CTA Data Processing

Image cleaning benchmark

Jérémie Decock

CEA Saclay - Irfu/SAp

October 23, 2016

Baseline

Subject

How to assess image cleaning algorithms ?

(preprocessing for Hillas parametrization)

Baseline

Algorithms for image cleaning

So far, 3 algorithms:

- "Tailcut"
- DFT
- Wavelet Transform

Baseline

Cleaning algorithms evaluation

Evaluations based on MC simulations (gamma photons only):

1. The mean distance of normalized “cleaned” images to the actual normalized “clean” images (i.e. images without NSB and instrumental noise)
2. The mean distance of Hilas parameters computed on “cleaned” images to those computed on the actual “clean” images
3. The mean distance of events features (position and energy) computed on “cleaned” images to those given to simulations

MonteCarlo simulations

MC simulations

We need MC simulations to assess algorithms:

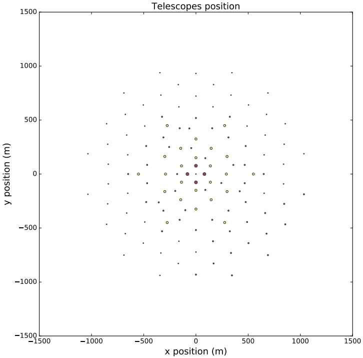
- So far, the priority is to make the dataset for the benchmark
- We can run MC simulations with Corsika/SimtelArray
  - The procedure is detailed on the SAp CTA wiki: `https://dsm-trac.cea.fr/cta/wiki/SAp`

Corsika/SimtelArray configuration

"CTA Prod3 demo" configuration

"CTA Prod3 demo" configuration

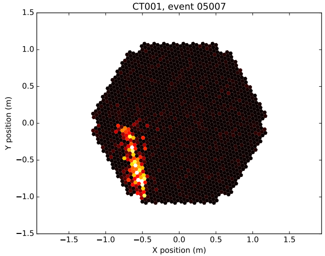
- "CTA Prod3 demo" configuration
  - 125 telescopes
  - Location: Paranal, Chile (altitude: 2150m)
- Cosmic rays: gamma photons
- Source: (20deg, 180deg)

