In [8]:

```
# Common useful packages
import matplotlib.pyplot as plt
from matplotlib import colors
import matplotlib.patches as mpatches
import importlib
import numpy as np
import copy
import spectral
import pickle
import os

# local imports
import hyperspectral_gta_data as hsi_data
from hsiViewer import hsi_viewer_layers as hvl
from hsiViewer import hsi_viewer_ROI as hvr
from hsiViewer import hsi_viewer as hsv
import microscene_functions as msf

# set the directory where data will be downloaded and accessed
# (This directory will be saved in a config file and used each time you use hypersp
# so you don't need to run the 'set_data_dir' function every time you use hyperspec
hsi_data.set_data_dir('C:/spectral_data')
```

Part 1: Open the image and load the data as a numpy array.

In [9]:

```
# Here are the datasets available through the hyperspectral_gta_data package
hsi_data.available_datasets()
```

```
Available Images:
   WashingtonDC
   MicroscenePolymers
   DetectionImagery_CookeCity_RIT
   VegBaccharisUPWINS
   PaintDetectionUPWINS
   indian_pines
   pavia_centre
   pavia_university
   AVIRIS_Cuprite
   ENMAP_Cuprite
Available Spectral Libraries:
   GHISACONUS_2008_001_speclib
   Microscene_Polymers_Library
   UPWINS_4_16_2024
   Detection_Library_Cooke_City_RIT
   USGS_splib07b
```

In [10]:

```
# To download and use the polymers image:
hsi_data.download('VegBaccharisUPWINS')
fname = hsi_data.get_fname('VegBaccharisUPWINS')
```

```
Available Files:
C:\spectral_data\spectral_images\Vegetation_Baccharis_halmifolia_UPWINS\Morven_Bacch
aris_h_or_ref
C:\spectral_data\spectral_images\Vegetation_Baccharis_halmifolia_UPWINS\Morven_Bacch
aris_h_or_ref.aux.xml
C:\spectral_data\spectral_images\Vegetation_Baccharis_halmifolia_UPWINS\Morven_Bacch
aris_h_or_ref.hdr
C:\spectral_data\spectral_images\Vegetation_Baccharis_halmifolia_UPWINS\Morven_Bacch
aris_h_or_ref_tiff.tfw
C:\spectral_data\spectral_images\Vegetation_Baccharis_halmifolia_UPWINS\Morven_Bacch
aris_h_or_ref_tiff.tif
Primary Image Filename:
C:/spectral_data\spectral_images\Vegetation_Baccharis_halmifolia_UPWINS/Morven_Bacch
aris_h_or_ref
```

