Accessing VLASS in the VO

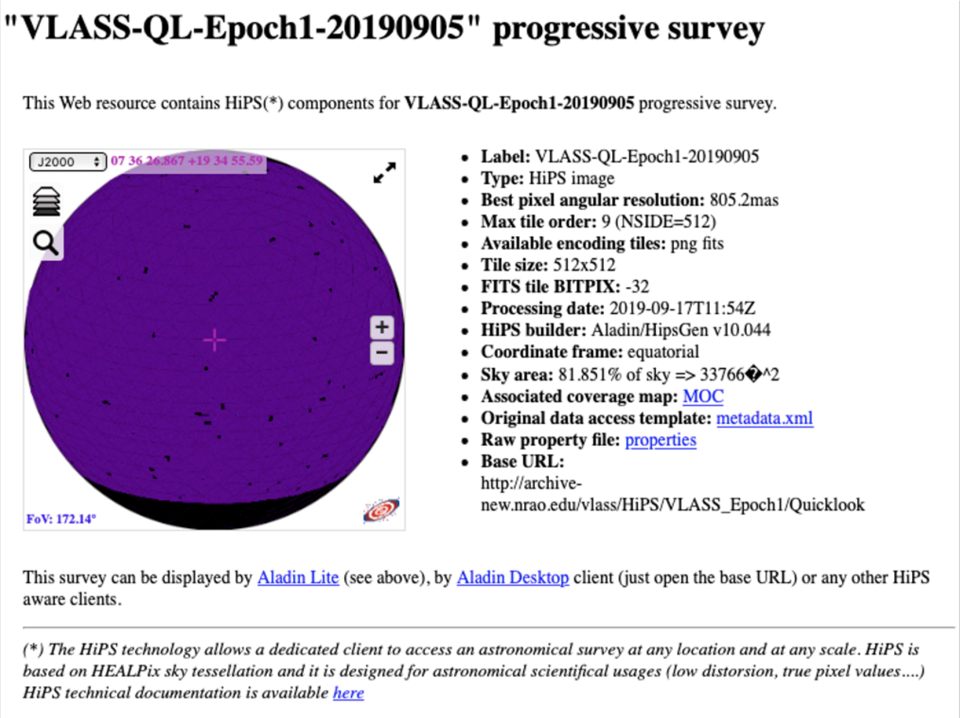
Mark Lacy for the VLASS team

Overview

- The VLA Sky Survey is a three epoch, 2.5” resolution survey at 3GHz made with the VLA covering the whole sky above Dec. -40 degrees.
- There are 34,000 images of each type (Quick Look, Single Epoch continuum and cubes over three epochs, plus combined epochs).
- Really need a good way to browse and get data/cutouts in bulk!
- The image data products are served from both NRAO and CADC, and catalogs via CDS.
- NRAO’s VO services are still being developed, so we use CADC and CDS for scripted access to the data products

HiPS service

(Will be registered soon!)



"VLASS-QL-Epoch1-20190905" progressive survey

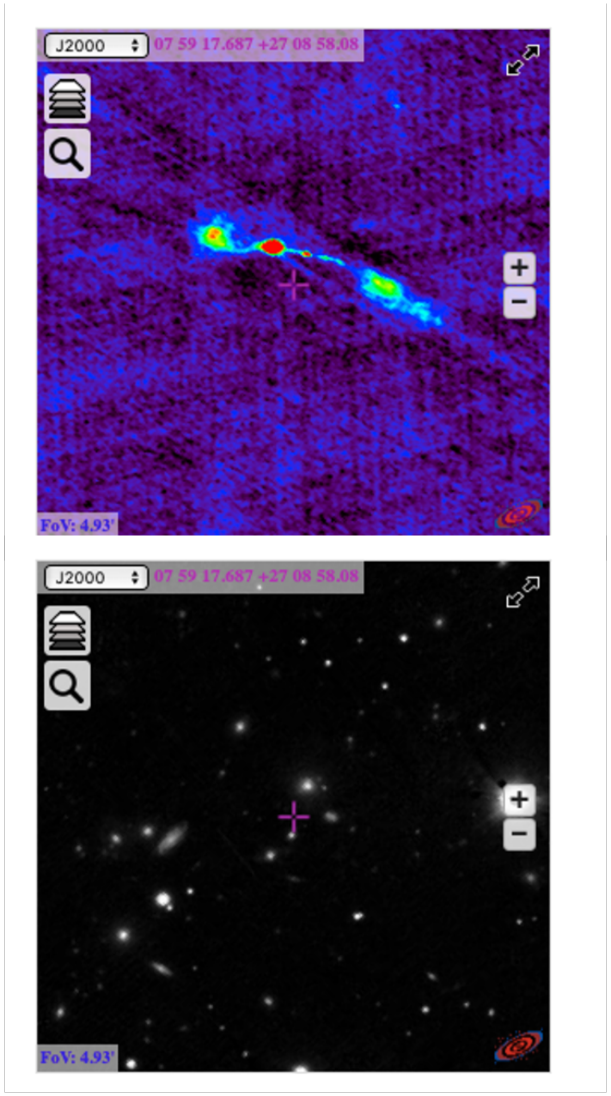
This Web resource contains HiPS(*) components for **VLASS-QL-Epoch1-20190905** progressive survey.