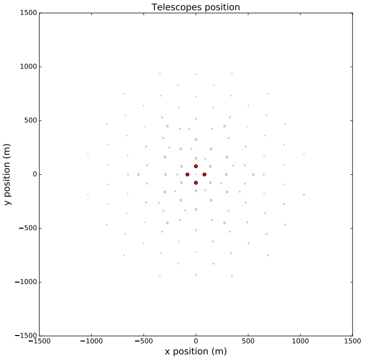


"CTA Prod3 demo" configuration

LSTCam

- Telescopes 1 to 4
- ? (hexagonal) pixels
- Optical focal length: ?m

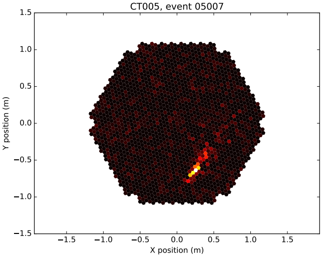


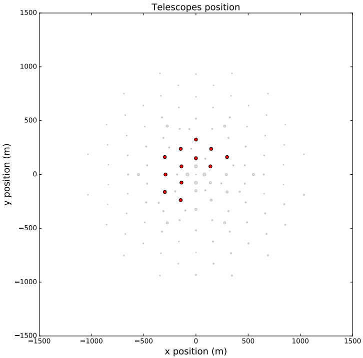


"CTA Prod3 demo" configuration

NectarCam

- Telescopes 5 to 16
- ? (hexagonal) pixels
- Optical focal length: ?m

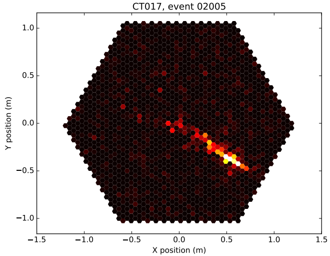


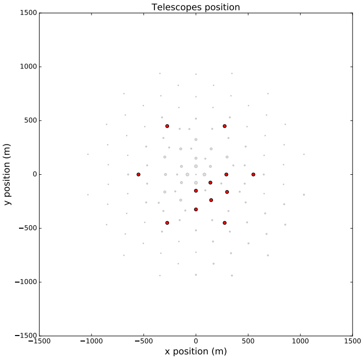


"CTA Prod3 demo" configuration

FlashCam telescopes

- Telescopes 17 to 28
- 1764 (hexagonal) pixels
- Optical focal length: 16.0m

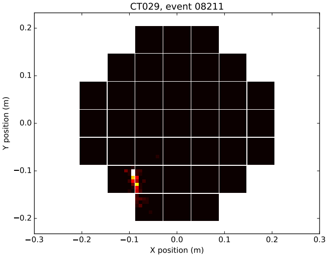


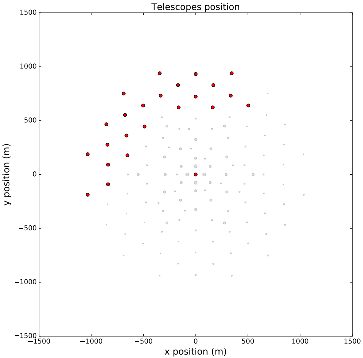


"CTA Prod3 demo" configuration

ASTRI telescopes

- Telescopes 29 to 52
- 2368 (rectangular) pixels
- Optical focal length: 2.15m

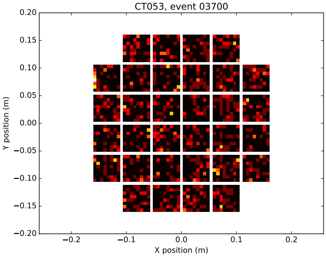


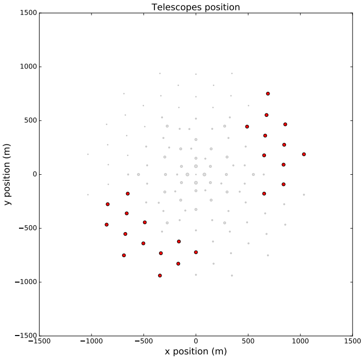


"CTA Prod3 demo" configuration

GATE

- Telescopes 53 to 76
- ? (rectangular) pixels
- Optical focal length: ?m

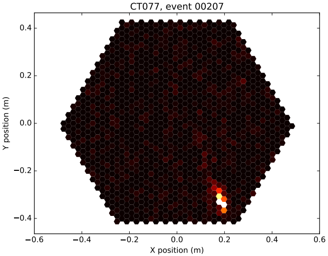


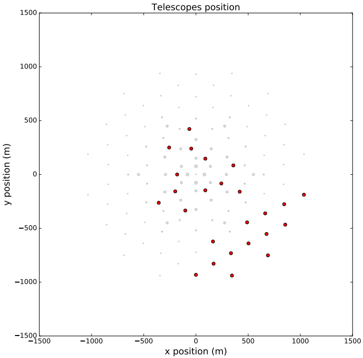


"CTA Prod3 demo" configuration

SST-1m

- Telescopes 77 to 101
- ? (hexagonal) pixels
- Optical focal length: ?m

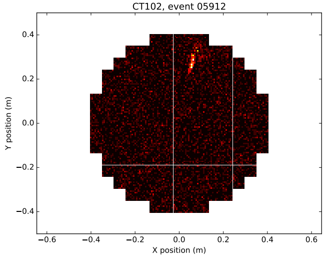


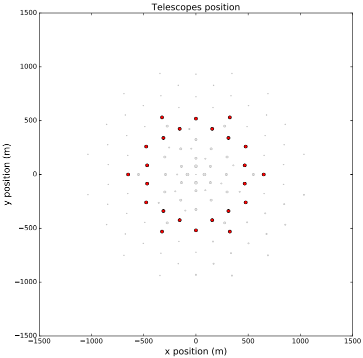


“CTA Prod3 demo” configuration

SCTCam

- Telescopes 102 to 125
- ? (rectangular) pixels
- Optical focal length: ?m



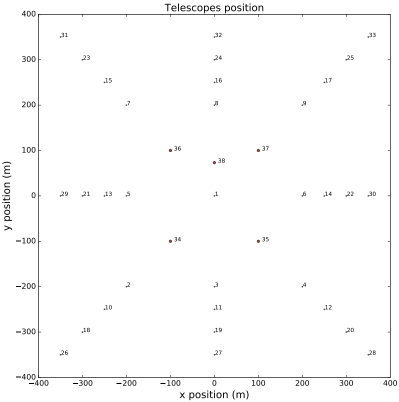


Corsika/SimtelArray configuration

"ASTRI mini-array" configuration

"ASTRI mini-array" configuration

- 38 telescopes
  - 33 ASTRI telescopes
  - 5 FlashCam telescopes
- Events:
  - 40204 γ-rays
  - 7580 protons
- γ-rays and protons are stored in separate files

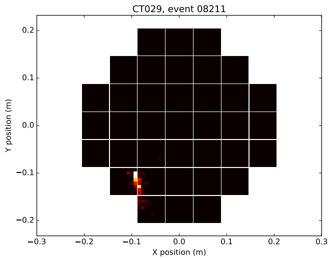


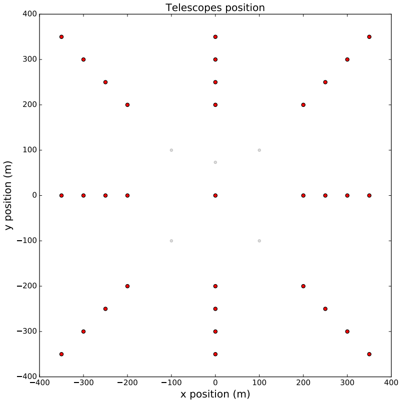
Simtel files are available on sapcta at /dsm/manip/cta/DATA/astri_mini_array/

