Full Pipeline with Sliding Window (on multiple tilesets) - Aug 2017

Created: 21 Aug 2018
Last update: 31 Oct 2018

Goal: Try pipeline but then using a 'sliding window'

---

1. Imports

In [1]:
```
# this will remove warnings messages
import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

# import
from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA, TruncatedSVD
from sklearn.pipeline import Pipeline
from sklearn.metrics import silhouette_score

import imgutils
```

In [2]:
```
# Re-run this cell if you altered imgutils
import importlib
importlib.reload(imgutils)
```

Out[2]:
```
<module 'imgutils' from 'C:\\JADS\\SW\\Grad Proj\\sources\\imgutils.py'>
```

---

2. Data Definitions & Feature Specification

In [3]:
```
# Data:
datafolder = '../data/Crystals_Apr_12/Tileset6_subset_1K'
n_tiles_x = 3  # mostly for visualization
n_tiles_y = 3


# Features to use:
#feature_funcs = [imgutils.img_mean, imgutils.img_std, imgutils.img_median,
#                 imgutils.img_mode,
#                 imgutils.img_kurtosis, imgutils.img_skewness]
feature_funcs = [imgutils.img_std, imgutils.img_relstd, imgutils.img_mean,
                 imgutils.img_skewness,  imgutils.img_kurtosis, imgutils.img_mode]
feature_names = imgutils.stat_names(feature_funcs)

# Size of the grid, specified as number of slices per image in x and y direction:
default_grid_x = 4
default_grid_y = default_grid_x
```

---

3. Import Data & Extract Features

Hyper parameters

In [4]:
```
# feature extraction
patch_size=(20,20)

# data hyper-parameters
default_n_clusters = 3

# algorithm hyper-parameters:
kmeans_n_init = 10
```

In [5]:
```
imgs = imgutils.getimgfiles(datafolder,'.tif')
```

In [6]:
```
img = imgutils.loadtiff(imgs[0])
print(img.shape)
img2 = imgutils.downsample_img(img, 2)
print(img2.shape)
```

```
(2000, 2000)
(1000, 1000)
```

In [7]:
```
import sklearn.feature_extraction.image as skimgfeat
import math
```

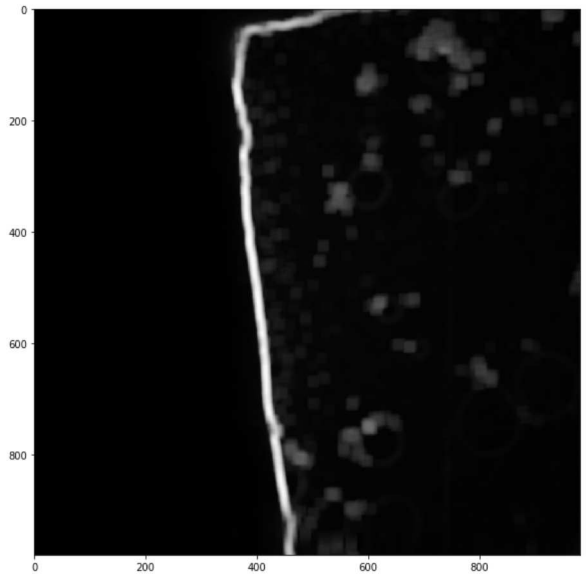
In [8]:
```
patches = skimgfeat.extract_patches_2d(img2, patch_size)
```

In [9]:
```
stds = np.empty(patches.shape[0])
```

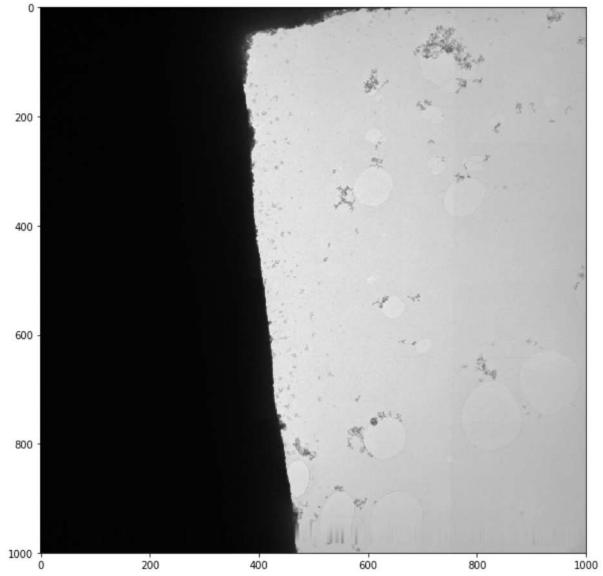
In [10]:
```
for i in range(patches.shape[0]):
    stds[i] = np.std(patches[i])
```

In [11]:
```
dim = (int)(math.sqrt(stds.shape[0]))
img3 = np.reshape(stds, (dim, dim))
```

In [12]:
```
imgutils.showimg(img3, fig_size=(10,10))
```



In [13]:
```
imgutils.showimg(img2, fig_size=(10,10))
```



In [14]:
```
patches = skimgfeat.extract_patches_2d(img2, patch_size)
patchstats = np.empty((patches.shape[0],5))
print(patch_size)
```

```
(20, 20)
```

In [15]:
```
print("Extracting features...")
for i in range(patches.shape[0]):
    patch = patches[i]
    patchstats[i,0] = np.mean(patch)
    patchstats[i,1] = np.median(patch)
    patchstats[i,2] = np.std(patch)
    patchstats[i,3] = np.max(patch)-np.min(patch)
    #patchstats[i,4] = np.percentile(patch,75)-np.percentile(patch,25)
```

```
Extracting features...
```

In [16]:
```
n_clusters = 4
print("CLustering...")
kmeans = KMeans(algorithm='auto', n_clusters=n_clusters, n_init=10, init='k-means++')
standardizer = StandardScaler()
pca = PCA()
pipeline = Pipeline([('scaler', standardizer), ('pca', pca), ('kmeans',kmeans)])
#pipeline = Pipeline([('scaler', standardizer),('kmeans',kmeans)])

y = pipeline.fit_predict(patchstats)
```

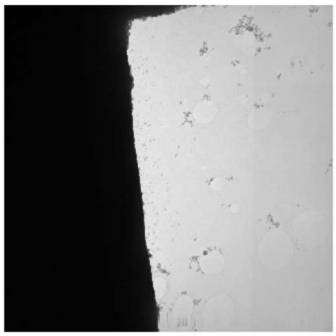
```
CLustering...
```

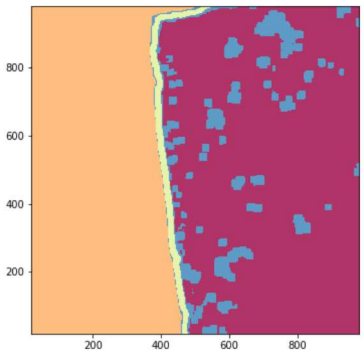
In [17]:
```
dim = (int)(math.sqrt(y.shape[0]))
img_clust = np.reshape(y, (dim, dim))
```

In [18]:
```
img_mean = np.reshape(patchstats[:,0], (dim, dim))
img_median = np.reshape(patchstats[:,1], (dim, dim))
img_std = np.reshape(patchstats[:,2], (dim, dim))
img_range = np.reshape(patchstats[:,3], (dim, dim))
```

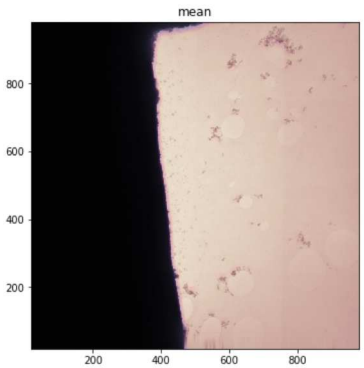
In [88]:
```
def plot_with_overlay(orgimg, overlayimg, fig_size=(6,6), show_org=True, show_overlay=True,
                      overlay_alpha=0.3, cmapname='Spectral', title=None):
    l = (orgimg.shape[0] - overlayimg.shape[0])
    t = (orgimg.shape[1] - overlayimg.shape[1])
    r = (orgimg.shape[0] - l)
    b = (orgimg.shape[1] - t)

    cmin = 1.1*np.min(overlayimg)
    cmax = 1.1*np.max(overlayimg)
    _ = plt.figure(figsize=fig_size)
    if show_org:
        plt.imshow(orgimg, cmap='gray', origin='upper', extent=[0,orgimg.shape[0], 0, orgimg.shape[1]])
        plt.axis('off')
    else:
        overlay_alpha=0.8
    if (show_overlay):
        plt.imshow(overlayimg, cmap=cmapname, alpha=overlay_alpha, vmin=cmin, vmax=cmax, origin='upper', extent=[l,r,t,b])
        plt.axis('off')
    if (title): plt.title(title)
    plt.show()
```

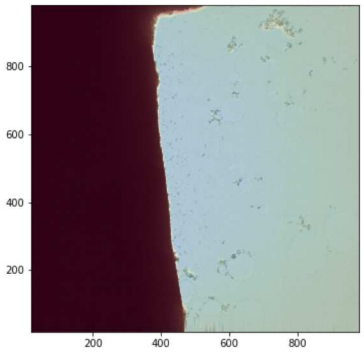
In [87]:
```
plot_with_overlay(img2, img_clust, show_overlay=False)
plot_with_overlay(img2, img_clust, show_org=False)
```



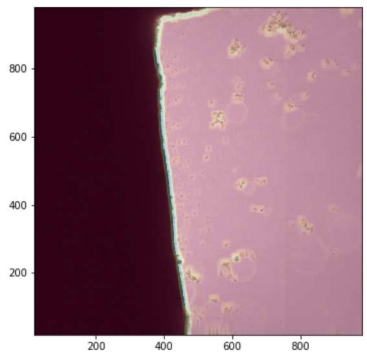


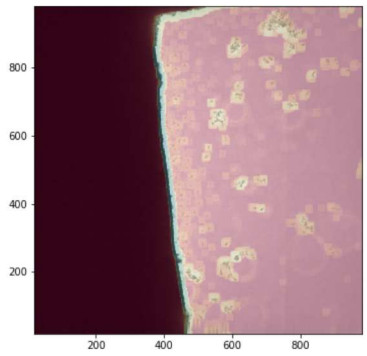
In [21]:
```
plot_with_overlay(img2, img_mean, title='mean', cmapname='magma')
plot_with_overlay(img2, img_median)
```



