HAWC CDR

February 1-2, 2001

Data Processing Software

Rhodri Evans, Joel Kastner and Bob Loewenstein

HAWC Data-Taking Hierarchy and Terms: Stare & Nod

- **HAWC detectors: "sample-rate data"**
  - HAWC outputs images at ADC sample rate (1-4 kHz)
- **Chopping secondary: "chop-rate data"**
  - telescope secondary chops between source and reference positions at 10-20 Hz
- **Telescope nodding: "nod pair data"**
  - telescope nods to switch source & reference sky beams at < 1 Hz
  - multiple nods for each position in a dither series
  - total integration time of a few tens of sec per nod position
- **Telescope dithering: "dither series"**
  - telescope moves through a pre-defined sequence of nod positions
  - offsets are small, non-integral numbers of pixels, to smooth over detector nonuniformities
- **Telescope mapping: "map"**
  - telescope executes a sequence of dithered stare & nod observations
  - images obtained for a grid of map points
  - large separations between points (of order 80% of detector dimensions)

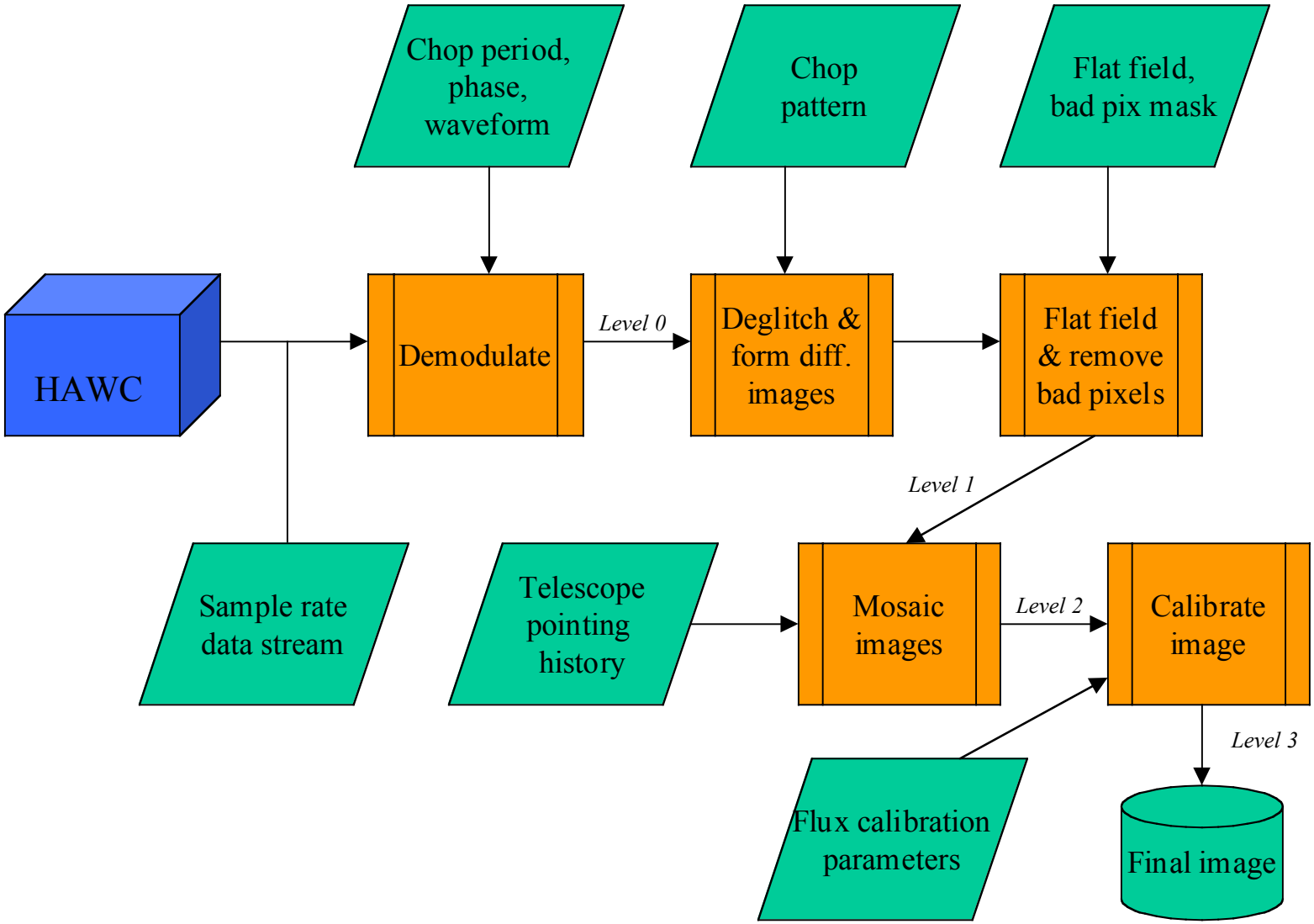
Table 1: HAWC Mapping (with dither) AOT

Parameter	Data Type	Example script line [2]	Observing Step (see above)
Observing mode	Mode	DITHER_OBSMODE set	1
Source position (RA)	(h, m, s, .ss)	23 34 45 00 S_RA set	1
Source position (dec)	char (north/south=+/-) integer (d, m, s, .ss)	+ 19 20 30 00 S_DEC set	1
Dither[3]	Pattern	1 DITHER set	7
Mapping offset (RA)	floating point, (arcsec)	175.0 M_RA set	1
Mapping offset (dec)	floating point, (arcsec)	175.0 M_DEC set	1
# Map Columns	integer	3 M_COL set	1
# Map Rows	integer	3 M_ROW set	1
Chop amplitude	floating point, (arcsec)	20.0 CHOP_RA_1 set	1
Chop angle	floating point, (degrees)	0.0 CHOP_ANGLE set	1
Chop mode	word	2POINT CHOP_MODE set	1
Aperture	integer, (selection)	1 APER set	2
Filter (LWC)	integer, (selection)	1 FSWC set	3
Chop cycles per frame	integer	10 CPF set	4
Frames per nod (***n***)	integer	10 FPN set	5
Integration time	floating point, (sec)	1.0 ITIME set	5
# Nod Pairs/position	Integer	2 NPAIRS set	6
Required Source S/N	integer	50 SSN set	11
Required Fluxcal S/N	integer	100 FSN set	11
Estimated Time of Observation[3]	Integer		

