Efficient Data Processing for Large Image Cube Visualisation

A. Comrie[1,2], R. Simmonds[1,2], A. Pinska[1,2], A. R. Taylor[1,2,3]

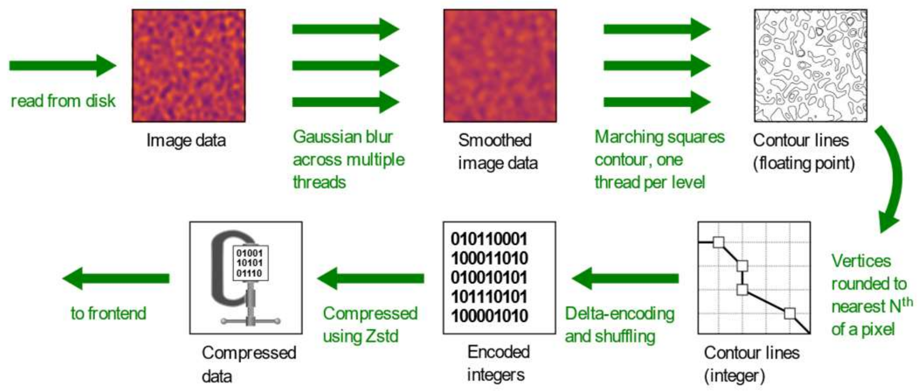
[1]*Inter-University Institute for Data Intensive Astronomy* [2]*University of Cape Town* [3]*University of the Western Cape*

Introduction

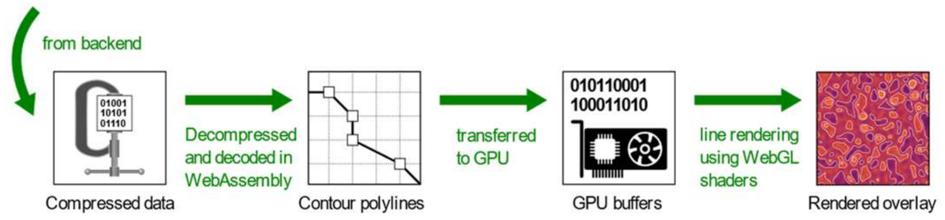
Image cubes created by the latest generation of telescopes are often too large to efficiently visualise on portable or desktop hardware. Instead, a remote visualisation system must be used. We present an approach developed to improve the efficiency of server-client visualisation tools. This approach has been implemented in the Cube Analysis and Rendering Tool for Astronomy (CARTA). We utilise a backend server with fast access to image data, communicating with a browser-based frontend app through a **WebSocket** connection for bi-directional streaming of control commands and data.

Calculating, compressing, delivering and rendering contours

Instead of rendering contours to a lossy image format before delivery to the frontend, contour polylines are generated by the backend across multiple CPU cores. A two-pass Gaussian filter is applied in parallel prior to contour tracing. Contour vertices are quantised to integers, before delta-encoding and byte shuffling, in order to drastically improve compression ratios with **Zstd**.



The frontend decompresses and decodes vertices using **WebAssembly** code. **WebGL** shaders are used to accelerate the rendering of contour polylines after they are loaded into GPU memory. GPU-accelerated frontend rendering allows the user to alter the render config of contours without requiring them to be re-generated or re-sent by the backend.



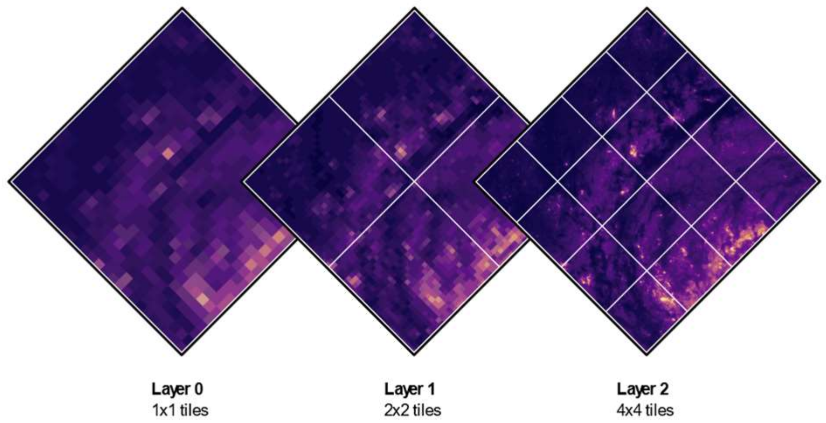
Utilising Modern Web Technology

WebAssembly allows us to compile custom code and common libraries written in c/c++, and execute them in the browser at near-native speeds. CARTA uses this for ZFP, Zstd and AST libraries.

