Промежуточное программное обеспечение на робототехническом контроллере ТРИК

Гагина Л.В., студентка кафедры системного программирования СПбГУ, lada.gagina@gmail.com

Мордвинов Д.А., аспирант кафедры системного программирования СПбГУ, dvvrd@gmail.com

Аннотация

При разработке программных систем для робототехнических платформ зачастую приходится решать множество рутинных задач, таких как управление данными сенсоров и приводов, сбор информации о состоянии робота, реализация часто используемых алгоритмов робототехники, избегание препятствий, детектирование объектов на изображении камеры или построение карты местности. Задачу повторного использования таких алгоритмов помогает решать робототехническое промежуточное программное обеспечение (robotics middleware). В данной работе рассмотрены выбор и адаптация программного обеспечения промежуточного уровня для контроллера ТРИК.

Введение

В настоящее время роботизированные системы проникают почти во все сферы нашей жизни, а робототехника интенсивно развивается. Роботы способны выполнять многие виды работы на порядки лучше человека, чем объясняется их возрастающая популярность.

Помимо развития аппаратных платформ происходит также интенсивное развитие ПО в сфере робототехники. Отличительной его особенностью является большое количество рутинных задач, таких как управление данными сенсоров и приводов, передача сообщений между процессами, а также типичных для робототехники алгоритмов, например, избегание препятствий, планирование движения, построение карты местности. Программное обеспечение промежуточного уровня на роботе (robotics middleware) берёт на себя эти задачи и позволяет переиспользовать множество уже реализованных алгоритмов.

Существует обширный набор различного открытого промежуточного ПО для роботов, которое можно переиспользовать на контроллере ТРИК. Наиболее известным и часто использующимся на данный момент в мире

является ROS [1], однако множество лабораторий в мире имеют интересные разработки, использующие альтернативное ПО промежуточного уровня. Таким образом, возникает задача анализа существующего промежуточного программного обеспечения для робототехнических платформ с целью выявления подходящего ПО для переиспользования на контроллере ТРИК. В данной статье будут вкратце рассмотрены варианты промежуточного программного обеспечения, а также результаты адаптирования некоторых из них для контроллера ТРИК.

Обзор

Существует несколько обзорных статей промежуточного программного обеспечения робототехники [2][3], по прочтении которых был сформирован первичный список для сравнения:

- ASEBA;
- CLARAty;
- MARIE;
- MIRO;
- OpenDaVINCI;
- OpenRDK [4];
- OPRoS;
- ORCA;
- OROCOS;
- Player [5];
- ROS [6][7];
- YARP [8];

Большинство из них работает под управлением Linux и позволяют использовать язык C++ (и иногда другие языки), а также координируют обмен сообщениями между компонентами приложений. Необходимо отметить, что ROS, как наиболее популярное промежуточное ПО, рассмотрен отдельно в других работах, поэтому в данной работе подробно рассмотрен не будет.

Критерии

Для выбора промежуточного программного обеспечения для адаптации на контроллере ТРИК были сформулированы критерии:

- открытый исходный код;
- популярность (количество статей, упоминающих данное промежуточное ПО);
- наличие поддержки (дата выхода последней версии или последней ревизии в системе контроля версий);
- количество готовых компонентов (необходимых компонентов по результатам опроса, проведённого в лаборатории робототехники)
- переносимость (по количеству платформ, на которые уже был выполнен перенос промежуточного ПО).

После применения критериев были оставлены для дальнейшего рассмотрения Player и OpenDaVINCI, которые впоследствии были адаптированы для контроллера ТРИК. В качестве примера приведем здесь некоторые детали адаптации системы Player.

Адаптация промежуточного ПО Player

Промежуточное ПО Player состоит из сервера для управления роботом, набора драйверов и утилит для доступа к данным драйвера. Поставляемые по умолчанию драйвера реализуют как доступ к низкоуровневым данным (данные с сенсоров), так и к высокоуровневым (применение алгоритмов к данным с сенсоров).

