ZTF Marshal

Yi Cao[1] & Lin Yan[2]

Abstract

This document describes major changes, functionalities and webpages of ZTF Marshal.

Major changes

The basic design of the ZTF marshal will be very similar to the PTF/iPTF marshal. Major changes will be made in order (1) to allow both private and public data and (2) for scalability of large data volumes (in terms of numbers of candidates, saved objects, and potential users).

In PTF/iPTF, two separate marshals were built for transients and variable stars. Apart from displaying light curves and spectra as well as coordinating follow-up observations, there are three major difference between these two marshals: (1) transients use subtracted light curves while variable stars need absolute light curves; (2) variable stars have a built-in periodogram; (3) transients and variables have their own naming conventions. Moving forward, ZTF will use a unified marshal to host both transients and variable stars. Point (1) can be easily resolved at the software level. The periodogram mentioned in Point (2) should be excluded in the ZTF marshal, because it requires significant computing resources. The naming convention for transients and variable stars should be discussed.

The backend database of the ZTF marshal will be also unified. Given the fact that different types of transients and variables have distinct sets of parameters and the fact that the marshal will also serve as a collector and presenter of both internal P48 data and external follow-up data, the dataset in the marshal database is not necessarily structured. As a result, a relational database may have difficulties to store these data. Such a database might also experience scalability issues once the number of transients and variables and the number of users grow rapidly. Based on these considerations, a non-relational database, such as **mongoDB**, is preferred. It is much more flexible in the data structure and has sufficient indexing capability for fast search at the same time. While it is free, it is well maintained and documented by a commercial company.

Regarding visual inspection of candidates, marshal will have scanning pages for transients and variable stars separately. Basically, a transient needs two detections to confirm while stars only need one epoch to calculate Δm with respect to its reference epoch.

Python 2 will be the primary programming language for development of the ZTF marshal. In order for better maintenance of the ZTF marshal, the marshal code should follow the LSST coding standard (`https://dev.lsstcorp.org/trac/wiki/PythonCodeStandards`) and well commented. Since the final location and domain of the ZTF marshal are to be determined, all absolute URLs or addresses, system-wide parameters and constants should be included in a configuration file and not explicitly appear in any code. All codes should be version-controlled using **git**. A repository will be set up soon.

ZTF marshal functionalities

The ZTF marshal is positioned as a web interface between users and data. It will provide functionalities of

- status of telescopes and projects;
- management of key programs and collaborations;

[1]Email: ycao@astro.caltech.edu
[2]Email: lyan@ipac.caltech.edu

- easy access and visualization of the survey data;
- basic and advanced searches to the candidate and marshal databases;
- cross-matching the survey data to external datasets;
- scanning and saving interesting sources;
- easy preparation and coordination of follow-up observations;
- collection and organization of follow-up data;
- protection of the raw and processed data.

Detailed webpages

In order to realize these functionalities, the following webpages are needed:

`Project status`

- the status of P48 and (optional) the P48 sky coverage of a given time range
- the status of image processing pipeline
- the survey plan

`Collaboration managements`

- user group and privileges: the marshal needs to define different levels of users with corresponding privileges. Naively, at least three levels are needed for access to objects: superuser, collaborator and public.
- key projects: each key project should have one project lead who may manage their working groups and sub-projects.

`Sources`

- scanning pages for different scientific goals: fast and young transients, slow transients, variable stars, etc..
- examine page for a given object or celestial location
- saving candidates
- generating finding charts
- calculating visibility plots
- cross-matching saved objects to external datasets (SDSS, PanSTARR, etc.)
- pulling out photometry of each saved object within **one week** of discovery
- receiving and prioritizing forced-photometry requests to pull longer histories of transients and variables
- sending out alerts to certain user groups
- receiving photometry from other telescopes, both automatically and manually

- displaying original and subtracted photometry for variables and transients
- assigning objects for follow-up observations
- receiving and displaying spectroscopic data both automatically and manually
- allowing users to download photometric and spectroscopic data
- user comments (with or without attachments)
- classification

`Search`

- cone search in a given time range
- advanced search
- preparing target lists for follow-up observations