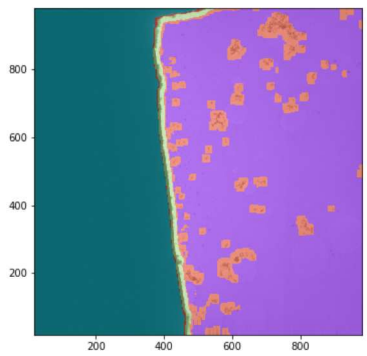


In [22]:
```
plot_with_overlay(img2, img_std)
plot_with_overlay(img2, img_range)
```





In [23]:
```
plot_with_overlay(img2, img_clust, cmapname='rainbow', overlay_alpha=0.5)
```



In [122]:
```
import sklearn.feature_extraction.image as skimgfeat
import matplotlib.pyplot as plt
import math
import sys

def run_new_pipeline(imgfilename, patch_size, n_clusters, downscale_factor=2, return_cluster_image = False,
                     algorithm='kmeans', show_results=True, show_diagnostics=False, show_diagnostics_extra=False,
                    fig_size=(6,6)):
    """ """

    print("Importing image(s)s...")
    img_full = imgutils.loadtiff(imgfilename)
    img = imgutils.downsample_img(img_full, downscale_factor)
    patches = skimgfeat.extract_patches_2d(img, patch_size)

    sys.stdout.write("Extracting features...")
    patchstats = np.empty((patches.shape[0],4))
    n_progress_update = (int)(patches.shape[0] / 1000)
    for i in range(patches.shape[0]):
        patch = patches[i]
        patchstats[i,0] = np.mean(patch)
        patchstats[i,1] = np.median(patch)
        patchstats[i,2] = np.std(patch)
        patchstats[i,3] = np.max(patch)-np.min(patch)
        if (i % n_progress_update == 0):
            progress = (int)(i*100.0 / patches.shape[0])
            sys.stdout.write("\rExtracting features... {:d} %".format(progress))
            sys.stdout.flush()
    sys.stdout.write("\rExtracting features... 100 %\n")
    sys.stdout.flush()

    print("Clustering into {} clusters...".format(n_clusters))
    kmeans = KMeans(algorithm='auto', n_clusters=n_clusters, n_init=10, init='k-means++')
    hierarch = AgglomerativeClustering(n_clusters=n_clusters, affinity='euclidean', linkage='complete')
    if (algorithm=='kmeans'):
        pipeline = Pipeline([('scaler', StandardScaler()), ('pca', PCA()), ('kmeans',kmeans)])
    elif (algorithm=='hierarchical'):
        pipeline = Pipeline([('scaler', StandardScaler()), ('pca', PCA()), ('hierarchical',hierarch)])
    else:
        raise ValueException("unsupported algorithm {}".format(algorithm))

    #x = patchstats
    x = patchstats[:,1:3]  # only mean, std and range
    y = pipeline.fit_predict(x)

    dim = (int)(math.sqrt(y.shape[0]))
    img_clust = np.reshape(y, (dim, dim))

    if show_results:
        print("Visualizing results...")
        plot_with_overlay(img, img_clust, title='cluster heatmap')
        plt.show()

    if show_diagnostics:
        print("Showing diagnostic images...")
        plot_with_overlay(img, img_clust, show_overlay=False, title='original image', fig_size=fig_size)
        plot_with_overlay(img, img_clust, show_org=False, title='local clusters', fig_size=fig_size)
        plt.show()

    if show_diagnostics_extra:
        print("Showing diagnostic feature images...")
        img_mean = np.reshape(patchstats[:,0], (dim, dim))
        img_median = np.reshape(patchstats[:,1], (dim, dim))
        img_std = np.reshape(patchstats[:,2], (dim, dim))
        img_range = np.reshape(patchstats[:,3], (dim, dim))
        plot_with_overlay(img, img_mean, title='local mean', fig_size=fig_size)
        plot_with_overlay(img, img_median, title='local median', fig_size=fig_size)
        plot_with_overlay(img, img_std, title='local standard deviation', fig_size=fig_size)
        plot_with_overlay(img, img_range, title='local range', fig_size=fig_size)

    if return_cluster_image:
        return img, img_clust
```

In [84]:
```
def get_single_cluster_image(cluster_img, cluster_num):
    return (cluster_img==cluster_num).astype(int)

def show_cluster_images(img, cluster_img, n_clusters, show_img=False, cmapname='tab10', fig_size=(6,6)):
    for i in range(n_clusters):
        img_clust_i = get_single_cluster_image(cluster_img, i)
        plot_with_overlay(img, img_clust_i, title='cluster {}'.format(i), show_org=show_img, cmapname=cmapname, fig_size=fig_size)

def show_single_cluster_image(img, cluster_img, cluster_to_show, show_img=True, opacity=0.5, cmapname='RdYlGn', fig_size=(6,6)):
    img_clust_i = get_single_cluster_image(cluster_img, cluster_to_show)
    plot_with_overlay(img, img_clust_i, title='cluster {}'.format(cluster_to_show), show_org=show_img, overlay_alpha=opacity, cmapname=cmapname,fig_size=fig_size)
```

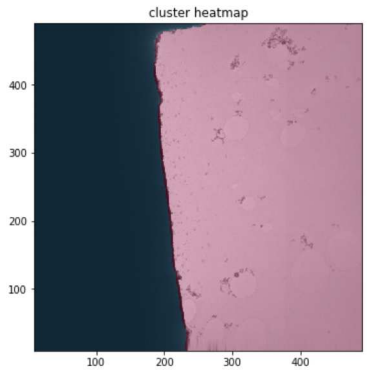
Try on multiple data sets - 2 Clusters

In [26]:
```
n_clust = 2
patch_size = (10,10)
```

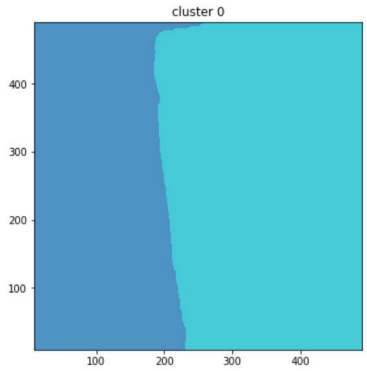
realxtals lm (hard)

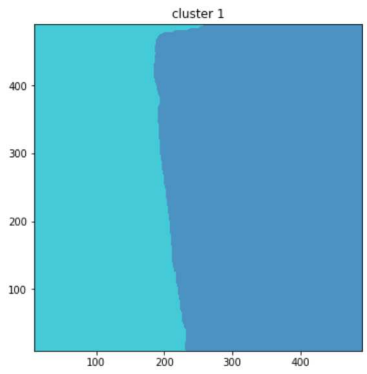
In [27]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset6_subset_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 2 clusters...
Visualizing results...
```



In [28]:
```
show_cluster_images(img, h, n_clust)
```

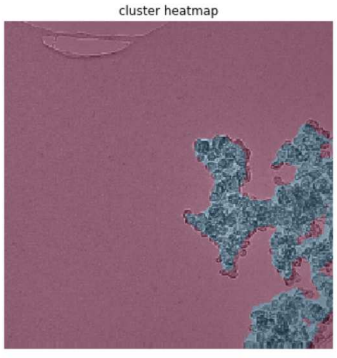




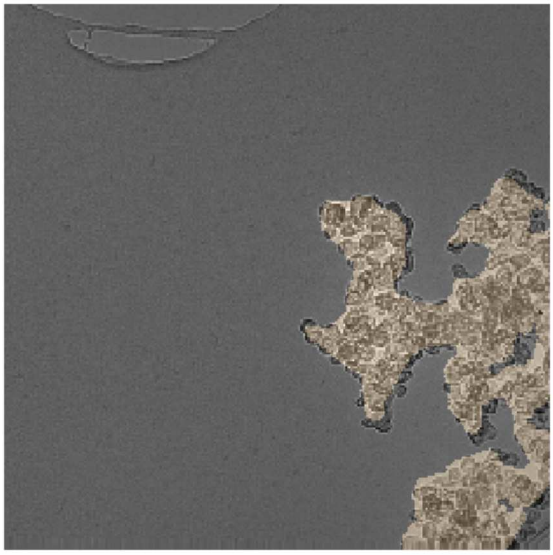
realxtals sa

In [115]:
```
n_clust = 2
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset7_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

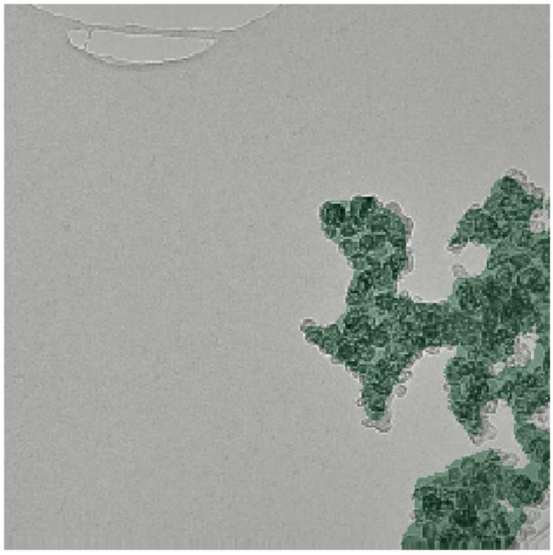
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 2 clusters...
Visualizing results...
```



In [116]:
```
plot_with_overlay(img, h, cmapname='magma', overlay_alpha=0.3, fig_size=(10,10))
```



In [98]:
```
plot_with_overlay(img, h, cmapname='Greens', overlay_alpha=0.3, fig_size=(10,10))
```

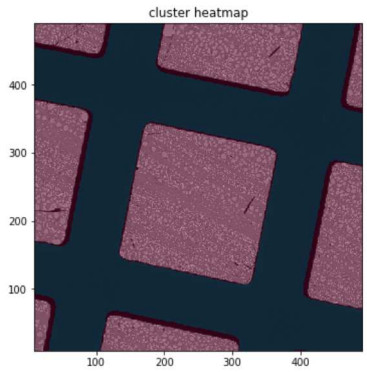


In [107]:
```
show_cluster_images(img, h, n_clust, fig_size=(10,10))
```

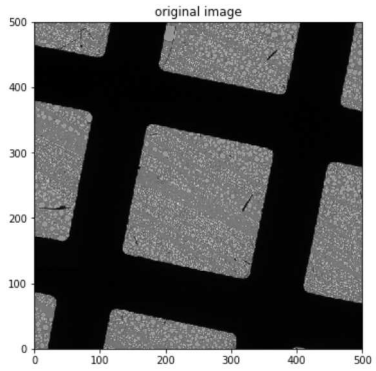
Asbestos LM

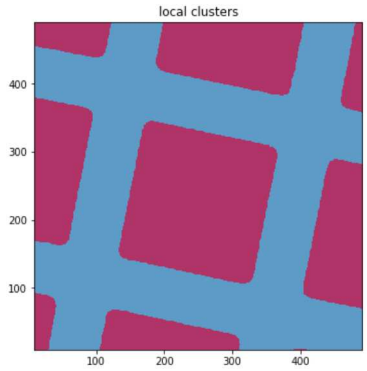
In [32]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/LM_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 2 clusters...
Visualizing results...
```

