# Diesel-Powered GPU Computing

Enabling a Real Time Radio Telescope in the Australian Outback

Richard Edgar
Initiative in Innovative Computing
Harvard University

# Talk Overview

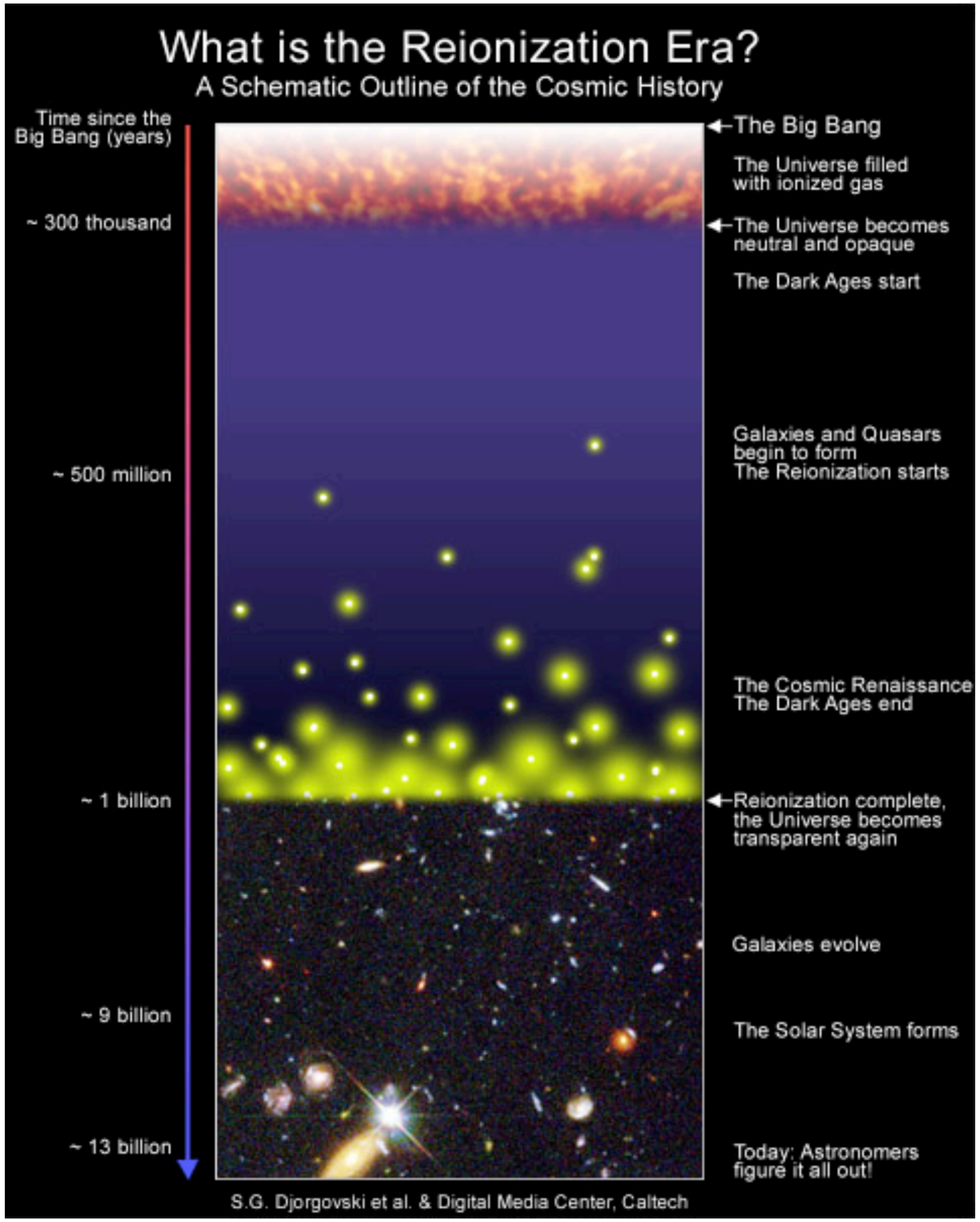
- Description of the Murchison Widefield Array
- MWA Challenges
- The Real Time System
- CUDA Acceleration
- Benchmarks
- Summary

# The

# The Murchison Widefield Array

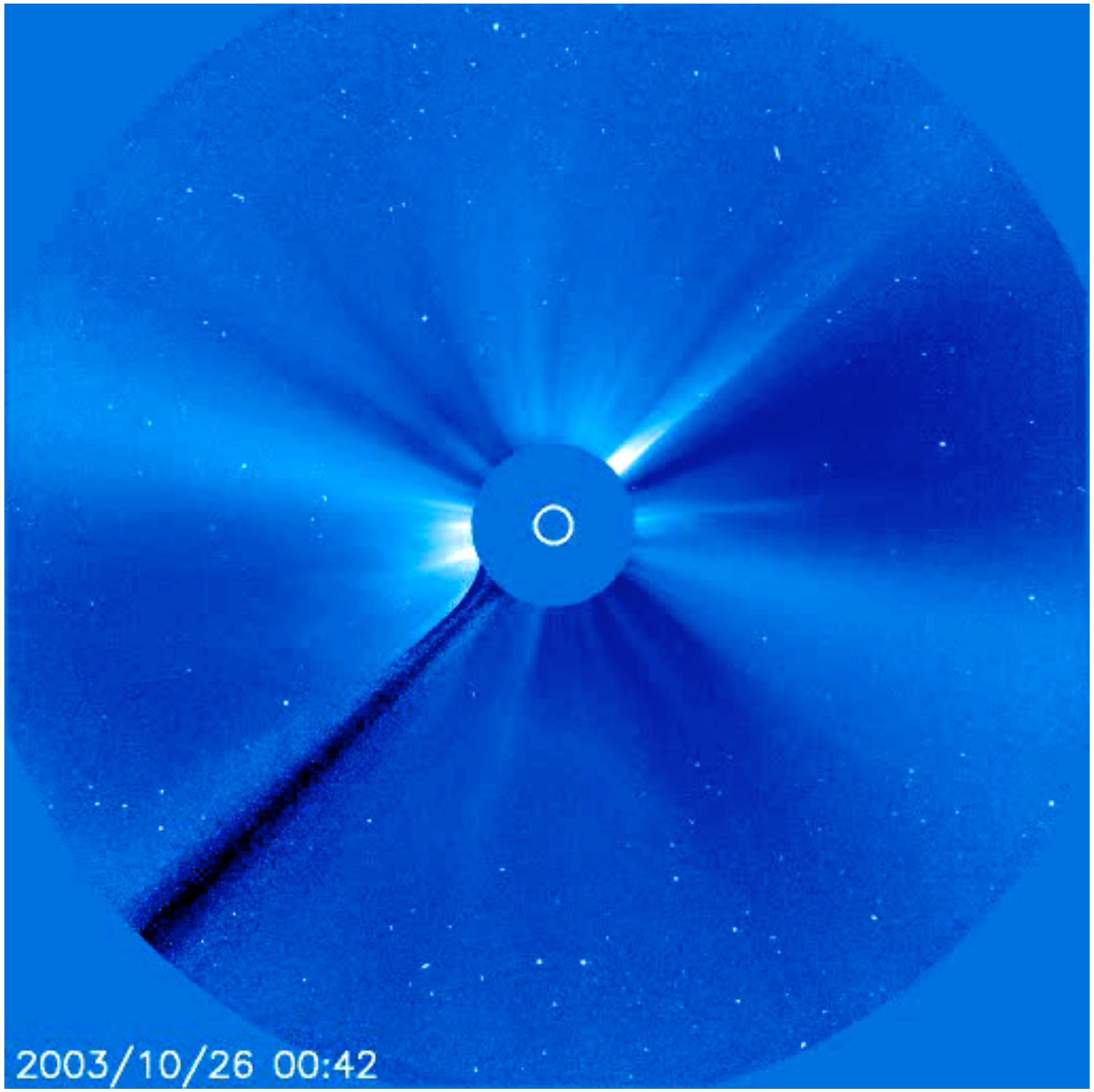
- Next Generation Radio Interferometer
  - Real time imaging
  - Wide field of view
- Key science goals are
  - Observing the Epoch of Reionisation
  - Solar, heliospheric & ionospheric observations
  - Surveying the transient radio sky

# Epoch of Reionisation




- Universe became neutral about 300,000 years after Big Bang
  - Observed by COBE, WMAP etc.
- Stars started to form 500 Myr later
  - Start of Epoch of Reionisation
- Reionisation complete 1Gyr after Big Bang
- EoR contains developing cosmic structures
  - Vital to understanding galaxy formation
- Best observed with redshifted H 21cm line

# Solar, Heliospheric & Ionospheric Observations

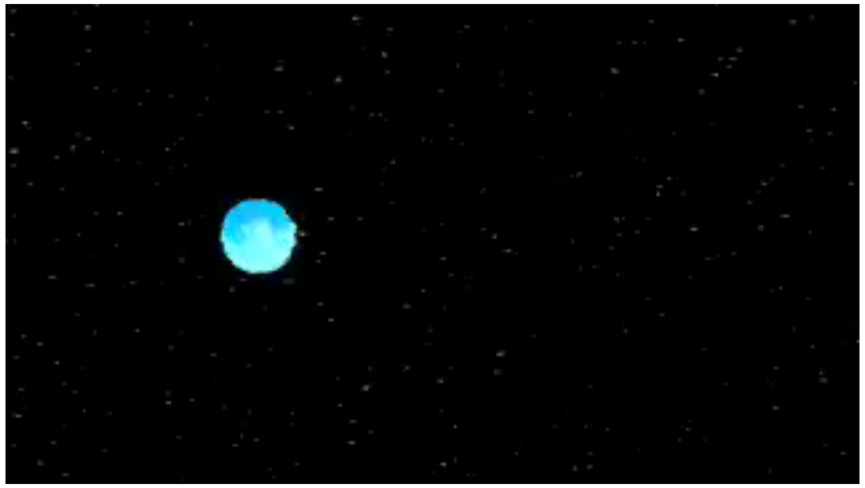



- Sun is an active body
  - Regular flares, coronal mass ejections and magnetic storms
- These affect
  - Satellites
  - Communications
  - Power grids
- Can reach Earth in 15 minutes
  - Need to predict in advance

