# How to use Camera Particle Detector

## Introduction

This program gives a simple count rate of ionisation events in the camera using simply black tape and the Astro Pi. The pictures are analysed, the number of events is counted, and the count rate is shown on the 8x8 display.

Initial tests were carried out on school sources: Sr-90 was used as a beta source, and Am-241 was used as an alpha and gamma source. This provided images for the original testing of edge-detection and other algorithms, as well as an insight into suitable framerates. The beta source, even though it has a low activity, produced striking tracks reminiscent of the Brownian motion of dust particles on water. There were no tracks with the Am-241 source, which is consistent with the literature (Simon Platt, 2008), as alpha particles are not penetrating enough to reach the lens and gamma radiation does not cause detectable ionisation in Charge Coupled Devices (CCDs). We were also able to capture images of neutron bombardment on Raspberry Pi Camera at the ISIS beamline at the Rutherford-Appleton laboratory at Harwell, in four sets: one set with no neutrons, one set with thermal neutrons at up to 120meV, and two sets with fast neutrons ranging up to much higher energies. The energies expected in the region of the ISS are in the mega electonvolt region (ESA). The higher energy neutrons caused a greater number of events, out of proportion with the number of neutrons: this was because many more decay paths are possible at these energies producing protons which cause wandering tracks (colloquially known as worms (Groom)). A brief summary of the program is below.

## Methods and algorithms

The *counter.py* module obtains the number of particle events in an image. The image is blurred using a Gaussian filter to remove noise, and Canny edge detection is employed to produce a binary image. Canny edge detection finds the intensity gradients in the image: where the gradient is above a high threshold, this is an edge; furthermore, if the gradient is above a low threshold, and the line is connected to a pixel above the high threshold, this is also included in the edge. *findContours* returns a list of the contours of this binary image; the length of the list is the particle count.

*pixcheck* is a method for discarding data due to dead pixels and persistent noise. It uses the *bitwise\_and* function, which finds the overlapping pixels between two images. This is applied to a gallery of the 4 most recently captured images, and data which appear in all four images are subtracted. It subtracts grayscale pixel values from the input images before further processing, accounting for hot pixels but not random telegraph error.

## Dependencies:

We use opencv for analysis and picamera for capturing files.

#### OpenCV

Installation instructions can be found here:

<http://docs.opencv.org/doc/tutorials/introduction/linux_install/linux_install.html>,

On the day of submission the OpenCV documentation was not accessible; there is a cached version available at:

<http://webcache.googleusercontent.com/search?q=cache:AlUodNYFolsJ:docs.opencv.org/doc/tutorials/introduction/linux_install/linux_install.html+&cd=1&hl=en&ct=clnk&gl=uk>

We recommend the following installation:

sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev

sudo apt-get install python-dev python-numpy libjpeg-dev libpng-dev libtiff-dev libjasper-dev

and then extract opencv to a temporary directory

cd <temp directory>

mkdir release

cd ./release

cmake -D CMAKE\_BUILD\_TYPE=RELEASE -D CMAKE\_INSTALL\_PREFIX=/usr/local/opencv ..

sudo make –j4

sudo make install/strip

#### Picamera

After OpenCV installation run install.sh to install picamera

## Usage:

1. The program can be launched by start.sh, it launches the run.py file with name parameter RadiationPi which is the folder to which the initial captures and faulty pixel file are written.
2. The camera must be optically isolated, if possible cover the lens with black tape but use a double layer over the lens so that the adhesive is not in contact with it. If the number of subtracted pixels is too high, the camera is probably overexposed.
3. When the program launches, it will display a white C as it adjusts the framerate, white balance and ISO, and records the first four captures which are saved for dead pixel detection and later analysis on the ground after the mission.
4. The program then begins a continuous loop in which it displays the count rate in red and number of pixels subtracted from the images captured in blue.

# Bibliography

ESA. (n.d.). *Radiation Environment.* Retrieved from ESA: http://www.esa.int/Our\_Activities/Space\_Engineering\_Technology/Space\_Environment/Radiation\_environment

Groom, D. (n.d.). (Unwanted) Radiation Events in Astronomical CCD Images.

Simon Platt, Z. T. (2008). *Charge-Collection and Single-Event Upset Measurements at the ISIS Neutron Source.*