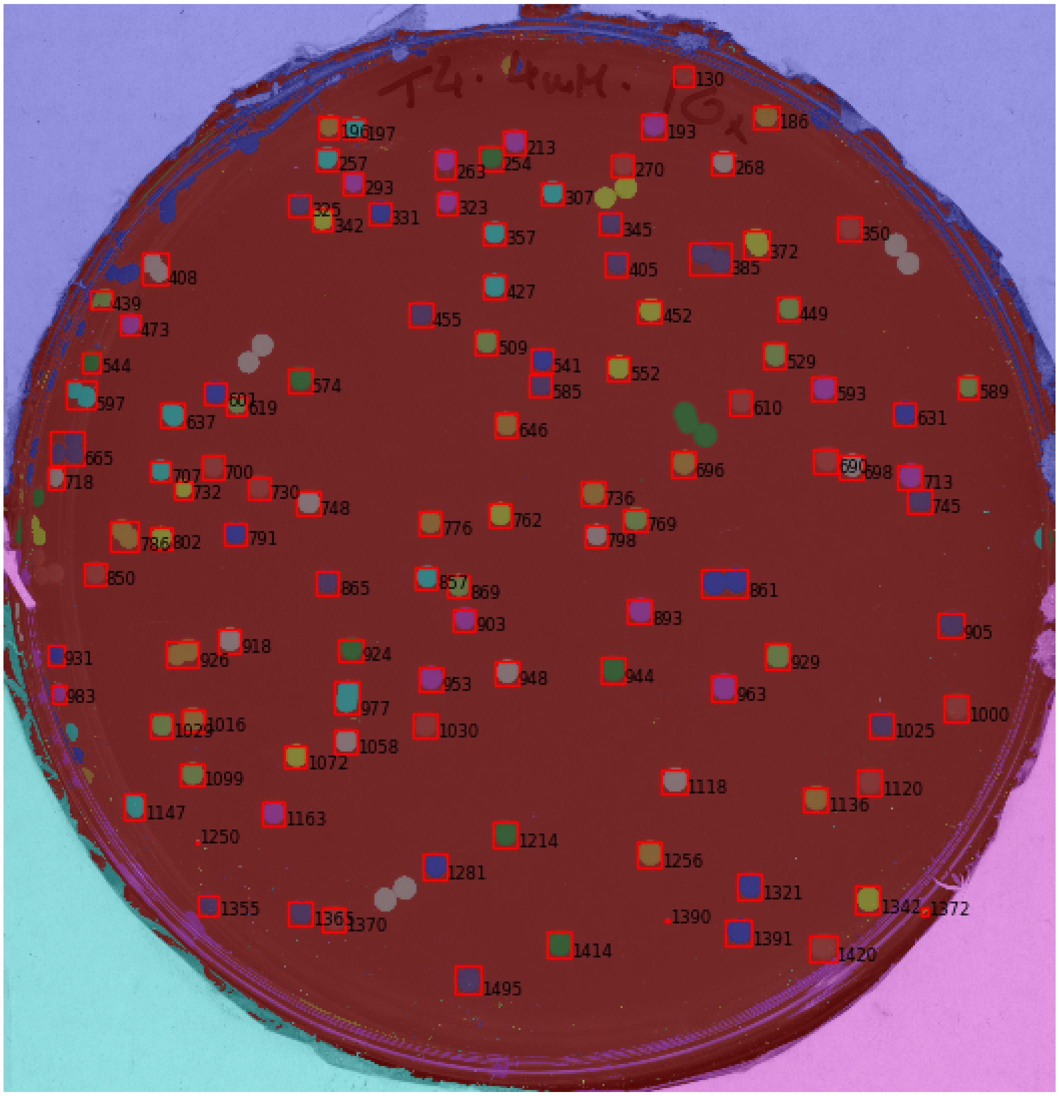
```
# find the good colonies
fig, ax = plt.subplots(figsize=(10, 6))
ax.imshow(cultureplates_manual[0].image)
for region in regionprops(cultureplates_manual[0].label):
    if 40 < region.area < 10000:
        minr, minc, maxr, maxc = region.bbox
        if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                if region.area>0.6*((maxr-minr)*(maxc-minc)):
                    rect = mpatches.Rectangle((minc, minr), maxc - mi
nc, maxr - minr,
                                              fill=False, edgecolor='red', li
newidth=1)
                    ax.add_patch(rect)
                    ax.text(maxc, maxr, str(region.label),
fontsize=6)
ax.set_axis_off()
plt.tight_layout()
plt.show()
```



