VO data access and visualization developments

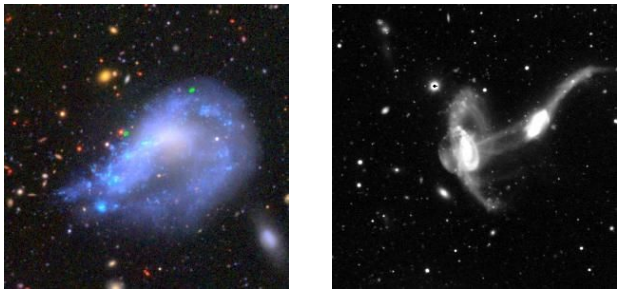
Matthieu Baumann, 4th February 2020, Strasbourg

VO data access: responding to community needs

- the CDS library ecosystem, getting ready for science platforms

- MOCPy:
  - Parses, manipulates and serializes MOC regions (group of HEALPix cells of different depths describing arbitrary regions)
- Astroquery.cds:
  - Sub-package of astroquery giving access to metadata of the datasets stored by the CDS:
    - HiPS, VizieR tables, Simbad database
  - E.g.: Query for surveys having observations in a complex region (i.e. a MOC).
- ipyaladin:
  - Widget allowing to run Aladin Lite in a Jupyter notebook.

- More to come: a new astroquery package to query **hips2fits** through Python (mainly giving a HiPS name and an astropy.WCS).

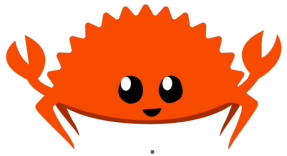


generated with HiPS2fits (T. Boch)

https://alasky.u-strasbg.fr/hips-image-services/hips2fits

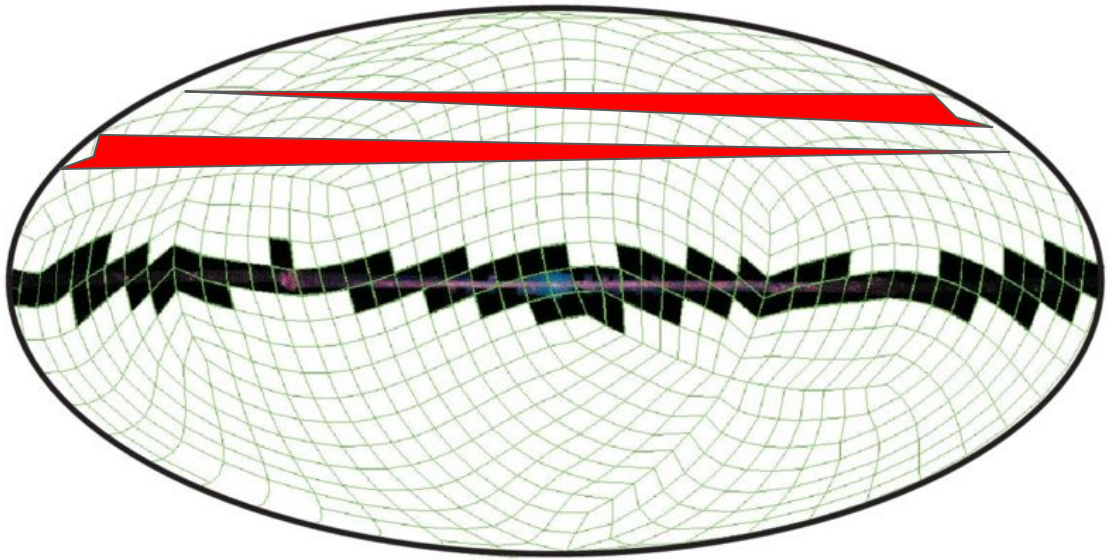
HiPS visualizer using WebGL & modern technologies

- WebGL2:
  - fully supported by firefox, chrome, Android OS web browsers
  - Compatible with iOS and Safari (experimental features currently but is working!)
- Rust:
  - Modern programming system language similar to C++ (compiled)
  - Performant, fast, new and appreciated by the developer community (most loved programming language on stackoverflow)
  - Good libraries ecosystem, wrapper around WebGL, etc..
- WebAssembly:
  - Bytecode compiled from C++/Rust and called from javascript code. It’s a W3C standard