Image courtesy of SOHO/EIT (ESA & NASA)

# Surveying Radio Transients

- The Universe is a dynamic place
- Full of short lived but dramatic events
  - Gamma Ray Bursts
  - Planetary and stellar radio bursts
  - Pulsar and Active Galactic Nuclei jets
- Combine observations with instruments at other wavelengths
  - Build up better models of these phenomena
- MWA will image 6 orders of magnitude deeper than previous radio surveys



Movie courtesy of NASA/Swift/Cruz deWilde

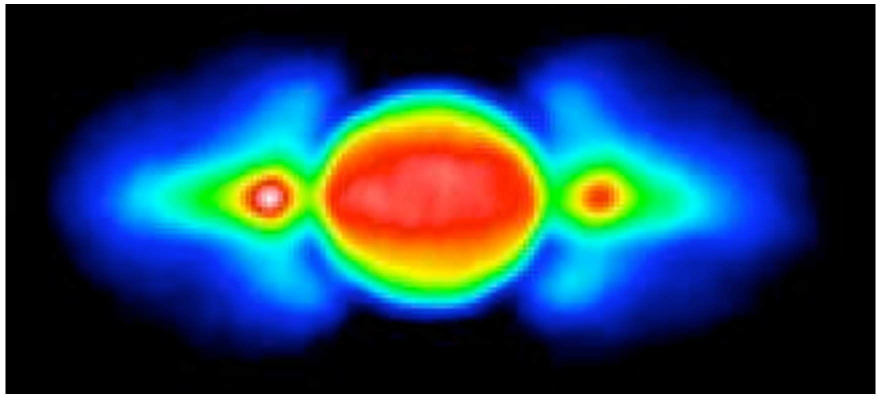
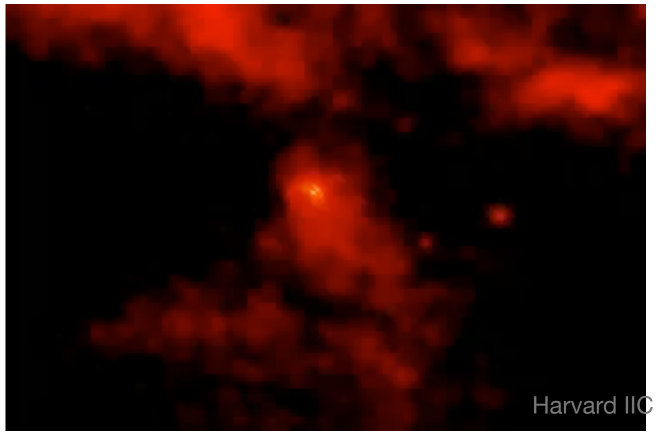


Image courtesy of CSIRO




Movie courtesy of NASA/CXC/Penn State/G.Pavlov et al.

# Radio Interferometry

- Interferometry used to overcome Rayleigh criterion
- Correlate signals from two detectors
  - Measures fourier transform of the sky
- Combine many detectors to measure entire sky
- Resolution controlled by longest baseline

$$\theta \approx \frac{\lambda}{d}$$

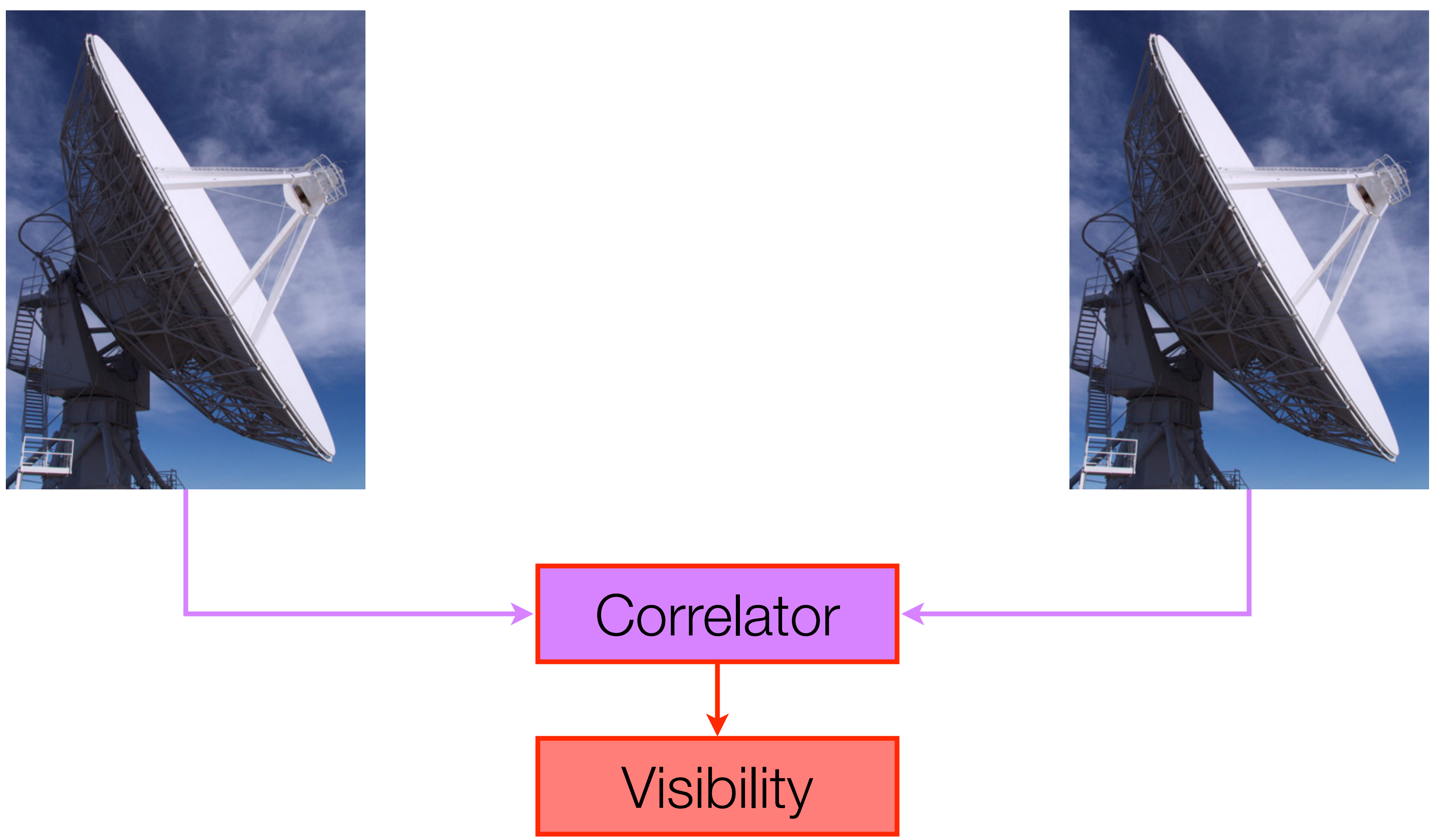
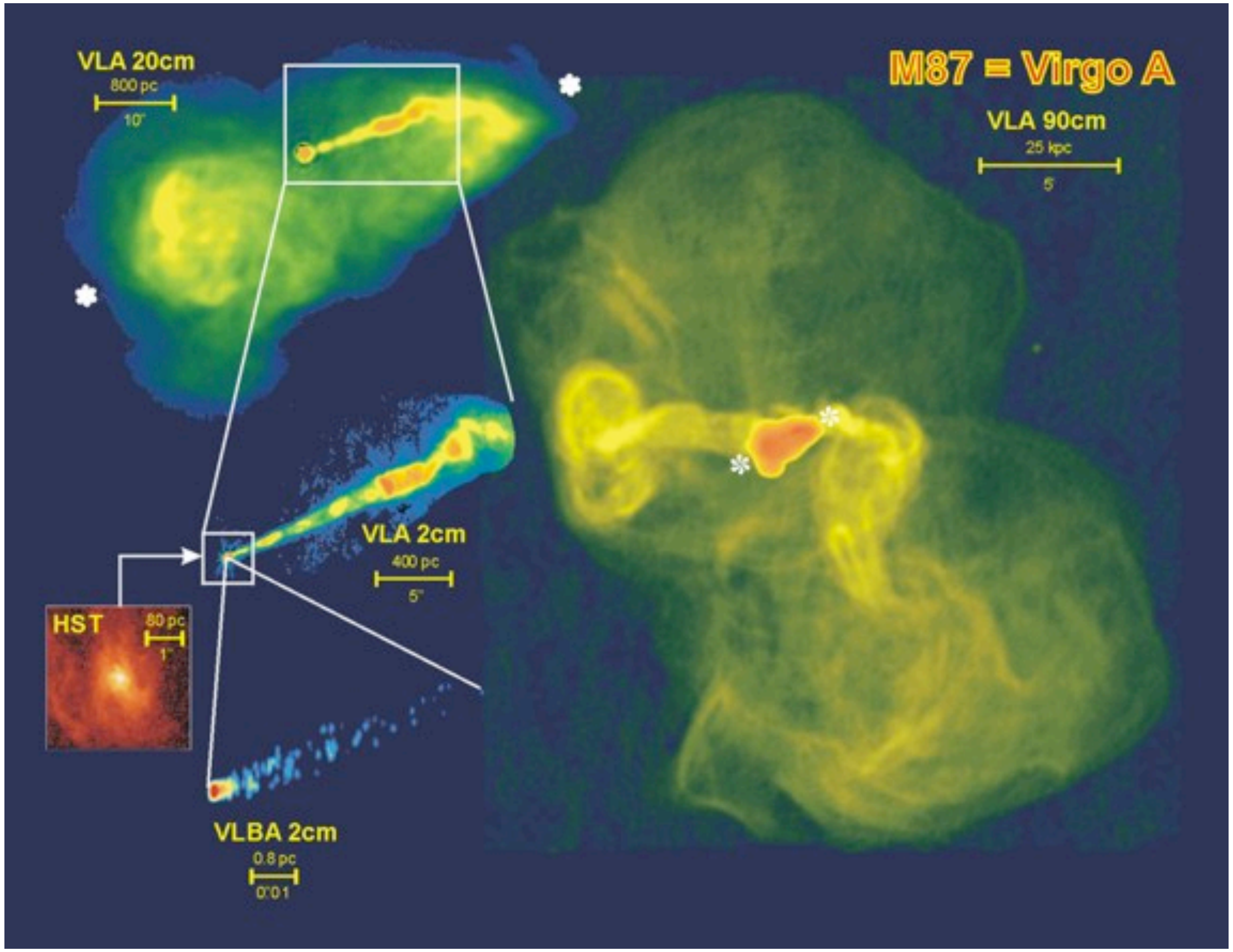