"ASTRI mini-array" configuration

ASTRI telescopes

- Telescopes 1 to 33
- 2368 (rectangular) pixels
- Optical focal length: 2.15m

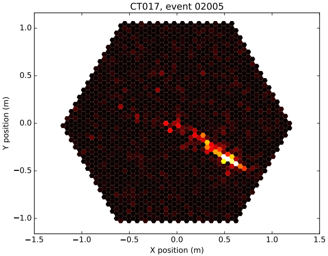


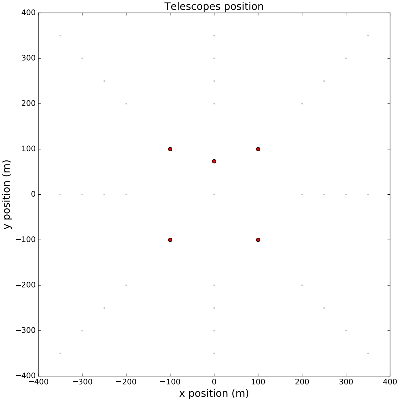


“ASTRI mini-array” configuration

FlashCam telescopes

- Telescopes 34 to 38
- 1764 (hexagonal) pixels
- Optical focal length: 16.0m





MC simulations

"ASTRI mini-array" configuration

Number of events per simtel files:

File	Num. events
gamma/run_1001.simtel.gz	4461
gamma/run_1002.simtel.gz	4567
gamma/run_1003.simtel.gz	4425
gamma/run_1004.simtel.gz	4401
gamma/run_1005.simtel.gz	4451
gamma/run_1006.simtel.gz	4451
gamma/run_1007.simtel.gz	4614
gamma/run_1008.simtel.gz	4423
gamma/run_1009.simtel.gz	4411

MC simulations

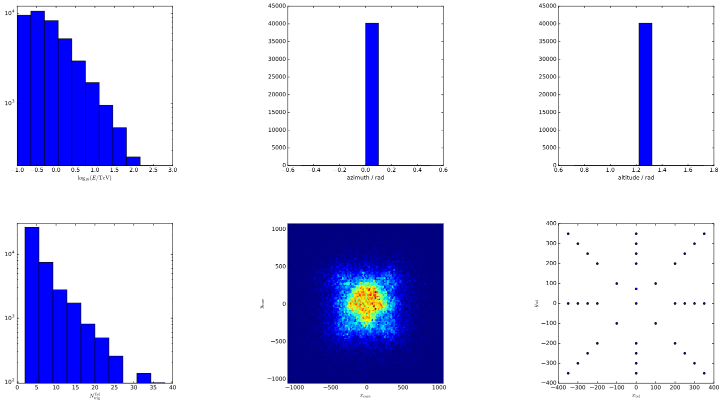
"ASTRI mini-array" configuration

Number of events per simtel files:

File	Num. events
proton/run_10000.simtel.gz	747
proton/run_10001.simtel.gz	680
proton/run_10002.simtel.gz	763
proton/run_10003.simtel.gz	792
proton/run_10004.simtel.gz	763
proton/run_10005.simtel.gz	776
proton/run_10006.simtel.gz	738
proton/run_10007.simtel.gz	749
proton/run_10008.simtel.gz	760
proton/run_10009.simtel.gz	812

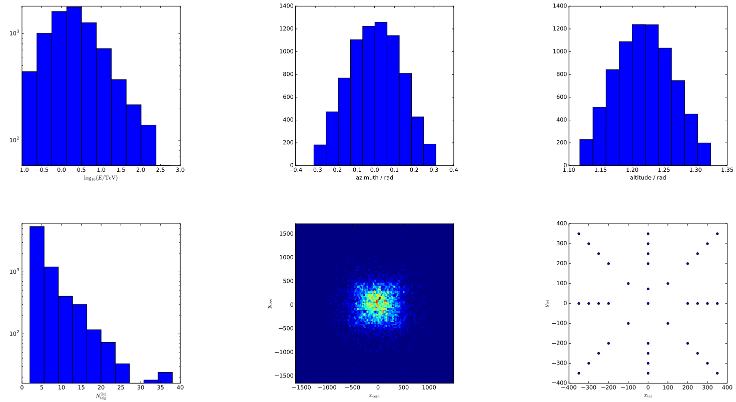
"ASTRI mini-array" configuration

Photons gamma



"ASTRI mini-array" configuration

Protons



Benchmark

First benchmark method

The mean distance of normalized images

The error function $\mathcal{E}$ is given by:

$$\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*) = \overline{|\varphi(\hat{\mathbf{s}}) - \varphi(\mathbf{s}^*)|}$$

Where:

- $\hat{\mathbf{s}}$ is the output image (the "cleaned" image) $\in \mathbb{R}^d$
- $\mathbf{s}^*$ is the reference image (the "clean" image) $\in \mathbb{R}^d$
- φ is a normalization function

$$\varphi(\mathbf{s}) = \frac{\mathbf{s} - \min(\mathbf{s})}{\max(\mathbf{s}) - \min(\mathbf{s})}$$

First benchmark method

The mean distance of normalized images

The reference image $\mathbf{s}^*$ is the best expected “cleaned” image, i.e. $\mathbf{s}^*$ is the raw input image $\mathbf{s}$ without:

- the instrumental noise
- the background noise

How do we get it from simulations ?