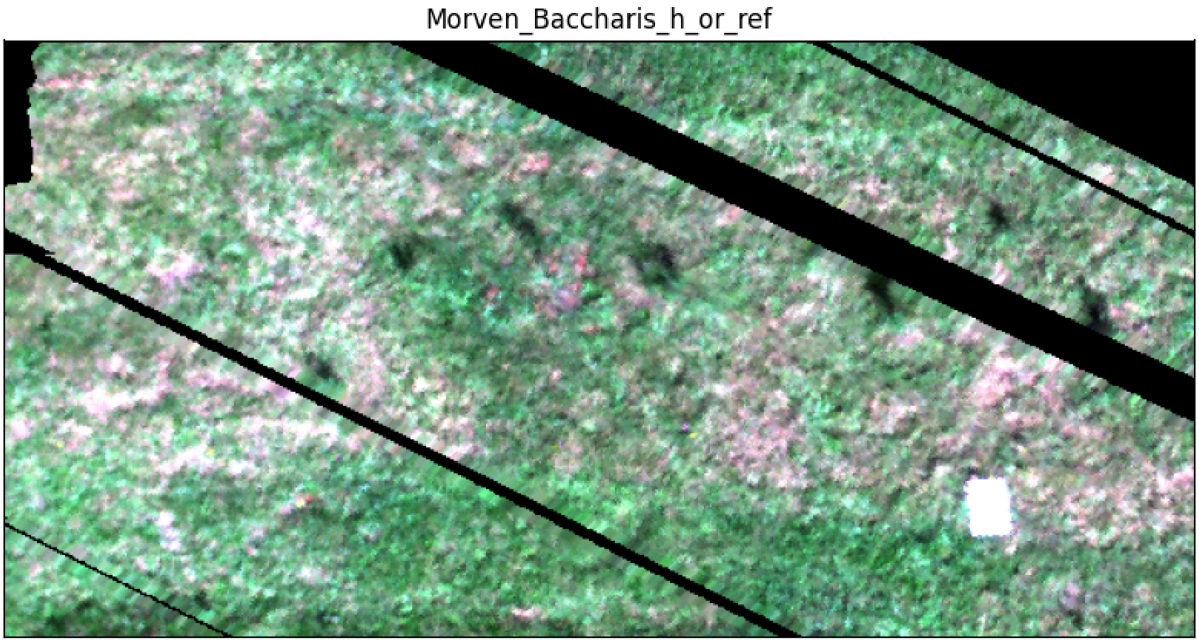
In [11]:

```python
# Open the image and read into an array
im = spectral.envi.open(fname+'.hdr', fname)
wl = np.asarray(im.bands.centers)
# Load the image into memory
im.Arr = im.load()
print(f'Shape of Im.Arr = {im.Arr.shape}')
im.List = np.reshape(im.Arr, (im.nrows*im.ncols, im.nbands))
print(f'Shape of im.List = {im.List.shape}')
dataList = im.List[np.sum(im.List, axis=1)>0, :]
print(f'Shape of dataList = {dataList.shape}')
nr = im.nrows
nc = im.ncols
nb = im.nbands
```

```
Shape of Im.Arr = (300, 600, 343)
Shape of im.List = (180000, 343)
Shape of dataList = (161247, 343)
```

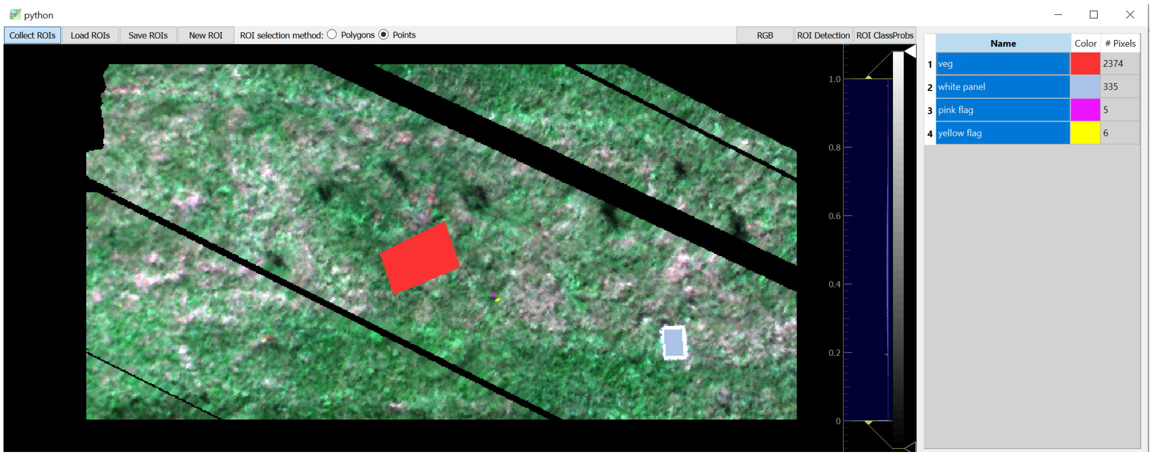
In [12]:

```python
# Plot an RGB image of the hyperspectral image
# make the RGB image from the hyperspectral image (select RGB bands and applying st
RGB_image = msf.make_rgb(im.Arr, wl, stretch = [2,98])
# Plot the image
plt.figure(figsize=(10,10))
plt.imshow(RGB_image)
plt.xticks([])
plt.yticks([])
plt.title(os.path.basename(fname));
```