Image courtesy of NRAO/AUI and David Andrew Gilder

# Radio Interferometry

Visibility Data




Ionospheric Distortion

Instrument Calibration

Image courtesy of NRAO/AUI

# Radio Interferometry

- Do not know instrument calibration or ionospheric distortion
- Have to use CLEAN algorithm
    - Solves for image, calibration & distortion simultaneously
- CLEAN is very successful, but
    - Iterative
    - Interactive
- Not suitable for real time use

# The Murchison Widefield Array

- MWA has 512 detectors over 1 $km^2$
  - Around 130,000 baselines
- Operating in 80-300 MHz waveband
- Channel bandwidth is 10 kHz, have 768 channels
- Detectors measure two polarisations
  - Have four correlation products (XX, XY, YX, YY)
- Integration time is 8 s
  - Set by ionospheric turnover time
- Acquire around 40 GB of data every 8 s

# MWA Challenges

# Real Time Operation

- Real time operation is new frontier for radio astronomy
- Required development of new algorithms
- Enabled by increase in computing power
  - Estimate 18 TFLOP for entire pipeline
  - Must be complete within 8 s

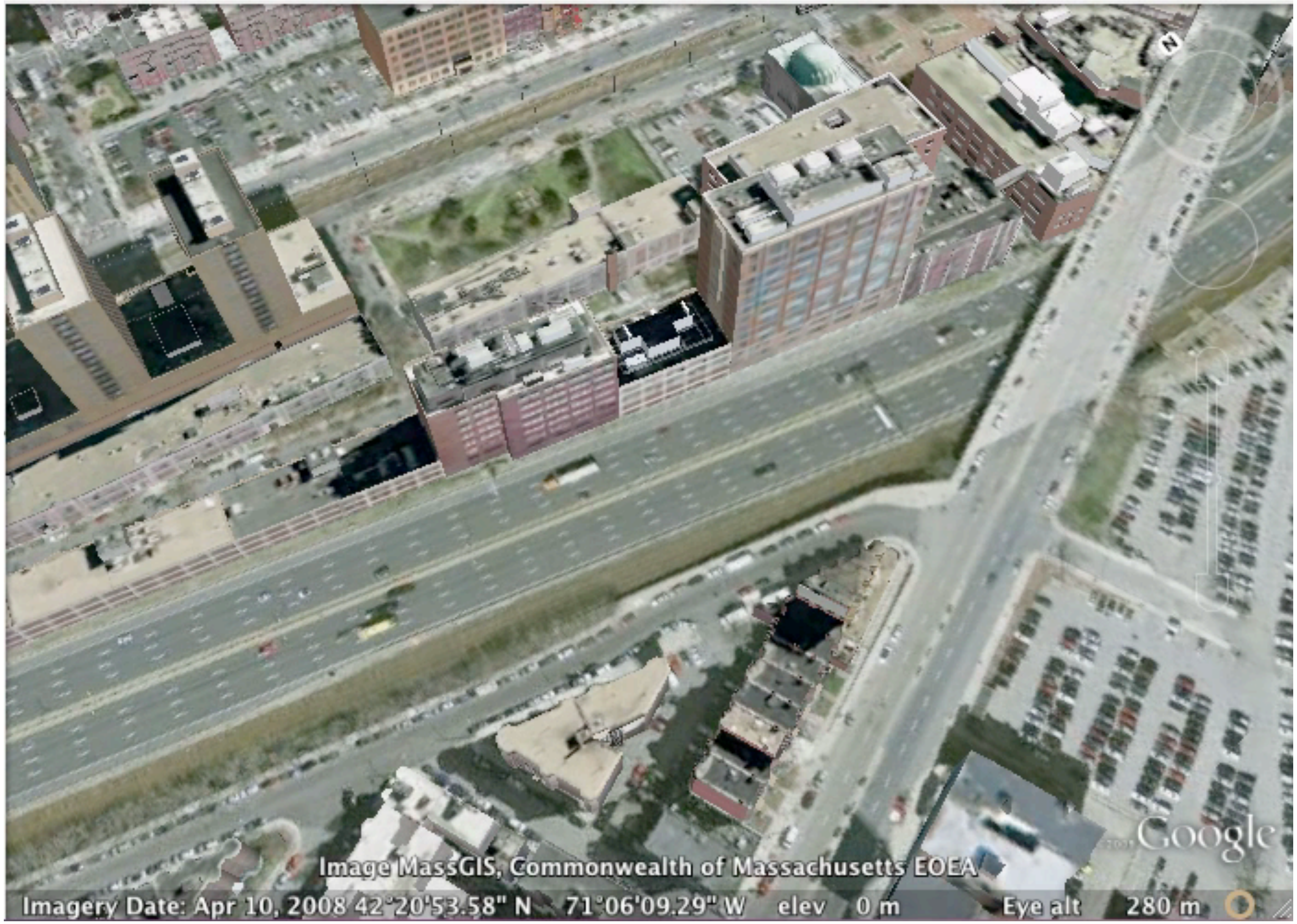
# The ‘A’ Challenge

- System has 512 tiles
  - Each has 16 dipole antennae
- Tiles are
  - Stationary
  - Non-directional
- Have to be steered in telescope software

# The ‘W’ Challenge

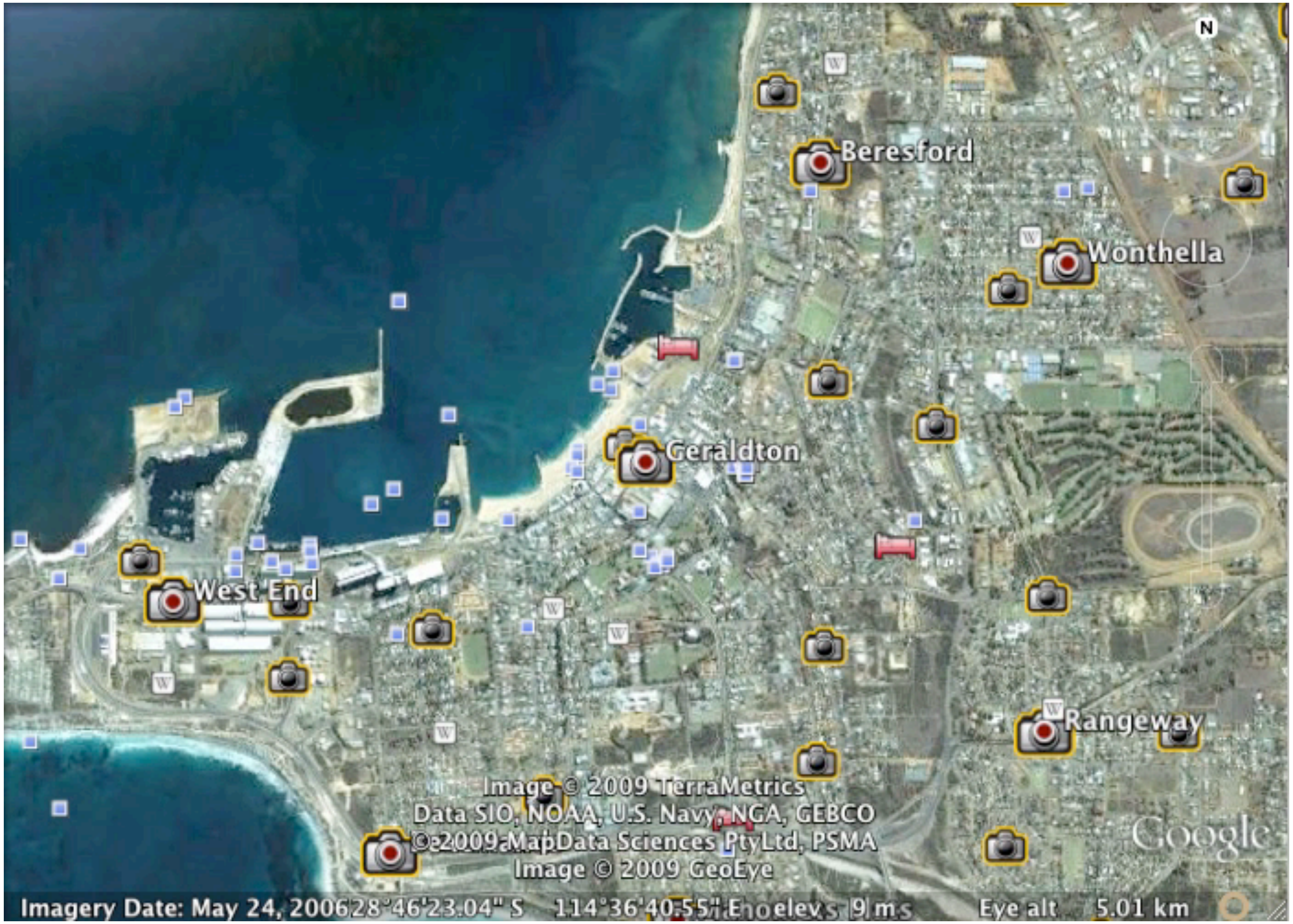
- MWA will observe very wide fields
- Many traditional approximations invalid
  - e.g. fourier transform not quite orthogonal
- Observations can get close to pole
  - Co-ordinate singularity
- Have to cope with these in software

