

Space and Time coverage maps in MOCPy

Matthieu Baumann, T. Boch, P. Fernique, A. Nebot, F.-X. Pineau



<https://github.com/bmatthieu3>



matthieu.baumann@astro.unistra.fr

What is a ST-MOC ?:



Source catalogs and image surveys all have a footprint on the sky. Taking into account a **new dimension** such as the **observational time** will allow to represent the time evolution of a spatial footprint.

Originally initiated by the **IVOA organization**, Space-Time coverage map (ST-MOC) is a **new efficient and consistent data structure** for storing observational positions along with their temporal information.

Please refer to <http://www.ivoa.net/documents/stmoc/> for more information.

Some Useful Links:

- GitHub : <https://github.com/cds-astro/mocpy>

- Documentation : <https://cds-astro.github.io/mocpy/>

- Notebooks : <https://mybinder.org/v2/gh/cds-astro/mocpy/master>

- PyPI : <https://pypi.org/project/MOCPy/>

pip install mocpy

New MOCPy Features:

- Create from a list of Astropy times and skycoords.
- Query by a time range to retrieve the spatial regions being observed within it.
- Perform **logical operations** (e.g. intersection of the XMM and Chandra ST coverages to find simultaneous observations).
- Filter an Astropy table containing position and time columns.
- Save to a FITS file.

Available ST-MOCs:



About **160** ST-MOCs generated for **VizieR** thanks to MOCPy and available as **FITS files** through : <http://alaska.u-strasbg.fr/footprints/STMOC/>
Open them with **MOCPy** or **Aladin Desktop** !

Implementation notes:



- Core functions have been fully rewritten in **Rust**. Rust is a **compiled programming language** comparable to C++ that is **safe, concurrent and performant**.
- Python code is now interfaced with Rust thanks to **PyO3**, a performant Python/Rust binder. **Numpy** has a wrapper in Rust too.
- The library is **multi-platform**. Binary wheels are generated for 32/64 bits Linux, MacOS and Windows architectures !

Performance:



- Rust exploits the full potential of your machine thanks to the **rayon** crate simplifying writing concurrent code.
- **2MASS** image survey has **4.8M** (position, time) tuples. On a i5-4690 CPU @ 3.50GHz (i.e. 4 physical cores), generating its ST-MOC takes **5.17s** CPU time for a **user time of 1.83s (~4x speedup)** !

