UNIVERSITY OF HAWAII AT MĀNOA
Institute for Astrononmy

Pan-STARRS Project Management System

PSPhot Sofware Design Description
The Pan-STARRS IPP Object Photometry Tool

Coop. Agreement No.	**: FA9451-06-2-0338**
Prepared For	**: IPP**
Prepared By	**: Eugene Magnier**
Document No.	**: PSDC-430-021-DR**
Document Date	**: April 24, 2007**
Revision	**: DR**

DISTRIBUTION STATEMENT
Approved for Public Release – Distribution is Unlimited

Submitted By:

[Insert Signature Block of Authorized Developer Representative] Date

Approved By:

[Insert Signature Block of Customer Developer Representative] Date

1 Introduction

1.1 Overview

The Institute for Astronomy at the University of Hawaii is developing a large optical synoptic survey telescope system, the Panoramic Survey Telescope and Rapid Response System (Pan-STARRS). The science goals, priorities, top-level concept of operations with associated operational requirements, and system performance drivers with associated system performance requirements are described in the Pan-STARRS Science Goals Statement (SGS). As described in this document, The system conceptual design for Pan-STARRS utilizes an array of four 1.8m telescopes each with a 7 degree^2 field of view, giving the system an étendue larger than all existing survey instruments combined (defined as the product of the collecting area A multiplied by the field-of-view solid angle Ω). Each telescope will be equipped with a 1.4 billion pixel CCD camera with low noise and rapid read-out, and the data will be reduced in near real time to produce both cumulative static sky and difference images from which transient, moving, and variable objects can be detected. Pan-STARRS will be able to survey up to $\approx 6,000$ degree^2 per night to a detection limit of approximately 24^{th} magnitude. This unique combination of sensitivity and sky coverage will open up many new possibilities in time domain astronomy including a major goal of surveying the Potentially Hazardous Object (PHO) population down to a diameter of ≈ 300 meters. In addition, the Pan-STARRS data will be used to investigate a broad range of astronomical problems of extreme current interest concerning the Solar System, the Galaxy, and the Cosmos at large. A prototype single telescope system, PS-1, is being developed as a preliminary step before construction of the complete four telescope system.

Project sponsor:	AFRL, United States Air Force
Acquirer:	University of Hawaii Institute for Astronomy
User:	Astronomical community
Developer:	University of Hawaii Institute for Astronomy, participating institutions, and associated subcontractors

The Pan-STARRS Image Processing Pipeline is responsible for the basic analysis of images from the Pan-STARRS telescopes Gigapixel Camera. The overall goals and requirements of the Image Processing Pipeline are described in the IPP System/Subsystem Design Description (SSDD; PSDC-430-XXX) and the IPP System Requirements Specification (SRS; PSDC-430-XXX). Among the Pan-STARRS project survey goals is a repeated all-sky survey in 5 filters, *grizy*, beginning with a pre-survey with the prototype telescope PS-1. The photometric and astrometric precision goals for the all-sky surveys, as well as the other survey components, are quite stringent:

- relative photometry: 10 millimagnitudes scatter for bright stars across the sky in the internal photometric system;
- relative astrometry; 10 milliarcseconds scatter for individual stars between repeated images.
- absolute astrometry: 100 milliarcseconds scatter for all ICRS reference stars (Tycho).

An additional constraint on the Pan-STARRS system comes from the high data rate. The prototype telescope alone is expected to produce typically ~ 700 GB per night of imaging data. These images will not be limited to high galactic latitudes, so large numbers of measurable stars can be expected in much of the data. The combination of the high precision goals of the astrometric and photometric measurements and the high data rate (and a finite computing budget) mean that the process of detecting, classifying, and measuring the astronomical objects in the image data stream will be a significant challenge.

In order to achieve these ambitious goals, the object detection, classification, and measurement process must be both precise and efficient. Not only is it necessary to make a careful measurement of the flux of individual objects, it is also critical to characterize the image point-spread-function, and its variations across the field and from image to image. Since comparisons between images must be reliable, the measurements must be stable for both photometry and astrometry.

1.2 Comparable Programs

A variety of astronomical software packages perform the basic object detection, measurement, and classification tasks needed by the Pan-STARRS IPP. Each of these programs have their own advantages and disadvantages. Below we discuss some of the most widely used of these other packages, highlighting the features of the programs which are particularly desirable, and noting aspects of the programs which are problematic for the IPP.

- DoPhot : analytical fitted model with aperture corrections. pro: well-tested, stable code. con: limited range of models, algorithm converges slowly to a PSF model, limited tests of PSF validity, inflexible code base, fortran (P. Schechter)
- DAOPhot : Pixel-map PSF model with analytical component. pro: well-tested, high-quality photometry. con: Difficult to use in an automated fashion, does it handle 2D variations well? (P. Stetson)
- Sextractor : pure aperture measurement with rudimentary object subtraction. pro: fast, widely used, easy to automate. con: poor object separation in crowded regions, PSF-modeling is only beta (psfex), what models are available? (E. Bertin)
- apphot : IRAF-based aperture photometry. pro: widely used. con: IRAF-based, aperture photometry. (???)
- galfit : detailed galaxy modeling. not a multi-object PSF analysis tool. con: does not provide a PSF model, not easily automated. very detailed results in very slow processing. only a galaxy analysis program. (C. Impey)
- SDSS phot : con: tightly integrated into the SDSS software environment. (R. Lupton)

discussion of these packages is insufficient: flesh out discussion and add in the references.

Add discussion of the lessons learned from experience with previous analysis programs

The Pan-STARRS IPP team decided that none of the existing packages met all of their needs, particularly given the very challenging goals of the project. We decided to redesign the photometry analysis from scratch, using the lessons learned from the existing photometry systems. In the process, the object analysis software would be written using the data analysis C-code library written for the IPP, `psLib`, and the components of the photometry code would be integrated into the IPP's mid-level astronomy data analysis toolkit called `psModules`. The result is 'PSPhot', which can be used either as a stand-alone C program, or as one of the high-level IPP components of `psModules`, available to programmers either via a C interface or through a SWIG interface in Perl (or potentially Python).

2 PSPhot Design Goals

PSPhot has a number of important requirements that it must meet, and a number of design goals which we believe will help to make usable in a wide range of circumstances. The critical requirements of the Pan-STARRS IPP which drive the requirements for PSPhot:

- **10 millimagnitude photometric accuracy**. For PSPhot, this implies that the measured photometry of stellar objects must be substantially better than this 10 mmag since the photometry error per image is combined with an error in the flat-field calibration and an error in measuring the atmospheric effects. We have set a goal for PSPhot of 3mmag photometric consistency for bright stars between pairs of images obtained in photometric conditions at the same pointing, ie to remove sensitivity to flat-field errors. This goal splits the difference between the three main

contributors and still allows some leeway. This requirement must be met for well-sampled images and images with only modest undersampling.

- **10 milliarcsecond astrometric accuracy**. Relative astrometric calibration depends on the consistency of the individual measurements. The measurements from PSPhot must be sufficiently representative of the true object position to enable astrometric calibration at the 10mas level. The error in the individual measurements will be folded together with the errors introduced by the optical system, the effects of seeing, and by the available reference catalogs. We have set a goal for PSPhot of 5mas consistency between the true source postion and the measured position given reasonable PSF variations under simulations. This level must be reached for images with 250 mas pixels, implying PSPhot must introduce measurement errors less than 1/50th of a pixel. **the choice of F32 parameters places a numerical limit of 1e-7 on the accuracy of a pixel relative to the size of a chip (since a single data value is used for X or Y). For the 4800^2 GPC chips, this yields a limit of about 0.25 milliarcsecond.**

- **processing time of 45 seconds** This requirement depends strongly on the hardware organization, the amount of time spent on other analysis steps, the density of stars per image, and the depth for a given type of image. For the sources at the faint limit (eg, 5σ), the average density of sources is expected to be roughly 3×10^5 per square degree, while sources at the 20 σ level may have densities of $\sim 5 \times 10^4$ per square degree. Allowing 30 seconds for the PSPhot portion of the analysis, of which 15 is used for careful analysis of the brighter sources, 10 seconds is used for PSF modeling and other overheads, and the remaining 5 seconds is used for the PSF fitting of the faintest source implies that the detailed modelling may take roughly 3msec per source, and the basic PSF fitting may be allowed 150 usec per source.

The design goals for PSPhot are chosen to make the program flexible, general, and able to meet the unknown usages cases future projects may require:

- **Flexible PSF model** Different image sources require different ways of representing the PSF. Ideally, both analytical and pixel-based versions should be possible.

- **PSF spatial variation** Most images result in some spatial PSF variations at a certain level. The PSF representation should naturally incorporate 2-D variations.

- **Flexible non-PSF models** PSPhot must be able to represent PSF-like objects as well as non-PSF sources. It must be easy to add new object models as interesting representations of sources are invented.

- **Clean code base** PSPhot should incorporate a high-degree of abstraction and encapsulation so that changes to the code structure can be performed without pulling the code apart and starting from scratch.

- **PSF validity tests** PSPhot should include the ability to choose different types of PSF models for diffent situations, or to provide the user with methods for assessing the different PSF models.

- **Careful aperture corrections** PSPhot must carefully measure and correct for the photometric and astrometric trends introduced by using analytical PSF models.

- **User Configurable** PSPhot should allow users to change the options easily and to allow different approaches to the analysis.

3 PSPhot Analysis Process

3.1 Overview

The PSPhot analysis is divided into several major stages:

- **Image preparation** Load data, characterize the image background, load or construct noise and mask images.
- **Initial object detection** Smooth, find peaks, measure basic properties
- **PSF determination** Select PSF candidates, perform model fits, build PSF model from fits, select best PSF model class.
- **Bright object analysis** Fit objects with PSFs, determine PSF validity, subtract PSF-like objects, fit non-PSF model(s), select best model class, subtract model.
- **Low S/N sources** Detect low-level sources, measure properties (aperture or PSF)
- **Aperture corrections** Measure the curve-of-growth, spatial aperture variations, and background-error corrections.
- **Output** Write out objects in selected format, write out difference image, noise image, etc, as selected.

Note that a given run of PSPhot **should** allow the user to perform any of these stages as an option. For example, the PSF model may already be available from external information, in which case the PSF modeling stage can be skipped. Or, when used as a library function, the image may have already been loaded and the mask and weight images constructed. In some implementations, it may be possible to skip the initial object detection stage because only supplied sources are measured. These are only some of the possible configurations. The use of these different configurations depends on the source of the image, the desired detail and speed of the processing, and the level of accuracy desired from the analysis.

3.2 Image Preparation

The first step is to prepare the image for detection of the astronomical objects. We need three separate images: the measured flux, the corresponding noise level, and a mask defining which pixels are valid and which should be ignored. For the stand-alone program, the input flux image is a required program argument. When it is loaded, it is converted by default to 32-bit floating point representation. In the function-call form of PSPhot, the image must be supplied by the user in 32-bit floating point format. The noise and mask images may either be provided by the user, or they may be automatically generated from the input image, based on configuration-defined values for the image gain, read-noise, saturation, and so forth. For the function-call form of the program, the flux image is provided in the API, and references to the mask and noise are provided in the configuration information. As in the stand-alone C-program, the noise and mask may be constructed automatically by PSPhot.

For the mask, we use an 8-bit image in which a value of 0 represents a valid pixel. We use each of the 8 bits to define different reasons a pixel should be ignored. This allows use to optionally respect or ignore the mask depending on the circumstance. For example, in some cases, we ignore saturated pixels completely while in other circumstances, it may be useful to know the flux value of the saturated pixel. In addition, the mask pixels are used to define the pixels available during a model fit, and which should be ignored for that specific fit. The initial mask, if not supplied by the user, is constructed by default from the image by applying three rules: 1) Pixels which are above a specified saturation level are

marked as saturated (configuration keyword: SATURATE). 2) Pixels which are below a user-defined value are considered unresponsive and masked as dead. 3) Pixels which lie outside of a user-defined window are considered non-data pixels (eg, overscan) and are marked as invalid. The valid window is defined by the configuration variables XMIN, XMAX, YMIN, YMAX.

Mask values are currently hard-wired numbers. We need a method for user-defined mask values to be supplied. PSLib needs to have a mask registration system.

The noise image, if not supplied is constructed by default from the flux image using the configuration supplied values of GAIN and READ_NOISE to calculate the appropriate Poisson statistics for each pixel. In this case, the image is assumed to represent the readout from a single detector, with well-defined gain and read noise characteristics. In some obvious cases, this assumption will not be valid. For example, if the input flux image is the result of an image stack with significantly variable number of input measurements per pixel, it will necessary to supply a noise image which accurately represents the noise as a function of position in the image.

3.3 Initial Object Detection

The objects are initially detected by finding the location of local peaks in the image. The flux image is smoothed with a very small circularly symmetric kernel using a two-pass 1D Gaussian. At this stage, the goal is only to detect the brighter sources, above a user defined S/N limit (configuration keyword: PEAK_NSIGMA). The detection efficiency for the brighter sources is not strongly dependent on the form of this smoothing function.

Is this smoothing needed? we could save time here by skipping it.

The local peaks in the smoothed image are found by first detecting local peaks in each row. For each peak, the neighboring pixels are then examined and the peak is accepted or rejected depending on a set of simple rules. First, any peak which is greater than all 8 neighboring pixels is kept. Any peak which is lower than any of the 8 neighboring pixels is rejected. Any peak which has the same value as any of the other 8 pixels is kept if the pixel X and Y coordinates are greater than or equal to the other equal value pixels. This simple rule set means that a flat-topped region will maintain peaks at the maximum X and Y corners of the region.

The current implementation ignores the S/N map in making the peak detection. This code must be modified (a la Kaiser) to be used for a peak-detection pass in a difference image or to re-find peaks in the image after the modeled objects have been subtracted.

Once a collection of peaks have been identified, basic properties of the objects are measured. First, the local sky flux is measured within a square annulus with user-defined dimensions (INNER_RADIUS and OUTER_RADIUS), using the sample median. This local background value is then used to calculate the object first and second moments within a small user-defined aperture (MOMENT_RADIUS). The first-order moments are a good representation of the object position, while the second-order moments are a measure of the object shape. The second-order moments are somewhat sensitive to the size of the aperture and the accuracy of the background measurement. The moment calculation is only performed using pixels which exceed a S/N of 1. If, in the process of calculating the source moments, the S/N limits reject all but **3** or fewer of the source pixels, the peak is identified as being suspect, and is not used for further analysis. If the measured centroid coordinates differ from the peak coordinates be a large amount (MOMENT_RADIUS), then the peak is again identified as being of poor quality and is rejected. In both of these cases, it is likely that the 'peak' was identified in a region of flat flux distribution or many saturated or edge pixels.

3.4 PSF Determination

3.4.1 PSF Model vs Object Model

PSPhot uses an analytical model to represent the shape and flux of an object. An important concept within the PSPhot code is the distinction between a model which describes an object on an image and a model with describes the point-spread-function (PSF) across an image.

Any object in an image may be represented by some analytical model, for example, a 2-D elliptical Gaussian:

$$f(x,y) = I_o exp(-z) + S \quad (1)$$

$$R = \frac{(x-x_o)^2}{2\sigma_x^2} + \frac{(y-y_o)^2}{2\sigma_y^2} + \sigma_{\rm xy}(x-x_o)(y-y_o) \quad (2)$$

The object model will have a variety of model parameters, in this case the centroid coordinates (x_o, y_o), the elliptical shape parameters ($\sigma_x, \sigma_y, \sigma_{\rm xy}$), the model normalization (I_o) and the local value of the background (S). A specific object will have a particular set of values for these different parameters.

The point-spread-function (PSF) of an image describes the shape of all unresolved objects in the image. In a typical image, the shape of point sources is not well described by a single functional form; rather, the shape will vary as a function of position in the image. The PSF model therefore must describe the parameter variation as a function of the position of the object on the image. Note that the object model consists of a certain number of parameters which are defined by the PSF model, and another set of parameters which are independent from object to object. For the case of the elliptical Gaussian model, the PSF parameters would be the shape terms ($\sigma_x, \sigma_y, \sigma_{\rm xy}$) while the independent parameters would be the centroid, normalization and local sky values (x_o, y_o, I_o, S). PSPhot uses a 2-D polynomial to specify the variation in the PSF parameters as a function of position in the image. In the case of the elliptical Gaussian, this implies that the parameters are each a function of the object centroid coordinates:

$$\sigma_x = f_1(x,y) \quad (3)$$

$$\sigma_y = f_2(x,y) \quad (4)$$

$$\sigma_{xy} = f_3(x,y) \quad (5)$$

$$(6)$$

PSPhot uses a single structure to represent the object model and another structure to represent the PSF model. The object model structure consists of the collection of measured object model parameters, carried as a `psLib` vector (`psVector`) along with an equal-length vector with the parameter errors. The structure also includes an integer giving the identifier of the model used in the particular case, as well as model fit statistics such as the Chi-Square of the fit and the magnitude representation of the ratio between the model flux and an aperture flux (see below for more details on this value).

The PSPhot representation of the PSF consists of an array of polynomials, each representing the variation in the object model PSF parameters (`psArray` of `psPolynomial2D`). The PSF model structure also includes the same integer used to identify which model corresponds to particular instance of the PSF. At the moment, the number of PSF parameters is a fixed number (4) fewer than the number of parameters of the corresponding object model. For example, the elliptical Gaussian model uses 7 parameters to represent the object and 3 for the PSF model.

PSPhot is written so that the object detection, measurement, and classification code does not depend on the specific form of the available object model functions. Access to the characteristics of the models is provided through a simple function abstraction method. Throughout PSPhot, there are many places where it is necessary for the code to refer to an aspect of the object or PSF model. Often, these quantities are needed deep within other parts of the code. For example, when

attempting to fit the pixel flux values for an object, it is necessary to generate a guess for the model parameters. Or, in order to limit the domain of the fit, it is necessary to determine an isophotal radius for a model.

In order to avoid having the code depend on the specific form of a model, the function calls needed in these types of circumstances are abstracted, and a method is provided to return the necessary function to the higher-level software. For example, each model type has its own function to define an initial guess for the model, or a function to determine the radius for a given flux level. These are then registered as part of the model function code. Another function is then used to return the appropriate function for a specific model type. For example, the `psModelLookup_GetFunction` will return the `psModelLookup` function for a given model type. This mechanism makes it very easy to add new model functions into the PSPhot code base. To add a new model function, the programmer simply defines a new model name (a string), the set of all necessary model lookup functions, and places the reference to the model code at the appropriate location in the psModelInit.c routine.

When a new model is provided to PSPhot, it is not necessary to specify the intended use of the object model function (ie, PSF-like object, galaxy, comet, etc). Any model can be used for the PSF model, or to describe the flux distributions of the non-PSF objects. The code currently uses a fixed translation between the object model parameters and the PSF model parameters. It also defines a specific order for the 4 independent parameters.

the code may also require that two of the PSF-like parameters represent the shape in some way.

3.4.2 PSF Candidate Object Selection

The first stage of determining the PSF model for an image is to identify a collection of objects in the image which are *likely* to be PSF-like. PSPhot uses the object moments to make the initial guess at a collection of PSF-like objects. At this point, the program has measured the second order moments for all objects identified by their peaks, as well as an approximate signal-to-noise ratio. All objects with a S/N ratio greater than a user-defined parameter (`PSF_SHAPE_NSIGMA` ???) are selected by PSPhot, though objects which have more than a certain number of saturated pixels are excluded at this stage. PSPhot then examines the 2-D plane of σ_x, σ_y in search of a concentrated clump of objects. To do this, it constructs an artificial image with pixels representing the value of σ_x, σ_y, using a user-defined scale for the size of a pixel in this artificial image (note that the units of the σ_x, σ_y plane are the size of the second-moment in pixels in the original image). A typical value for the bin size is approximately 0.1 image pixels. The binned σ_x, σ_y plane is then examined to find a peak which has a significance greater than XXX. Unless the image is extremely sparse, such a peak will be well-defined and should represent the objects which are all very similar in shape. Other objects in the image will tend to land in very different locations, failing to produce a single peak. To avoid detecting a peak from the unresolved cosmic rays, objects which have second-moments very close to 0 are ignored. The only danger is if the PSF is very small and too many of these objects are rejected as cosmic rays.

Once a peak has been detected in this plane, the centroid and second moments of this peak are measured. All objects which land within XXX σ of this centroid are selected as likely PSF-like objects in the image.

3.4.3 PSF Candidate Object Model Fits

All candidate PSF objects are then fitted with the selected object model, allowing all of the parameters (PSF and independent) to vary in the fit. PSPhot uses the Levenberg-Marqardt process for the non-linear fitting. Non-linear fitting can be very computationally intensive, particularly for if the starting parameters are far from the minimization values. PSPhot uses the first and second moments to make a good guess for the centroid and shape parameters for the PSF models. In order to minimize the impact of close neighbors, the noise values used in the fit are enhanced by a fraction of the deviation of the particular pixel value from the model guess. Any objects which fail to converge in the fit are flagged as invalid.

does the noise enhancement introduce too much bias?

discuss the convergence criteria, model parameter guesses

For the resulting collection of object model parameters, the PSF-dependent parameters of the models are all fitted as a function of position to a 2-D polynomial. The order of this polynomial is (should be?) a user-defined parameter. The fitting process for these polynomials is iterative, and rejects the $3-\sigma$ outliers in each of three passes. This fitting technique results in a robust measurement of the variation of the PSF model parameters as a function of position without being excessively biased by individual objects which fail drastically. Objects whose model parameters are rejected by this iterative fitting technique are also marked as invalid and ignored in the later PSF model fitting stages.

All of the PSF-candidate objects are then re-fitted using the PSF model to specify the dependent model parameter values for each object. For example, in the case of the elliptical Gaussian model, the shape parameters ($\sigma_x, \sigma_y, \sigma_{xy}$) for each object are set by the coordinates of the object centroid and fixed (not allowed to vary) in the fitting procedure. The resulting fitted models are then used to determine a metric which tests the quality of the PSF model for this particular image.

The metric used by PSPhot to assess the PSF model is the scatter in the differences between the aperture and fit magnitudes for the PSF objects. The difference between the aperture and fit magnitudes (*ApResid*) is a critical parameter for any PSF modeling software which uses an analytical model to represent the flux distribution of the objects in an image. An approximate correction is measured here, with a more detailed correction measured after all object analysis is performed. The PSF model with the best consistency of the aperture correction is judged to be the best model.

3.4.4 Basic Deblending

The collection of identified peaks is examined to find peaks which are 'blended', that is, they are close enough together to make the analysis of one of the sources difficult if performed in isolation. Saturated stars also result in additional peaks which are likely to be invalid; it is useful to restrict a saturated star to a single primary position with associated neighboring peaks.

The deblending process first searches for any peaks which are within the image cell of another peak. All such groups are examined, starting with the brightest source. An isophot is found about the primary peak which is at least `DEBLEND_SKY_NSIGMA` times the sky sigma above the local background and which is otherwise `DEBLEND_PEAK_FRACTION` of the primary peak central pixel flux. Any secondary sources which are contained within this isophot are considered to be blended peaks associated with the primary peak.

3.5 Bright Source Analysis

After a PSF model has been determined, PSPhot performs the analysis of the bright objects in the image. In this stage, all of the objects with an estimated signal to noise (based on the moments analysis) greater than a user-set threshold are analysed and subtracted from the image. An optional successive stage then finds fainter sources and measures them as well (see Faint Source Analysis, Section **??**). In the bright source analysis stage, two major varients are available. In the primary version, all objects are examined (in decending order of brightness) and an appropriate models is determined for each object which is then subtracted; in the alternate version, the objects are examined (in decending order of brightness) and the PSF-like objects subtracted first, then the extended objects are analysed on a second pass.

3.5.1 Fast Ensemble PSF Fitting

Before the detailed analysis of the objects is performed, it is convenient to subtract off all of the sources, at least as well as possible at this stage. We make the assumption that all sources are PSF-like. We also assume their position can be taken as the peak of a 2D quadratic function fitted to the peak pixel and its surrounding 8 pixels. A single linear fit is used to simultaneously measure all source fluxes. Since the local sky has been subtracted, this measurement assumes the local sky is zero.

$$\chi^2 = \sum_{\text{pixels}} (F_{x,y} - \sum_{\text{sources}} A_i PSF[x,y])^2$$

Minimizing this equation with respect to each of the A_i values results in a matrix equation:

$$M_{i,j}\bar{A}_i = \bar{F}_j$$

where $\bar{A}_i$ is the vector of A_i values, the elements of $M_{i,j}$ consist of the dot product of the unit-flux PSF for source i and source i, and $\bar{F}_j$ is the dot product of the unit-flux PSF for source i with the pixels corresponding to source i. The dot products are calculated only using pixels within the source apertures. Since most sources have no overlap with most other sources, this matrix equation results in a very sparse, mostly diagonal square matrix. The dimension is the number of sources, likely to be 1000s or 10,000s. Such a matrix does not lend itself to direct inversion. However, an interative solution quickly yields a result with sufficient accuracy. In the iterative solution, a guess at the solution is made; the guess is multiplied by the matrix, and the result compared with the observed vector $\bar{F}_j$. The difference is used to modify the initial guess. This proces is repeated several times to achieve a good convergence.

Once a solution set for A_i is found, all of the objects are subtracted from the by applying these values to the unit-flux PSF.

3.5.2 PSF Model applied to detected objects

Once a PSF model has been selected for an image, PSPhot attempts to fit all of the detected objects, above a user-defined signal-to-noise ratio (**KEYWORD**) with the PSF model. For these fits, the dependent parameters are fixed by the PSF model and only the 4 independent object model parameters are allowed to vary in the fit. PSPhot again uses the Levenberg-Marqardt process for the non-linear fitting. The objects are fitted in their S/N order, starting with the brightest and working down to the user-specified limit.

Once a solution has been achieved, PSPhot attempts to judge the quality of the PSF model as a representation of the object shape. To do this, it calculates the next step of the minimization *allowing the shape parameters to vary*. This step, essentially the Gauss-Newton minimization distance from the current local minimum, should be very small if the object is well represented by the PSF, but large if the PSF is not a good representation of the object flux. The model quality is judged by the change in the two shape parameters which represent the 2D size of the object. For the case of the elliptical Gaussian, these two parameters are σ_x and σ_y. For a generic model, the shape parameters may be defined differently, but the should always be two parameters which scale the object size in two dimensions (what about a polar-coordinate form?) Currently, PSPhot requires the two relevant shape parameters to be the first two dependent parameters in the list of model parameters (ie, parameters 4 & 5).

The expected distribution of the variation of the two shape parameters will be a function of the signal-to-noise of the object in question and the value of the shape parameter itself. The expected standard deviation on the shape parameter is, eg, σ_x/SN. If the object is well-represented by the PSF, then the shape parameter values should be close to their minimization value. We can thus ask, for each object, given the measured amplitude of the Gauss-Newton step, how many standard

deviations from the expected value (of 0.0) is this particular value? Objects for which the variation in the shape parameters is a large positive number of standard deviations are likely to be better represented by a larger flux distribution than the PSF (eg, a Galaxy or Comet, etc). Objects for which the variation in the shape parameters is a large negative number of standard deviations are likely to be better represented by a smaller flux distribution than the PSF (ie, a cosmic ray or other defect). A user-defined number of standard deviations is used to select these two cases, and to flag the object as a likely galaxy (really meaning 'extended') or as a likely defect.

At this stage of the analysis, PSPhot uses two additional indicators to identify good and poor PSF fits. The first of these is the signal-to-noise ratio. It is possible for the peak finding algorithm to identify peaks in locations which are not actually a normal peak. Some of these cases are in the edges of saturated, bleeding columns from bright stars, in the nearly flat halos of very bright stars, and so on. In these cases, a local peak exists, with a lower nearby sky region. However, the fitted PSF model cannot converge on the peak because it is very poorly defined (perhaps only existing in the smoothed image). The fit can either fail to converge or it can converge on a fit with very low or negative peak flux / flux normalization. PSPhot will flag any non-convergent PSF fit and any object with PSF S/N ratio lower than a user-defined cutoff. It is also useful to identify very poor fits by setting a maximum Chi-Square cutoff for objects.

As the objects are fitted to the PSF model, those which survive the exclusion stage are subtracted from the image. The subtraction process modifies the image pixels (removing the fitted flux, though not the locally fitted background) but does not modify the mask or the noise images. The signal-to-noise ratio in the image after subtraction represents the significance of the remaining flux. If the subtractions are sufficiently accurate models of the PSF flux distribution, the remaining flux should be below 1 σ significance. In practice the cores of bright stars are poorly represented and may have larger residual significance. **in future work, we may choose to enhance the noise to minimize detection of objects in the residuals of brighter objects**.

3.5.3 Blended Sources

Sources which are blended with other sources are fitted together as a set of PSFs. A single multi-object fit is performed on all blended peaks. The resulting fits are evaluated independently and any which are determined to be PSFs are subtracted from the image.

3.5.4 Double Sources

Sources which are judged to be non-PSF-like are confronted with two possible alternative choices. First, the object is fitted with a double-source model. In this pass, the assumption is made that there are two neighboring sources, but the peaks are blended together, or otherwise not distinguished. The initial guess for the two peaks is made by splitting the flux of the single source in half and locating the two starting peaks at +/- 2 pixels from the original peak along the direction of the semi-major axis of the sources, as measured from the second moments. In order for the two-source model to be accepted, both sources must be judged as a valid PSF source. Otherwise, the double-PSF model is rejected and the source is fitted with the available non-PSF model or models.

better description of the acceptance criteria; the FLT model is tried before the DBL is accepted or rejected.

3.5.5 Non-PSF Objects

Once every object (above the S/N cutoff) has been confronted with the PSF model, the objects which are thought to be galaxies (extended) can now be fit with appropriate models for the galaxies (or other likely extended shapes). Again,

the fitting stage starts with the brightest sources (as judged by the rough S/N measured from the moments aperture) and working to a user defined S/N limit.

PSPhot will use the user-selected galaxy model to attempt the galaxy model fits. In the configuration system, the keyword `GAL_MODEL` is set to the model of interest. All suspected extended objects are fitted with the model, allowing all of the parameters to float. The initial parameter guesses are critical here to achieving convergence on the model fits in a reasonable time. The moments and the pixel flux distribution are used to make the initial parameter guess. Many of the object parameters can be accurately guessed from the first and second moments. The power-law slope can be guessed by measuring the isophotal level at two elliptical radii and comparing the ratio to that expected.

For each of the galaxy models (in fact for all object models), a function is defined which examines the fit results and determines if the fit can be consider as a success or a failure. The exact criteria for this decision will depend on the details of the model, and so this level of abstraction is needed. For example, in some case, the range of valid values for each of the parameters must be considered in the fit assessment. In other cases, we may choose to use only the parameter errors and the fit Chi-Square value.

All galaxy model fits which are successful are then subtracted from the image as is done for the successful PSF model fits. Of course, the background flux is retained, with the result that only the object is subtracted from the image. Again, the noise image is (currently) not modified.

we have no code yet to select the best of several models for a given objects. The relative value of the Chi-Square is the obvious test in this case.

3.6 Faint Sources

this is not done : should use the ensemble PSF fitting to fit just the new significant peaks

After a first pass through the image, in which the brighter sources above a high threshold level have been detected, measured, and subtracted, PSPhot optionally begins a second pass at the image. In this stage, the new peaks are detected on the image with the bright objects subtracted. In this pass, the peak detection process uses the noise image to test the validity of the individual peaks. All peaks with a significance greater than a user-defined minimum threshold are accepted as objects of potential interest.

The objects which are measured in this faint-object stage are clearly low significance detections. A typical threshold for the bright object analysis would S/N of 5 - 10. The lower limit cutoff for the faint object analysis would typically be S/N of 2 - 4. In this stage, PSPhot can perform one of three types of analysis. The difference between these options is one of speed vs detail.

In the first option, PSPhot can repeat the analysis described above in sections XXX and XXX, performing a PSF fit followed by a non-PSF fit to the objects which are not PSF-like, and subtracting them. The advantage of this option is that the faint objects are treated identically to the bright objects, and all potential parameters are measured, even for marginally extended sources. The disadvantage of this option is that the low signal-to-noise of the objects in this stage limits the amount of information which can be extracted from them. The marginal gain may not be worth the large expense of processing time.

In the second option, PSPhot can perform only the PSF model fit to the remaining peaks, but ignore any further questions of the shape of the objects. The advantage of this option is that it is substantially faster than performing the more complex (and less stable) multi-parameter non-linear fits on all faint objects. On the downside, less information is learned about the objects.

Finally, PSPhot can simply ignore the fitting process and instead glean information about the fainter sources on the basis

of the peak characteristics. In this option, the image is smoothed with the PSF model, and the peak for each object is measured. The peak flux and the local peak curvature theoretically give sufficient information to recover the object flux, the centroid coordinates, and the centroid errors. The advantage of the stage is speed, especially for the very faintest levels: if the lower limit is not sufficiently faint, the time spent in performing the smoothing (3 FFTs) cannot make up for the time which would have been spent applying the PSF model to the peaks. The downside of this method is an increased sensitivity to the local sky model (the local sky must be correctly subtracted) and fewer constraints on the quality of the detection (no Chi-Square is measured, for example).

In the ideal case, if we were only interested in detecting PSFs, and we had a good model for the PSF, we could optimally find the sources by smoothing the image and the noise image with the PSF model. *write out the description of Nick's optimal PSF finding*.

PSPhot allows the user to select between these three options for the analysis of the faint sources. Three separate user-controlled signal-to-noise ratio limits are defined. One specifies the depth to which the PSF / non-PSF analysis is performed. A second (which must be smaller) specifies the depth to which only the PSF is fitted. A third specifies the depth to which the analysis is performed using on the peak statistics. If two of these are identical, then certain of these options are simply skipped. For example, if the peak analysis threshold is set to the same value as the PSF-only threshold, no peak analysis is performed.

3.7 Aperture Correction Measurement

The important concept here is that an analytical model will always fail to describe the flux of the objects at some level. In the end, all astronomical photometry is in some sense a relative measurement between two images. Whether the goal is calibration of a science image taken at one location to a standard star image at another location, or the goal is simply the repetitive photometry of the same star at the same location in the image, it is always necessary to compare the photometry between two images. If this measurement is to be consistent, then the measurement must represent the flux of the stars in the same way regardless of the conditions under which the images were taken, at least within some range of normal image conditions. So, for example, two images with different image quality, or with different tracking and focus errors, will have different PSF models. Since an analytical model will always fail to represent the flux of the star at some level, the measured flux of the same object in the two images will be different (even assuming all other atmospheric and instrumental effects have been corrected). The amplitude of the error will by determined by how inconsistently the models represent the actual object flux. For example, if the first image PSF model flux is consistently 10% too low and the second is 5% too high, then the comparison between the two images will be in error by 15%.

Aperture photometry avoids these problems, by trading for other difficulties. In aperture photometry, if a large enough aperture is chosen, the amount of flux which is lost will be a small fraction of the total object flux. Even more importantly, as the image conditions change, the amount lost will change by an even smaller fraction, at least for a large aperture. This can be seen by the fact that the dominant variations in the image quality are in the focus, tracking and seeing. All of these errors initially affect the cores of the stellar images, rather than the wide wings. The wide wings are largely dominated by scattering in the optics and scattering in the atmosphere. The amplitude and distribution of these two scattering functions do not change significantly or quickly for a single telescope and site.

The difficulty for aperture photometry is the need to make an accurate measurement of the local background for each object. As the aperture grows, errors in the measurement of the sky flux start to become dominant. If the aperture is too small, then variation in the image quality are dominant. The brighter is the object, the smaller is the error introduced by the large size of the aperture. However, the number of very bright stars is limited in any image, and of course the brighter stars are more likely to suffer from non-linearity or saturation.

this discussion sucks: put in some more details of my point: amplitude of systematic vs random sky errors

How important is this effect? Consider a typical bright object with a flux of (say) 40,000 counts in an image of background 1000 counts per pixel, with FWHM of 4 pixels. In principle, the flux of this object should be measurable with an accuracy of roughly 0.57% ($\frac{\sqrt{40000+1000\times12}}{40000}$). However, the measurement of the sky is limited at some finite level by Poisson statistics. If we are required to use an aperture of (say) 25 pixels in radius (eg, 5 arcseconds for an 0.2 arcsec / pixel detector), and we have an annulus of twice this radius to measure the local sky, then we will have an error of XXX.

outline the variation of *ApResid* as a function of magnitude.

PSPhot measures the aperture correction (*ApResid*) for every PSF candidate object, then calculates the trend of this correction as a function of the magnitude. This trend is fitted with a line. The resulting function can be used to determine the effective aperture correction for an infinite flux object and the average bias inherent in the sky measurement for the image. The scatter of the PSF-candidate object measurements about this trend is a measure of how well we can measure photometry from the image by applying the specific PSF model. The slope of this trend is a measure of the bias in the local sky measurment for each object. In principal, the measured sky levels could be modified by this bias. More generally, the measured bias in a collection of images could be used to improve the model fitting or sky fitting portion of the software the remove the bias term.

PSPhot allows a collection of PSF model functions to be tried on all PSF candidate objects. For each model test, the above corrected ApResid scatter is measured. The PSF model function with the smallest value for the ApResid scatter is then used by PSPhot as the best PSF model for this image. The number of models to be tested is specified by the configuration keyword `PSF_MODEL_N`. The configuration variables `PSF_MODEL_0`, `PSF_MODEL_1`, through `PSF_MODEL_N - 1` specify the names of the models which should be tested.

3.7.1 Types of Object / PSF models currently available

the discussion of the model types needs to be extended

- GAUSS : Pure elliptical Gaussian
- PGAUSS : polynomial approximation to a Gaussian (PGAUSS)
- QGAUSS : power law with variable exponential term
- SGAUSS :

discuss the stability issues with the galaxy model(s)

3.8 Output Options

need to discuss tests

need to discuss failings and holes

4 Alternative Scenarios

4.1 Trailed Sources

4.2 Fixed / Known-position Sources

4.3 Difference Images

much of this discussion is theoretical: PSPhot can incorporate these modifications, but it currently does not.

The noise map for a difference image must be generated from the two images use to construct the difference. Otherwise, the low sky level will automatically result in inconsistent interpretation of the noise.

For a difference image, both positive and negative objects will be present. The basic peak detection algorithm will only trigger for the positive sources. One solution is to simply apply PSPhot to both the difference image and its negative value. **do we want to code in an automatic switch to get both positive and negative excursions in the single pass?**.

In the case of a difference image, the PSF model construction stage will probably fail for lack of valid sources. It is better in these cases to provide PSF model from some other source. For example, the two images which are combined to generate the difference image represent the PSF. Presumably, one or both have been convolved with a PSF-matching kernel. The images which result from the convolution should be used to measure the PSF model.

The object classification scheme defaults to the galaxy models for objects which are not well represented by the PSF model. In a properly-constructed difference image, galaxies are unlikely to remain behind as significant sources. Most real objects in the difference image will be PSF-like and will consist of photometrically variable objects (flare stars, supernovae, etc) or astrometrically variable objects (high-proper motion stars or solar-system objects). There are three likely classes of objects which will not be well represented by the PSF model. 1) Fast-moving solar-system objects will appear as short streaks. For example, a fast solar system object would have an apparent rate of 0.5 degrees per hour, translating to 15 arcseconds in a 30 second exposure. Even a main belt asteroid at roughly 1 AU would have reflect motion of approximately 1 degree per day, equivalent to 1.25 arcsec in a 30 second exposure, and could be noticeably smeared and non-PSF-like. A trailed-star model can be used to characterize these types of objects. 2) Small offset stars, either due to atmospheric / color effects or modest proper motion will appear as PSF dipoles in the difference images. The positive and the negative images will have stellar profiles, but they will be significantly offset and will not subtract well. The two components may not have the same amplitude. A PSF-dipole model can be used to fit these types of objects, with free parameters of the two centroids and the two fluxes. 3) Comets will appear in the difference images as a non-PSF objects. Their 2-D structure includes both the flux from the coma (with a typical power-law profile) and flux from the tail (with a more complex flux distribution). A comet flux model can be used to characterize these objects in difference images. A major difficulty in applying these three types of models is in making a robust test of which model should be used. This problem is akin to the issue of selecting and distinguishing between multiple galaxy models, as discussed in the section on Galaxy models.

5 PSPhot Structures and Data Elements

The following structures are described in detail in the document ‘Pan-STARRS PS-1 Image Processing Pipeline Modules Supplementary Design Requirements’ (psModules SDRS; PSDC-430-012).

```
pmModel
pmModelGroup
```

```
pmGrowthCurve
pmPSF
pmPSFTry
pmSource
pmPeak
pmMoments
```

psphot is supposed to operate on individual readouts, and use the techniques used by ppImage to extract header-related metadata. currently, psphot uses an alternative to the psReadout until the ppImage code can be folded together with psphot.

5.1 Top-Level APIs

```
psMetadata     *psphotArguments (int *argc, char **argv);
```

Load the command-line arguments, parse the configuration file, and place the configuration information on a single metadata structure. This function searches for the following command line option flags, and places their corresponding values on the output metadata with the given name. These options override any such values in the configuration file.

```
-mask (filename)      : MASK_IMAGE
-weight (filename)    : WEIGHT_IMAGE
-resid (filename)     : RESID_IMAGE
-region [x0:x1,y0:y1] : ANALYSIS_REGIONP
-photcode (code)      : PHOTCODE
-psf (filename)       : PSF_INPUT_FILE
-modeltest x y        : TEST_FIT_X, TEST_FIT_Y
-model (name)         : TEST_FIT_MODEL
-fitmode (name)       : TEST_FIT_MODE
-fitset (name)        : TEST_FIT_SET
```

The following option flags can be used to set any option:

```
-D  (key) (value)      : any string value
-Df (key) (value)      : any F32 value
-Di (key) (value)      : any S32 value
```

The function next examines the remaining command-line arguments and complains if there are not exactly 3 arguments, reporting the program usage. It sets default configuration variables, and then loads the configuration file specified as the third command-line option. Finally, it sets the `IMAGE` and `OUTPUT_FILE` config options to arguments 1 and 2, respecitively.

```
eamReadout     *psphotSetup (psMetadata *config);
```

This function examines the configuration data in `config` and loads the image into memory. It constructs the weight and mask images if they have not been specified, or loads the specified images. The weight image is built based on the read noise and gain of the image, as extracted from the header or from the configuration options directly. It defines the mask based on the selection image region, the values for saturation and the `min_VALID_PIXEL`.

```
bool            psphotModelTest (eamReadout *imdata, psMetadata *config);
```

This function is an optional test mode for psphot. If the test mode has been selected, this function will attempt to fit a single object with the requested model. It writes out subimage containing the source, the difference, the mask, and the weight. This function may load a PSF model or fit a floating model.

```
psStats        *psphotImageStats (eamReadout *imdata, psMetadata *config);
```

Measure the basic image properties: median sky, expected sky sigma

```
psPolynomial2D *psphotImageBackground (eamReadout *imdata, psMetadata *config, psStats *sky);
```

Model the image background as a 2D polynomial and subtract from the image. The should use a more sophisticated model and return the subtracted image.

```
psArray        *psphotFindPeaks (eamReadout *imdata, psMetadata *config, psStats *sky);
```

Create a smoothed image and find all local peaks above the threshold level (uses: `PEAKS_SMOOTH_SIGMA, PEAKS_SMOOTH_NS` `PEAKS_NSIGMA_LIMIT, PEAKS_OUTPUT_FILE`)

```
psArray        *psphotSourceStats (eamReadout *imdata, psMetadata *config, psArray *allpeaks);
```

Create the basic source structures for all peaks, define the initial pixels, measure the local sky (sky offset) and the source moments.

```
bool            psphotRoughClass (psArray *sources, psMetadata *config);
```

Find the PSF clump and make the first cut source identifications

```
bool            psphotBasicDeblend (psArray *sources, psMetadata *config, psStats *sky);
```

Find all blended peaks and tag, group with single primary source.

```
pmPSF          *psphotChoosePSF (psMetadata *config, psArray *sources, psStats *sky);
```

Try each of the selected PSF models on a subset of likely PSF stars. Measure the metric (aperture residual scatter) for each PSF model and choose the best model.

```
bool                psphotEnsemblePSF (eamReadout *imdata, psMetadata *config, psArray *sources, pmPSF *ps
```

Perform simultaneous fitting to all sources in the array using a linear fitting process which assumes all sources are PSFs and their positions are fixed. Set the positions based on the bilinear interpolation of the peak implied by the 3x3 square of pixels containing the peak. Local sky is also assumed to be correctly subtracted.

```
bool            psphotFullFit (eamReadout *imdata, psMetadata *config, psArray *sources, pmPSF *psf, psStat
```

Fit all sources in sequence starting from the brightest, and subtracting the sources as they are fitted. This function only attempts single PSF and single EXT models and chooses between them. The sources are assumed to have been subtracted in advance (ie, using psphotEnsembleFit). The models which do not succeed are re-subtracted using the prior model.

```
bool            psphotBlendFit (eamReadout *imdata, psMetadata *config, psArray *sources, pmPSF *psf, psSta
```

Fit all sources in sequence starting from the brightest, and subtracting the sources as they are fitted. This function attempts a multi-source fit for blended sources, or a single PSF if it is not a blend, followed by both EXT and DBL models and chooses between them. The sources are assumed to have been subtracted in advance (ie, using psphotEnsembleFit). The models which do not succeed are re-subtracted using the prior model.

```
bool            psphotReplaceUnfit (psArray *sources);
```

After models have been attempted for all sources, this function replaces the sources which were temporarily subtracted, but which did not succeed or converge on a good solution.

```
bool            psphotApplyPSF (eamReadout *imdata, psMetadata *config, psArray *sources, pmPSF *psf, psSta
```

Attempt to fit the PSF model to all sources in brightness order, subtracting the resulting model if successful. Only attempts single PSF models.

```
bool            psphotFitExtended (eamReadout *imdata, psMetadata *config, psArray *sources, psStats *skySt
```

Attempt to fit the PSF model to all sources in brightness order, subtracting the resulting model if successful. Only attempts single EXT models.

```
bool            psphotApResid (eamReadout *imdata, psArray *sources, psMetadata *config, pmPSF *psf);
```

Measure the curve-of-growth and the aperture correction trend.

```
void            psphotOutput (eamReadout *imdata, psMetadata *config, psArray *sources, pmPSF *psf, psStats
```

Write out data in various formats as selected.

6 User's Guide

6.1 Configuration Parameters

```
FAINT_SN_LIM
FIT_MAX_CHI
FIT_MIN_SN
FIT_NSIGMA
FIT_PADDING
FIT_RADIUS
GAIN
```

```
GAL_MODEL
GAL_MOMENTS_RADIUS
INNER_RADIUS
INPUT
MASK
NOISE
NSUBSET
OUTER_RADIUS
OUTPUT
OUTPUT_MODE
PEAK_NSIGMA
PSF_MODEL_N
PSF_MOMENTS_RADIUS
PSF_SHAPE_NSIGMA
RDNOISE
SATURATE
SMOOTH_NSIGMA
SMOOTH_SIGMA
XMAX
XMIN
YMAX
YMIN
```

6.2 Command-Line Arguments and Options

6.3 Input & Output Data Formats

7 Sample Tests

8 Further Work to be Completed

- convert to pmCell as input data
- loop over all readouts in a pmCell
- write out multiple files?
- better method for defining the recipe?
- additional options for image background
- image background should return a background image