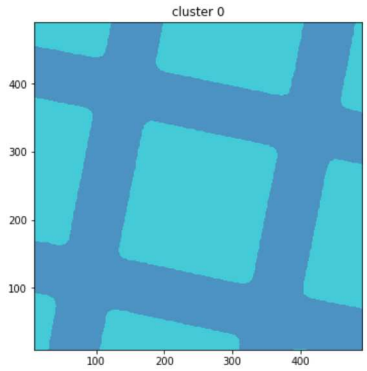


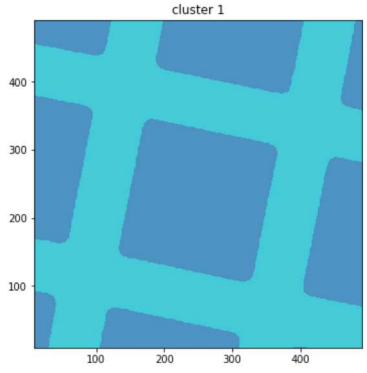

```
Showing diagnostic images...
```





In [33]:
```
show_cluster_images(img, h, n_clust)
```

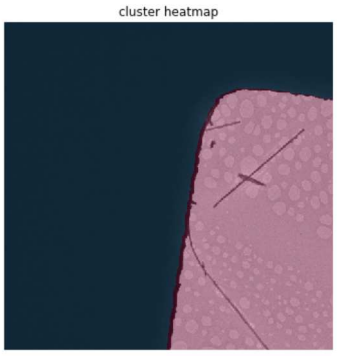




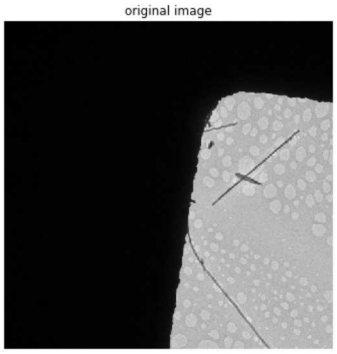
Asbestos SA

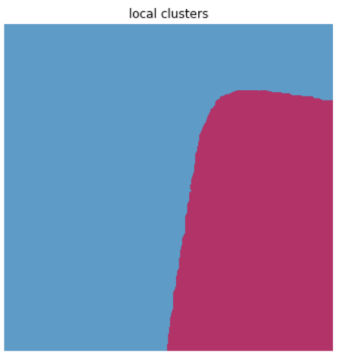
In [108]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/SA_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 2 clusters...
Visualizing results...
```

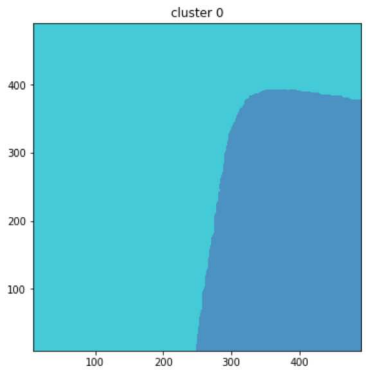


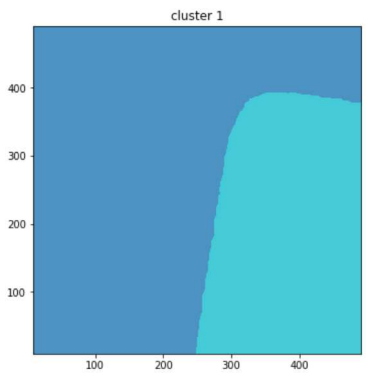

```
Showing diagnostic images...
```





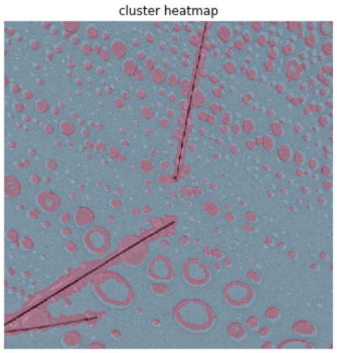
In [35]:
```
show_cluster_images(img, h, n_clust)
```



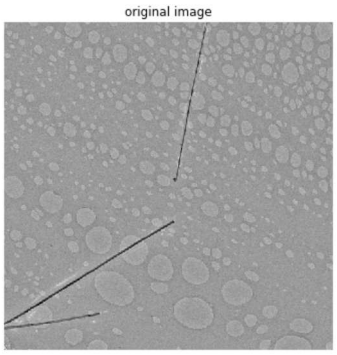


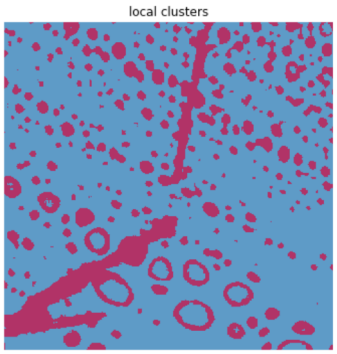
In [109]:
```
imgfilename = imgfiles[9]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 2 clusters...
Visualizing results...
```

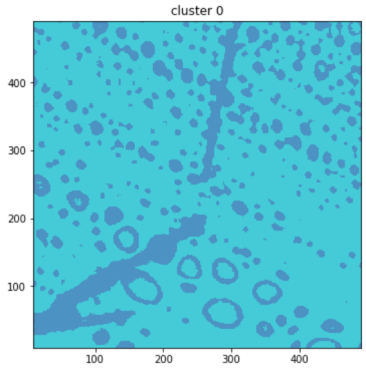


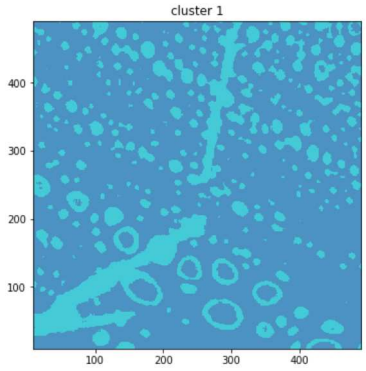

```
Showing diagnostic images...
```





In [37]:
```
show_cluster_images(img, h, n_clust)
```





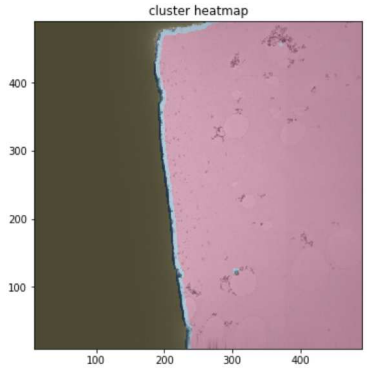
Try on multiple data sets - 3 Clusters

In [129]:
```
n_clust = 3
patch_size = (10,10)
```

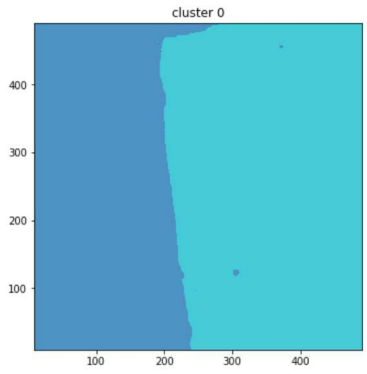
realxtals lm (hard)

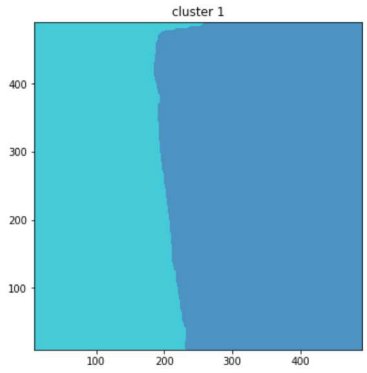
In [39]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset6_subset_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

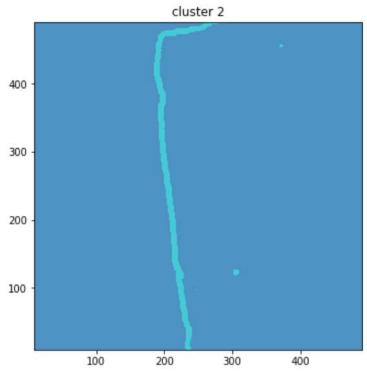
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 3 clusters...
Visualizing results...
```



In [40]:
```
show_cluster_images(img, h, n_clust)
```



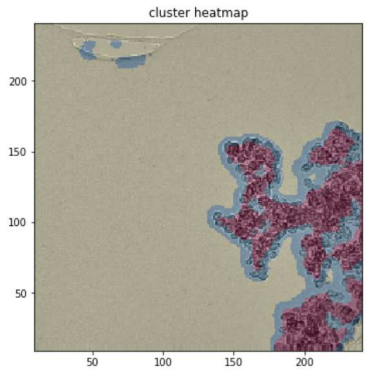




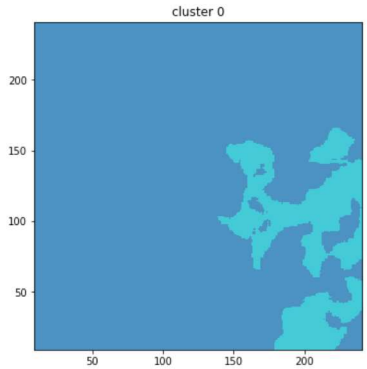
realxtals sa

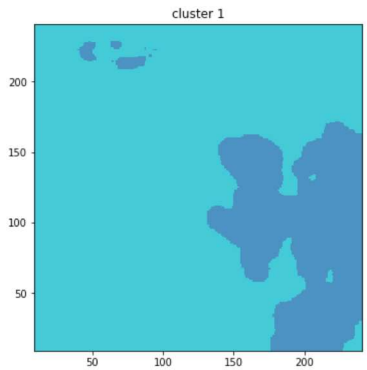
In [41]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset7_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