- **Label:** VLASS-QL-Epoch1-20190905
- **Type:** HiPS image
- **Best pixel angular resolution:** 805.2mas
- **Max tile order:** 9 (NSIDE=512)
- **Available encoding tiles:** png fits
- **Tile size:** 512x512
- **FITS tile BITPIX:** -32
- **Processing date:** 2019-09-17T11:54Z
- **HiPS builder:** Aladin/HipsGen v10.044
- **Coordinate frame:** equatorial
- **Sky area:** 81.851% of sky => 33766�^2
- **Associated coverage map:** MOC
- **Original data access template:** metadata.xml
- **Raw property file:** properties
- **Base URL:** http://archive-new.nrao.edu/vlass/HiPS/VLASS_Epoch1/Quicklook

This survey can be displayed by Aladin Lite (see above), by Aladin Desktop client (just open the base URL) or any other HiPS aware clients.

(*) The HiPS technology allows a dedicated client to access an astronomical survey at any location and at any scale. HiPS is based on HEALPix sky tessellation and it is designed for astronomical scientifical usages (low distorsion, true pixel values....) HiPS technical documentation is available here

archive-new.nrao.edu/vlass/HiPs



Scripted query of multiple sources using SODA

- The pyVO python package can be used to make image and catalog searches, and also TAP queries (see https://gitlab.nrao.edu/mlacy/vlass_vo)
- Focus on extracting cutouts from a list of sources via the CADC’s SODA service.
- Steps:

1. Import the source list as csv
2. Create tuples of RA, Dec, radius in degrees and make searches for the cutouts: cutout='CIRCLE='+str(poslist[i][0])+'%20'+str(poslist[i][1])+'%20'+str(poslist[i][2])
3. Run an SIA2 search on the position and VLASS collection: sia2_return.append(vo.dal.imagesearch2("https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/sia/v2query",pos=poslist[i],collection="VLASS"))
4. Use datalink to get the link to the primary product - datalink1=next(row.getdatalink().bysemantics('#this’)),
5. *Fudgy bit* – split out the link url from the datalink description (dataset=datalink1['description'].split(' ')[1]), sanitize it (urllib.parse.quote(dataset,safe=’’) and concatenate it with the base path to the SODA service (sodaurls.append(pathurl+path+'&'+cutout)

In [6]:

```
#The SODA service path was obtained from https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops
#To identify the primary image data ('#this' in the semantics) we can use the semantics and just get the image data
pathurl='https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/sync?ID='
sodaurls=[]
vlassimgs=[]
for i in range(nsrc):
    cutout='CIRCLE='+str(poslist[i][0])+'%20'+str(poslist[i][1])+'%20'+str(poslist[i][2])
    nrows=np.shape(sia2_return[i])[0]
    for j in range(nrows):
        row=sia2_return[i][j]
        datalink2=next(row.getdatalink().bysemantics('#this'))
        dataset=datalink2['description'].split(' ')[1]
        path=urllib.parse.quote(dataset,safe='')
        sodaurls.append(pathurl+path+'&'+cutout)
        vlassimgs.append(src[i]+'_'+datalink2['description'].split('/')[1])

nimgs=len(vlassimgs)
print('Number of images to attempt to download ',nimgs)
```

```
Number of images to attempt to download  4
```

