Research and Implementation of a Rapid Automatic Response Observation System for Transient Sources (Postprint)

Authors: Fu Xiachuan, Wu Chao, Huang Maohai, Zhang Mo, Lu Xiaomeng

Date: 2017-09-26T00:00:00+00:00

Abstract

Follow-up observations of astronomical transient sources are crucial for the identification and characterization of these sources. The stochastic nature of transient events and their rapid photometric variability necessitate that follow-up observation systems possess rapid automated response capabilities. We present the research and implementation of a rapid automated response observation system for astronomical transient sources developed in Python, whose core comprises database interaction and module scheduling functionalities based on the Django framework. The system's primary functions include alert message transmission based on VOEvent, a web-based status monitoring and interactive platform, automated data processing and automated result feedback, as well as a telescope control interface. Tests of message and data transmission between the China-France Astronomical Satellite Science Center in Beijing and the Xinglong Observatory, along with actual triggered observation tests, demonstrate that the system satisfies the basic requirements for automated follow-up observations of transient sources by the ground-based observation system of the China-France Astronomical Satellite project. Meanwhile, the system demonstrates good portability and can be adapted for other similar automated response observation systems.

Full Text

Research and Implementation of a Transient Quick Response and Automatic Observation System

Authors: Fu Xiaochuan[123], Wu Chao[123], Huang Maohai[123], Zhang Mo[123], Lu Xiaomeng[1]

Affiliations:

1. National Astronomical Observatories, Chinese Academy of Sciences, Beijing

100012, China
2. Key Laboratory of Space Astronomy and Technology, National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100012, China
3. University of Chinese Academy of Sciences, Beijing 100049, China

Abstract

Follow-up observation of astronomical transient sources is crucial for their identification. The random occurrence and rapid brightness evolution of transients demand that follow-up observing systems possess rapid automatic response capabilities. This paper describes the research and implementation of a Transient Quick Response and Automatic Observation System based on Python, with core functionality centered on database interaction and module scheduling implemented through the Django framework. The system' s main features include VOEvent-based alert messaging, a web-based status monitoring and interaction platform, automatic data processing with result feedback, and telescope control interfaces. Tests conducted between the SVOM Chinese Science Center in Beijing and the Xinglong Observation Base, including actual triggered observation trials, demonstrate that the system meets the basic requirements for automatic transient source follow-up observations in the SVOM project' s ground-based observation system. The system also exhibits excellent portability for adaptation to other similar automatic response observation systems.

Keywords: Astronomical data processing; Transient source; Automatic observation; VOEvent; Python; Django

1. System Overall Framework Design

Based on analysis of the data processing workflow, the Transient Quick Response and Automatic Observation System must complete the entire pipeline from receiving transient alerts at the science center (or from international alert networks) to generating observation plans, interrupting regular observations if necessary, controlling the telescope for CCD observations, processing the data, and automatically uploading results back to the remote science center. The system architecture comprises several key modules: alert monitoring, telescope control, automatic data processing, and result feedback, with system operation monitoring and interaction implemented primarily through web services.

This paper focuses on the development of the alert monitoring, central scheduling control, and user monitoring/interaction modules, along with integration of data processing components. The system structure is illustrated in [Figure 1: see original paper].

The system development leverages Python for its rich ecosystem of third-party astronomical software packages, friendly hardware control interfaces, and user

monitoring capabilities. The Django framework provides object-oriented development convenience for module scheduling, parameter passing, raw data and data product management, and web-based services. The system achieves three primary functions: alert triggering, telescope observation control, and data processing/transmission, while also providing monitoring and interaction capabilities to notify the broader astronomical community and facilitate follow-up observations with larger international telescopes.

Similar international automatic telescope systems such as BOOTES, MASTER, ROTSE, and RAPTOR have more comprehensive functional requirements, though much of their operational software remains unpublished. The SVOM project demands more stringent data processing procedures and timing constraints. Combining scientific and engineering requirements, we have researched and implemented this transient quick response automatic observation system.