# The ‘M’ Challenge

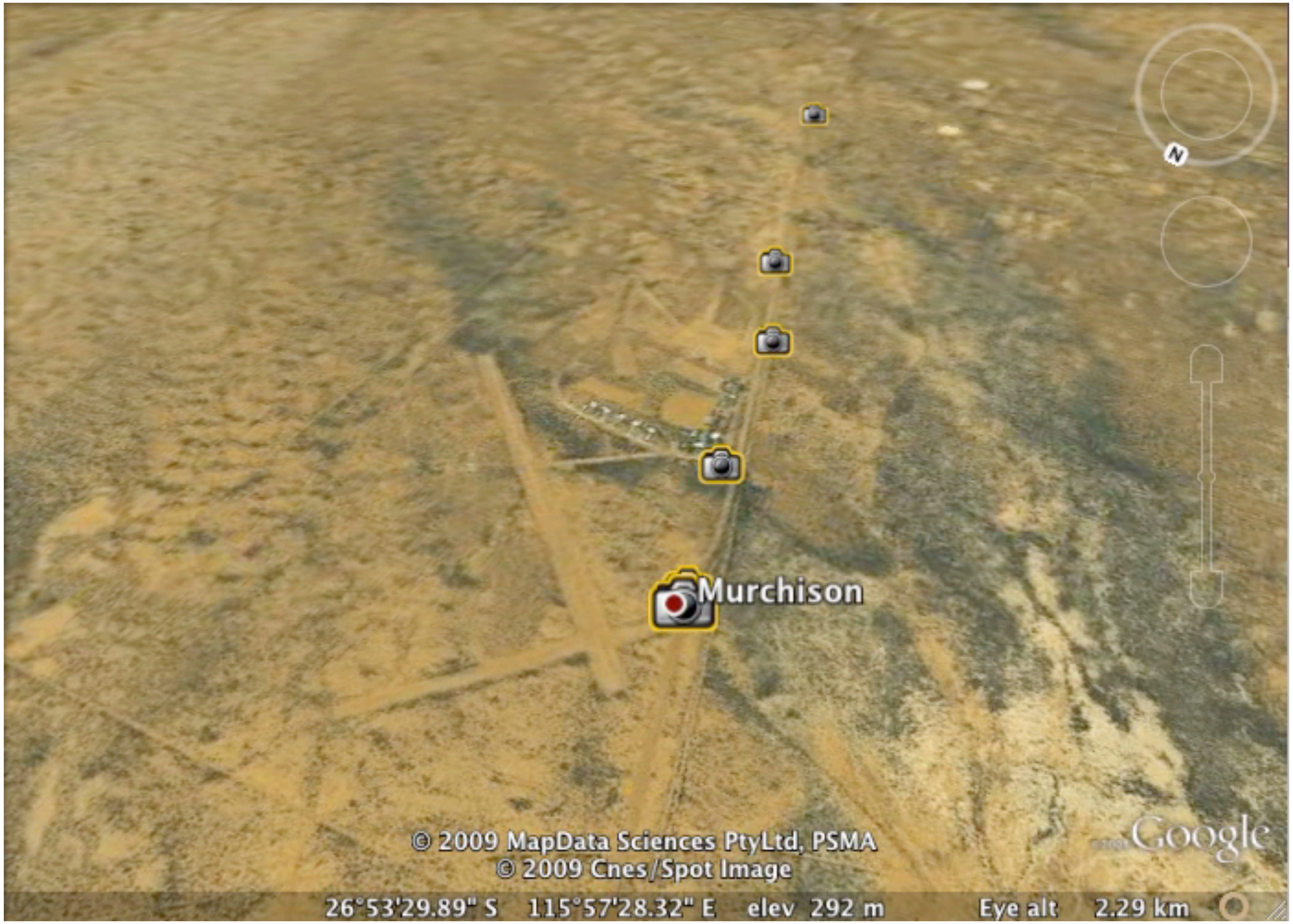


# The ‘M’ Challenge

# The ‘M’ Challenge



# The ‘M’ Challenge



# The ‘M’ Challenge

# The ‘M’ Challenge

# The ‘M’ Challenge

# The ‘M’ Challenge

# The ‘M’ Challenge

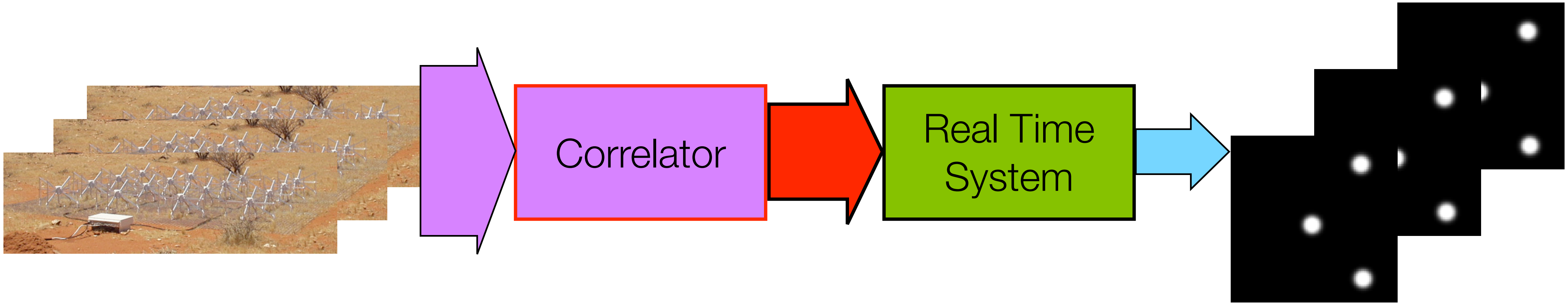
# The Need for GPUs

- MWA pipeline is very computationally intensive
  - Doubtful CPUs could meet real time deadlines
- MWA has limited power available
  - CPUs certainly too power hungry
- Requires use of GPUs