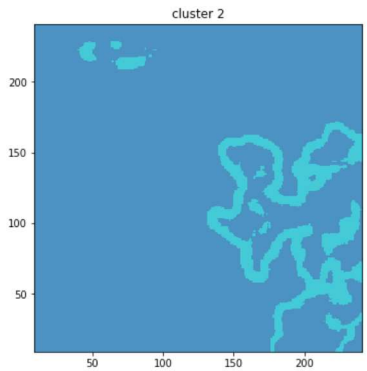
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 3 clusters...
Visualizing results...
```



In [42]:
```
show_cluster_images(img, h, n_clust)
```



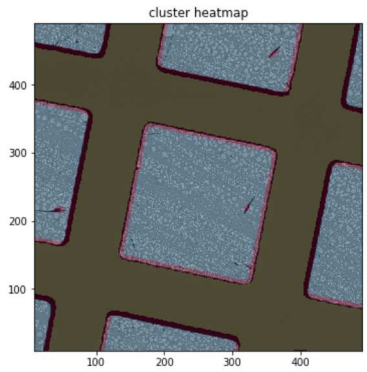




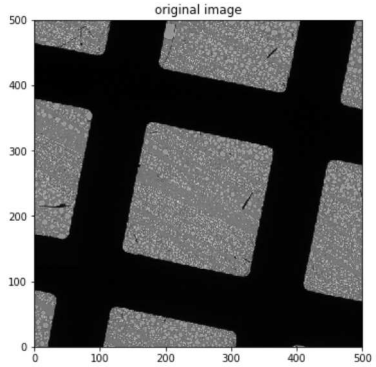
Asbestos LM

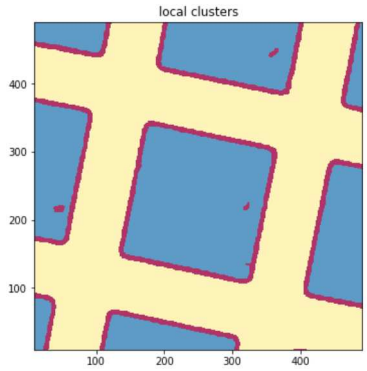
In [43]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/LM_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 3 clusters...
Visualizing results...
```

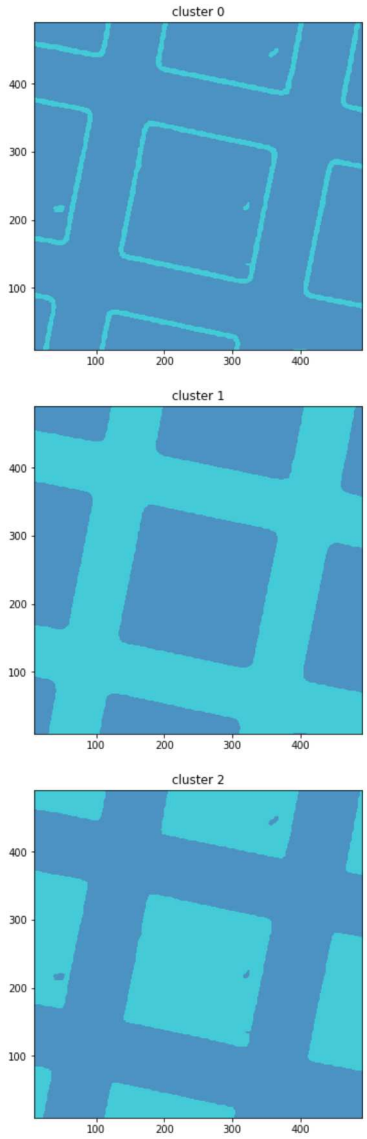



```
Showing diagnostic images...
```





In [44]:
```
show_cluster_images(img, h, n_clust)
```

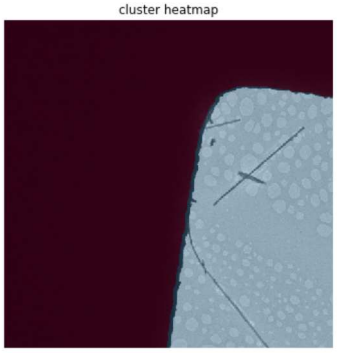


Asbestos SA

In [119]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/SA_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

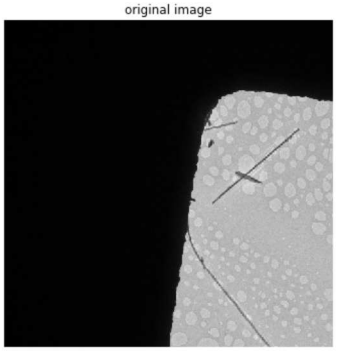
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 2 clusters...
Visualizing results...
```

cluster heatmap



```
Showing diagnostic images...
```

original image

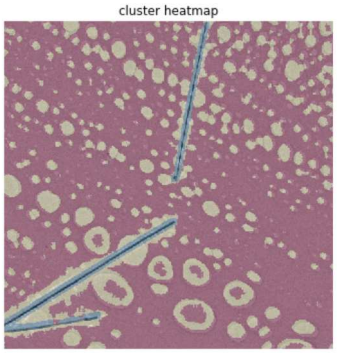


local clusters

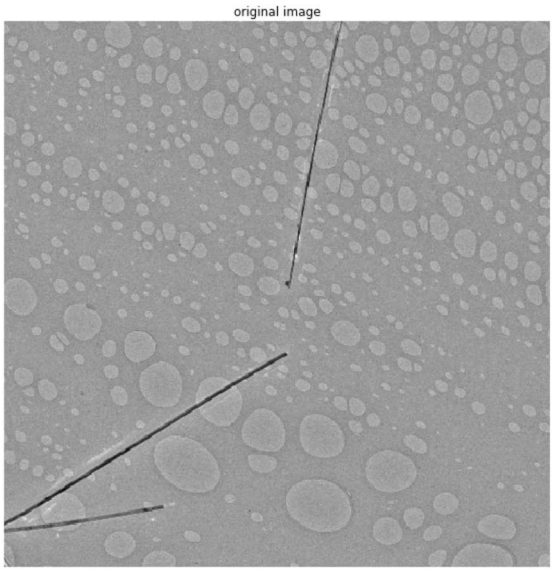
In [ ]:
```
show_cluster_images(img, h, n_clust)
```

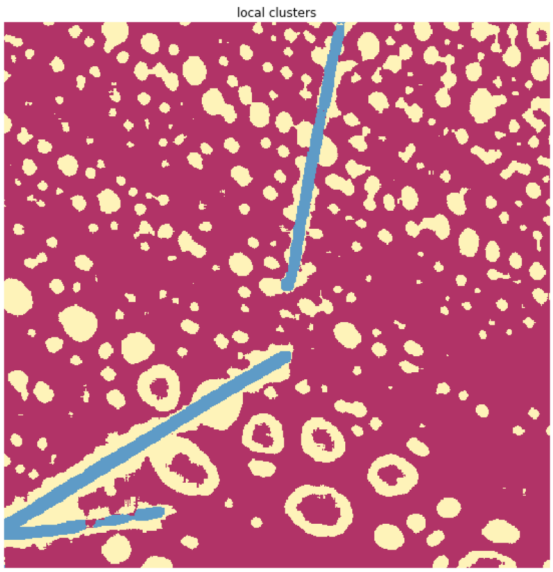
In [130]:
```
imgfilename = imgfiles[9]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4, fig_size=(10,10))
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 3 clusters...
Visualizing results...
```



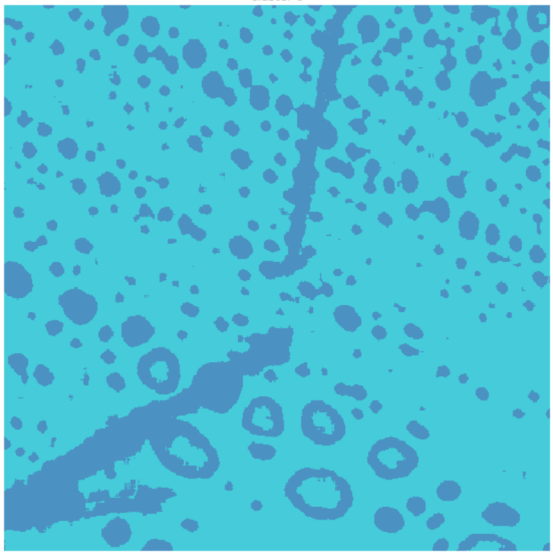