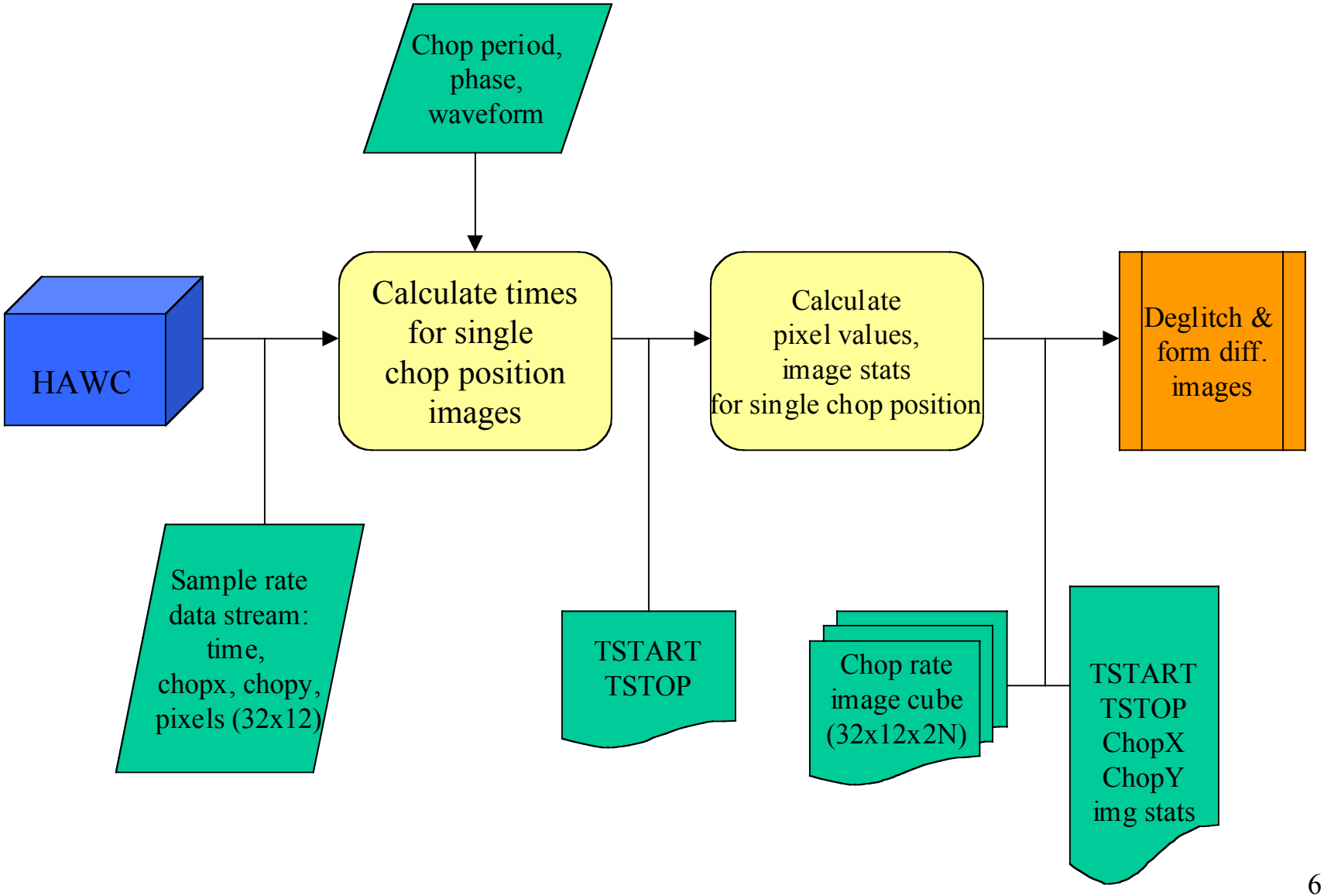
HAWC Data Processing Pipeline

- **Demodulate sample-rate (ADC rate) data to produce chop-rate image data**
- **Deglitch chop-rate data and generate chop-rate difference images**
- **Co-add chop-rate difference images for each telescope pointing in a "stare & nod" sequence**
- **Process coadded chop-rate difference images to remove detector artifacts & nonuniformities**
- **Form nod pair difference image for each position of a dithered series of telescope pointings**
- **Combine dithered image series into a mosaic & establish image coordinate system**
- **Combine dithered mosaics obtained over map grid into a map image**
- **Calibrate map image pixel intensities**

HAWC Data Pipeline: Stare & Nod Mode



Demodulation Process



Pipeline Data Products

Level 0: **Chop-rate image cube**, a 32 x 12 x 2N image (where N is the number of chop cycles per nod position)

Level 1: **Nod pair difference images** , integrated over M nod pairs (where M is large enough to cover substantial integration time)

Level 2: **Mosaiced image(s),** result of dithered set of stare/nod observations at one or more map positions

Level 3: **Calibrated map image** , result of ensemble of dithered images taken over map grid (flux-calibrated, with world coordinate system specified)