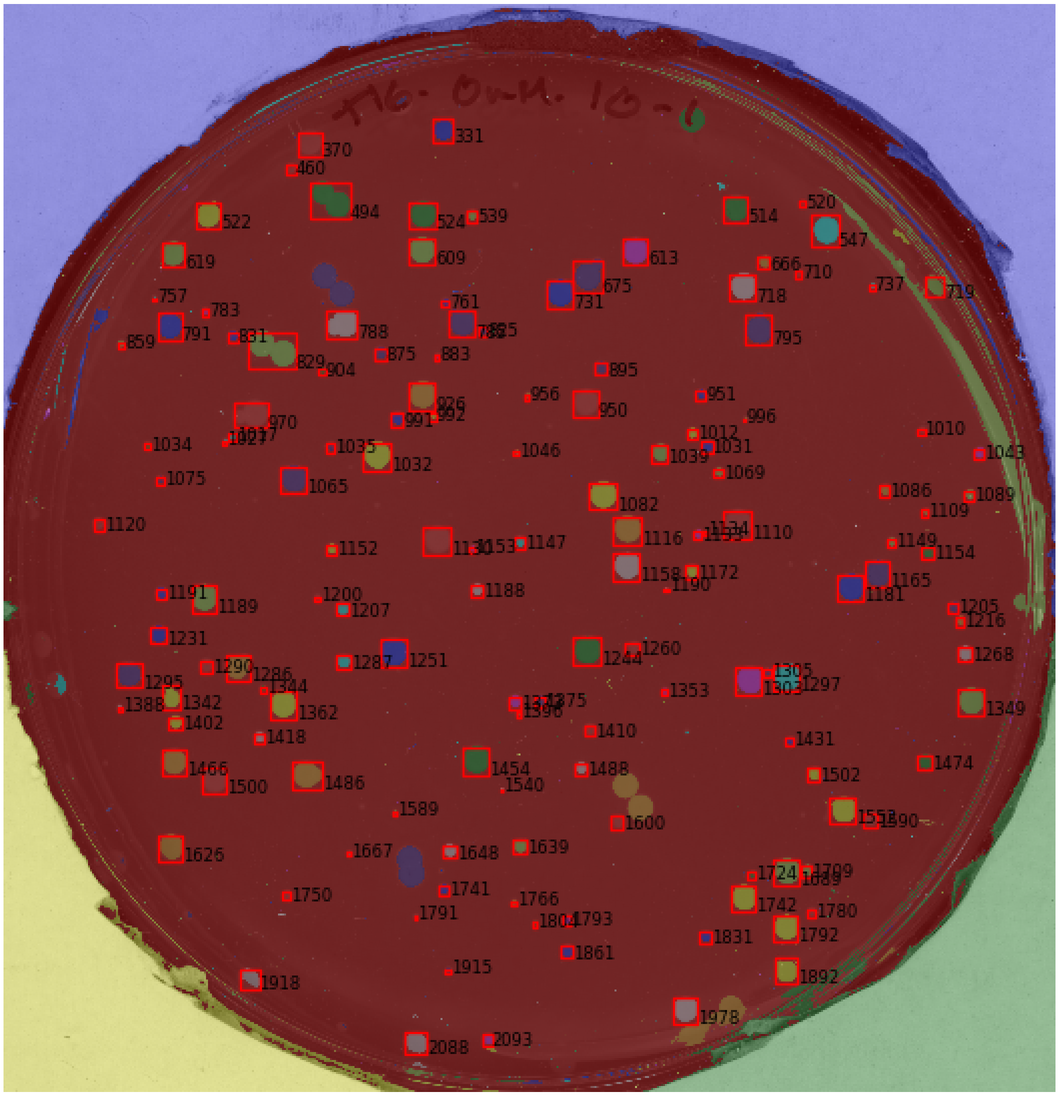
```
In [31]: fig, ax = plt.subplots(figsize=(10, 6))
         ax.imshow(cultureplates_manual[1].image)
         for region in regionprops(cultureplates_manual[1].label):
             if 40 < region.area < 10000:
                 minr, minc, maxr, maxc = region.bbox
                 if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                         if region.area>0.6*((maxr-minr)*(maxc-minc)):
                             rect = mpatches.Rectangle((minc, minr), maxc - mi
         nc, maxr - minr,
                                                   fill=False, edgecolor='red', li
         newidth=1)
                             ax.add_patch(rect)
                             ax.text(maxc, maxr, str(region.label),
         fontsize=6)
         ax.set_axis_off()
         plt.tight_layout()
         plt.show()
```