WebGL HiPS visualizer: Support of multiple projections

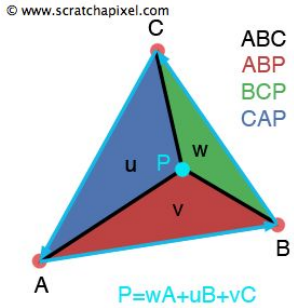
- Aladin-Lite uses canvas rendering:
  - Vertices of the HEALPix tiles being in the field of view are projected to the screen
  - This works for orthographic projection but for 2D projections like Hammer-Aitoff this introduces artifacts at the bounds!
  - One vertex of the tile is projected on the other side of the screen! **The tile is stretched and it's not good**

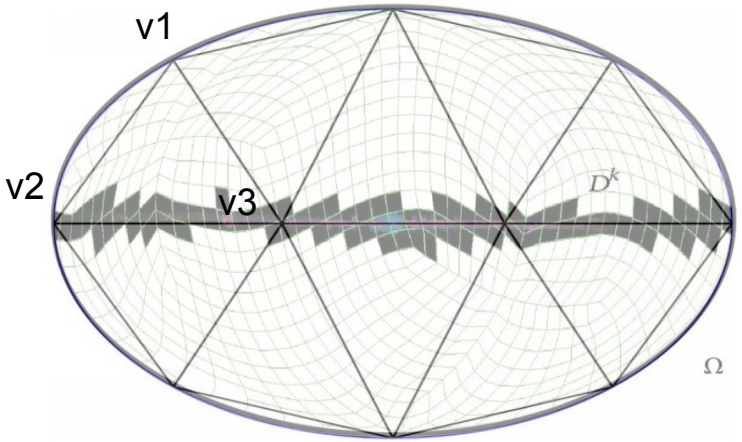


WebGL HiPS visualizer: Support of multiple projections

WebGL offers a way to “work” pixel by pixel and not tile by tile!:

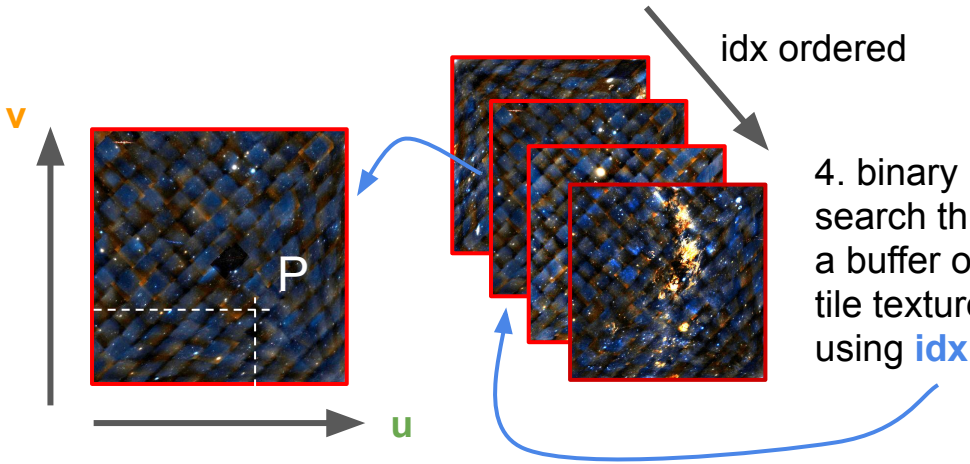
1. A set of (xp, yp) points in the projected space and their respective (ra, dec) in the world are computed one time at the beginning and given to the GPU
2. For each pixel, the GPU knows in which triangle the pixel is and can perform an interpolation between the 3 vertices of the triangle to know the (ra, dec) and (xp, yp) of the pixel.