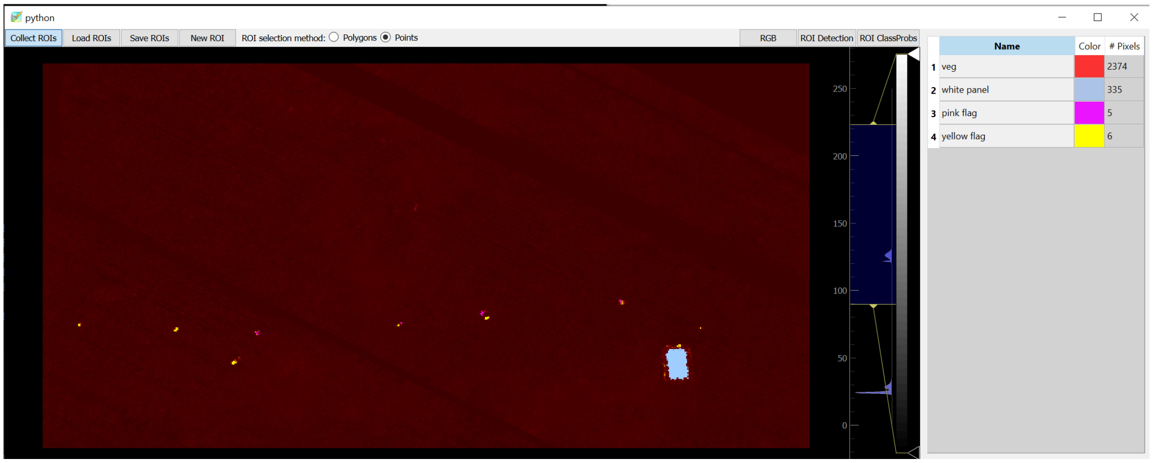


Part 2: Create and save ROIs.

NOTE: To get a class probabilities image, highlight all ROIs you want to use by clicking on the 'Name' cell above all the ROI name cells, then click the 'ROI ClassProbs' button as shown below. If you highlight the full rows (instead of just the name), an error will result.



	Name	Color	# Pixels
1	veg	🟥	2374
2	white panel	🟦	335
3	pink flag	🟪	5
4	yellow flag	🟨	6



ROI selection method: ○ Polygons ● Points

	Name	Color	# Pixels
1	veg	🟥	2374
2	white panel	🟦	335
3	pink flag	🟪	5
4	yellow flag	🟨	6

In [13]:
```
importlib.reload(hvr)
hvr.viewer(im, stretch=[0,99.5], rotate=True)
```

Out[13]:
```
<hsiViewer.hsi_viewer_ROI.viewer at 0x2d6bf30dd80>
```

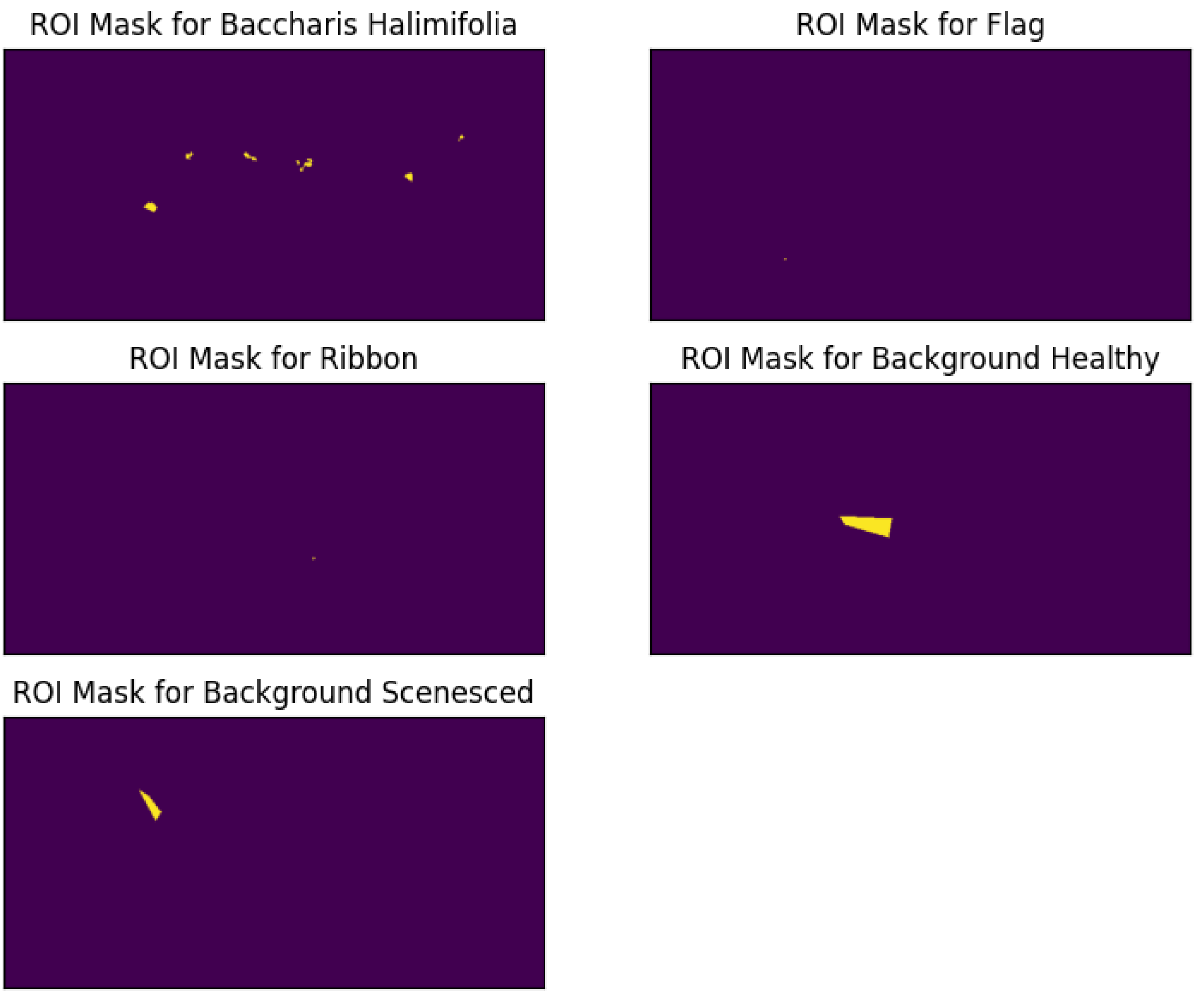
Part 3: Initial Evaluation of ROIs

In [14]:
```
# Unpickling the dictionary
fname = 'C:\\spectral_data\\spectral_images\\UPWINS Baccharis Halimifolia.pkl'
with open(fname, 'rb') as f:
    roiData = pickle.load(f)
```

In [15]:
```
df = roiData.df # a DataFrame holding all the data for the ROIs
roi_names = roiData.names # the names of the ROIs
roi_colors = roiData.colors # a Python dictionary of colors, where colors[class_nam
roi_masks = roiData.masks # a Python dictionary of location masks, where colors[cla
nClasses = len(roi_names)

# Notes:
#   Each mask is an array of bollean values of the same dimensions as the image,
#   and True represents a pixel in the ROI while False represents not in the ROI.
```

In [16]:
```
# determine how many rows of images (assuming each row has 2 masks)
nRowsPlots = int(np.ceil(nClasses/2)) # (change size of figure as needed)
plt.figure(figsize=(8, nRowsPlots*2))
for i in range(nClasses):
    name = roi_names[i]
    # create the subplot
    plt.subplot(nRowsPlots, 2, i+1)
    plt.imshow(roi_masks[name])
    plt.xticks([])
    plt.yticks([])
    plt.title(f'ROI Mask for {name}');
    plt.tight_layout()
```