First benchmark method

The mean distance of normalized images

We simply use *photoelectron* images

Each image **s** in simtelarray files have an equivalent *photoelectron* image that we use as reference $\mathbf{s}^*$

In ctapipe:

- **s** := event.dl0.tel[tel_id].adc_sums[channel]
- $\mathbf{s}^*$:= event.mc.tel[tel_id].photo_electrons

First benchmark method

The mean distance of normalized images

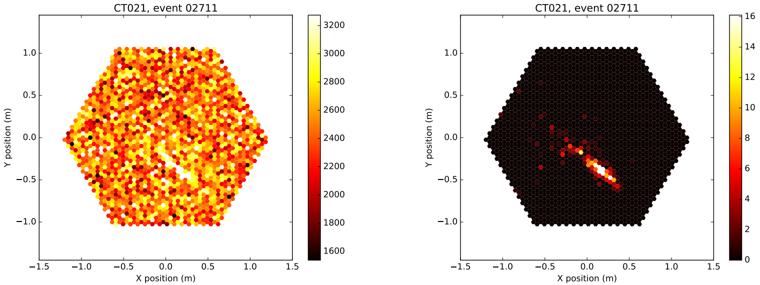


Figure: An example of input image **s** (left) and the corresponding reference image **s*** (right)

Second benchmark method

The mean distance of Hillas parameters

The error function $\mathcal{E}$ is given by:

$$\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*) = |H(\hat{\mathbf{s}}) - H(\mathbf{s}^*)|$$

Where:

- $H(\mathbf{s})$ returns the vector of Hillas parameters of $\mathbf{s}$
- $\hat{\mathbf{s}}$ is the output image (the "cleaned" image) $\in \mathbb{R}^d$
- $\mathbf{s}^*$ is the reference image (the "clean" image) $\in \mathbb{R}^d$

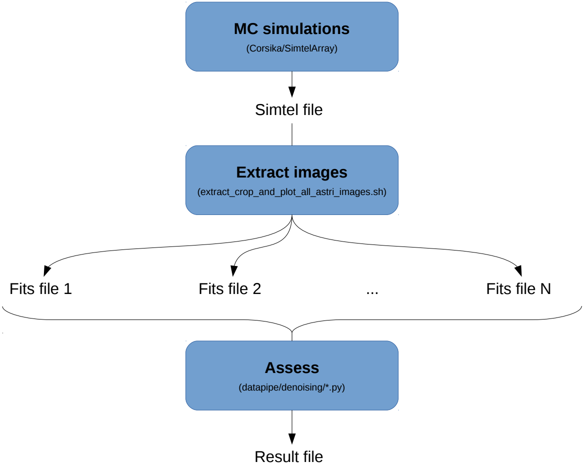
Optimize cleaning algorithms parameters

For each image cleaning algorithm f, compute its optimal parameter $\boldsymbol{\eta}^*$ on a training set of simulations

$$
\begin{aligned}
\boldsymbol{\eta}^* &= \arg\min_{\boldsymbol{\eta}} \left[\mathbb{E}_{\boldsymbol{\kappa},\boldsymbol{\Omega}} \left(\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*)\right)\right] \\
\hat{\mathbf{s}} &= f_{\eta}(\mathbf{s}) \\
\mathbf{s}^* &= \mathrm{MC}(\boldsymbol{\kappa}, \boldsymbol{\omega}_{shower}) \\
\mathbf{s} &= \mathbf{s}^* + \boldsymbol{\omega}_{NSB}(.) + \boldsymbol{\omega}_{instrument}(.) \\
\boldsymbol{\Omega} &:= (\boldsymbol{\omega}_{shower}, \boldsymbol{\omega}_{NSB}, \boldsymbol{\omega}_{instrument})
\end{aligned}
$$

with MC the simulation function, $\boldsymbol{\kappa}$ simulations parameters and $\boldsymbol{\Omega}$ random variables.

Workflow


MC simulations
(Corsika/SimtelArray)
Simtel file
Extract images
(extract_crop_and_plot_all_astri_images.sh)
Fits file 1
Fits file 2
...
Fits file N
Assess
(datapipe/denoising/*.py)
Result file