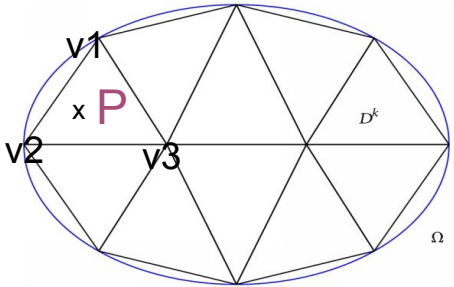
WebGL HiPS visualizer: Support of multiple projections

1. For each pixel we have its (ra, dec), we can get the HEALPix cell in which it is located and its (xt, yt) position on the tile!
   a. This uses the *HEALPix* **ang2pix** function which has been implemented in GLSL to run on GPU (F.-X. Pineau)
2. A buffer of 128 tile textures ordered by HPX idx is given to the GPU each frame. We retrieve the texture corresponding to the HPX idx returned by ang2pix! (binary search on a 128 sized array).



2. GPU interpolation between v1, v2 , v3 → (θ, δ)

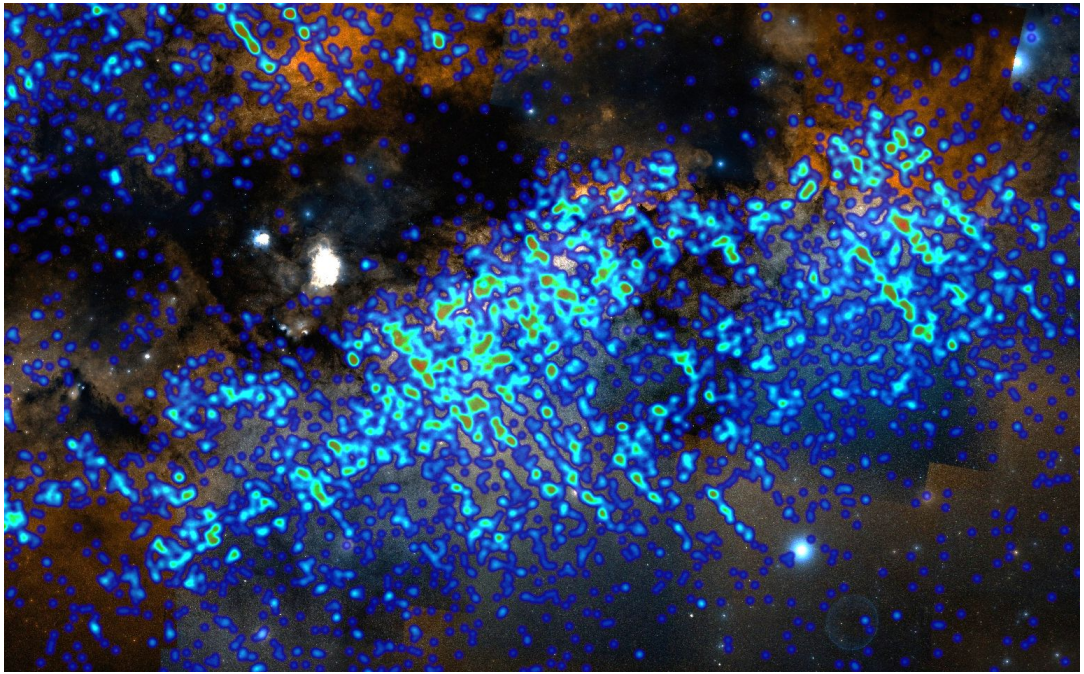
1. Color of P



3. ang2pix(θ, δ) → (**idx**, **u**, **v**)

WebGL HiPS visualizer: Plotting tables of ~100k sources

- WebGL allows to use the GPU to do “instancing”, i.e. plotting a large number of sources, up to several 100k in real time.
- Tables can be rendered as **heat-maps**.



1. Sources in the field of view are retrieved
2. A gaussian kernel is plotted over each of these sources. This gives a black/transparent image.
3. The image is passed in the GPU again which maps a color map on it.
4. Parallax, mag can be mapped to the size/opacity of the kernel.