```
Showing diagnostic images...
```



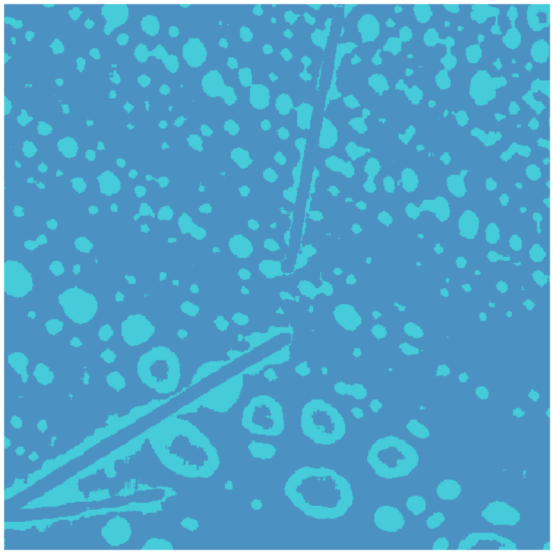


In [134]:
```
show_cluster_images(img, h, n_clust, fig_size=(10,10))
```

cluster 0

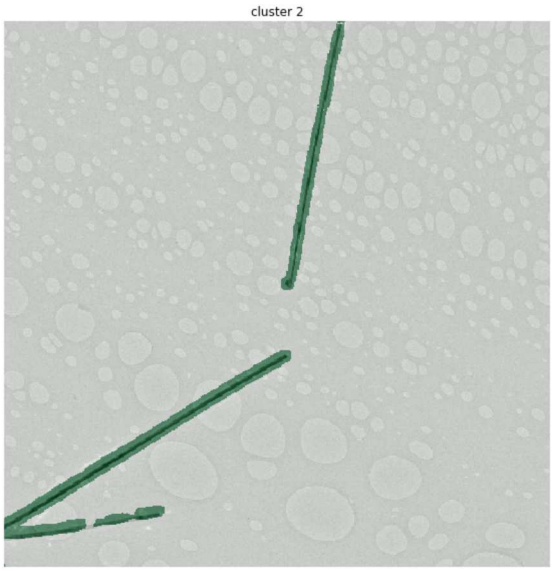


cluster 1

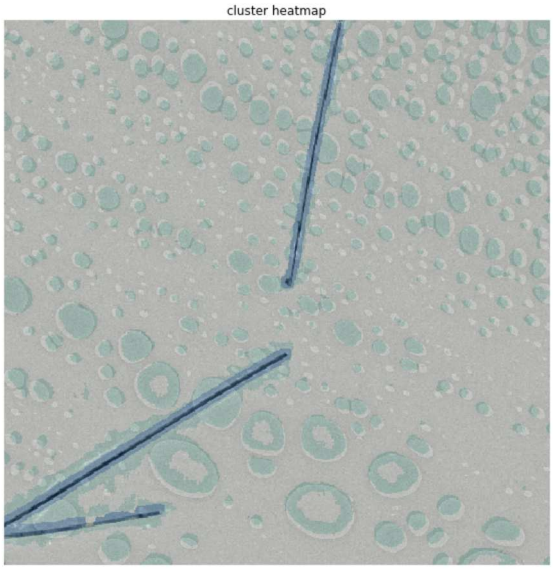


cluster 2

In [139]:
```
show_single_cluster_image(img, h, 2, opacity=0.5, cmapname='Greens', fig_size=(10,10))
```



In [159]:
```
plot_with_overlay(img, h, title='cluster heatmap', cmapname='GnBu', fig_size=(10,10))
```



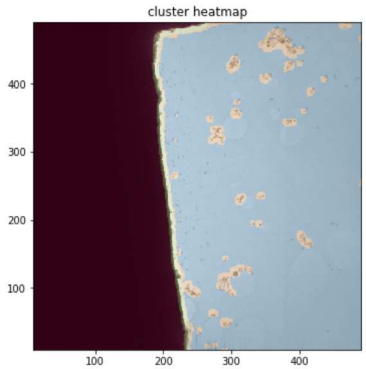
Try on multiple data sets - 4 Clusters

In [50]:
```
n_clust = 4
patch_size = (10,10)
```

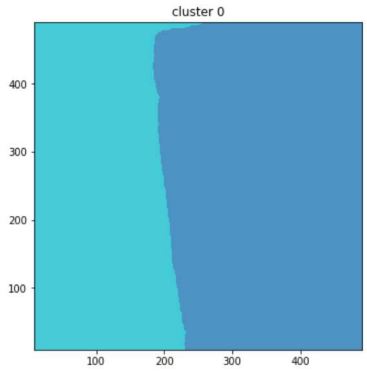
realxtals lm (hard)

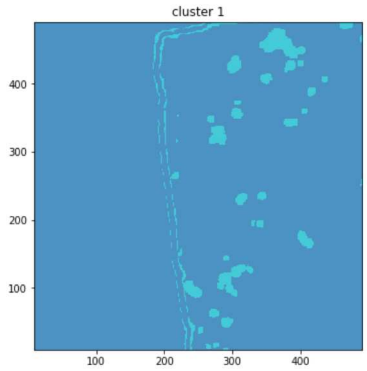
In [51]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset6_subset_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

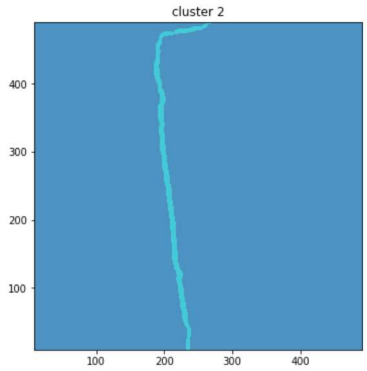
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 4 clusters...
Visualizing results...
```

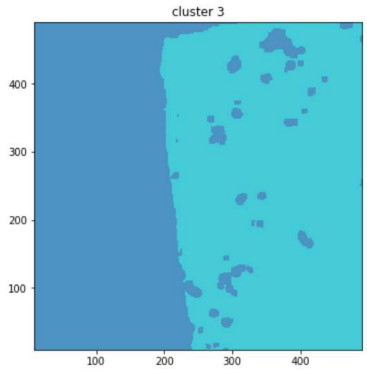


In [52]:
```
show_cluster_images(img, h, n_clust)
```





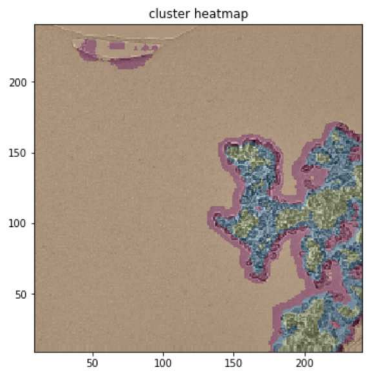




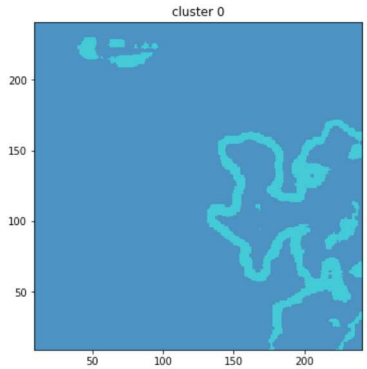
realxtals sa

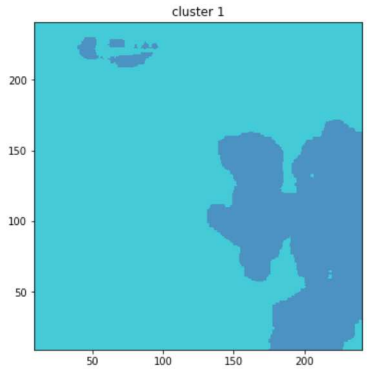
In [53]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset7_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

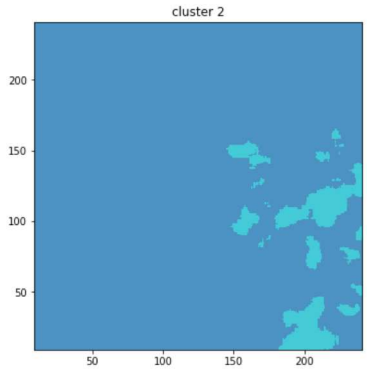
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 4 clusters...
Visualizing results...
```

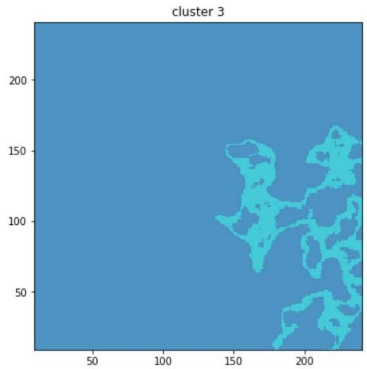


In [54]:
```
show_cluster_images(img, h, n_clust)
```





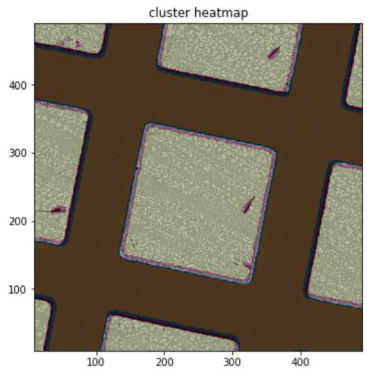




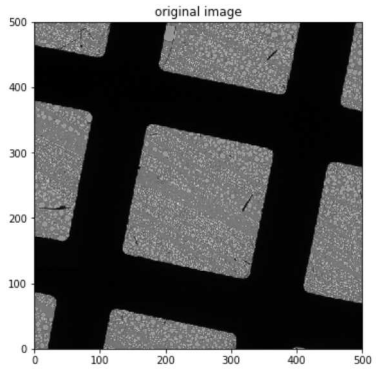
Asbestos LM

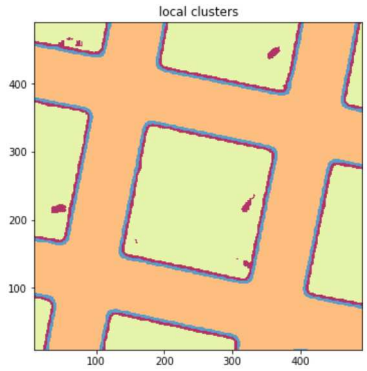
In [55]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/LM_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 4 clusters...
Visualizing results...
```