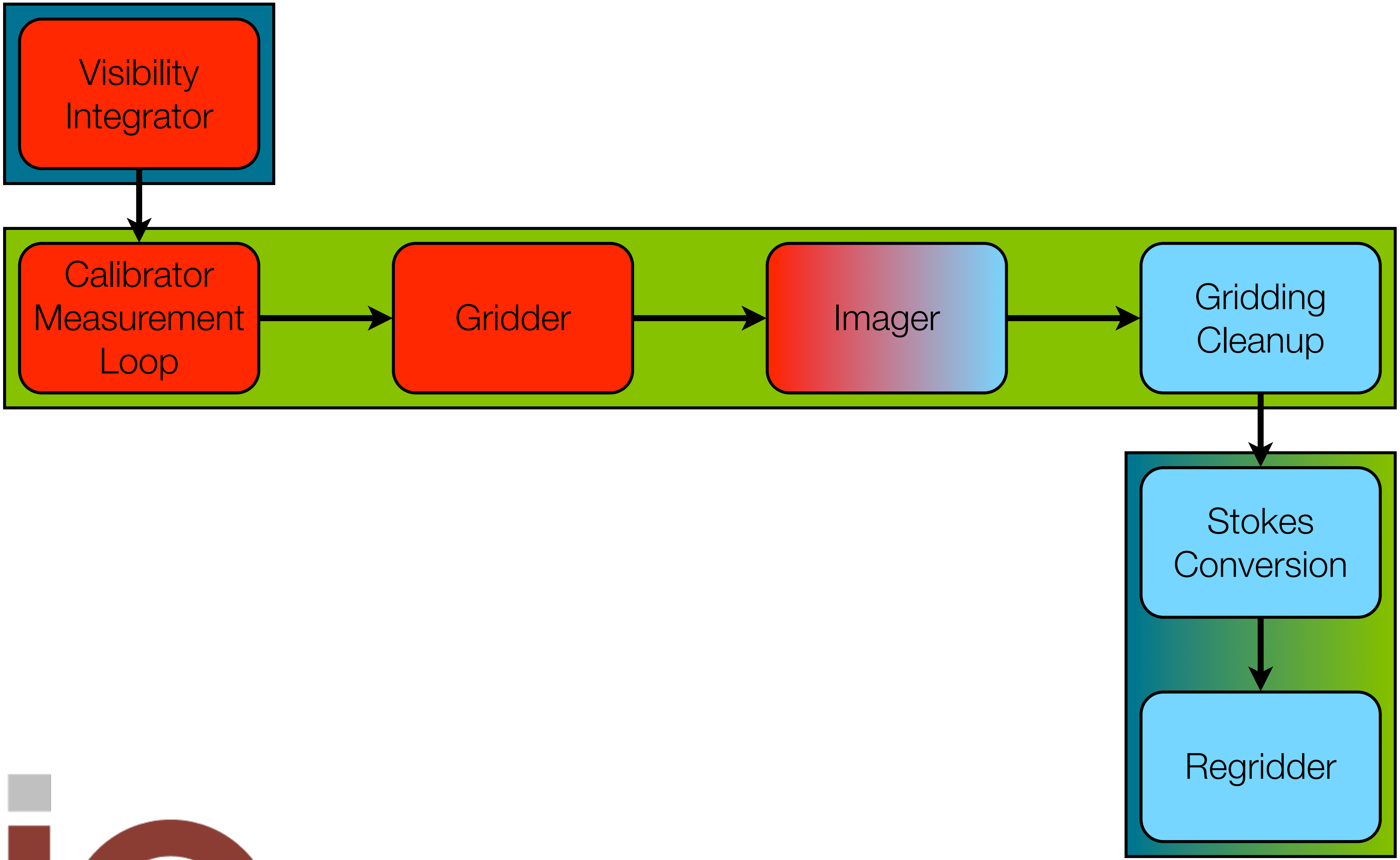
# The Real Time System

# The Real Time System

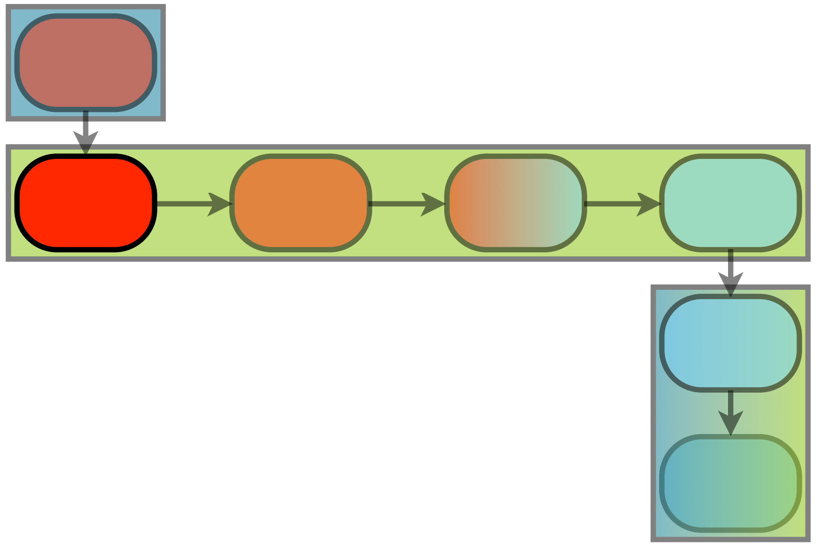
- The Real Time System (RTS)
  - Takes visibilities from the correlator
  - Produces final images
- Runs on the Real Time Computer (RTC)



# RTS Overview

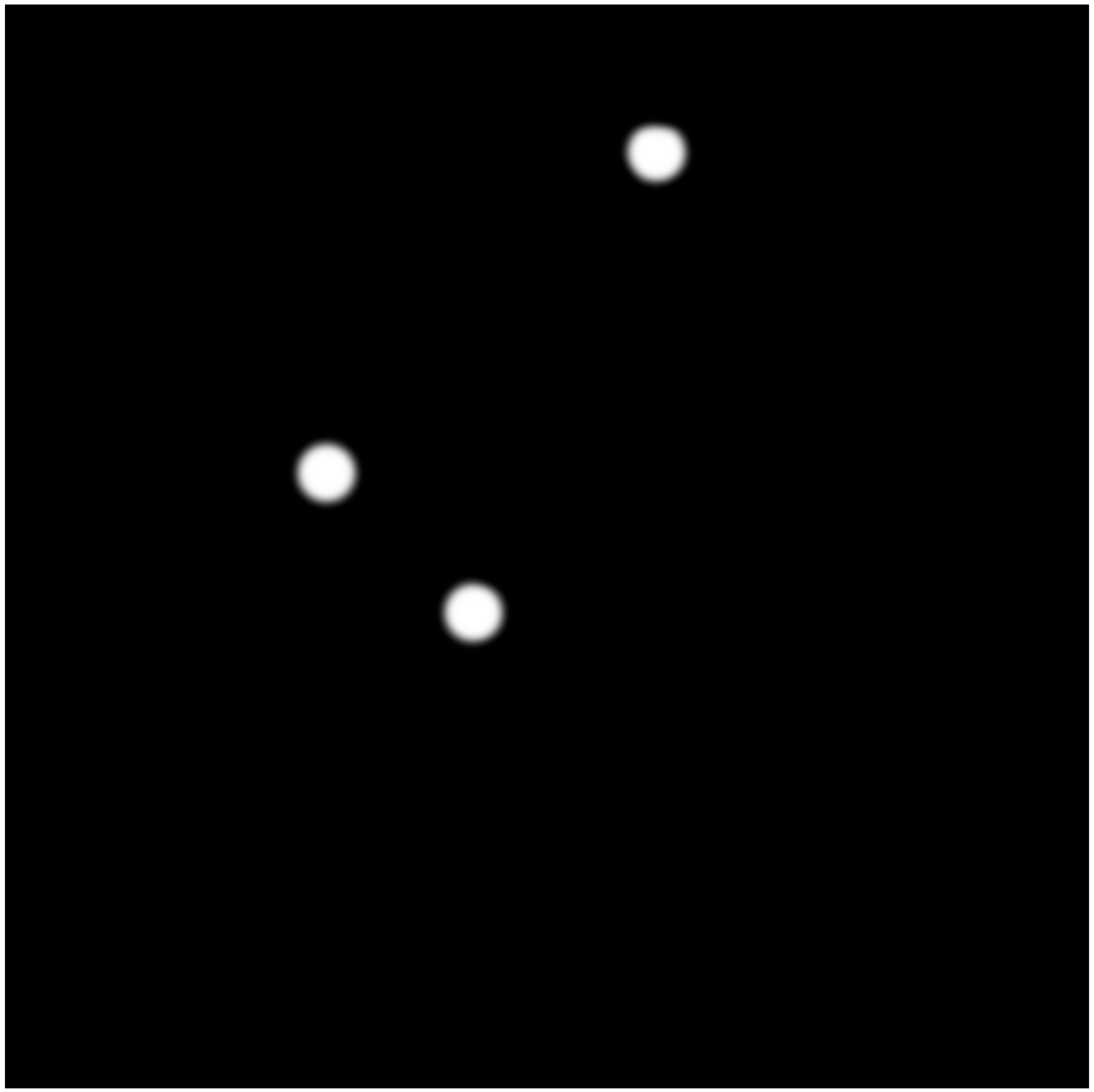


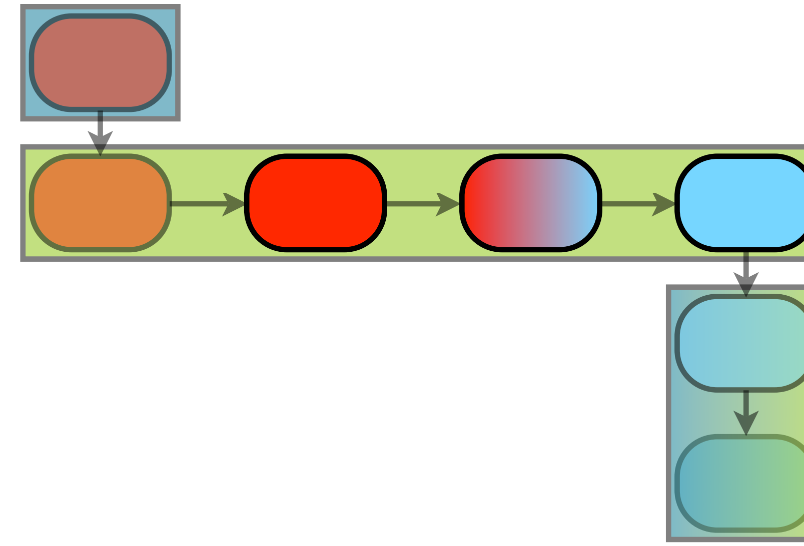
# Calibrator Measurement Loop



- The CML is responsible for
  - Determining tile gains
  - Measuring the ionospheric distortion
- Uses list of known calibration sources
- Only point in RTS where channels communicate
- Calibration works best on consecutive channels
  - Assume linear frequency response
- Ionospheric response known function of frequency
  - Best to use widely separated channels
- Each node has consecutive channels
  - MPI communication for ionospheric fit