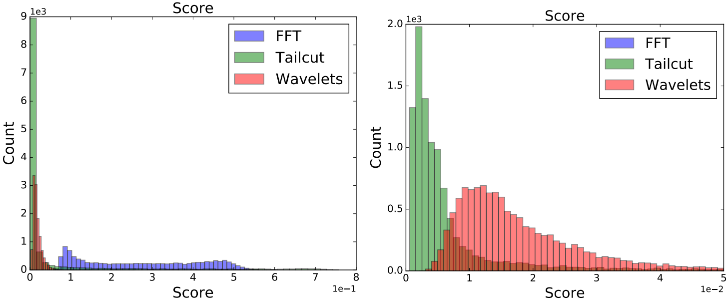
Results

Score

ASTRI mini-array on sapcta

First benchmark method

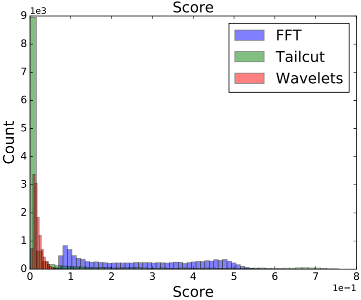
(Very) early results

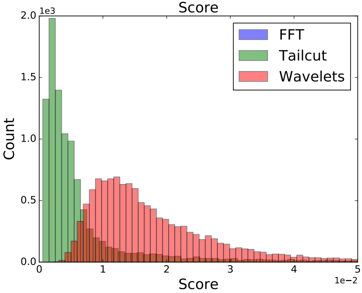


- ASTRI mini-array (**uncalibrated data**) - Telescopes 1 to 9 only
- **Non-optimal parameters** (quickly and poorly chosen)
- Polychromatic event set

First benchmark method

(Very) early results

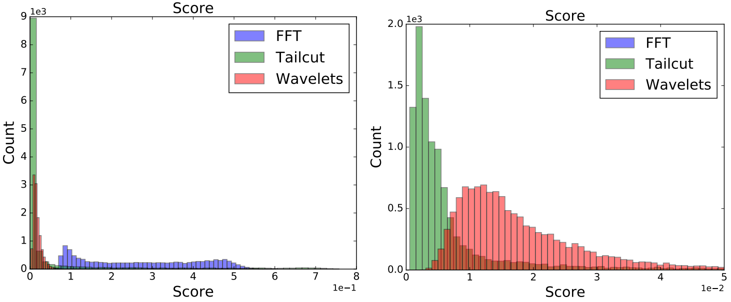




- Tailcut: JD's implementation
- FFT: Numpy implementation
- Wavelets: Cosmostat Sparce2D (mr_transform) b-Spline wavelet transform

First benchmark method

(Very) early results



- Hardware: Intel 24 cores @ 2.0GHz (unknown model - KVM) - 32Go
- Input files: sapcta:/dsm/manip/cta/DATA/astri_mini_array/fits/gamma/
- Num samples: 12016 images

Execution time

Time constraints

The official trigger rate in CTA North is $30000s^{-1}$ (source TDR)

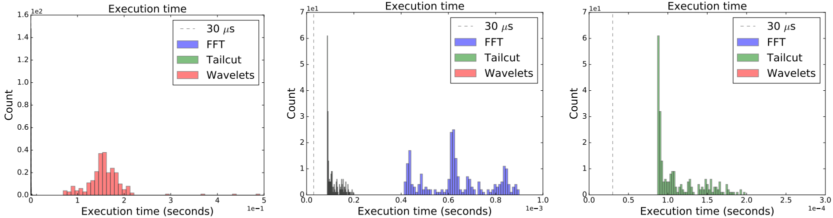
Thus 30 microseconds are available to have the event denoised and a rough reconstruction performed

Execution time

Prod3 demo on laptop (MacBook Pro)

Execution time

Measurements (MacBook Pro 9,2)



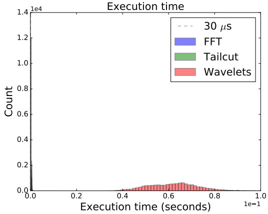
- Hardware: Intel Core i7 Ivy Bridge @ 2.9GHz - 8Go DDR3 1600MHz
- Input files: "Prod3 demo / Sim 13" (not shared)
- Num samples: 241 images

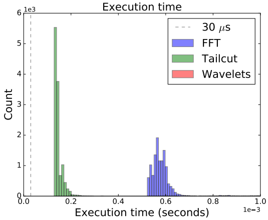
Execution time

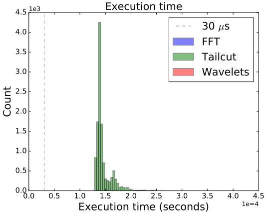
ASTRI mini-array on sapcta

Execution time

Measurements (sapcta)



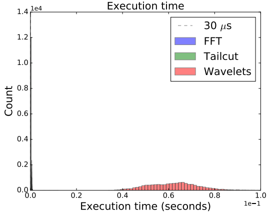


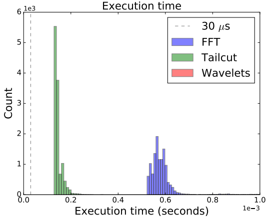


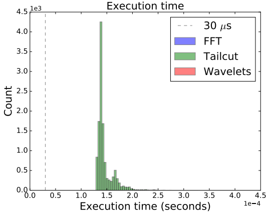
- ASTRI mini-array (**uncalibrated data**)
- Telescopes 1 to 9 only
- Polychromatic event set

Execution time

Measurements (sapcta)



