The user wants me to translate a Chinese academic paper title: "Distributed Control Software for the Active Surface System of Tian-ma Radio telescope 后印本" Let me break this down: - "Distributed Control Software" - This is straightforward technical terminology - "for the Active Surface System" - Al···

Authors:

Date: 2023-09-20T00:00:00+00:00

Abstract

The user wants me to translate a Chinese academic text to English. Let me analyze the requirements:

1. Preserve ALL LaTeX commands and placeholders exactly
2. Preserve ALL citation commands ([?], ??, (??), etc.) exactly
3. **PRESERVE ALL ···paragraph wrapper tags EXACTLY**
4. Maintain academic tone and formal scientific writing style
5. Use precise technical terminology - obey the glossary strictly
6. Preserve section structure and paragraph organization
7. Translate all content completely - do not omit any information

The glossary provided is: - 机器学习 => machine learning - 深度学习 => deep learning

The input text is:

```
The Tian-ma Radio Telescope (TMRT) applies an Active Surface System (ASFS), which corrects f
```

Wait, this is already in English. Let me re-read the instructions. The user says "Translate from Simplified Chinese to English". But the provided text is already in English. Let me check if there might be a mistake.

Looking at the text again: "The Tian-ma Radio Telescope (TMRT) applies an Active Surface System (ASFS), which corrects for large-scale deformations due to gravity and thermal on the primary reflector···"

This is definitely English. But the instruction says to translate from Simplified Chinese to English. This is confusing. Let me check if there’ s any Chinese text that I missed.

No, the entire input is in English. The user might have made a mistake in the prompt, or they might be testing if I can recognize that no translation is needed.

However, the instructions are very specific about preserving tags, LaTeX commands, etc. The current input doesn’ t have any of these

Full Text

Authors and Affiliations

Dong Zhang[1,2], Rong-Bing Zhao[1]*, Zhen Yan[1], Wei-Hua Shang-Guan[1,2], Qian Ye[1], Zhi-Qiang Shen[1], Qing-Hui Liu[1], Jin-Qing Wang[1], Li Fu[1], Chu-Yuan Zhang[1,2], Dong-Fang Song[1,2], Si-Yu Liu[1,3]

[1] Shanghai Astronomical Observatory, Chinese Academy of Sciences, Shanghai 200030, China
[2] University of Chinese Academy of Sciences, Beijing 100049, China
[3] Chongqing Jiaotong University, Chongqing 400074, China

*Corresponding author: zhaorb@shao.ac.cn

Received 2023 May 31

Abstract

The Tian-ma Radio Telescope (TMRT) employs an Active Surface System (ASFS) to correct for large-scale deformations of the primary reflector caused by gravitational and thermal effects. Centralized and automated management of the ASFS through software has presented a significant challenge, which we have addressed by developing the TMRT Active Surface System Control Software (TASCS). This paper describes the design and implementation of TASCS, which provides functionalities for device control, status monitoring, human-computer interaction, and data management. TASCS adopts the open-source Tango Controls framework and distributed middleware technology to achieve real-time automated adjustment of the primary reflector through remote centralized control of numerous actuators. The system has been successfully deployed on TMRT and has played an important role in Event Horizon Telescope (EHT) observations.

Key words: Tian-ma Radio Telescope (TMRT) —Active Surface System (ASFS) —Distributed Control Software —Tango Controls

1 Introduction

Due to their large size and weight, large-aperture antennas are susceptible to large-scale deformations caused by gravitational and thermal effects, which reduces antenna efficiency. Currently, major international radio telescopes such as the Green Bank Telescope (GBT), Sardinia Radio Telescope (SRT), and Tianma Radio Telescope (TMRT) have adopted Active Surface Systems (ASFS) to address this issue. ASFS can restore the ideal shape of the primary reflector by adjusting actuators, thereby improving antenna efficiency.

TMRT is currently the largest fully steerable Cassegrain-type radio telescope system in Asia. Its ASFS is equipped with 1104 actuators that provide real-time adjustments to the 1008 panels of the primary reflector. Figure 1 [Figure 1: see original paper] (Dong et al. 2018) demonstrates significant improvement in antenna efficiency when the ASFS operates in the Q-band (35–50 GHz).

The control software serves as the "brain" of the ASFS, responsible for coordinating all devices and monitoring their operational status. Given the large number of devices and the complex structure of the ASFS, we adopted a distributed control framework for software design and development. Distributed control frameworks offer numerous advantages: they improve development efficiency by hiding low-level mechanism details such as message queues, asynchronous communication, thread pools, and lifecycle management; they provide unified resource sharing and invocation for the system by standardizing device attributes and behaviors such as data structures, interfaces, protocols, and state machines. Internationally, several major radio telescopes have adopted distributed control frameworks for their ASFS, including the Alma Common Software (ACS) (Chiozzi et al. 2004; Shen et al. 2011), Ygor (Ford & Clark 2002; Ford et al. 2014), and Tango Controls (BAFFA et al. 2019).

In this paper, we describe the design and implementation of the TMRT Active Surface System Control Software (TASCS) using Tango Controls. As an open-source, cross-platform, object-oriented distributed system framework, Tango Controls offers excellent scalability, rich application programming interfaces (APIs), and a suite of practical toolkits (Verdier et al. 2011). It enables users to implement device control, data acquisition, and real-time transmission functionalities, making it an ideal choice for ASFS control software development.

The remainder of this paper is organized as follows: we first briefly introduce the ASFS hardware, then describe the software from three perspectives: design, implementation, and testing.

2 Hardware

The ASFS of TMRT consists of actuators, control buses, subcabinets, and control software. The system is equipped with a total of 1104 actuators installed on the back structure of the antenna (Dong et al. 2016). The antenna panel position is adjusted by controlling four adaptive screws on each actuator. The

ASFS utilizes an RS-485 bus to connect all actuators to the Ethernet (see Figure 2 [Figure 2: see original paper]) and is divided into 24 fan sections. Each section is controlled by a subcabinet that includes three buses, resulting in 72 buses total, with each bus connecting 15 or 16 actuators.

3.1 Function Design

TASCS is designed to meet the diverse needs of its users, including astronomers and observation engineers, and therefore provides the following functionalities:

1. **Monitoring and control:** Continuously collect and analyze device status information and raise alarms for system abnormalities. Read the Antenna Deformation Model (ADM) from the database, calculate the optimal displacement for each actuator at the current elevation, and move the actuators to their desired positions.

2. **Human-computer interaction:** Provide remote access for multiple users through a graphical user interface (GUI). Display real-time device status and provide a device debugging interface for users.

3. **Data management:** Provide access to the ADM, store device status data, and record log files.

3.2 Architecture Design

TASCS adopts the open-source Tango Controls framework, which utilizes CORBA (Common Object Request Broker Architecture) middleware technology (Götz et al. 2003). The core concept of Tango Controls is based on devices and device classes. A device, as the control object, represents either a physical hardware device or a logical device. Devices have unified features including attributes, properties, pipes, and commands, which are defined by the device class. Device objects are instantiated within a device server that acts as a container process (Tan 2015).

The architecture includes Actuator-Bus Device Servers, Coordinator Device Servers, and Subcabinet Device Servers. Clients include device control, data visualization, and debugging functions. Communication between device servers and clients is accomplished using the Tango Middleware Protocol. The database consists of the Tango Database and Model Database.

3.3 Model Design

We use the Unified Modeling Language (UML) to model the software structure, as shown in Figure 4 [Figure 4: see original paper]. Actuator-Bus inherits from the DeviceImpl base class and implements initial configuration, state acquisition, and operational control of the actuator through customized code. Actuator-BusClass defines the unified interface characteristics of Actuator-Bus, including attributes, commands, and properties. The attributes include actuator position, speed, temperature, current, and alarm status; the properties include actuator address and number; the commands include actuator startup, shutdown, start operation, stop operation, and abnormality recovery.

4 Implementation

TASCS comprises modules including the Actuator-Bus Device Server, Coordinator Device Server, State Machine, GUI, Database, and Logging. The specific implementation of each module is described below.

4.1 Actuator-Bus Device Server

The Actuator-Bus Device Server is the core component of the entire software, using a bus as the control object. As shown in Figure 5 [Figure 5: see original paper], the Actuator-Bus Device Server first reads the ADM from the model database and calculates the optimal displacement of each actuator according to commands, then calls the device driver interface to control the actuator. Simultaneously, it collects actuator status data in real time, pushes it to clients, and records log files.

4.2 Coordinator Device Server

Due to the numerous devices in a radio telescope, we use the Coordinator Device Server (hereinafter referred to as the coordinator) to synchronize various telescope devices during the observation process (see Figure 6 [Figure 6: see original paper]).

A complete observation process generally consists of three stages: preparation, observation, and termination. During the preparation stage, the antenna must move toward the target, the receiver must be configured with parameters, and the ASFS must adjust panel positions. To ensure all devices are ready before observation begins, the coordinator calculates the Synchronization Start Time (SST) based on the preparation time required by each device and distributes the SST to all device servers. Subsequently, each device server autonomously manages its corresponding device—for example, the antenna points to the target, the ASFS adjusts actuators to optimal positions, and the Data Acquisition System (DAS) prepares for data recording. After reaching the SST, the antenna begins tracking, the ASFS starts real-time adjustment, and the DAS begins recording data. During the termination stage, all devices stop running and return to their initial states.

4.3 State Machine

To unify actuator behavior and ensure operation according to a strict timing sequence, we employ a state machine to describe the various working states of actuators and the transition methods between different states. As shown in Figure 7 [Figure 7: see original paper], after power-on, actuators are initialized and configured with parameters to transition to the "Ready" state. Upon receiving commands from the coordinator, each actuator enters the "Activate" state, during which all actuators move toward their target positions. After reaching the synchronization start time, they enter the "Running" state and make real-time adjustments based on the current elevation. During operation, if system abnormalities occur—such as device failure or transmission error—the actuators stop running and enter the "Standby" state.

4.4 Graphical User Interface (GUI)

The GUI is developed using Python, PyQt5, and the OpenGL library. The interface consists of three parts: Device Control, Data Visualization, and Debugging (see Figure 8 [Figure 8: see original paper]). Device Control provides users with an interface for device operation. Data Visualization displays real-time device status information. Debugging provides users with a device debugging interface for analyzing and troubleshooting device faults.

4.5 Database

The database consists of the Tango database and model database. The Tango database provides device configuration information and runtime catalogs for both clients and device servers (Tan 2015). The ADM is stored in the model database, which records the displacement of each actuator at every elevation.

4.6 Logging

The logging module records the historical states of the ASFS. It records the position and running mileage of all actuators and the state information of each device for system recovery and troubleshooting.

5 Measurement Results

Currently, TASCS has been successfully deployed on TMRT and plays an important role in Event Horizon Telescope (EHT) observations (Akiyama et al. 2019). It is primarily used in two cases during observation:

In the first case, the surface error of the primary reflector is measured using the Out-of-Focus (OOF) holography technique (Nikolic et al. 2007; Dong et al. 2014) prior to observation. TASCS converts the surface error into displacement values for the actuators and makes corresponding adjustments.

In the second case, the Antenna Control Software (ACS) guides the antenna to track the target during observation. It sends the current elevation to TASCS, which reads the ADM and adjusts panel positions in real time.

The control logic diagram for the ASFS is shown in Figure 9 [Figure 9: see original paper].

After enabling TASCS, we conducted surface measurements on the primary reflector and achieved a surface accuracy of RMS 201 microns (see Figure 10 [Figure 10: see original paper]), meeting our expectations.

6 Conclusion and Discussion

This paper describes the design and implementation of the Tian-ma Telescope Active Surface System Control Software (TASCS), which uses Tango Controls as the distributed control framework, follows a client-server design pattern, and employs C++ and Python languages. TASCS supports multi-user remote control and monitoring, facilitates easy upgrades, and enables flexible cross-platform deployment. During actual operation, TASCS automatically adjusts the primary reflector via actuators according to the ADM, thereby improving surface accuracy. Given the advantages of Tango Controls in building distributed control systems, we plan to adopt it for additional radio telescope projects in the future.

Acknowledgements This work was supported by the National SKA Program of China (No. 2020SKA0120104) and the Natural Science Foundation of Shanghai (No. 20ZR1467600).

References

2015, Tango Controls, https://www.tango-controls.org/, accessed: July 19, 2023

Akiyama, K., Alberdi, A., Alef, W., et al. 2019, *The Astrophysical Journal Letters*, 875, L4

BAFFA, C., GIANI, E., & VELA NUNEZ, M. 2019, *Astronomical Data Analysis Software and Systems XXVI*, 521, 185

Chiozzi, G., Jeram, B., Sommer, H., et al. 2004, in *Advanced Software, Control, and Communication Systems for Astronomy*, Vol. 5496, SPIE, 205

Dong, J., Jin, H., Ye, Q., et al. 2016, in *Software and Cyberinfrastructure for Astronomy IV*, Vol. 9913, SPIE, 49

Dong, J., Juan, L., Ya-jun, W., & Jin-qing, W. 2014, *Annals of Shanghai Astronomical Observatory, CAS*, 1, 51

Dong, J., Zhong, W., Wang, J., Liu, Q., & Shen, Z. 2018, *IEEE Transactions on Antennas and Propagation*, 66, 2044

Ford, H. A., Anderson, R., Belousov, K., et al. 2014, in *Ground-based and Airborne Telescopes V*, Vol. 9145, SPIE, 102

Ford, J. M., & Clark, M. H. 2002, in *Advanced Telescope and Instrumentation Control Software II*, Vol. 4848, SPIE, 12

Götz, A., Taurel, E., Pons, J., et al. 2003, ICALEPCS2003, Gyeongju, October

Nikolic, B., Prestage, R., Balser, D., Chandler, C., & Hills, R. 2007, *Astronomy & Astrophysics*, 465, 685

Shen, T., Ibsen, J., Olguin, R., & Soto, R. 2011, in *Proceedings of ICALEPCS*

Tan, Y. 2015, *Tango Controls Documentation*

Verdier, P., Pons, J., Poncet, F., & Leclercq, N. 2011, in *Proceedings of*

ICALEPCS