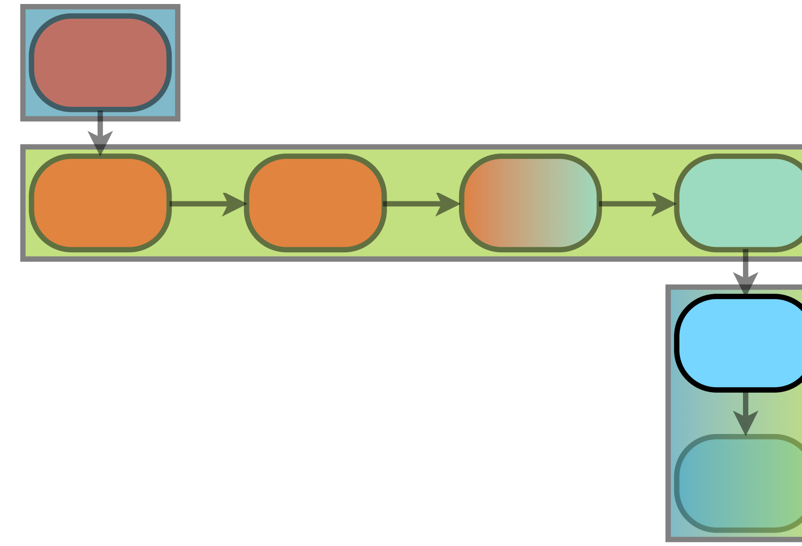
# Gridder, Imager & Gridding Cleanup



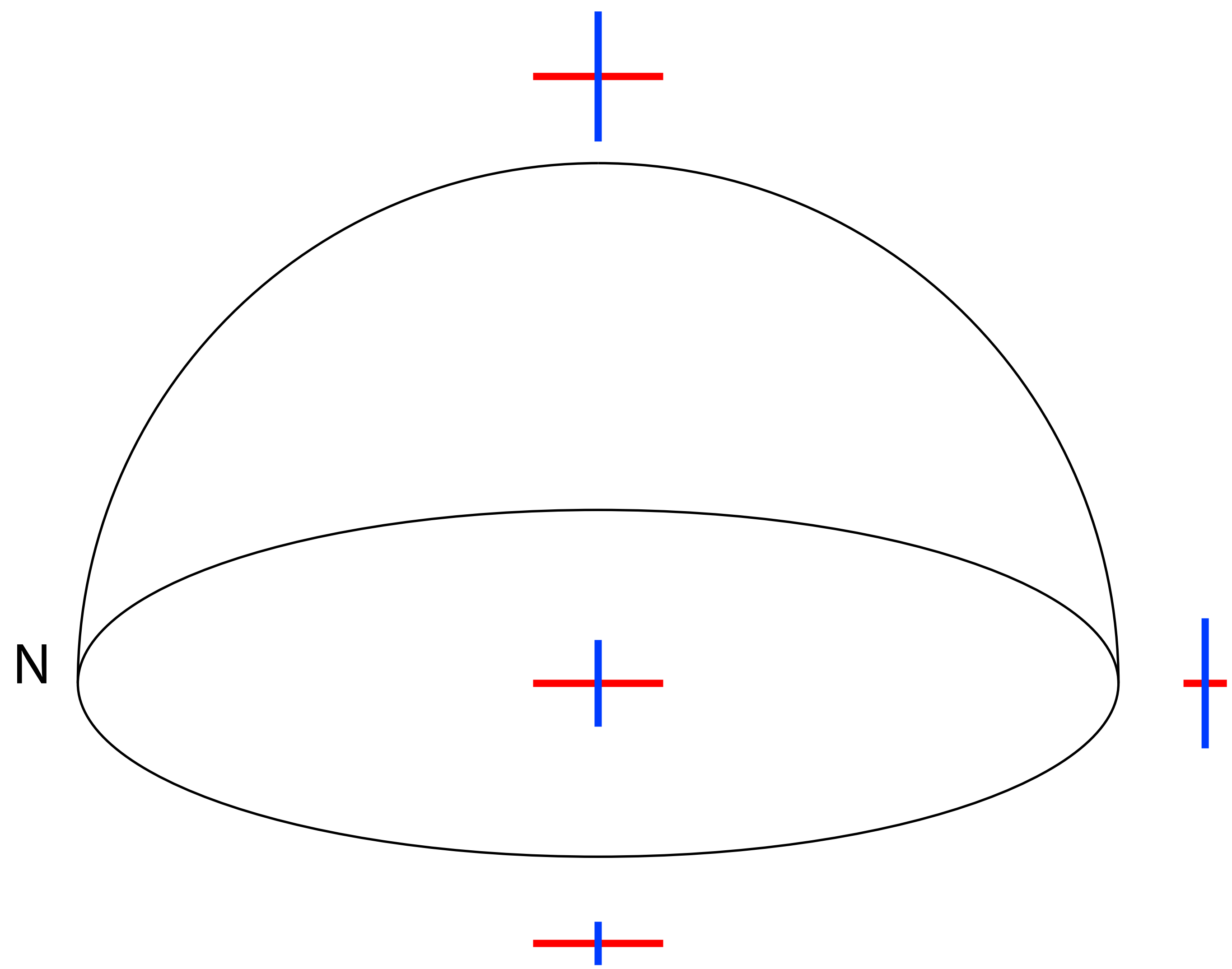


- Visibilities are points on fourier plane
- Gridder interpolates onto regular grid
  - Convolves with compact kernel
- Imager performs FFT
- Gridding clean up divides out kernel

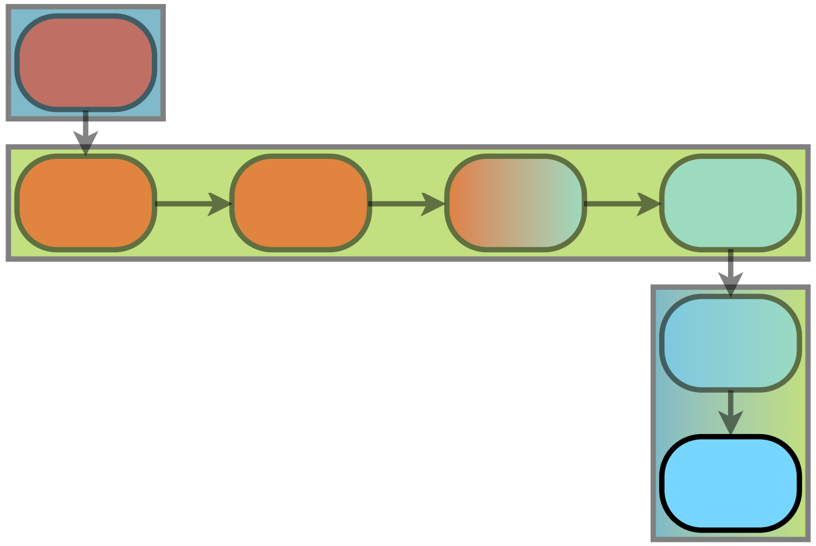
# Stokes Conversion



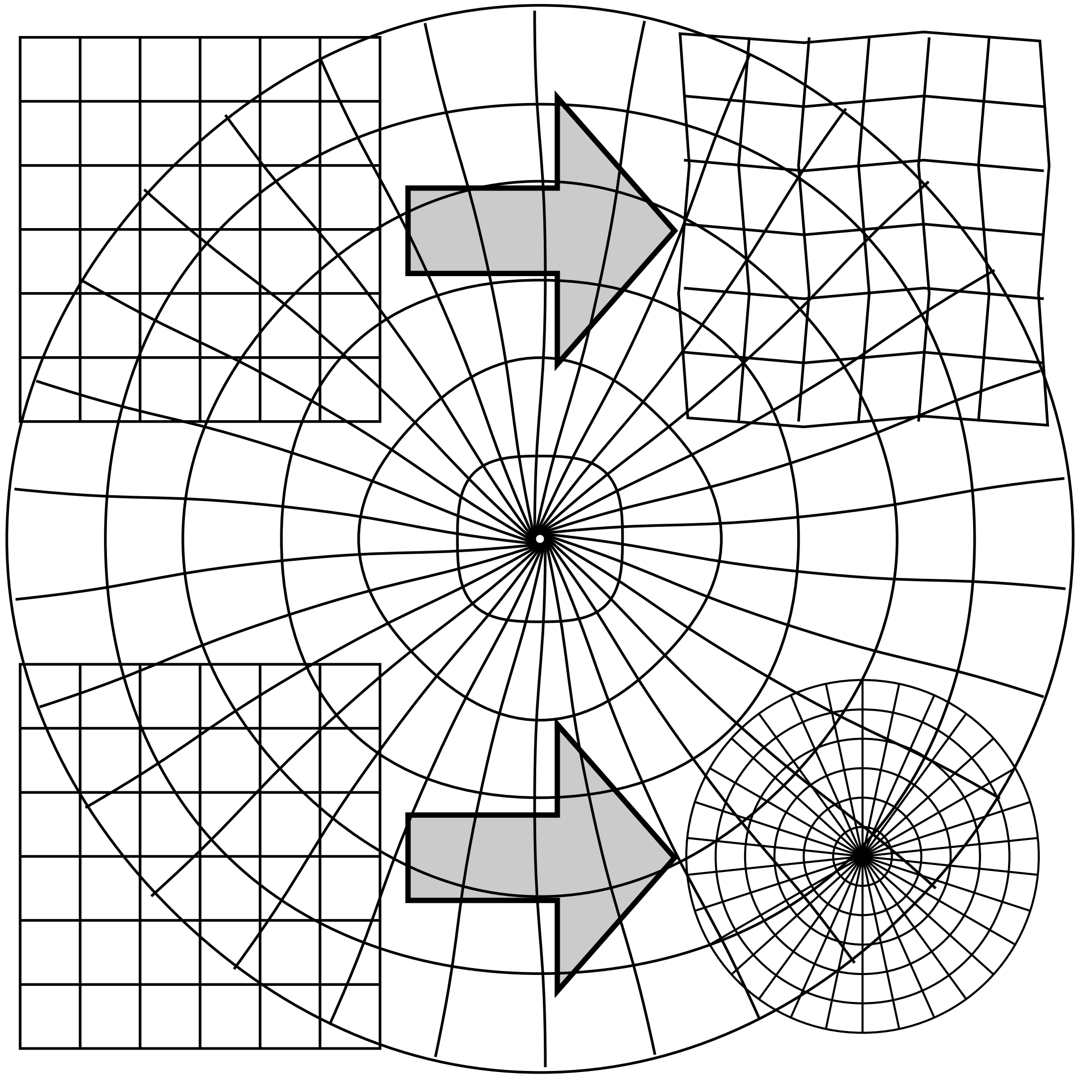
- Have four polarisations in ground frame
- Want polarisations in sky frame
- Each pixel is 4 element vector
  - Multiply by 4x4 matrix
- Matrices computed on the CPU
  - Applied on GPU