WebGL HiPS visualizer: Some other features in development

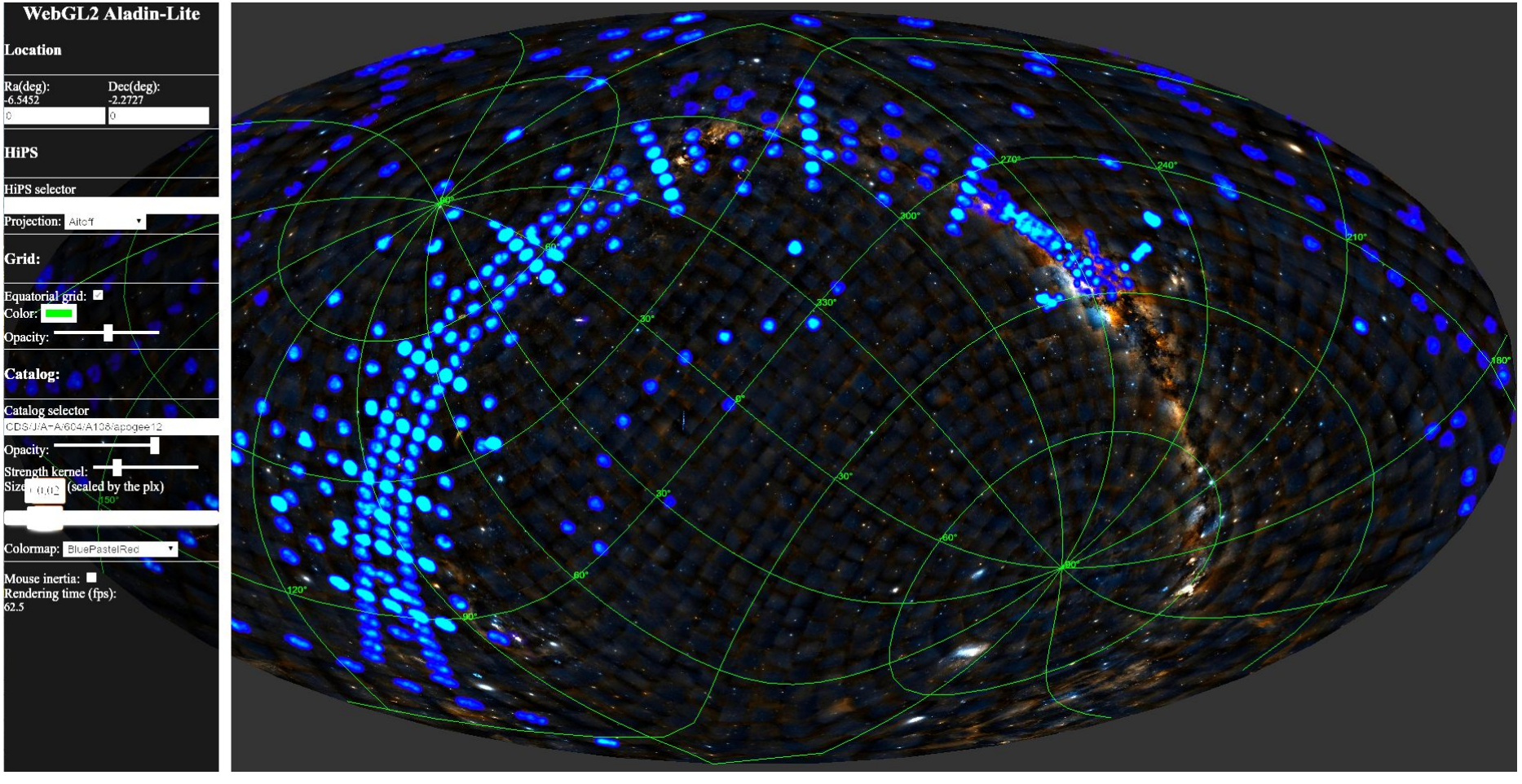
- Texture blending between tiles that have just been loaded
- Equatorial grid (not fully implemented) adaptable to all projections
- Mouse inertia when the mouse button/finger is released
- Moving animation between 2 locations on the sky

...

Nothing is better than a...

WebGL HiPS visualizer demo

http://cdsportal.u-strasbg.fr/webgl/



What are the next steps ?

- Preparing for different types of platforms...
- Lots of work to do:
  - Responsive js/css UI for a mobile compatible version
  - Render FITS images on top of a HiPS
  - Real-time texture generation by blending FITS tiles from several HiPSes
  - Support more projections
  - (future potential for planet surface rendering using height map HiPS)
- Integrate WebGL code into Aladin-Lite as a improved visualization interface
  - Will not tend to fully replace the existing Aladin-Lite
  - Aladin Lite API will remain the same

VO data access and visualization developments

Thank you!