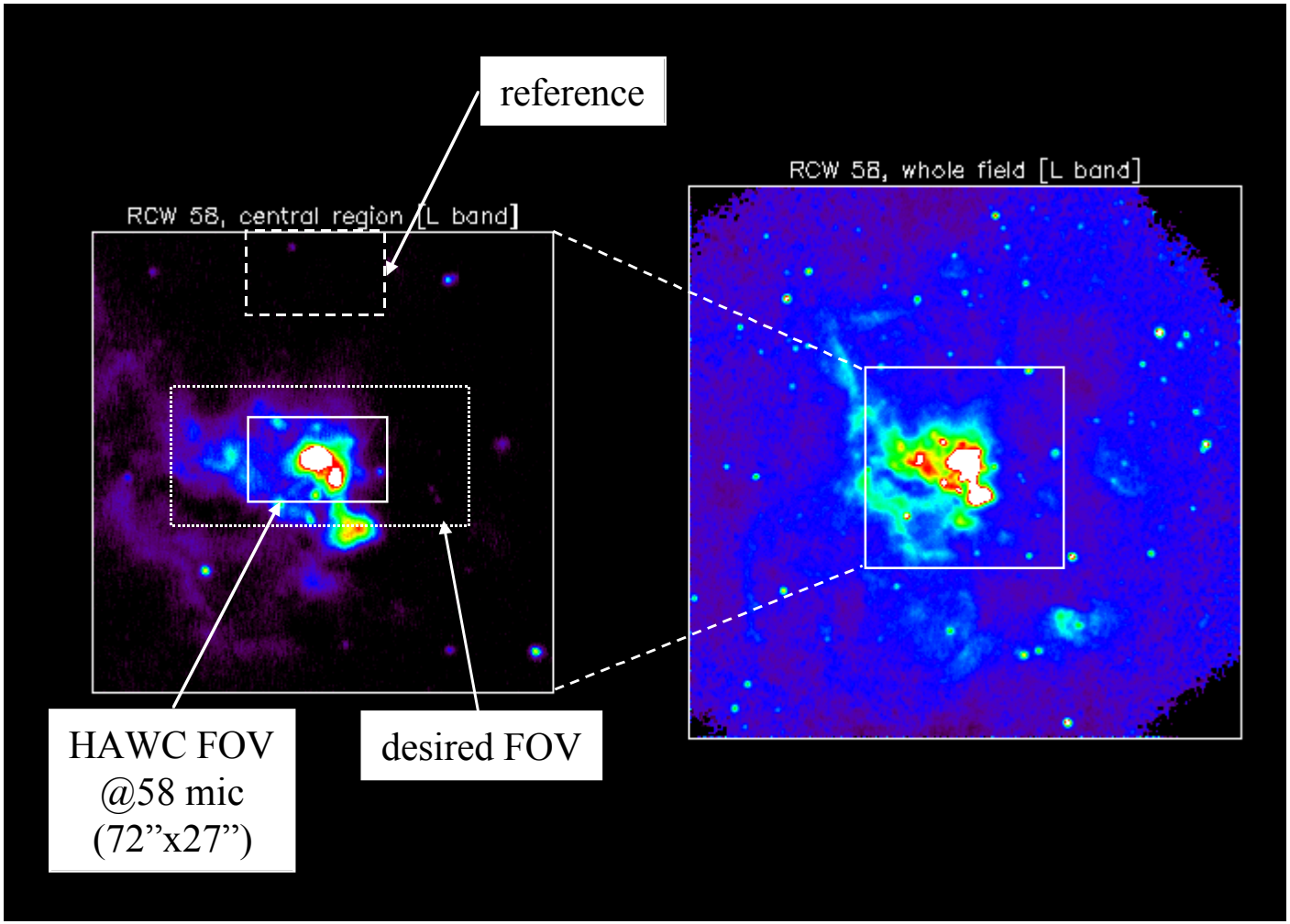
Pipeline Algorithm Development: Approach

- **Modular, extensible**
  - Pipeline as a set of stand-alone tools w/ well-defined inputs & outputs
  - isolation of I/O from data processing
- **Emphasis on rapid prototyping of algorithms**
  - IDL as prototyping language
  - migration to other languages (e.g., C or C++) where necessary (speed, memory considerations, e.g.)
- **reuse/adapt existing algorithm designs, code where feasible**
  - draw on extensive KAO experience (e.g., for deglitching schemes)
  - draw on experience with South Pole IR data system
  - investigating overlap with FORCAST pipelines
- **Use simulation tools to test & refine algorithms**