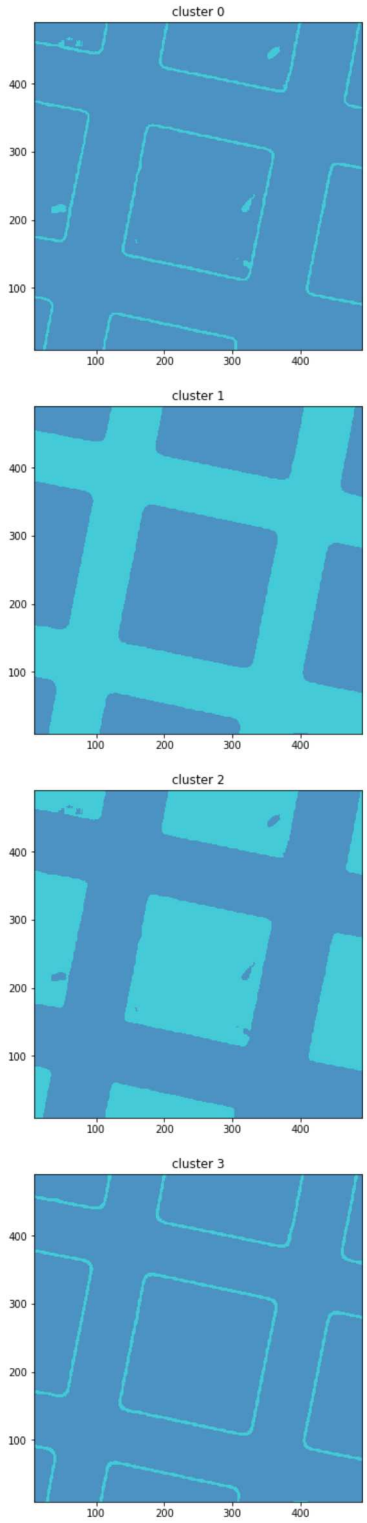



```
Showing diagnostic images...
```





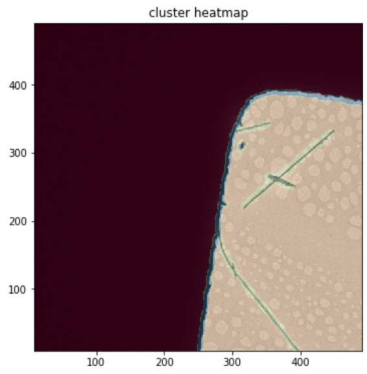
In [56]:
```
show_cluster_images(img, h, n_clust)
```



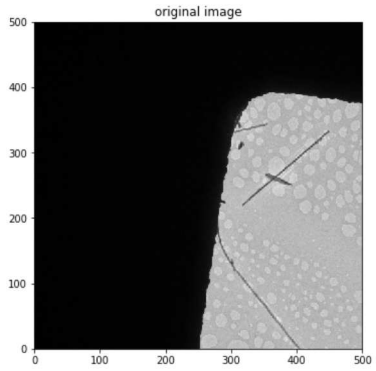
Asbestos SA

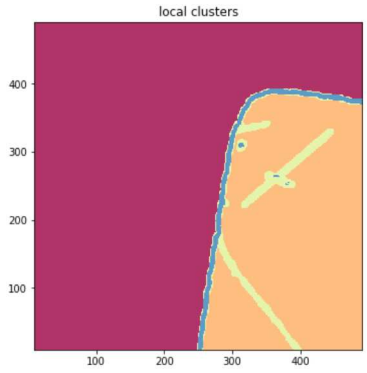
In [57]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/SA_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 4 clusters...
Visualizing results...
```

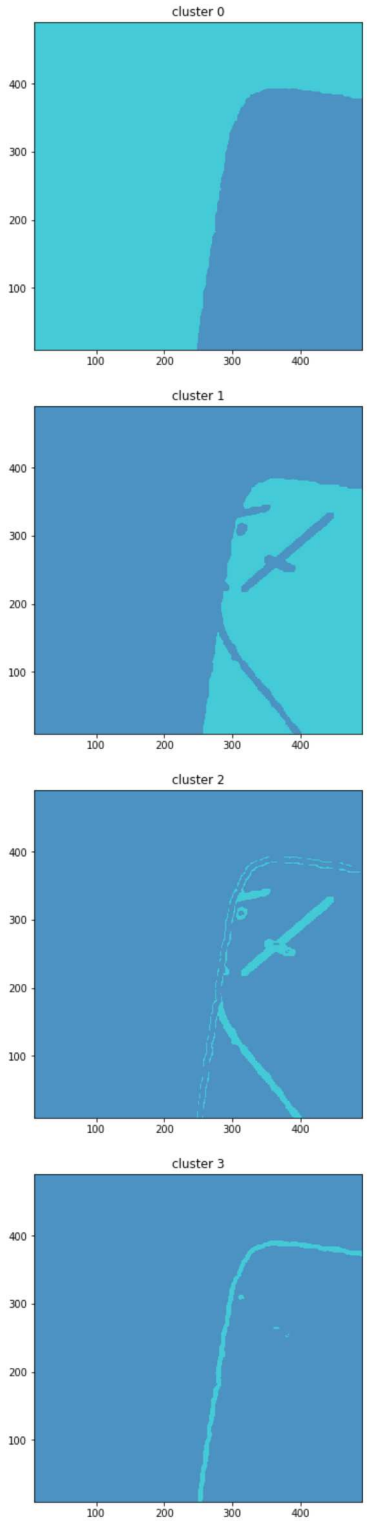



```
Showing diagnostic images...
```



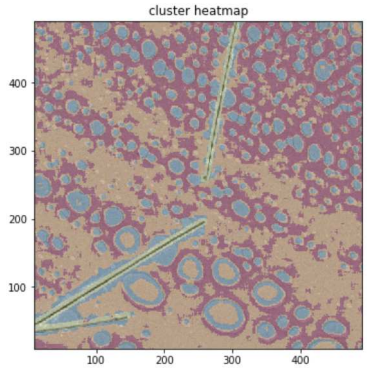


In [58]:
```
show_cluster_images(img, h, n_clust)
```

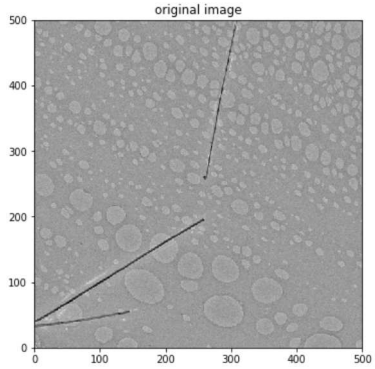


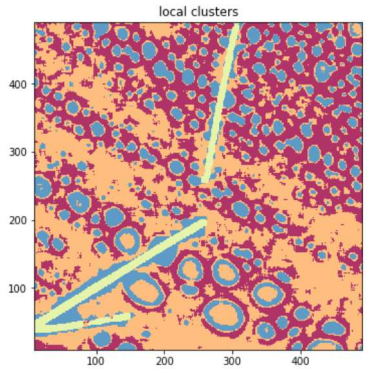
In [59]:
```
imgfilename = imgfiles[9]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 4 clusters...
Visualizing results...
```

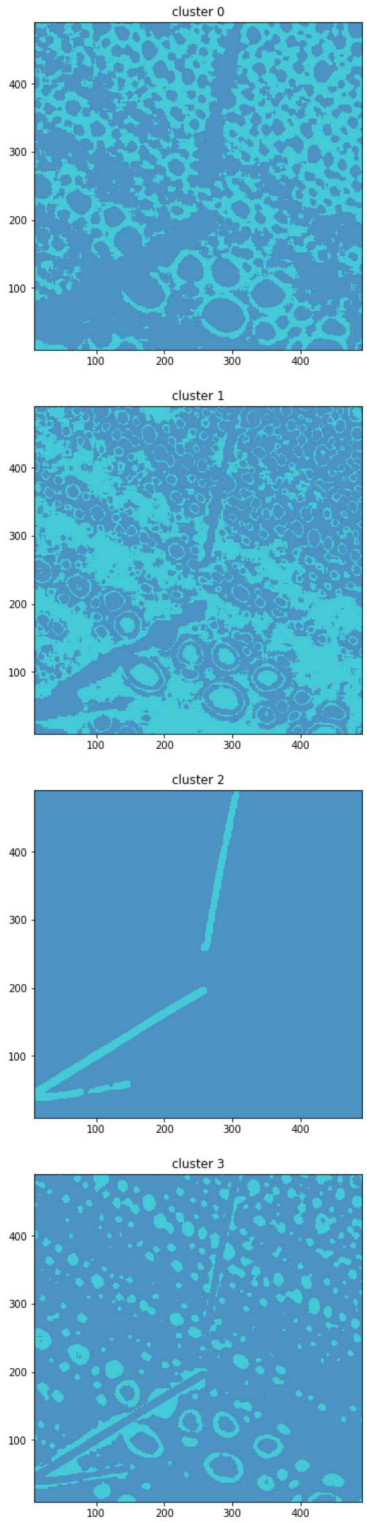



```
Showing diagnostic images...
```





In [60]:
```
show_cluster_images(img, h, n_clust)
```



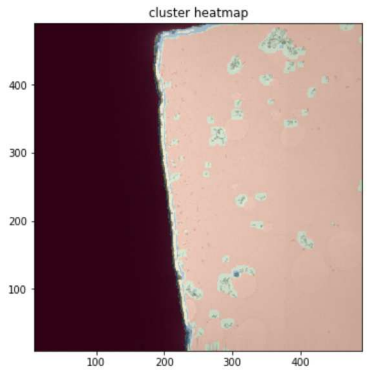
Try on multiple data sets - 5 Clusters

In [61]:
```
n_clust = 5
patch_size = (10,10)
```

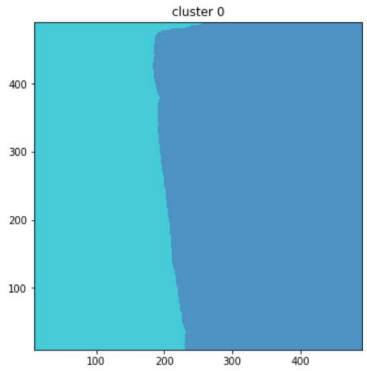
realxtals lm (hard)

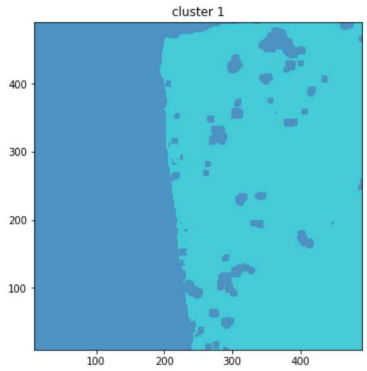
In [62]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset6_subset_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

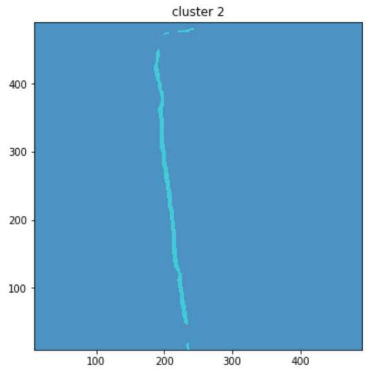
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 5 clusters...
Visualizing results...
```