Now we get the images and save them to disk. Note that some URLS return as invalid, probably becuase the cutout is outside of the imaged area (due to overlaps some images can appear near the edges of more than one image), so we put the image getting and saving in a try/except structure:

In [7]:

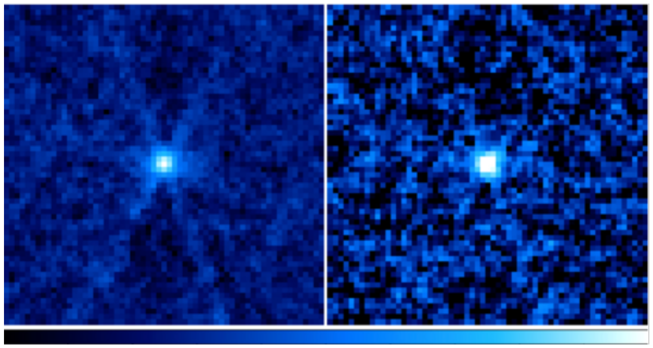
```
for i in range(nimgs):
    try:
        with urllib.request.urlopen(sodaurls[i]) as g:
            hdu2=fits.open(g)
            data=hdu2[0].data
            hdr=hdu2[0].header
            print('Writing image ',vlassimgs[i])
            fits.writeto(vlassimgs[i],data,hdr,overwrite=True)
    except:
        print('Cutout not found, moving on')

print('Done downloading files')
```

```
Writing image  test_VLASS2.2.ql.T18t07.J050939+313000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
Writing image  test_VLASS2.2.ql.T18t07.J050915+303000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
Writing image  test_VLASS1.2.ql.T18t07.J050915+303000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
Writing image  test_VLASS1.2.ql.T18t07.J050939+313000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
Done downloading files
```

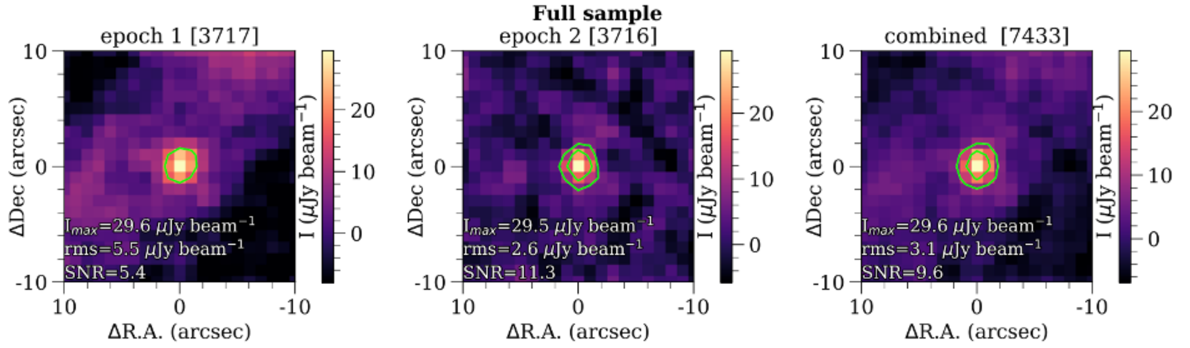
Results

- Return will include all VLASS epochs available, plus any overlaps within an epoch (might need to do more filtering).
- Images can be streamed directly into memory, no need to save as FITS files (unless you want to).
- CIRADA cutout service (cutouts.cirada.ca) limited to 200 positions and requires download as FITS files, much clicking…
- Thus, this SODA service makes it feasible to obtain cutouts for >O(10^3) objects for use in stacking, studies of larger samples, machine learning training etc.
- Added to VLASS user-contributed software page: https://science.nrao.edu/vlass/user-contributed-software-and-scripts

Example science use case - stacking



Stacking of blue (left) vs red (right) quasars (Glikman+22)



Stacking of z>4 AGN (Perger+23)

Future work

Frequency/velocity and polarization cubes

- Radio images can be multi-dimensional.
- VLASS, for example, will produce cubes of data with Stokes axes (I, Q, U) and will serve 16 frequency channels (48 planes).
- Other radio telescopes will produce much larger cubes, both in Stokes I and full polarization.
- High-level products such as spectral index maps and Faraday synthesis cubes will also be produced.
- ObsCore can probably deal OK with these, but some thought will need to be given to how to treat them in terms of cutouts etc so the user can easily request what they need and what makes sense for science.