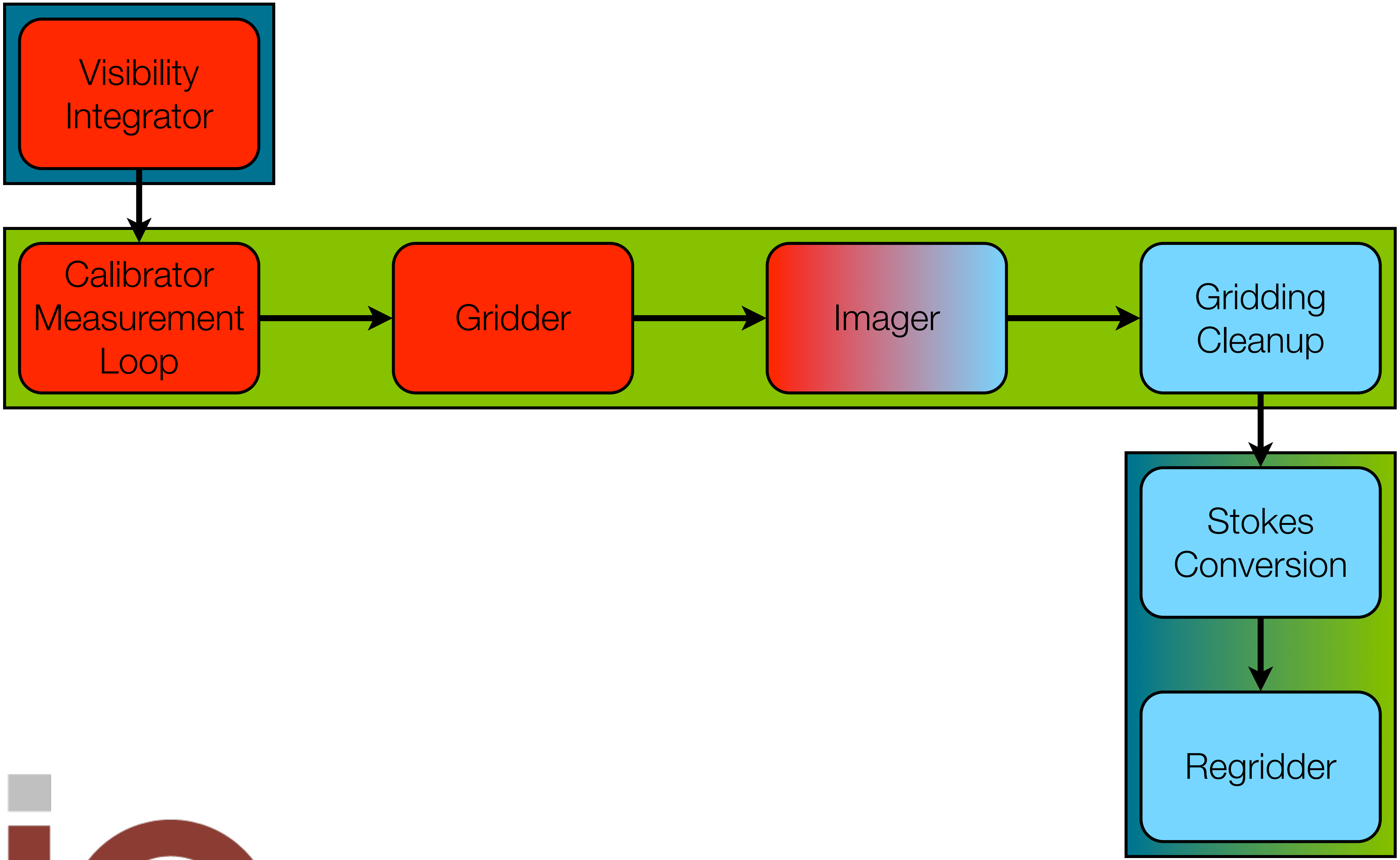
# Regridding



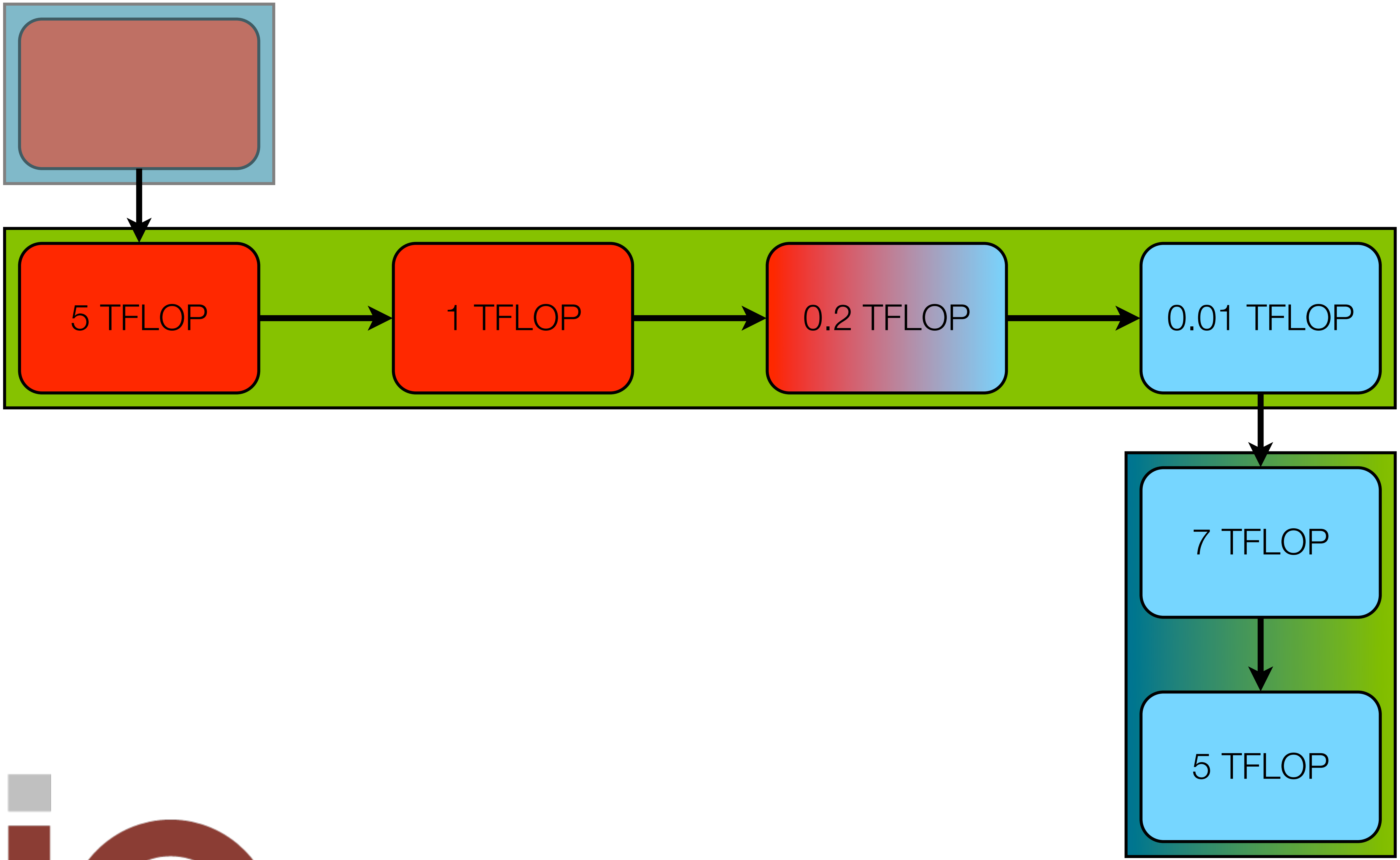
- Still have two effects to handle
  - Ionospheric distortion
  - Sky curvature
- Use HEALPIX library
- Precomputation on CPU
  - Applied on GPU



# RTS Computational Load



# RTS Computational Load



# CUDA Acceleration

# CUDA Acceleration

- Existing code base written in C
  - CUDA a natural choice
- Previous feasibility study by Dale
- CUDA code needed to be updated & integrated
- Goal was unified codebase
  - GPU acceleration compile-time option

# CUDA Acceleration

- Basic procedure
  - Identify large loop
  - Loop body becomes kernel
  - Loop control becomes grid of threads
- Concentrated on basic implementation to date

# CUDA Acceleration

- Main problem is locating data
  - Flatten C arrays
  - Extract data from embedded structures
- Set up ‘mirror’ device arrays within data structures
- Add routines to transfer data between CPU and GPU
  - These are expensive

```
vis[i][j][k]
d_vis[k+nk*(j+nj*i)]
theArray->tile[i].coord.x
theArray->d_x[i]
```

**Need full CUDA pipeline**

# CUDA Acceleration

- Approach generally successful
- Loop bodies became kernels quite easily
  - Get good speed ups
- Some areas more troublesome
- Gridder was particularly problematic

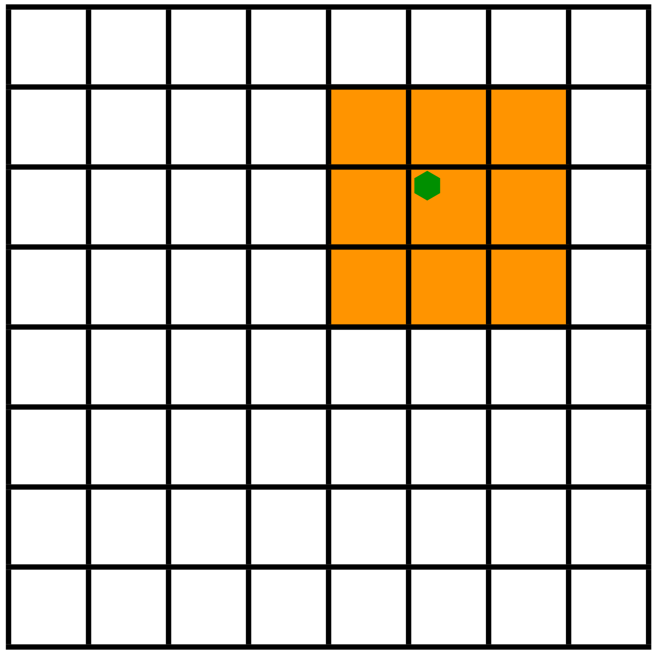
# Gridder

- Responsible for interpolating visibilities onto regular grid
- Convolves each visibility with compact kernel
- Can be written as scatter or gather