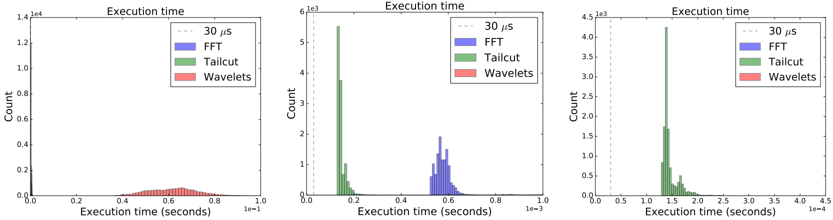




- Tailcut: JD's implementation
- FFT: Numpy implementation
- Wavelets: Cosmostat Sparce2D (mr_transform) b-Spline wavelet transform

Execution time

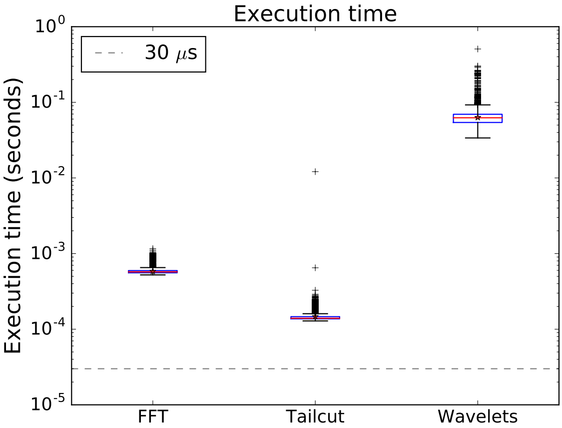
Measurements (sapcta)



- Hardware: Intel 24 cores @ 2.0GHz (unknown model - KVM) - 32Go
- Input files: sapcta:/dsm/manip/cta/DATA/astri_mini_array/fits/gamma/
- Num samples: 12016 images

Execution time

Measurements (sapcta)



Execution time

Axis of improvement

- Check execution time on sapcta [ok]
- Check execution time of mr_recons and mr_filter
- Use mr_transform as a library [work in progress]
- Compile mr_transform with the best optimization flags
- Use fast linear algebra libraries in Sparce2D (like Blas)
  - Compare Blas, Eigen, ...
  - Test Blas with GPGPU ?
- Use OpenMP in Sparce2D (e.g. parallelize planes making)
- Check some other wavelet function (default=2 bspline WT)
- Check some other library
- Check FWT (Fast Wavelet Transform) algorithm ?

Execution time

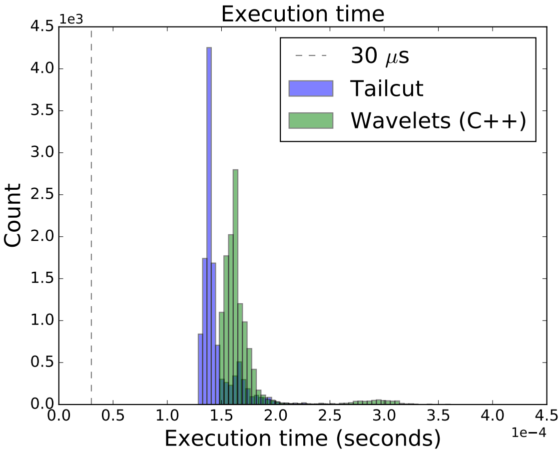
A more accurate measurement for wavelets

```
    struct timeval timev_start1, timev_end, timev_diff1;

    // Start benchmark here
    gettimeofday(&timev_start1, NULL);

    // Perform the transformation
    MR_Data.transform(Dat);

    // Stop benchmark here
    gettimeofday(&timev_end, NULL);
    timersub(&timev_end, &timev_start1, &timev_diff1);
    fprintf(stdout, "DELTATIME1 %ld.%06ld\n",
            timev_diff1.tv_sec, timev_diff1.tv_usec);
```

sapcta:/dsm/manip/cta/bin/ISAP_V3.1/cxx/sparse2d/src/mr_transform.cc

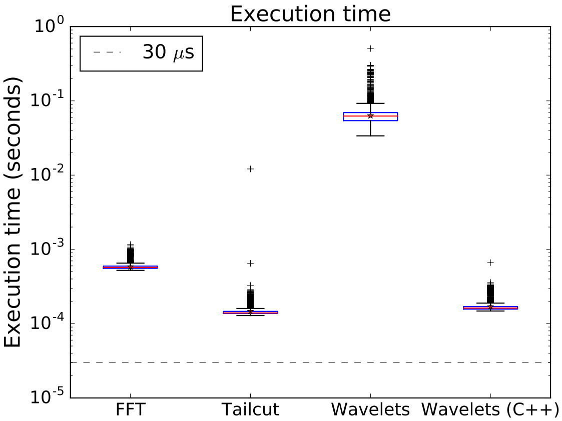
Execution time

A more accurate measurement for wavelets



Execution time

A more accurate measurement for wavelets



External papers

Papers

"Hadron suppression using Wavelet Transformations for the H.E.S.S. Telescope system" (2002, Stefan Funk)

Stefan's Paper

Subject

- Uses Wavelets for γ-ray/hadron separation
- Mention a little bit image cleaning but no experiments (e.g.section 3.3 and conclusion)