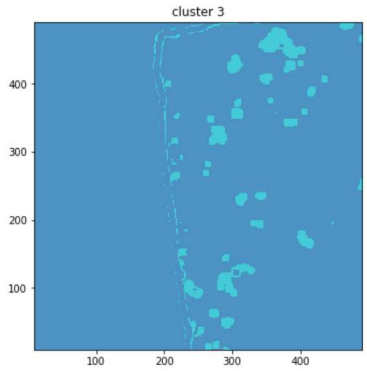


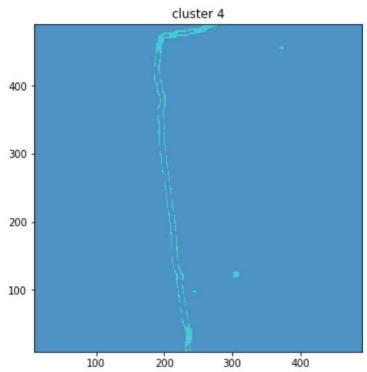
In [63]:
```
show_cluster_images(img, h, n_clust)
```







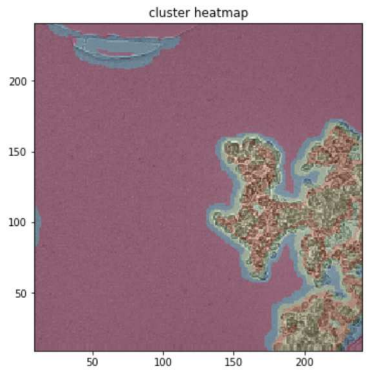




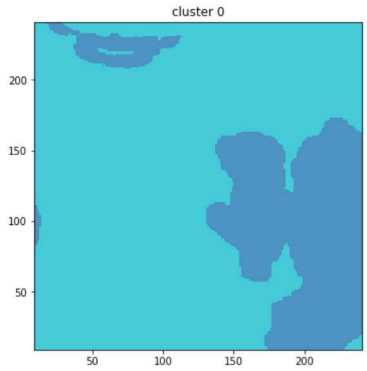
realxtals sa

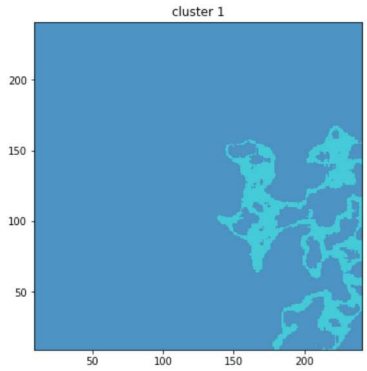
In [64]:
```
imgfiles = imgutils.getimgfiles('../data/Crystals_Apr_12/Tileset7_1K','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True, downscale_factor=4)
```

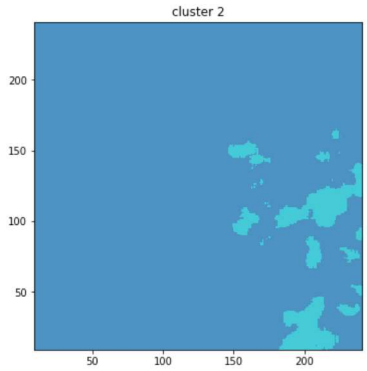
```
Importing image(s)s...
Extracting features... 100 %
Clustering into 5 clusters...
Visualizing results...
```

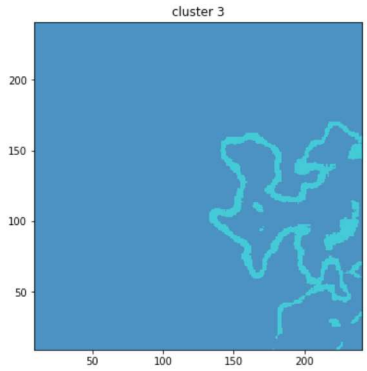


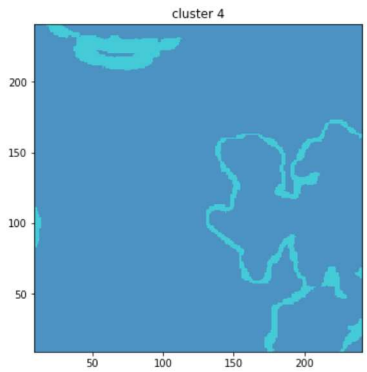
In [65]:
```
show_cluster_images(img, h, n_clust)
```







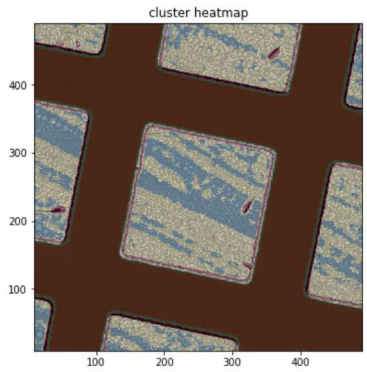




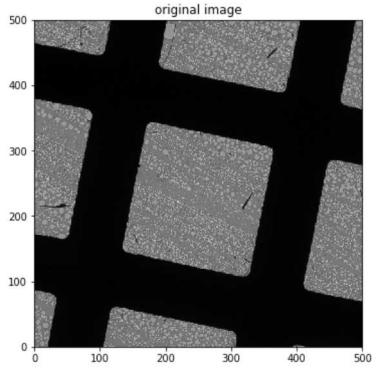
Asbestos LM

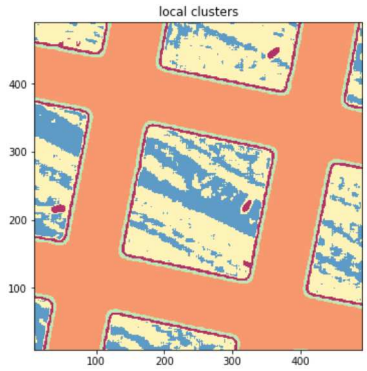
In [66]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/LM_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 5 clusters...
Visualizing results...
```

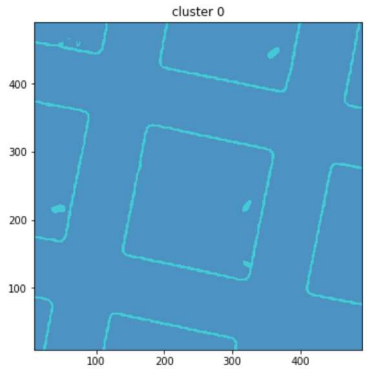


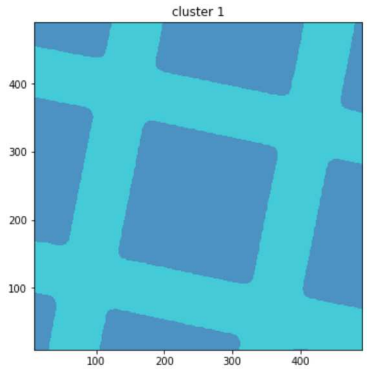

```
Showing diagnostic images...
```

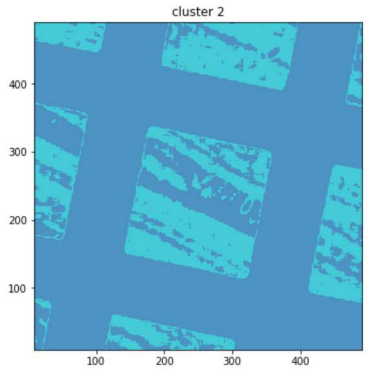


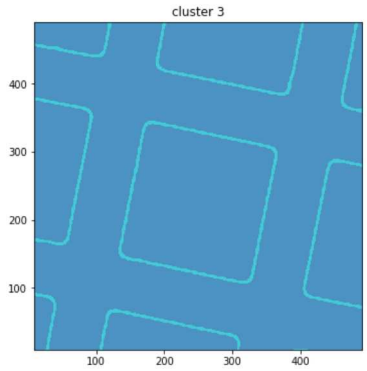


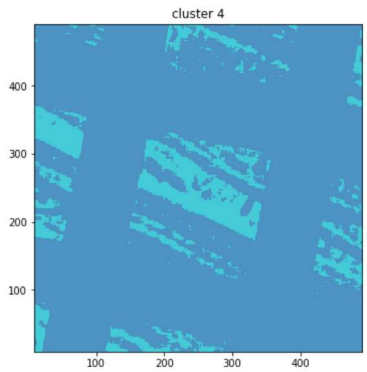
In [67]:
```
show_cluster_images(img, h, n_clust)
```







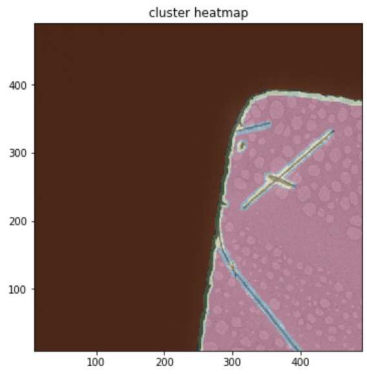




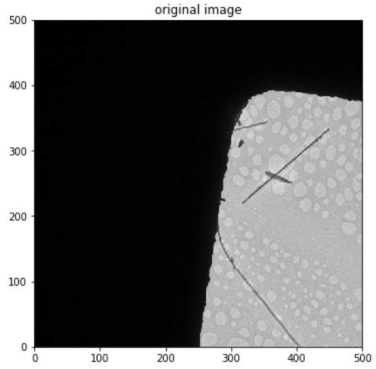
Asbestos SA

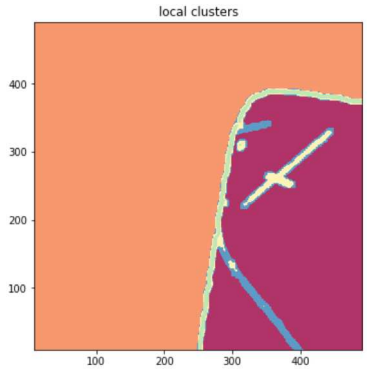
In [68]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/SA_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 5 clusters...
Visualizing results...
```