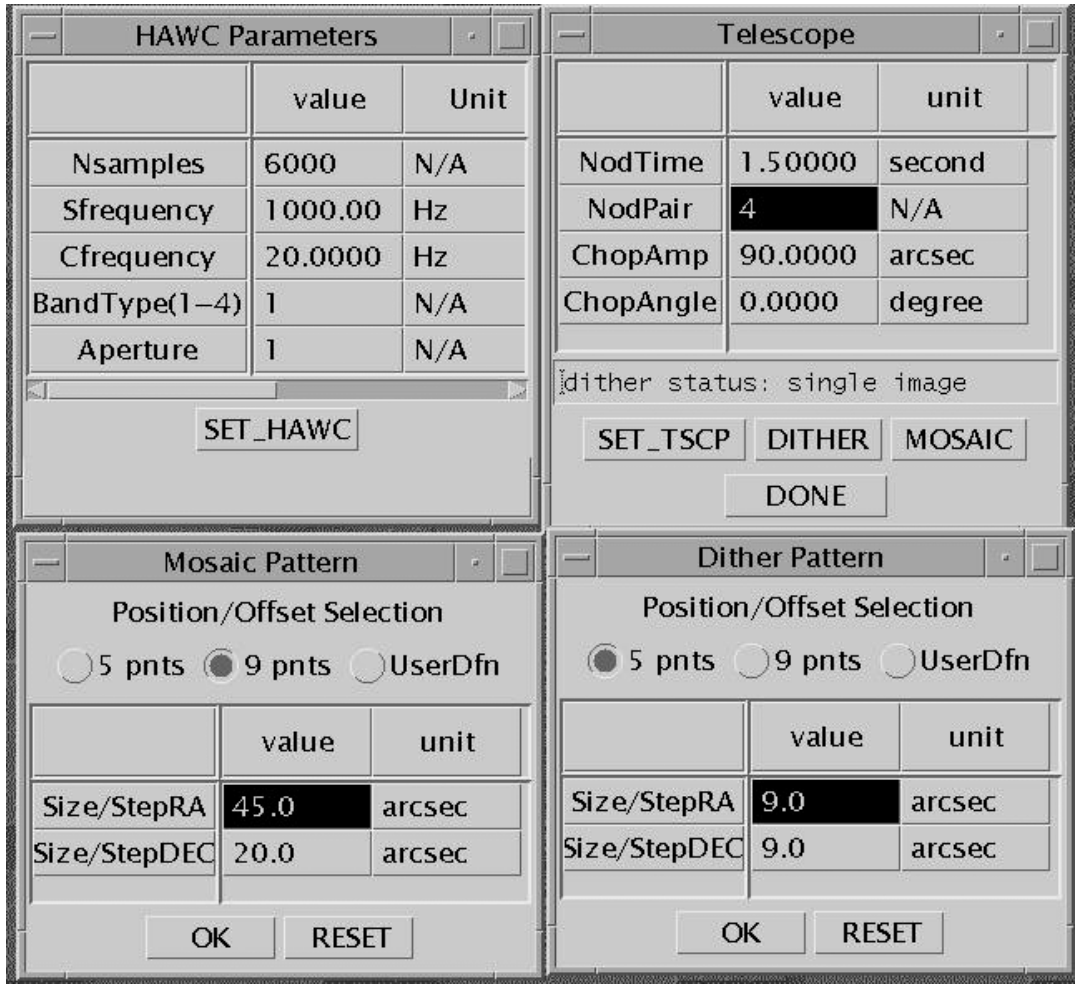
Pipeline Algorithm Development: Status

- **Simulator under development (in IDL)**
- **Prototype demodulator written**
  - prototyped in IDL; migrated to C to optimize speed & portability
- **Prototype image generator written**