Stefan's Paper

Methodology

1. Add margins on the input image
2. Map the orthogonal camera coordinates into a hexagonal coordinate system
3. Apply the hexagonal wavelets to the hexagonal grid ; get wavelets coefficients for each scale
4. Compute the standard deviation of wavelet coefficients for each plane
5. Give these moments to the neural network used to discriminate γ-rays to hadrons (in addition to Hillas parameters)

Stefan's Paper

Baseline

- Neural Network with Hillas parameters as input
- Neural Network with Hillas parameters + Wavelet coefficients moments as input

The neural network is a Feed Forward Neural Network with 2 hidden layers (i.e. 4 layers in total)

Stefan's Paper

Baseline

Topologies of the used artificial neural networks (Table 3.1)

Name	NN topology (I-H-H-O)
wave6net	7 - 13 - 5 - 1
wave8net	9 - 15 - 5 - 1
hillnet	9 - 11 - 5 - 1
hillwave6net	11 - 17 - 5 - 1
hillwave8net	13 - 19 - 5 - 1
hillpointnet	6 - 12 - 8 - 1
hillwave6pointnet	12 - 16 - 8 - 1
hillwave8pointnet	14 - 18 - 8 - 1

Stefan's Paper

Baseline

Input parameters for the different neural networks: see Table 3.2

Stefan's Paper

Data

- Showers simulated with Corsika
- Between 100 GeV to 20 TeV for γ-rays
- Between 300 GeV to 30 TeV for protons
- Zenith angles: 0, 20 and 40 degree
- 3 classes of events to discriminate:
  - γ-rays
  - Isotropical protons (simulate a single telescope system)
  - Point source protons (simulate stereoscopy)
- Nearly 15000 events simulated per class (nearly 5000 for each zenith angle)

Stefan's Paper

Data

Input images:

- 960 pixels
- 32x32 hexagonal grid enlarged to 64x64 and 256x256 hexagonal grid

Stefan's Paper

Results

Hexagonal wavelets vs orthogonal wavelets:

- Hexagonal wavelets are OK
- "nearly no anisotropies between hexagonal wavelets and orthogonal wavelets"

Stefan's Paper

Results

Classification one large scales (planes 0, 1, 2):

- Can easily discriminate:
  - isotropically arriving protons
  - point source protons
- Can't easily discriminate:
  - point source protons
  - γ-ray sources

Classification one small scales (planes 4, 5, 6, 7):

- Can easily discriminate:
  - protons
  - γ-rays

Stefan's Paper

Results

Isotropically arriving protons:

- γ/hadron discrimination quality factor increase up to 80% with Wavelets

Point source protons:

- γ/hadron discrimination quality factor increase by nearly 10% only with Wavelets

Stefan's Paper

Execution time

Execution time on a Pentium III @ 800MHz (c.f. p.50):

- 64x64 pixels: 35Hz (28 ms/image)
- 256x256 pixels: 1.5Hz (666 ms/image)

16 times more pixels $\rightarrow$ 24 times more time

TODO

- What is the complexity class (i.e. computation time with respect to inputs) of Wavelets Transforms ?
- $O(n)$ for Fast Wavelets Transform (versus $O(n \log_2(n))$ for FFT) ?

Stefan's Paper

Implementation

Why does this work is not used in the HESS project ?

- Because this thesis has been written too late (the year HESS went into operation) ?
- Because the execution time is not compatible with the trigger rate ?
- Other reasons ?

Stefan's Paper

Discussion

- Point source protons vs isotropical protons: point source protons have additionnal Hillas parameter (impact point source) which is very discriminative
- Influence of zenith angle: high zenith angle $\leftarrow$ more energy $\leftarrow$ protons produces more subshowers (more discriminative)
-
-
-

Stefan's Paper

Ideas

- Use directly second order moments of the Wavelet coefficients distribution as an input of the classifier in addition to the Hillas parameters
- To shorten computation time: S.Funk suggest to apply multi-scale analysis only to small scales ("scales 4 to 7")
- To "minimize boundary effects": add margins to the image (multiply the size of the image by 2 to 16 on each dimension)
- In section 3.3 – about image cleaning (not experimented) – S.Funk applies the same threshold to each plane

Stefan's Paper

Question

- What values are used in image margins ? Random values ? Which distribution ? Fixed values ? 0 ?
- How NN are trained ? Batch method ? Iterative method ?
- What is the stopping criteria ?
- Does results have been obtained with only one run ? What about stochasticity ?
- The dataset seems to small to avoid cross validation.

Conclusion

Work in progress and TODOs

- Get actual “clean” images [ok]
- Make an initial dataset [work in progress]:
  - Manage hexagonal pixels and non rectangular camera shapes [ok]
  - Calibrate telescopes [work in progress]
  - Run sanity checks on MC simulations [work in progress]
- Implement the 1st benchmark method [ok]
- Implement the 2nd benchmark method
- Optimize cleaning algorithms parameters
- Improve execution time for wavelet transforms (30 μs)

Issues

First benchmark method

Normalization function used in the first benchmark method

$$\varphi(\mathbf{s}) = \frac{\mathbf{s} - \min(\mathbf{s})}{\max(\mathbf{s}) - \min(\mathbf{s})}$$

Issue with this normalization function

Outliers may significantly change $\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*)$.