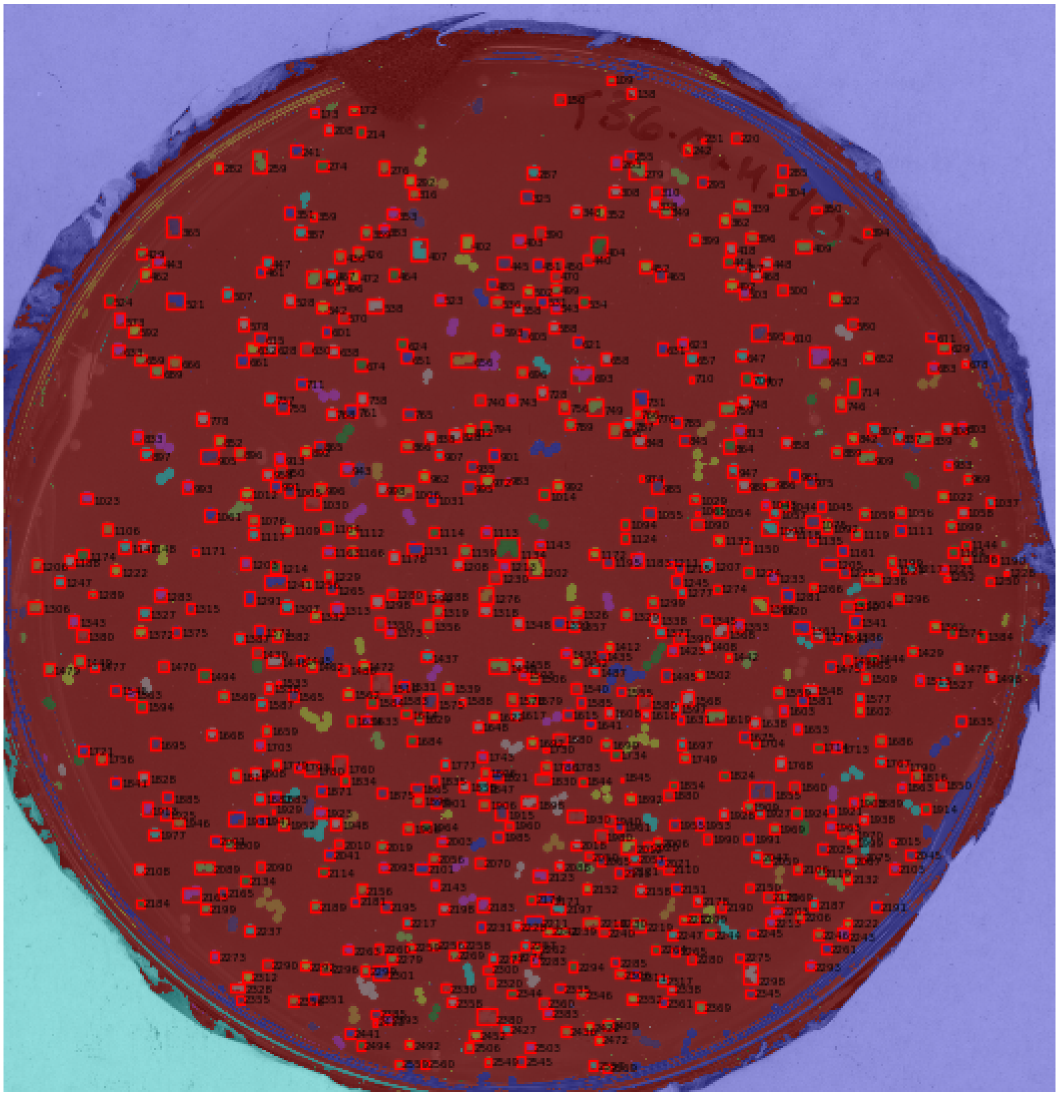
```
In [32]: fig, ax = plt.subplots(figsize=(10, 6))
         ax.imshow(cultureplates_manual[2].image)
         for region in regionprops(cultureplates_manual[2].label):
             if 40 < region.area < 10000:
                 minr, minc, maxr, maxc = region.bbox
                 if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                         if region.area>0.6*((maxr-minr)*(maxc-minc)):
                             rect = mpatches.Rectangle((minc, minr), maxc - mi
         nc, maxr - minr,
                                                      fill=False, edgecolor='red', li
         newidth=1)
                             ax.add_patch(rect)
                             ax.text(maxc, maxr, str(region.label),
         fontsize=6)
         ax.set_axis_off()
         plt.tight_layout()
         plt.show()
```



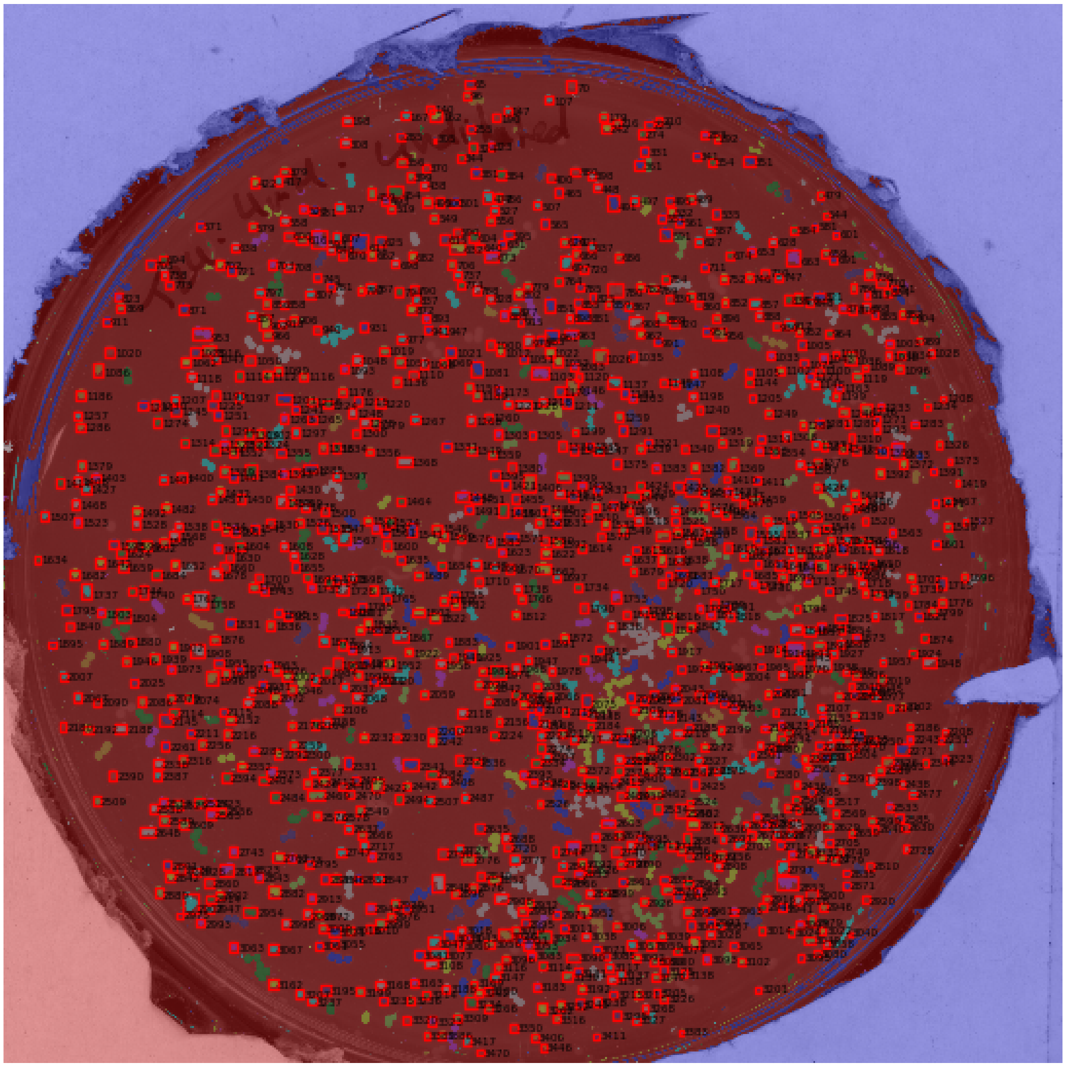
```
In [33]: fig, ax = plt.subplots(figsize=(10, 6))
         ax.imshow(cultureplates_manual[3].image)
         for region in regionprops(cultureplates_manual[3].label):
             if 40 < region.area < 10000:
                 minr, minc, maxr, maxc = region.bbox
                 if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                         if region.area>0.6*((maxr-minr)*(maxc-minc)):
                             rect = mpatches.Rectangle((minc, minr), maxc - mi
         nc, maxr - minr,
                                                   fill=False, edgecolor='red', li
         newidth=1)
                             ax.add_patch(rect)
                             ax.text(maxc, maxr, str(region.label),
         fontsize=6)
         ax.set_axis_off()
         plt.tight_layout()
         plt.show()
```



