Fitting PSFs across multiple images simultaneously with *fitpsf*

John Krist's *fitpsf* software has the ability to fit the same stellar source across multiple images simultaneously. In this mode, the software allows each PSF a unique x-tilt, y-tilt, background, and focus value but fits all other terms in common across the images.

Set up

The first several steps of the multi-image mode are exactly the same as the usual process for fitting PSFs in single images:

1. Begin by calling the `collect` command in an IDL session to create the list of input files.
2. Using `wfit` (or your own procedure for detecting a source in each image and defining the background level and bad pixels), produce the required `.dat` and `.cos` files for each input image.

New Input Files

At this point, we'd normally call `mkfocusin` to copy over a template `.in` file for the desired camera and chip as well as to define a `.pro` file with a series of sequential calls to `fitpsf`. We still need these files, but now they're defined slightly differently.

The `.in` file gains new tip, tilt, background, and focus entries for each of the input images (all other parameters are shared across images). For example, to run `fitpsf` on 4 WFC3UVIS1 images, the `.in` file should take the form:

```
      Array dimensions      512
            Wavelength   0.4107
        Fitting method        1
   Merit function power  2.00000
     Merit wing damping  0.00500
           Camera mode    WFC3UVIS1
          Zernike type OBSCURED
Y    1st file from parax focus (um)       0.0000
Y File  2 dFocus from 1st file (um)       0.0000
Y File  3 dFocus from 1st file (um)       0.0000
Y File  4 dFocus from 1st file (um)       0.0000
Y                  X-coma (microns)       0.0000
Y                  Y-coma (microns)       0.0000
Y           X-astigmatism (microns)       0.0000
Y           Y-astigmatism (microns)       0.0000
Y               Spherical (microns)      -0.0110
N                X-clover (microns)       0.0035
N                Y-clover (microns)      -0.0065
I X-spherical astigmatism (microns)       0.0000
I Y-spherical astigmatism (microns)       0.0000
N               X-ashtray (microns)       0.0045
N               Y-Ashtray (microns)      -0.0065
I   Fifth order spherical (microns)       0.0000
Y            Star  1 Background*1e4       0.0000
Y            Star  2 Background*1e4       0.0000
Y            Star  3 Background*1e4       0.0000
Y            Star  4 Background*1e4       0.0000
Y          Star  1 X-tilt (microns)       0.0000
```

```
Y           Star  1 Y-tilt (microns)       0.0000
Y           Star  2 X-tilt (microns)       0.0000
Y           Star  2 Y-tilt (microns)       0.0000
Y           Star  3 X-tilt (microns)       0.0000
Y           Star  3 Y-tilt (microns)       0.0000
Y           Star  4 X-tilt (microns)       0.0000
Y           Star  4 Y-tilt (microns)       0.0000
Y                                  Blur       0.4500
```

Note that the focus terms are now defined in terms of physical despace at the secondary (rather than in terms of RMS wavefront error at the camera), and the focus values after the first file are defined relative to the first file.

And the `.pro` file becomes a single call to fitpsf:

```
fitpsf, '@image_list',0,1,/EE
```

where the `image_list` file is a new input containing each image file per line (without the file extension). For example:

```
ict8p2ikq_flt
ict8p2iuq_flt
ict8p2j4q_flt
ict8p2jfq_flt
```

Since `image_list` is now the primary input to fitpsf, the `.in` file we defined above should be named to match: `image_list.in`.

Running *fitpsf*

fitpsf can now be called in IDL with `@do_image_list.pro`. This will write out `image_list.sum` and `image_list.par` files, which are identical to the single-image outputs, with the addition of a few new lines. These files can be parsed with a modified version of `focusresults.pro`.

Python wrappers

I've written Python code to automate defining the multi-image inputs and parsing the outputs. It also generalizes to the single-image case, so a single wrapper suffices for both modes. This functionality is documented in TBD.