Для использования Player с готовыми драйверами достаточно написать конфигурационный файл, с указанием того, какие драйверы будут использованы, и код на одном из языков программирования (C, C++, Python, Ruby), в котором посредством прокси-серверов можно получить доступ к данным драйвера и использовать их (см. Рис. 1). Такой подход сработал без дополнительных манипуляций для драйвера UVC камеры, подключенной к контроллеру ТРИК, что позволило, к примеру, получать набор координат объектов, детектированных на изображении.

В случае с силовыми моторами контроллера ТРИК было необходимо реализовать драйвер для управления позицией и скоростями робота и сбора информации о текущем местоположении робота. Для этого был

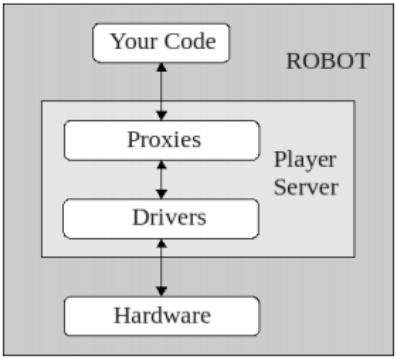

Рис. 1: Схема работы промежуточного ПО Player

использован базовый класс ThreadedDriver, в наследнике которого был реализован конструктор и методы для подключения сервера к драйверу, переопределены методы MainSetup и MainShutdown, использующиеся при подключении новых клиентов к драйверу, и методы ProcessMessage и UpdateData, использующиеся для отправки команд, поступающих от пользовательского кода, на моторы, и для получения новых данных о местоположении робота, соответственно.
Для взаимодействия с моторами была использована функциональность библиотек trikRuntime. При этом обработка внутренних событий от периферии происходит в отдельном потоке. Аналогичным образом были реализованы драйвера датчиков расстояния и света.

Заключение

В результате работы было рассмотрено несколько вариантов промежуточного программного обеспечения робототехники и выделены отдельные системы для использования на контроллере ТРИК.
На данном этапе работы инструментарий Player успешно запущен на контроллере с использованием драйверов, получающих изображение с камеры, подключенной к контроллеру, и детектирующих объекты на нём. Также реализованы драйверы для силовых моторов и датчиков расстояния контроллера, что позволяет системе Player управлять движением робота с учетом препятствий.

Литература

[1] ROS/Introduction http://wiki.ros.org/ROS/Introduction (Дата обращения: 26.04.2016)

[2] *Ayssam Elkady and Tarek Sobh*. Robotics Middleware: A Comprehensive Literature Survey and Attribute-Based Bibliography. http://hindawi.com/journals/jr/2012/959013/

[3] *Nader Mohamed, Jameela Al-Jaroodi, and Imad Jawhar*. Middleware for Robotics: A Survey. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4681485

[4] *Daniele Calisi, Andrea Censi, Luca Iocchi, Daniele Nardi*. OpenRDK: a modular framework for robotics software development. http://www.dis.uniroma1.it/ĩocchi/publications/iocchi-iros08.pdf

[5] *Radu Bogdan Rusu, Alexis Maldonado, Michael Beetz*. Player/Stage as Middleware for Ubiquitous Computing. https://www.researchgate.net/profile/Matthias_Kranz/publication/220031570_Playerstage_as_middleware_for_ubiquitous_computing/links/0fcfd509a3fda48014000000.pdf

[6] *Stefan Diewald, Luis Roalter, Andreas Möller, Matthias Kranz*. Simulation of Tangible User Interfaces with the ROS Middleware. https://www.researchgate.net/profile/Stefan_Diewald/publication/260035267_Simulation_of_Tangible_User_Interfaces_with_the_ROS_Middleware/links/0deec52fa4b00b152b000000.pdf

[7] *Veli BAYAR, Bora AKAR, Uğur YAYAN, H.Serhan YAVUZ, Ahmet YAZICI*. Fuzzy Logic Based Design of Classical Behaviors for Mobile Robots in ROS Middleware. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6873613

[8] *Giorgio Metta, Paul Fitzpatrick and Lorenzo Natale*. YARP: Yet Another Robot Platform. http://cdn.intechweb.org/pdfs/4161.pdf