Subject: drizzle function (?)
Posted by Helder Marchetto on Fri, 02 Dec 2016 14:23:23 GMT
View Forum Message <> Reply to Message

Hi,
I must make first a short intro: I'm not an astronomer and I found recently out that the Hubble telescope used a drizzle algorithm to achieve an imaging resolution higher then the pixel accuracy. As I would say, they have gone below the Nyquist frequency (1/2l with l being the pixel spacing).
This is useful when the imaging system delivers an image with higher resolution than what the detector can resolve *and* when the image can shifted arbitrarily on the detector with subpixel resolution.

Ok, so I would like to try that. If anybody has some code that does that, it would be super-cool, but assuming that I'm not that lucky:

- Anybody have a good reference where the math/physics of the image reconstruction process is described?
- Anybody have "practical" experience: how accurate does the subpixel shift have to be? Signal-to-Noise ratios?

I came across an article on David's page, but apparently David's article deals with array decimation and Wayne Landsman mentions:
- that there aren't any "drizzle or other flux conserving algorithms available in IDL.
- there is some C-code that can be linked to IDL. Anybody tried this???

Half way down I thought that this is an astronomy or math question rather than IDL, but at the end of the day I want to implement this in IDL, not C or something else.

Also, I don't use satellite images. I would be dealing with a set of n-images of (nx,ny) pixels and relative n-shifts (dx,dy). I can freely choose (dx,dy), but generally blow 1 pixel, if necessary higher.

Thanks for any help and have a nice weekend,
Helder

Subject: Re: drizzle function (?)
Posted by Russell[1] on Thu, 08 Dec 2016 21:12:03 GMT
View Forum Message <> Reply to Message

On Friday, December 2, 2016 at 9:23:26 AM UTC-5, Helder wrote:
> Hi,
> I must make first a short intro: I'm not an astronomer and I found recently out that the Hubble telescope used a drizzle algorithm to achieve an imaging resolution higher then the pixel accuracy. As I would say, they have gone below the Nyquist frequency (1/2l with l being the pixel spacing).
> This is useful when the imaging system delivers an image with higher resolution than what the detector can resolve *and* when the image can shifted arbitrarily on the detector with subpixel

resolution.
>
> Ok, so I would like to try that. If anybody has some code that does that, it would be super-cool, but assuming that I'm not that lucky:
>
> - Anybody have a good reference where the math/physics of the image reconstruction process is described?
> - Anybody have "practical" experience: how accurate does the subpixel shift have to be? Signal-to-Noise ratios?
>
> I came across an article on David's page, but apparently David's article deals with array decimation and Wayne Landsman mentions:
> - that there aren't any "drizzle or other flux conserving algorithms available in IDL.
> - there is some C-code that can be linked to IDL. Anybody tried this???
>
> Half way down I thought that this is an astronomy or math question rather than IDL, but at the end of the day I want to implement this in IDL, not C or something else.
>
> Also, I don't use satellite images. I would be dealing with a set of n-images of (nx,ny) pixels and relative n-shifts (dx,dy). I can freely choose (dx,dy), but generally blow 1 pixel, if necessary higher.
>
> Thanks for any help and have a nice weekend,
> Helder

Hi Helder.

I am an astronomer working with HST. I can point you to a few references, but before I do, let me just say a few things so we're talking the same language here. You probably already know this stuff, but again just so we're on the same page.

First "drizzle" refers to the algorithm for resampling the image. There are many resampling algorithms you might like (bilinear or Lancozs come to mind). You might resample for a host of reasons, such as the detector grid is highly distorted and you want a rectified image. Or maybe you want to shrink the pixels to improve the spatial resolution. To do this, particularly the latter, you need to "dither" the images. Here you take a multiple images which are offset from one another by tiny amounts (fractions of pixels), to inform you about scales smaller than the pixel scale (presumably this is what you mean about the dx,dy). For HST we're in both camps, distorted images and under sampled images, but before we can apply the "drizzle" algorithm, we must correct a few things to the native images first (such as aligning them and correcting for varying brightness of the background, and others).

The current implementation of this is whole package is called DrizzlePac (written mostly, if not entirely, in python). The website is:

http://drizzlepac.stsci.edu

but the manual for the package is:

http://documents.stsci.edu/hst/HST_overview/documents/DrizzI ePac/drizzlepac.pdf

I think what you're looking for is Chapter 2 of the manual. Though keep in mind this is highly astronomy-centric. In particular, how to deal with cosmic rays, aligning images on the sky (for example, how accurate are your dx,dy estimates), and applying weight images to the arithmetic operations. However, I think the nuts and bolts are there.

As for an IDL or C implementation.... Well, there's far less there. Alas, IDL is falling out of favor among many astronomers, and so drizzle hasn't entirely been ported. I've written a great deal of stuff, but it's highly tailored to my specific use case. Should you decide to embark on such a project, then I can tell you that there are 2-3 major pieces you'll need. First is the thing on Fanning's webpage about pixel collection. Second is something that does the computational geometry (ie. compute overlapping area of two pixel grids), for that look for JD Smith's thing called polyclip. The other things you'll need are more pedestrian, like how to keep track of what input pixel goes to what output pixel, and how to combine sets of input pixels into the output (i mean, do you want to average them, median them, etc.). For the astronomy purposes, this last step gets really tricky with dealing with transient things, like cosmic rays.

The final thing I"ll caution you with... By construction, you will correlate your output pixels. This will depend on parameters of the drizzle algorithm (for example in drizzle pac, they call this pixfrac). This correlated noise may or may not be bad for you, depending on the typical brightness of the things you're imaging. For astronomy, particularly when looking at very very faint things, this correlated noise becomes the dominate source of noise (here our objects have signal-too-noise ratios of order 3). So this sub pixel stuff will definitely affect your output noise properties (and hence the signal-t-noise ratios), but that's for you to judge. You can very very roughly consider this a trade --- you're trading clean noise properties for improved spatial resolution. THere's no free lunch here, you have to give something if you want to get something.

I hope this helps.
russell