  - makes use of algorithms from SPIREX/Abu (near-IR image) processing pipeline
- **Prototypes successfully integrated into the DCS**
  - integrated simulator and pipeline demonstrated at DCS Prototype Review (Nov. ‘00)

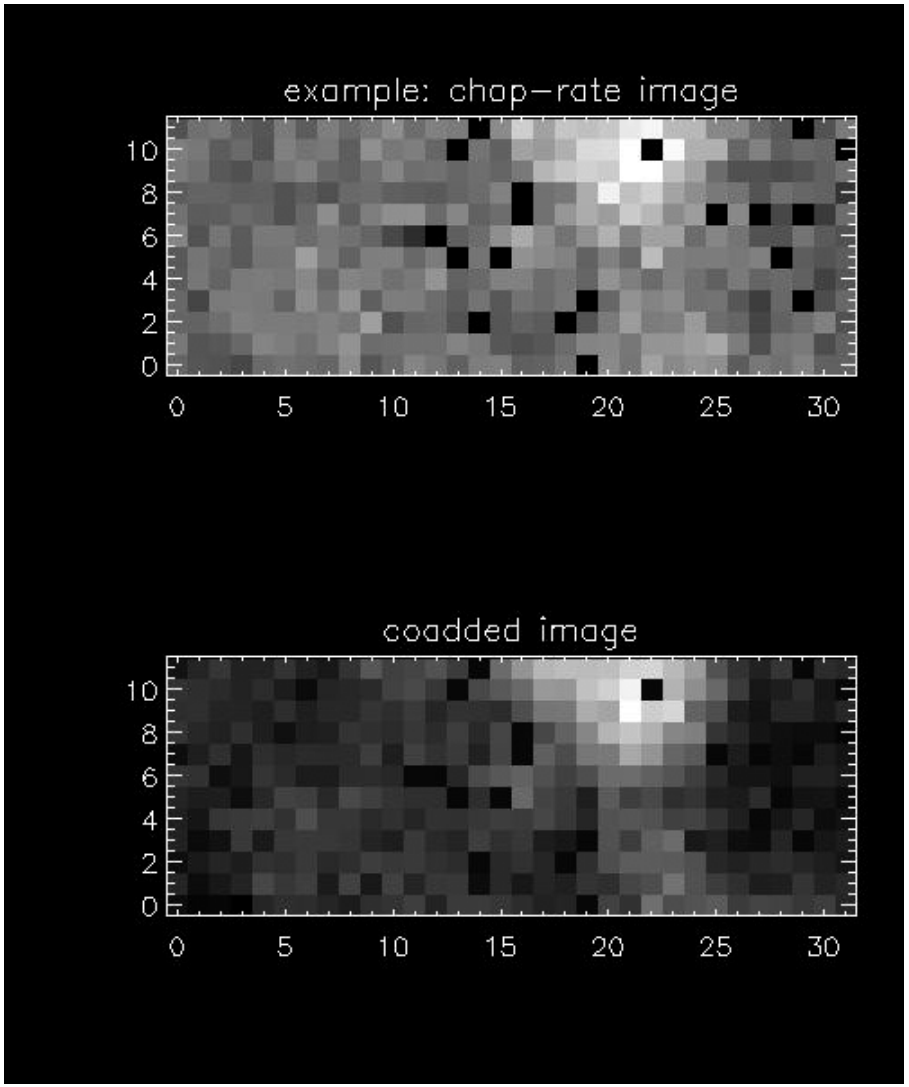
Example: Processing data from the HAWC Simulator



HAWC Simulator AOT GUI



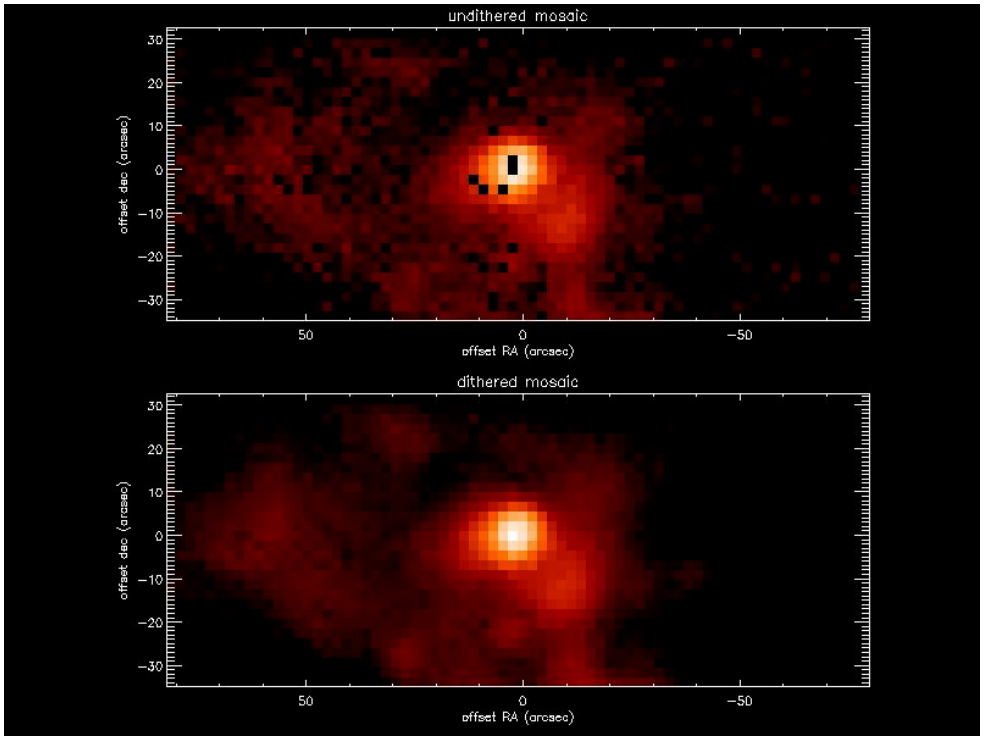
Level 0 & 1 output from the prototype pipeline



Top: difference of Level 0 (chop-rate) images

Bottom: Level 1 (nod pair difference) image

Level 2 output from the prototype pipeline



Shown: comparison of mosaics constructed from individual, coadded images (top) vs. dithered sets of coadded images (bottom)

The Level 3 HAWC data product: a *flux-calibrated* far-infrared image

- **Calibration models and algorithms under development at Yerkes**
  - as part of our filter optimization work we have already developed algorithms which take a source spectrum and determine the W/pixel at the detector after the source has passed through a model atmosphere, telescope and HAWC optics
- **Pipeline will incorporate calibration algorithms**
  - Yerkes models to serve as basis for algorithms
  - algorithms (code) & parameters (calibration data files) kept separate to enable improvements in data products as calibration knowledge improves
- **Simulator will facilitate calibration algorithm testing**
  - supports introduction of incrementally improving source, atmosphere, and instrument models