Issues

A possible solution to improve the first benchmark method

Replace:

$$\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*) = \overline{\text{abs}(\varphi(\hat{\mathbf{s}}) - \varphi(\mathbf{s}^*))}$$

with:

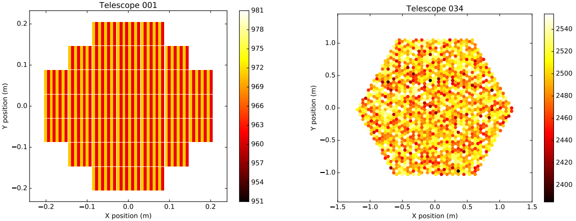
$$\mathcal{E}_1(\hat{\mathbf{s}}, \mathbf{s}^*) = \overline{\text{abs}\left(\frac{\hat{\mathbf{s}}}{\sum_i \hat{\mathbf{s}}_i} - \frac{\mathbf{s}^*}{\sum_i \mathbf{s}^*{}_i}\right)}$$

$$\mathcal{E}_2(\hat{\mathbf{s}}, \mathbf{s}^*) = \overline{\text{abs}\left(\sum_i \hat{\mathbf{s}}_i - \sum_i \mathbf{s}^*{}_i\right)}$$

$$\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*) = \left(\mathcal{E}_1(\hat{\mathbf{s}}, \mathbf{s}^*), \mathcal{E}_2(\hat{\mathbf{s}}, \mathbf{s}^*)\right)^T$$

Issues

ASTRI mini-array simulations



Pedestal:

- The pedestal seems very weird for all ASTRI telescopes: [971, 961, 971, 961, ..., 971, 961, 971, 961]
- Other telescopes seems OK

Issues

ASTRI mini-array simulations

FWI (for debug), the ADC is probably computed in sim_telarray/common/sim_signal.c at lines 1247 and 1260 (function create_pm_signals):

```
1247            signal = (int) (ch->sensitivity[ichan]*
1248                 rawsignal[ibin] + ch->pedestal[ichan] + pedestal_sysoff);
```

listings/sim_signal.c

I guess:

- signal := ADC
- sensitivity := gain
- rawsignal := PE

What is the difference between pedestal and pedestal_sysoff ?

Issues

Hillas parameters in ctapipe: bug ?

```
    # INIT PLOT ################################################################

    x, y = event.meta.pixel_pos[tel_num]
    foclen = event.meta.optical_foclen[tel_num]
    geom = ctapipe.io.CameraGeometry.guess(x, y, foclen)

    disp = ctapipe.visualization.CameraDisplay(geom, title='CT%d' % tel_num)

    # GET PHOTOELECTRON IMAGE (CLEAN IMAGE) ####################################

    disp.image = event.mc.tel[tel_num].photo_electrons

    # COMPUTE HILLAS PARAMETERS ################################################

    image = disp.image.copy()
    hillas = hillas_parameters(geom.pix_x, geom.pix_y, image)

    # PLOT #####################################################################

    disp.set_limits_percent(70)
    disp.overlay_moments(hillas, linewidth=3, color='blue')
    plt.show()
```

listings/plot_events_photoelectron_image_hillas.py

Tools and Documents

- Ctapipe: https://github.com/cta-observatory/ctapipe
- PyHessio: https://github.com/cta-observatory/pyhessio
- Cosmostat tools (iSAP/Sparse2D): http://www.cosmostat.org/software/isap/
- My scripts to manage simtel files: https://github.com/jdhp-sap/snippets
- My image cleaning scripts: https://github.com/jdhp-sap/data-pipeline-standalone-scripts
- Tino's scripts: https://github.com/tino-michael/tino_cta
- My other related presentations: https://github.com/jdhp-sap-docs

References I

Stefan Funk, *Hadron suppression using wavelet transformations for the hess telescope system*, Master's thesis, 2002.

Draft

First benchmark method (bis)

The mean distance ratios of images

The error function $\mathcal{E}$ is given by:

$$\mathcal{E}(\hat{\mathbf{s}}, \mathbf{s}^*) = \text{mean}\left(\frac{\text{abs}(\hat{\mathbf{s}} - \mathbf{s}^*)}{\mathbf{s}^*}\right)$$

(well of course it have to be adapted to avoid div by 0...)

Second benchmark method (bis)

Take into account the energy level

The error function $\mathcal{E}$ is given by:

$$\Lambda \sim \mathcal{X}_{\boldsymbol{\eta}}(\hat{\mathbf{s}}, \mathbf{s}^*).\mathcal{K}(\hat{E}, E)$$

$$\mathcal{X} \simeq \sum_i (\hat{\mathbf{s}}_i - \mathbf{s}^*{}_i)^2$$

$$\mathcal{K} \simeq \frac{\sqrt{(\sum_i \hat{\mathbf{s}}_i - \sum_i \mathbf{s}^*{}_i)^2}}{\sum_i \mathbf{s}^*{}_i}$$