In [35]:
```
fig, ax = plt.subplots(figsize=(10, 6))
ax.imshow(cultureplates_manual[4].image)
for region in regionprops(cultureplates_manual[4].label):
    if 40 < region.area < 10000:
        minr, minc, maxr, maxc = region.bbox
        if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                if region.area>0.6*((maxr-minr)*(maxc-minc)):
                    rect = mpatches.Rectangle((minc, minr), maxc - mi
nc, maxr - minr,
                                          fill=False, edgecolor='red', li
newidth=1)
                    ax.add_patch(rect)
                    ax.text(maxc, maxr, str(region.label),
fontsize=4)
ax.set_axis_off()
plt.tight_layout()
plt.show()
```



In [37]:
```
fig, ax = plt.subplots(figsize=(10, 6))
ax.imshow(cultureplates_manual[5].image)
for region in regionprops(cultureplates_manual[5].label):
    if 40 < region.area < 10000:
        minr, minc, maxr, maxc = region.bbox
        if 0.6<(maxr-minr)/(maxc-minc)<1.4:
                if region.area>0.6*((maxr-minr)*(maxc-minc)):
                    rect = mpatches.Rectangle((minc, minr), maxc - mi
nc, maxr - minr,
                                              fill=False, edgecolor='red', li
newidth=1)
                    ax.add_patch(rect)
                    ax.text(maxc, maxr, str(region.label),
fontsize=4)
ax.set_axis_off()
plt.tight_layout()
plt.show()
```



In [ ]:

```
# now, loop through all the plates to get the timecourse data
# if you're absolutely certain the experiment is over,
# sequester the last 3 timepoints in another folder
# to prevent anomalous flatlines at the end of growth
```

In [38]:

```
out = open('/home/cmarx/Documents/Jessica/imageprocessing/imageproces
sing_4mM_170924/scanner2_plate_IDs_170924.csv', 'w')
for eachplate in cultureplates_manual:
    out.write(str(eachplate.name)+','+str(eachplate.pos)+','+str(each
plate.scanner)+'\n')
out.close()
```

In [39]:

```
path = "/home/cmarx/Documents/Jessica/images/deathat4mM_170917/scanne
r2/scan_*.tiff"
out = open('/home/cmarx/Documents/Jessica/imageprocessing/imageproces
sing_4mM_170924/scanner2_colony_trajectories_170924.csv', 'w')
for eachplate in cultureplates_manual:
    topbound = eachplate.boundaries[0]
    bottombound = eachplate.boundaries[2]
    leftbound = eachplate.boundaries[1]
    rightbound = eachplate.boundaries[3]
    for filename in glob.glob(path):
        image1 = io.imread(filename)[topbound:bottombound,
leftbound:rightbound,0]
        image_thresh = closing(image1 > thresh, square(3))
        colony_pixel_counts = dict()
        for colony in eachplate.colonies.keys():
            whitepix = 0
            for pixel in eachplate.colonies[colony]:
                x_coord=pixel[0]
                y_coord=pixel[1]
                if image_thresh[x_coord, y_coord]==True:
                    whitepix = whitepix+1
            colony_pixel_counts[colony] = (whitepix)
        for colony2 in colony_pixel_counts.keys():
            out.write(str(eachplate.name)+','
                      +str(colony2)+','
                      +str(colony_pixel_counts[colony2])+','
                      +str(filename[-24:-5])+'\n')
out.close()
```

In [ ]: