LSST Pixel Coordinate Definitions

There are three issues here:

> 1/ What's the index of the bottom-left pixel? 2/ What is meant by the bottom-left pixel? 3/ Where within that pixel is the point (0.0, 0.0)

I propose the we:

> 1/ Use 0-based coordinates, so the bottom-left pixel is (0, 0)
>
> 2/ Define the bottom left by:
>
> > Orientate the CCD so segment 00 (see Appendix) is at the bottom left corner of the device when looking at the sky; take its bottom left corner pixel to be the bottom-left pixel.
>
> This definition is LSST-centric; when using other devices we'll need to find the equivalent of the Appendix to define our geometry
>
> 3/ Use the FITS convention that the pixel centre is (0.0, 0.0), so the first pixel physically extends from -0.5 to 0.5

Notes:

1. This results in a right-handed coordinate system.
2. All CCD data must be stored in such a way that it can be mapped to the "silicon" coordinates define by the figure in the appendix by translations (no rotations or flips). When dealing with data from segments 10 --- 17 it is likely that someone will have to flip the data left-to-right; we need to define if this is a camera or DM responsibility.
3. These pixel coordinates run from 0 to NPIX-1 (ignoring overscan). There's no guarantee that there's a smooth mapping from them to millimeters in the focal plane; for example, the gap between segments is unlikely to be zero or even an integral number of pixels (and binning some detectors [e.g. SDSS] can lead to fractional-pixel offsets between segments). We should therefore expect that the WCS class will have to do something special (e.g. a lookup-table) to allow for such effects before fitting smooth functions.

Rationale:

Frank Valdes and Abi Saha point out the FITS IAU standard as a precedent for defining coordinate systems, adopting the bottom left pixel centre as (1.0, 1.0).

To address the first issue, while FITS is 1-based, that's basically because Fortran is. Almost all other languages (in particular C/C++/Python) aren't, and I think that that's a stronger argument; having to add 1 to all array indices is error prone as well as ugly.

On the second point. FITS takes the pixel centre to be (0.0, 0.0) which (as Abi points out) makes sense when

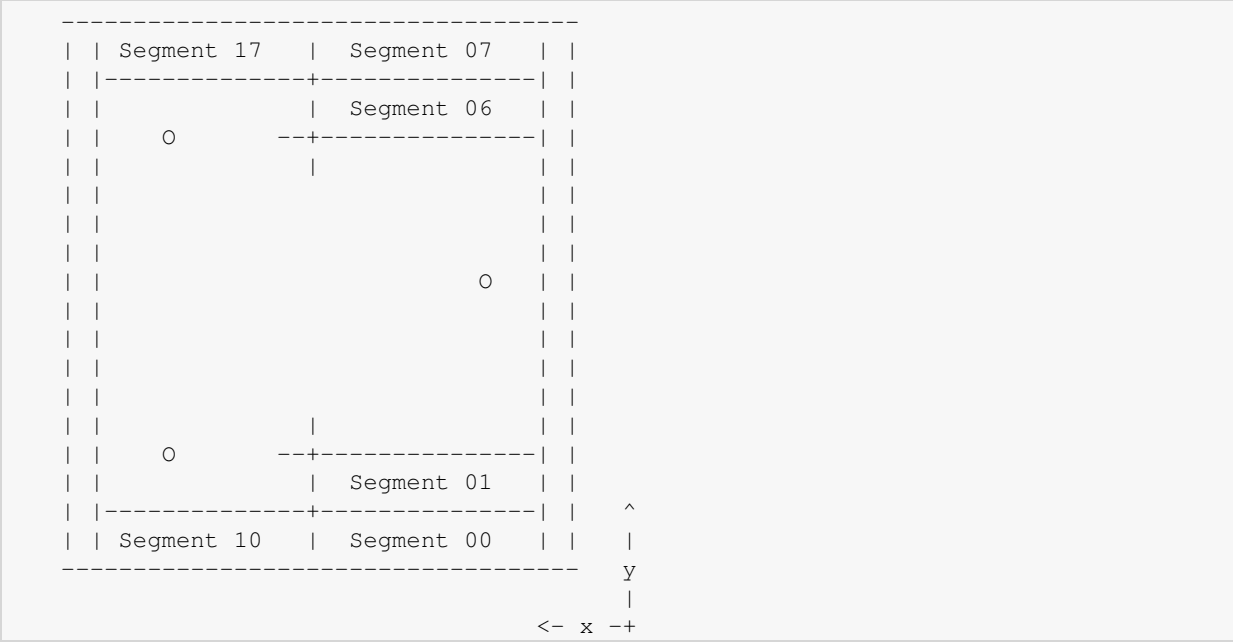
you realise that a CCD image samples a continuous (pixel-convolved) image at the position of each pixel centre.

SDSS went with the pixel centre being (0.5, 0.5) which means that you always have to add 0.5 to a pixel position to get the centre (and truncate, not round, a floating position to get a pixel index). We had no strong reasons to do this, and it was causing trouble. Jim Gunn was not happy (he still isn't) that we took this decision.

N.b. as regards e.g. ds9, it's easy to add an extra WCS to display pixel coordinates in any adopted convention

Appendix: Physical layout of CCD segments

Kirk Gilmore referred me to a document written by Martin Nordby (nordby@?) with a sketch of a CCD as viewed from L1, which is referred to as the sensor "back side":



The "O"s are sensor package mounts and the "| |" gutters at each side are the serial registers.

These are back-side illuminated chips with three mirrors, so that means that the looking-at-the-sky coordinate system is right-handed, with segment 00 at the bottom left.

Comments

Comment by ktl on Thu 26 Jun 2008 10:43:40 PM CDT

I'm a bit concerned about the "non-smoothness" of the mapping in the third "Note" above. Why is the coordinate system CCD-based and not amplifier-based (which would presumably be at least closer to "smooth"? For that matter (acting as devil's advocate), why is it not FOV-based? I can imagine that for some ISR computations there will be a need to work with physical image-plane coordinates; which pixel coordinates provide the simplest mapping for this?

Comment by rowen on Mon 14 Jul 2008 11:33:45 AM CDT

It seems strange to use a convention that is based on the whole CCD rather than the amplifier, yet does not have a smooth translation between pixel position and x,y position on the CCD. I can see alternatives:

- Use the amplifier-based convention: the pixel with index (0,0) is the first pixel read out. Then a full CCD image is a mosaic of amplifier images, each with its own WCS. I believe this is the conventional approach. This approach has the advantage that it is easy to handle overscan in the ISR pipeline, and it will simplify handling data from other surveys.
- Specify the position of the lower left corner of the CCD as (0.0,0.0), but increase pixel position linearly with actual physical position of the CCD. Thus the center position of pixels on the next amplifier will probably not end in .0. One will have to use a function to translate pixel index to pixel position (which I have long advocated) but the offset to this function will be stored in the metadata for each amplifier image (rather than being a project-wide constant), and a lookup table for this for a full CCD image. I suspect this proposal and the original proposal both will have very similar headaches if one is trying to process an entire CCD image at once (if the algorithm cares about position).

Comment by rowen on Thu 04 Sep 2008 12:02:08 PM CDT

In my opinion LSST code should NOT assume a convention for the mapping between pixel index and pixel position. There are inline functions in afw for converting between index and position, plus a constant describing the difference between the two. In my opinion all LSST code should be required to use these. This avoids the following common problems:

- A coder assumes the wrong convention and codes accordingly. RHL notes many examples of this. There are two conventions already in common use and RHL is proposing to adopt a third, which will exacerbate the problem.
- A coder assumes the right convention but codes it wrong. In particular, the transformation from position to index is notoriously easy to get wrong.
- A coder forgets the difference between index and position. This leads to code that is hard to understand and is often subtly incorrect.

Add comment

Your email or username: