Astronomy and Computing 18 (2017) 47–53

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

PHOTOMETRYPIPELINE: An automated pipeline for calibrated photometry

M. Mommert

Northern Arizona University, Department of Physics and Astronomy, Flagstaff, AZ 86011, USA

ARTICLE INFO

Article history:
Received 7 August 2016
Accepted 30 November 2016
Available online 3 January 2017

Keywords:
Methods
Data analysis
Techniques
Photometry
Astrometry

ABSTRACT

PHOTOMETRYPIPELINE (PP) is an automated pipeline that produces calibrated photometry from imaging data through image registration, aperture photometry, photometric calibration, and target identification with only minimal human interaction. PP utilizes the widely used Source Extractor software for source identification and aperture photometry; SCAMP is used for image registration. Both image registration and photometric calibration are based on matching field stars with star catalogs, requiring catalog coverage of the respective field. A number of different astrometric and photometric catalogs can be queried online. Relying on a sufficient number of background stars for image registration and photometric calibration, PP is well-suited to analyze data from small to medium-sized telescopes. Calibrated magnitudes obtained by PP are typically accurate within $\leq$0.03 mag and astrometric accuracies are of the order of 0.3 arcsec relative to the catalogs used in the registration. The pipeline consists of an open-source software suite written in Python 2.7, can be run on Unix-based systems on a simple desktop machine, and is capable of realtime data analysis. PP has been developed for observations of moving targets, but can be used for analyzing point source observations of any kind.

1. Introduction

Telescopes across the globe acquire massive amounts of imaging data every night. While the underlying science goals vary widely in these observations – from deep observations of extragalactic targets to short observations of rapidly spinning asteroids — the immediate objective of most observations is similar: obtaining reliable and calibrated brightness measurements of usually faint point sources. This objective requires not only good seeing and transparency conditions, as well as more or less extensive planning in order to address the science goal in the most efficient way, but also a sophisticated and accurate reduction and analysis of the acquired data.

Large observatories often provide support in the reduction and analysis of their data. However, the majority of imaging data have been – and still is – acquired with telescope apertures of a few meters or smaller. Smaller telescopes are usually easier to access because they are more numerous, but the observer is often left alone in the data reduction and analysis process. This factor leads to large amounts of imaging data from smaller telescopes being left unanalyzed as their proper analysis is not considered worth the effort, or because observing conditions were not ideal. The availability of an automated and robust software pipeline to exploit these data would simplify access to this data treasure trove.

I present PHOTOMETRYPIPELINE (PP), a Python-based, open-source software suite that provides automated and calibrated point-source photometry of imaging data, specifically designed for small to medium-sized telescopes. PP provides image registration, photometric analysis and calibration for both fixed and moving targets with only minimal user interaction. The pipeline can be run on Unix-based systems, ranging from desktop machine to larger and more capable machines. PP was originally designed to obtain photometry of asteroids, but can be applied to observations of any point-sources, including stars, extragalactic sources, artificial satellites, and space debris. Due to its modular and flexible design, it can be modified to work with data from nearly any professional telescope.

PP is available for download on GitHub.[1] Since PP is still evolving, refer to the online documentation[2] for up-to-date information. This document describes the functionality of PP Version 1.0 as of 30 November 2016. Also refer to the documentation for installation guides and additional support.

E-mail address: michael.mommert@nau.edu.

1 https://github.com/mommermi/photometrypipeline.
2 http://mommermi.github.io/pp/index.html.

http://dx.doi.org/10.1016/j.ascom.2016.11.002
2213-1337/

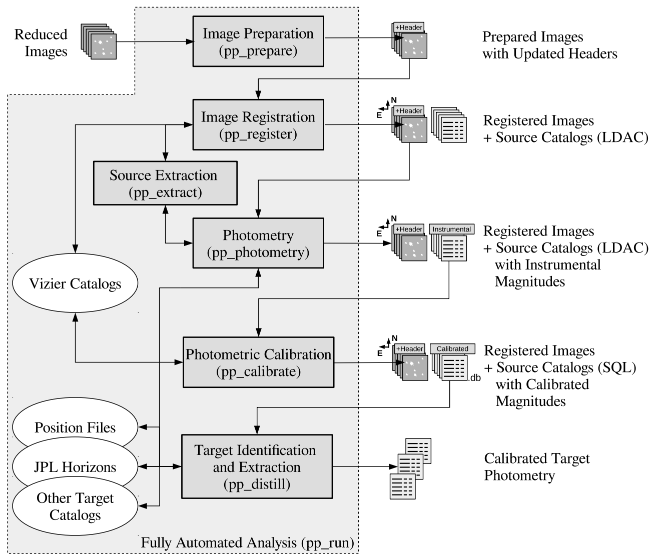

Fig. 1. PP work flow diagram. The shaded area indicates the sequence of tasks run by the automatic analysis routine `pp_run`; the same sequence should be used if the individual tasks are run separately. White ovals indicate resources that are required in the analysis. The individual pipeline steps are discussed in Section 2.

2. Methods and implementation

2.1. Overview

The pipeline is implemented as a suite of Python 2.7 scripts. It makes use of Python packages that are freely available and easy to install through the Python Package Index[3]; required packages include NumPy,[4] SciPy,[5] astropy,[6] matplotlib,[7] and CALLHORIZONS.[8] For specific tasks, open-source software has been integrated into the pipeline. That software is called within the Python environment, but has to be installed on the machine prior to running the pipeline. The required auxiliary software is Source Extractor[9] (Bertin and Arnouts, 1996) and SCAMP[10] (Bertin et al., 2002), which are introduced below. Both Source Extractor and SCAMP are well-tested and widely used within the community, providing a robust foundation for the most important pipeline features.

PP consists of a number of stand-alone Python scripts that can be called separately, or can be run automatically. Fig. 1 presents the work flow of the automated pipeline routine (`pp_run`). The same sequence should be used if the individual tasks are called separately in order to meet the respective file dependencies. `pp_run` is designed to run automatically on the majority of all data provided; this implies that it might not run successfully on non-ideal data (see Section 4.3 for a discussion). Running the individual pipeline tasks separately, using fine-tuned parameters differing from the defaults used by `pp_run`, might be necessary to improve the outcome for non-ideal datasets.

The runtime of the pipeline depends on a number of different parameters, including the memory and computing power of the machine, the image size, the number of images, the background star density, and the number and nature of targets in each field. For instance, running the example data presented in Section 3 (79 700 pixel×700 pixel images, one target in the field) through the default pipeline (using `pp_run`) on a quad-core 1.9 GHz laptop running Ubuntu Linux 16.04 takes less than 20 min. With minor modifications, PP is able to provide real-time data analysis.

Every step of the image analysis process is thoroughly documented and summarized in a "diagnostics" HTML file that is created on-the-fly. This webpage allows for inspection of each pipeline process and serves as a validation of the data quality. Furthermore, it allows for identification of data affected by background sources, artifacts, and target mis-identification.

In order to provide the best possible results, PP should be run on fully reduced image data, which includes flat fielding and bias correction, as well as trimming of the data. It is mandatory that all frames coming from the same instrument have the same image dimensions.

2.2. Pipeline tasks

The individual pipeline tasks shown in Fig. 1 are described in detail below. Refer to the documentation provided online for more details on how to use these tasks.

2.2.1. Image data preparation—`pp_prepare`

In order to allow for the degree of automation provided by PP, it heavily relies on properly populated FITS image headers. Since every instrument/telescope combination uses slightly different formats, each combination has to be set up before data can be run through the pipeline. This setup consists of tailored parameter files for Source Extractor and SCAMP (see below), as well as a dictionary that translates header information into a format that is readable by the pipeline. In order to exploit its full potential, PP requires information on the telescope pointing and the date and time of the observations, as well as the detector pixel scale, detector binning, the official Minor Planet Center[11] (MPC) identifier of the observatory, the used photometric filter, and the target name to be present in each FITS image header.

Pipeline task `pp_prepare` identifies the used instrument and then reads, translates, and modifies the required FITS header keywords into a common format that is independent of the instrument and readable by all pipeline tasks. It also removes existing plate solutions in the World Coordinate System (WCS, see Greisen and Calabretta, 2002; Calabretta and Greisen, 2002) format from the header and implants a zero-th order solution based on the provided image coordinates, the detector pixel scale, and the typical image orientation for the respective telescope/instrument combination. This step is crucial for proper image registration (see Section 2.2.2).

This approach grants a high degree of flexibility to the pipeline, making it applicable to a large range of telescopes and instruments.

2.2.2. Image registration—`pp_extract` and `pp_register`

A plate solution for each input image is found using SCAMP, which computes astrometric solutions based on coarse WCS information in the FITS image header (provided by `pp_prepare`), a catalog of all sources in the field, and a reference catalog. SCAMP works completely automatic. Field source catalogs are generated using Source Extractor in the binary Leiden Data Analysis Center (LDAC) catalog format. Source Extractor identifies field sources based on their signal-to-noise ratio (SNR) and a minimum number of connected pixels that exceed that SNR threshold. For each source, Source Extractor provides the position in image and – if available – WCS coordinates, different flavors of photometry,

3 https://pypi.python.org/pypi.
4 http://www.numpy.org/.
5 https://www.scipy.org/.
6 http://www.astropy.org/.
7 http://matplotlib.org/.
8 https://github.com/mommermi/callhorizons and Appendix.
9 http://www.astromatic.net/software/sextractor.
10 http://www.astromatic.net/software/scamp.
11 http://minorplanetcenter.net.