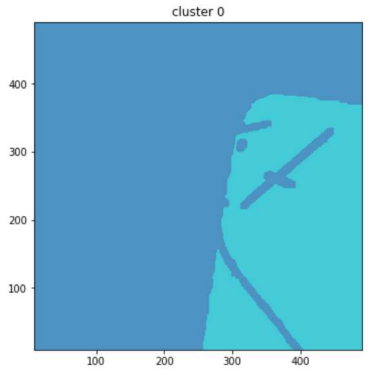


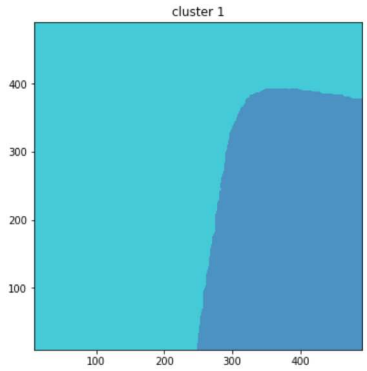

```
Showing diagnostic images...
```

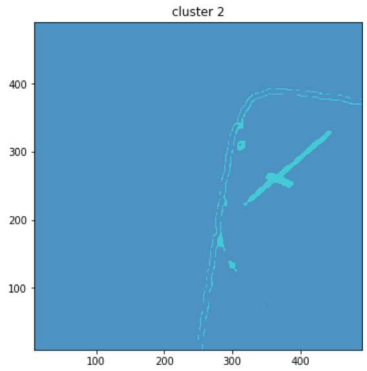


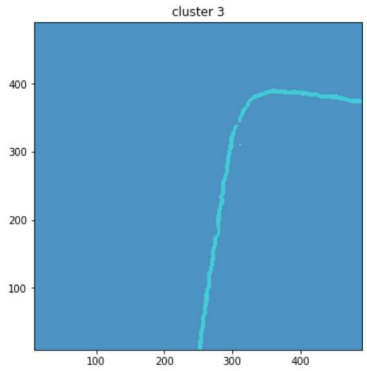


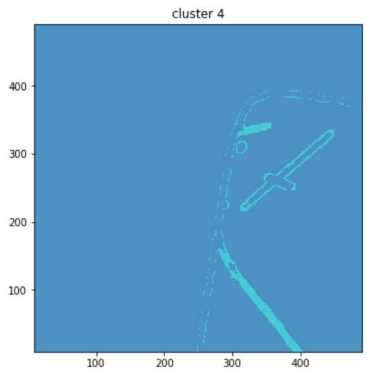
In [69]:
```
show_cluster_images(img, h, n_clust)
```





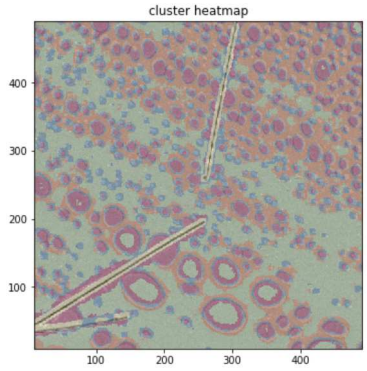




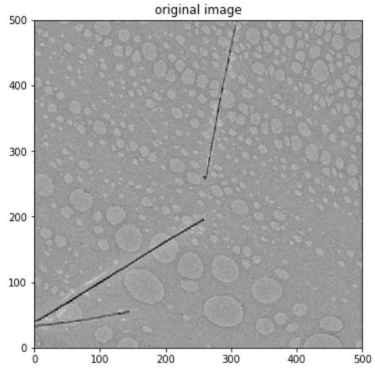


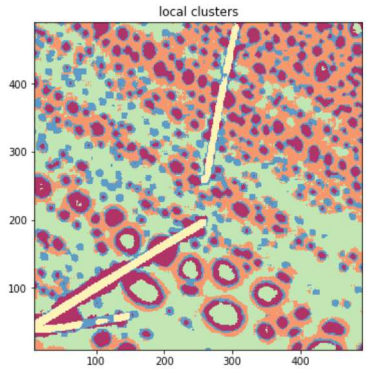
In [70]:
```
imgfilename = imgfiles[9]
img, h = run_new_pipeline(imgfilename, patch_size, n_clust, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 5 clusters...
Visualizing results...
```

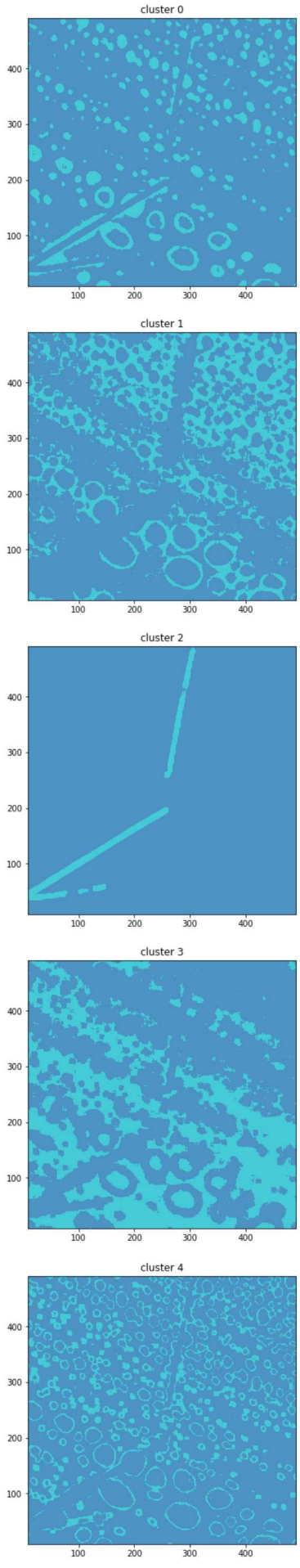



```
Showing diagnostic images...
```





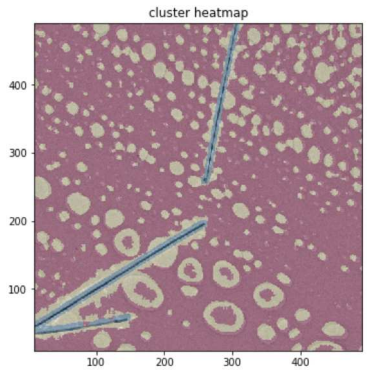
In [71]:
```
show_cluster_images(img, h, n_clust)
```



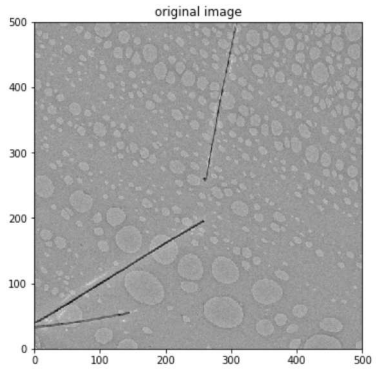
3 clusters, choose one as overlay

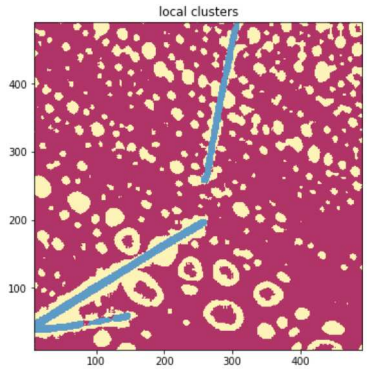
In [72]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/SA_Tileset','.tif')
imgfilename = imgfiles[9]
img, h = run_new_pipeline(imgfilename, patch_size, 3, return_cluster_image=True,show_diagnostics=True, downscale_factor=4)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 3 clusters...
Visualizing results...
```

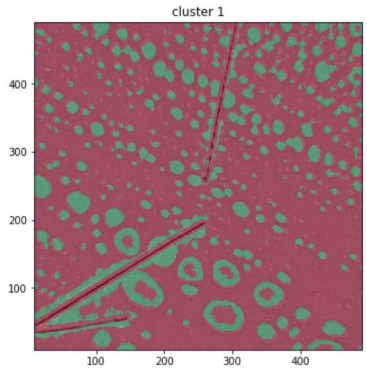



```
Showing diagnostic images...
```





In [73]:
```
show_single_cluster_image(img, h, 1)
```

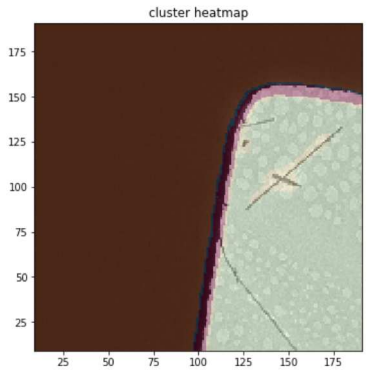


run asbestos SA with black once more but with hierarchical:

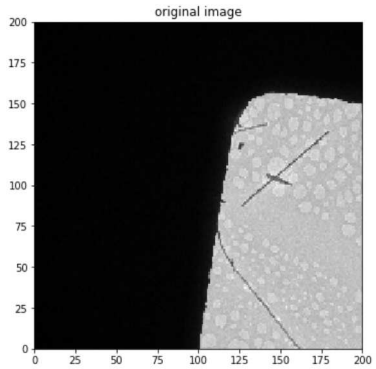
(needs more downscaling as hierarchical cannot run on large sets)

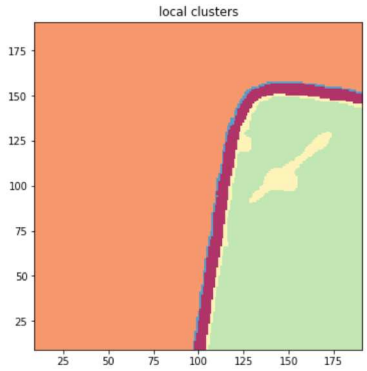
In [74]:
```
imgfiles = imgutils.getimgfiles('../data/Asbestos_Aug30/SA_Tileset','.tif')
imgfilename = imgfiles[0]
img, h = run_new_pipeline(imgfilename, (10,10), n_clust, return_cluster_image=True, algorithm='hierarchical', show_diagnostics=True, downscale_factor=10)
```

```
Importing image(s)s...
Extracting features... 100 %
Clustering into 5 clusters...
Visualizing results...
```




```
Showing diagnostic images...
```





In [75]:
```
show_cluster_images(img, h, n_clust)
```