ROI Mask for Baccharis Halimifolia

ROI Mask for Flag

ROI Mask for Ribbon

ROI Mask for Background Healthy

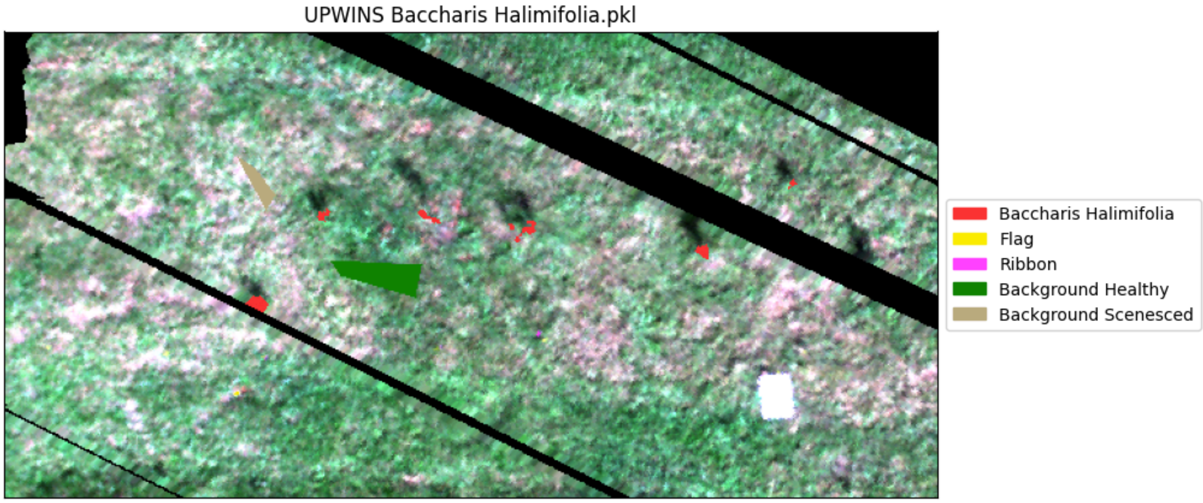
ROI Mask for Background Scenesced

In [17]:

```
# Examine the locations
# create an RGB image
RGB_image_with_ROIs = msf.make_rgb(im.Arr, wl, stretch = [2,98])
# Add the ROI locations
for name in roi_names:
    # get the color for this ROI, converted from hex to (r,g,b)
    color = colors.to_rgb(roi_colors[name])
    mask = roi_masks[name]
    # set the pixel values for this ROI to (0,0,0)
    for i in range(3):
        RGB_image_with_ROIs[:,:,i] = RGB_image_with_ROIs[:,:,i]*(mask==False)
    # set the pixel values for this ROI to the color for this ROI
    for i in range(3):
        RGB_image_with_ROIs[:,:,i] = RGB_image_with_ROIs[:,:,i] + mask*color[i]

# Plot the image
plt.figure(figsize=(10,10))
plt.imshow(RGB_image_with_ROIs)
plt.xticks([])
plt.yticks([])
plt.title(os.path.basename(fname));
# Create legend patches: each patch is a colored box with the ROI name.
legend_patches = [mpatches.Patch(color=roi_colors[name], label=name) for name in ro
# Add the legend to the right of the image
plt.legend(handles=legend_patches, loc='center left', bbox_to_anchor=(1, 0.5))
```

```
Out[17]: <matplotlib.legend.Legend at 0x2d6c69fa510>
```



```
In [18]: # Examine the dataframe
         df
```

Out[18]:

	Name	Color	Pixel_x	Pixel_y	399.096008	400.854004	402.612	404.369995
0	Baccharis Halimifolia	#fa3232	175	155	0.0282	0.0368	0.0366	0.0344
1	Baccharis Halimifolia	#fa3232	174	156	0.0282	0.0368	0.0366	0.0344
2	Baccharis Halimifolia	#fa3232	175	156	0.0334	0.0369	0.0309	0.0318
3	Baccharis Halimifolia	#fa3232	173	157	0.0282	0.0368	0.0366	0.0344
4	Baccharis Halimifolia	#fa3232	174	157	0.0332	0.0369	0.0311	0.0319
...	...	...	...	...	...	...	...	...
249	Background Scenesced	#baae80	108	172	0.0547	0.0529	0.0493	0.0510
250	Background Scenesced	#baae80	104	173	0.0451	0.0459	0.0377	0.0331
251	Background Scenesced	#baae80	105	173	0.0460	0.0515	0.0426	0.0450
252	Background Scenesced	#baae80	106	173	0.0425	0.0466	0.0452	0.0468
253	Background Scenesced	#baae80	105	174	0.0406	0.0417	0.0425	0.0402

1410 rows × 347 columns