WebWorkers run JavaScript or WebAsesmbly code in a separate thread, allowing us to offload compute-intensive tasks to other CPU cores. CARTA uses this for tile decompression.

WebGL allows us to utilise GPU-acceleration when rendering in the browser, by writing shaders in the GL shader language (GLSL). CARTA uses WebGL shaders to render image and contour data.

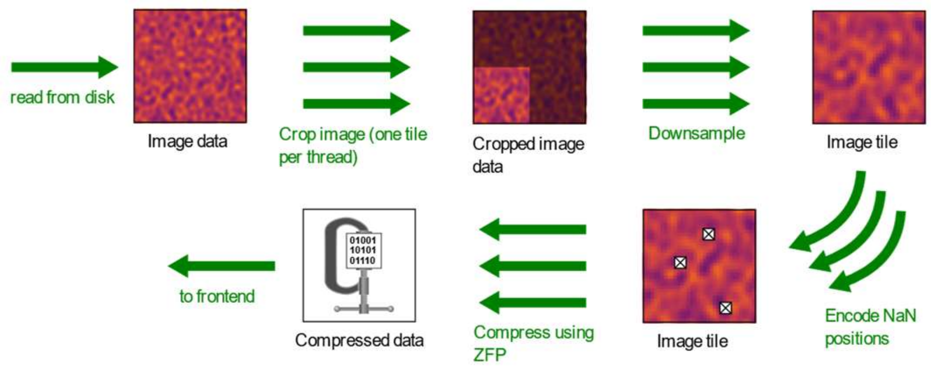
Tiled Rendering



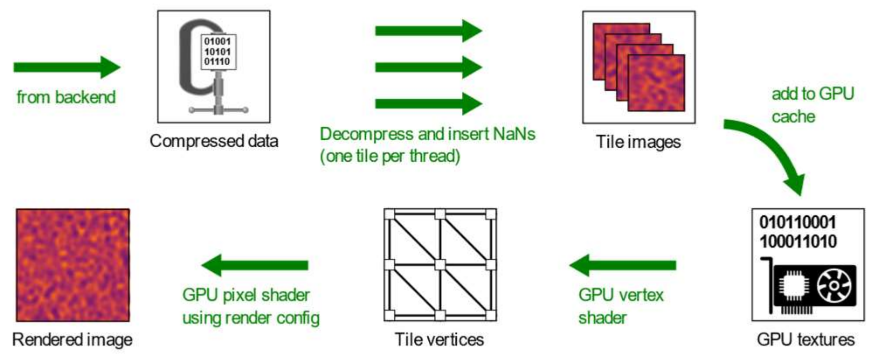
Images are delivered to the frontend in a series of fixed-size **tiles**. Each tile belongs to a **layer**, with each subsequent layer doubling the number of tiles in each dimension. As the user zooms into the image, tiles from higher-resolution layers are requested from the backend. Only those tiles that are visible in the frontend are sent. Tiles are **cached** by the frontend in both system memory (as compressed data) and GPU memory (as uncompressed textures) for efficient rendering, and to prevent duplicate tile requests.

Compressing, delivering and rendering images

When visualising large image cubes, a single channel is loaded into memory at a time. The backend sends only the sections of the image that are visible in the user's viewport, down-sampled to an appropriate resolution. Image data is sent as lossy compressed floating-point data using the **ZFP** library. Image tile are prepared and compressed in parallel.



Tiles are decompressed in parallel using **WebWorker** threads. After loading tiles into GPU memory as floating-point textures, **WebGL** shaders are used to apply clip bounds, scaling and colour maps, transforming the floating-point data into a colour image. If the render configuration changes, the image is updated instantly without requesting new data from the backend.



CARTA Software Package

The Cube Analysis and Rendering Tool for Astronomy (CARTA) is a server-client software package for efficiently visualising and performing analysis on large image cubes. It is developed by a collaboration from the four institutes shown below. Version 1.3 of CARTA will support contour overlays.

See P4-3 for details

https://cartavis.github.io

https://github.com/CARTAvis

Author Information

Dr. Angus Comrie is a visualisation software developer at IDIA, and the lead developer of the CARTA project. Feature requests are welcome!

angus@idia.ac.za

https://github.com/veggiesaurus