2. Real-Time Message and Data Transmission

For communication between the Chinese Science Center (CSC) and the Transient Quick Response and Automatic Observation System, we designed two transmission paths for different data types: (1) Instant data communication based on the Virtual Observatory Event (VOEvent) protocol for time-critical messages, enabling second-level message delivery; (2) A large-data transfer channel using the File Transfer Protocol (FTP) for transmitting large-volume scientific observation data that does not require real-time delivery.

VOEvent is a standard data format in the astronomical transient observation field, encapsulating information such as coordinates in XML files. It can be used for telescope pointing control, observation plan transmission, and result dissemination [7]. To ensure reliable message and data delivery, the VOEvent transport protocol incorporates a message acknowledgment mechanism where all received messages generate a response acknowledgment.

The transmission flow proceeds as follows: Upon receiving a gamma-ray burst trigger, the Chinese Science Center sends a VOEvent alert to the Transient Quick Response and Automatic Observation System. After receiving the alert, the system acknowledges it, processes the alert content, and controls the telescope observation. Following exposure completion, the system automatically processes the raw observation data. If a gamma-ray burst is identified (with precise coordinates, etc.), the system generates a Finding Chart (Fchart) and sends a VOEvent with the upload information. This process repeats until observation completion. The VOEvent Transport Protocol (VTP) uses password authentication to ensure data integrity [8]. FTP is a bidirectional file transfer protocol between client and server that, while transferring large image files, facilitates content verification and confirmation.

3. Central Scheduling Control

Module coupling strength depends on interface complexity. To reduce system complexity and improve design quality, we adopted a database-based approach for inter-module parameter passing. This design allows modules to communicate by simply passing simple values, with each module retrieving additional data independently from the database, greatly reducing coupling.

Database-based parameter passing offers several advantages: it facilitates real-time system status tracking and analysis by monitoring modules, aids automated system management, enhances stability, and increases transparency while avoiding potential conflicts from multiple modules having write access to the same data. During trial operation, the system has matured continuously, though future research may explore message middleware for parameter passing.

System operation centers on central scheduling control. When the alert monitoring module detects an alert, Django framework functions create a new database record with Trigger_ID as the primary key to store alert information. The telescope control module reads alert and observation strategy information from the database using Trigger_ID upon receiving a trigger. The raw data interface module reads configuration files and monitors based on database table information, writing changes back to the database. The data processing system reads calibration files and processing paths using Trigger_ID, creating new records upon completion. This workflow is illustrated in [Figure 3: see original paper].

Automated data processing requires comprehensive data management. Database data types include: (1) System parameters like observation plans and telescope configuration information, stored as key-value pairs; (2) Calibration files and scientific products, where filenames and paths are written to the database while actual data remains on disk; (3) Critical values such as magnitudes, stored numerically. Except for manual observation plan adjustments via web interface, all other information is managed automatically by the system.

The database schema comprises ten tables: observation information, system status, observation plans, raw calibration files, calibration file associations, processed calibration files, raw science images, processed science images, science image calibration associations, and science products. The Entity-Relationship (E-R) diagram is shown in [Figure 4: see original paper], with table details provided in .

4. Status Monitoring and Interaction

To enable remote system status monitoring and manual observation strategy modification, we developed a web-based interaction platform using Django' s Model-View-Controller framework. The platform features real-time log display and observation strategy management.

The log display interface refreshes system operation logs in real-time using Asynchronous JavaScript and XML (AJAX) technology, enabling data exchange with

the server without reloading the entire page. The observation strategy management page provides three functions: (1) displaying currently active observation plans with names and schedules; (2) showing stored observation plan names and schedules for flexible activation; (3) adding, deleting, or modifying observation plans as needed.

Observation plans are stored in the database as two-dimensional tables. To optimize storage and reduce read operations, exposure sequences and timing are encoded into one-dimensional strings. The platform decodes this information for display and encodes it upon submission, enabling single-operation database transactions. When users access the page, the browser sends a request, Django processes it and retrieves current log information or observation plans from the database, then returns the rendered template. This browser-based approach provides cross-platform accessibility.

The interaction platform has completed development of observation strategy management and real-time log monitoring interfaces. Users access the platform by entering a specific address in their browser, which Django matches to corresponding view functions that access the database through the ORM mapper. Static page display is implemented through HTML, CSS, and JavaScript templates. The actual interface is shown in [Figure 6: see original paper].

5. System Testing

Testing covered both functional and performance aspects. Functional tests included alert transmission, telescope automatic control, raw data interface reception, data processing, and result upload. Performance tests measured alert response time, total triggered observation workflow completion time, and data processing result accuracy.

The test framework simulated the actual application scenario: the Beijing-based science center generated test VOEvent alerts sent to the alert scheduling server, which forwarded them to the Xinglong Observation Base' s Transient Quick Response and Automatic Observation System. After parsing the alert, the system controlled automatic telescope observation, processed the acquired data, and uploaded results back to the Beijing science center. Data transmission utilized both FTP and VOEvent real-time links. The test flowchart is shown in [Figure 7: see original paper].

Over two observation nights with multiple tests, the system demonstrated no missed or false alerts. VOEvent servers and messages transmitted data completely and accurately. All expected functions operated correctly, with average response time from alert transmission to telescope control command execution being approximately 90 seconds. Statistical analysis of system logs showed the complete workflow from alert reception to data upload completion required less than 150 seconds maximum, including telescope exposure time. With typical 100-second exposures, the system' s operational time (excluding exposure) was less than 100 seconds maximum. These results are illustrated in [Figure 8: see

original paper] and [Figure 9: see original paper].

To verify observation execution correctness, we compared uploaded results with reference images from the SAO-DSS server. The comparison confirmed that the automatic observation system correctly executed observation commands, producing scientifically accurate images. [Figure 10: see original paper] shows the system' s automatic observation results alongside the corresponding SAO-DSS field.

6. Conclusion and Outlook

The Transient Quick Response and Automatic Observation System was developed based on SVOM project requirements for ground-based follow-up telescopes. The system automatically responds to science center observation alerts, retrieves observation plans, controls telescope pointing, acquires data, and uploads results in real-time, while providing manual monitoring and interaction interfaces for remote operation and observation plan adjustment.

Built on Python' s rich astronomical data processing software and VOEvent messaging capabilities, the Django framework provides web services, data processing applications, and a friendly object-oriented database interface. Using a database backend for support, the system achieves both automated inter-module parameter passing and comprehensive data/product management. The web-based monitoring and interaction approach significantly enhances remote operation and user-friendliness.

Testing under actual ground-based follow-up telescope scenarios verified alert response time, data reporting time, and response authenticity. The system responds to alerts in less than 100 seconds on average, with observation result reporting completed within 160 seconds. Comparison with SAO-DSS images confirms correct pointing and data processing execution.

The system demonstrates unattended operation, rapid automatic processing, and real-time data transmission. For robustness, modules are designed to: (1) log status and handle unexpected conditions gracefully; (2) provide emergency startup interfaces allowing direct parameter passing to bypass database connections when necessary. With no special dependencies and clear input/output interfaces, the system offers strong portability and can be adapted to other telescopes with minimal interface modifications, making it applicable to other similar transient automatic response observation systems.

Acknowledgments: We thank Dr. Zhou Lixiao for suggestions and discussions on system testing, and SVOM project engineering and observation staff for assistance in system construction and testing.

References

[1] Paul J, Wei J, Basak S, et al. The Chinese-French SVOM mission for gamma-ray burst studies. *Comptes Rendus Physique*, 2017, 298-308.

[2] Castro-Tirado AJ. Robotic astronomy and the BOOTES network of robotic telescopes. *The Astronomical Journal*, 1901-1913.

[3] et al. MASTER prompt and follow-up GRB observations.

[4] et al. ROTSE all-sky surveys for variable stars. I. test fields.

[5] Balsano R, Gomboc A, et al. The automatic real-time Gamma-Ray Burst pipeline. *The Publications of the Astronomical Society of the Pacific*, 549-552.

[6] Casperson DJ, et al. RAPTOR closed-loop monitoring of the night sky. *Astronomische Nachrichten*, 288-296.

[7] Swinbank J. Comet: a VOEvent broker. *Astronomy and Computing*, 12-26.

[8] Huang Maohai, Wu Chao. Prototype VOEvent network systems based on VTP and XMPP for the SVOM Chinese science center. *International Conference on Space Operations*, 2016.

Note: Figure translations are in progress. See original paper for figures.