**Scatter**

For each visibility

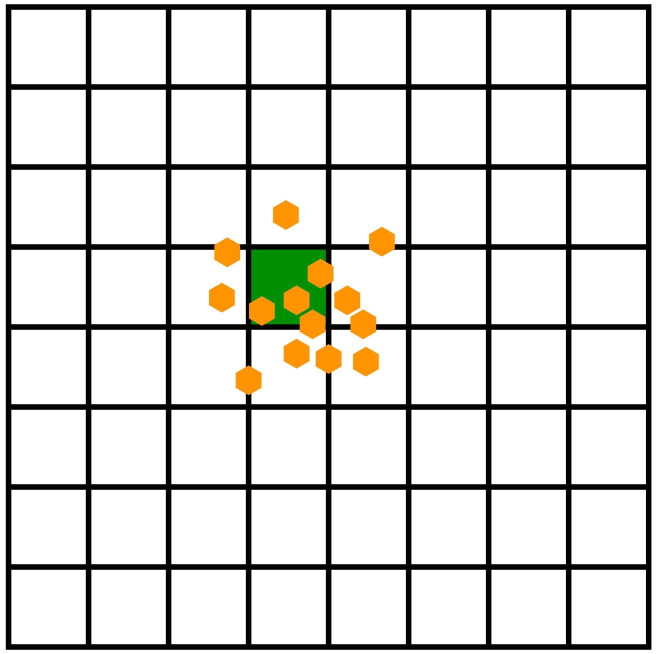
1. Compute affected pixels
2. Distribute convolved values



**Gather**

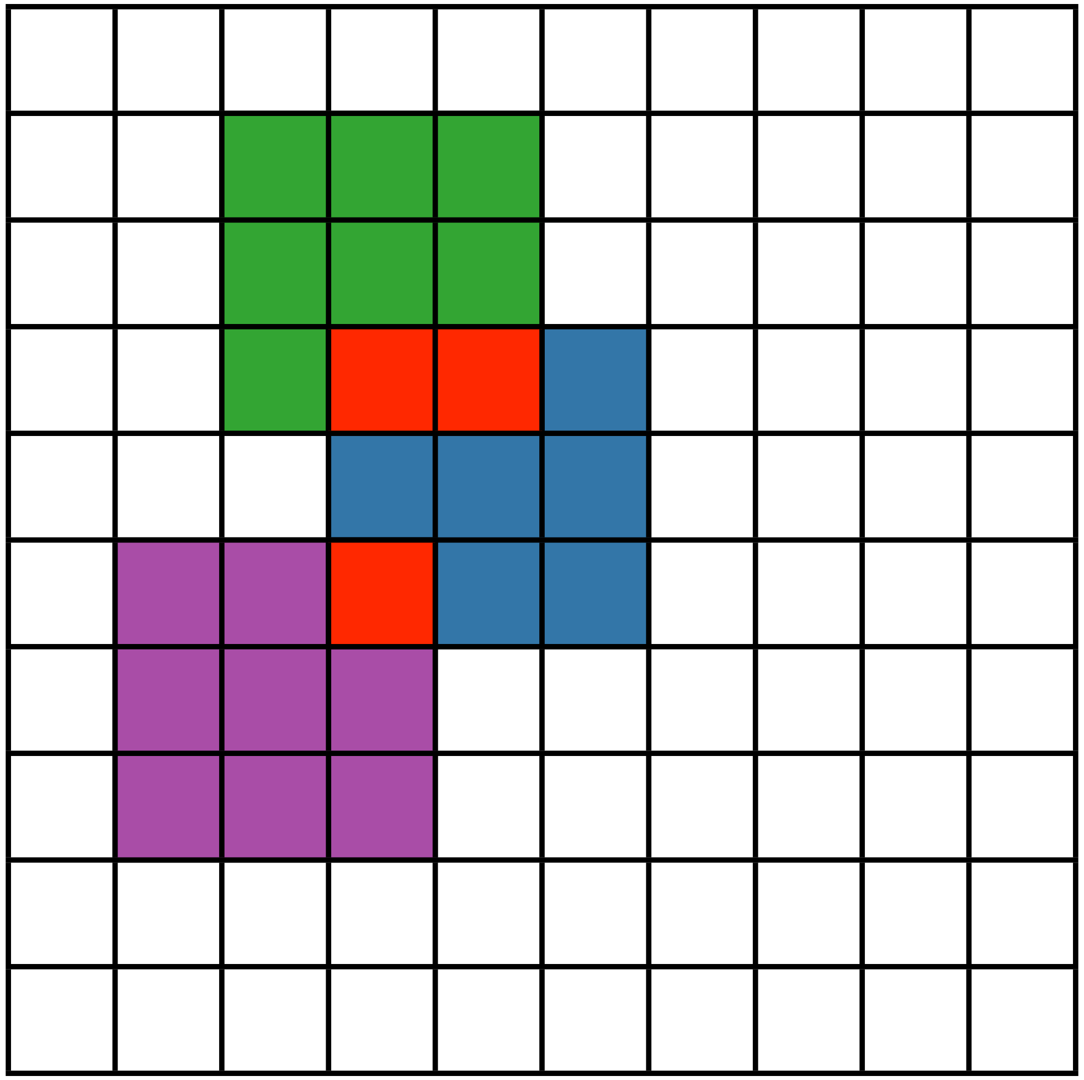
For each pixel

1. Compute visibilities in range
2. Pull in convolved values



# Scatter Race Condition

- Original CPU gridder written as a scatter
- Race condition in parallel
- Had to convert to a gather



# Gridder - Parallel Gather

- Easy to code a simple ‘gather’ algorithm
- No race condition, but slower than CPU
- Consider how many visibilities affect each pixel
    - 130,000 visibilities
    - 24x24 pixel kernel
    - 1600x1600 pixel image
    - Find about 30 visibilities per pixel
- Most checks are wasted

# Gridder - Parallel Gather

- Solution is to bin the data
  - Assign visibilities to 24x24 pixel bins
  - Sort visibilities by bin
  - Make tables of ‘first’ and ‘last’ visibilities in a bin
- Threads use this to pare visibility list
- Extra set up cost, but runs much faster
- `thrust` library a great help

# Benchmarks

# Benchmarking Hardware

- Benchmarks performed on
  - 3 GHz Intel Xeon E5462 (Harpertown)
  - NVIDIA Tesla S1070
- Benchmarks for
  - Single channel
  - 5 calibration sources
  - 30 degree field of view ($1600^2$ pixels)

# Benchmarks - CML

| Stage | CPU (ms) | GPU (ms) |
|---|---|---|
| Clear Groups | 12.1 | 12.5 |
| Unpeel | 1489.6 | 9.5 |
| Rotate & Accumulate | 1397.2 | 10.3 |
| Scale | 70.9 | 1.1 |
| Measure Ionospheric Offset | 349.7 | 17.7 |
| Ionospheric Correction | 97.6 | 1.3 |
| Measure Tile Response | 1116.8 | 46.6 |
| Peel | 506.3 | 5.9 |
| **Total** | **5569.6** | **104.6** |

# Benchmarks - Gridder

| **Stage** | **CPU (ms)** | **GPU (ms)** |
|---|---|---|
| Prepare Spheroid | | 5.1 |
| Memory | | 18.2 |
| Locations | 41.6 | 0.3 |
| Bin | | 0.4 |
| Sort | | 6.8 |
| Reorder | | 1.5 |
| Lookup Table | | 0.1 |
| Convolve | 1282.7 | 152.0 |
| **Total** | **1324.3** | **186.6** |

# Benchmarks - Imager

| Stage | CPU (ms) | GPU (ms) |
|---|---|---|
| Conjugates | 79.4 | 1.8 |
| Send | 55.4 | 2.0 |
| FFT | 304.7 | 29.9 |
| Receive | 145.7 | 8.6 |
| **Total** | **587.9** | **42.3** |

# Benchmarks - Gridding Cleanup

| Stage | CPU (ms) | GPU (ms) |
|---|---|---|
| Make Corrector | 26.8 | 10.0 |
| Apply Corrector | 98.1 | 1.2 |

# Stokes Conversion

| **Stage** | **CPU (ms)** | **GPU (ms)** |
|---|---|---|
| Apply Transform | 438.1 | 6.6 |
| Retrieve Image | | 21.6 |
| **Total** | **438.2** | **28.2** |

# Regridder

| Stage | CPU (ms) | GPU (ms) |
|---|---|---|
| Send Regridding Information | | 54.6 |
| Perform Regridding | 730.7 | 26.9 |
| Retrieve Image | | 23.2 |
| **Total** | **730.7** | **104.7** |

# Benchmarks

- Overall speed up 18.7x
  - CPU failed to meet 8 s deadline
- Performance/$ improvement 11.7x
- Performance/W improvement 10.2x

# Summary

# Summary

- Have full CUDA pipeline
- Speedups impressive
  - Main problem was locating data
  - Further optimisations possible
- Code is MPI parallel
- Prototype deployment underway
  - Full deployment through 2010

# Summary

- MWA is new frontier in radio astronomy
  - Real time
  - Wide field of view
  - Huge data volume
- MWA science enabled by GPUs
  - High FLOP/sec
  - Low power draw
- MWA also pathfinder for Square Kilometer Array
  - Estimated 1 EFLOP pipeline

# Collaborators

- Center for Astrophysics
  - Daniel Mitchell
  - Stephen Ord
  - Randall Wayth
  - Lincoln Greenhill
- School of Engineering & Applied Sciences
  - Kevin Dale
  - Hanspeter Pfister