Язык программирования роботов в терминах потоков данных

Г.А. Зимин
Санкт-Петербургский государственный университет
Кафедра системного программирования
Email: zimin.grigory@gmail.com

Д.А. Мордвинов
Санкт-Петербургский государственный университет
Кафедра системного программирования
Email: mordvinov.dmitry@gmail.com

***Аннотация*—В статье описывается язык программирования роботов в терминах потоков данных на основе модельно-ориентированного подхода. Дается краткий обзор языков программирования роботов и оговариваются причины создания нового языка. Рассматриваются подходы к реализации систем управления роботами. В заключение представлено решение типовой задачи создания системы управления на описанном языке.**

I. Введение

Взаимодействие с роботами и языки программирования для них — это актуальная тема исследований: статьи, освещающие связанные с ними вопросы постоянно обозреваются на крупных конференциях, к примеру ICRA[1], IROS[2], MORSE[3].

Последние три десятилетия продолжается активное изучение и исследование возможностей применения визуальных языков программирования — каждый год проводится крупные конференции, такие как VL/HCC[4], MDKE[5]. В робототехнике визуальные языки программирования также нашли свое применение[?], [?], [?], [?], [?]. Они позволяют быстрее создавать и нагляднее отображать системы управления роботами, обучать робототехнике школьников — примерами этого могут служить различные среды программирования роботов для начинающих, такие как NXT-G[6], TRIK Studio[7], ROBOLAB[8].

Программы управления роботом по своей природе реактивны: они обрабатывают данные, непрерывно приходящие с множества датчиков и посылают команды на приводы. В силу этого для программирования роботов хорошо подходят реактивные языки, или языки программирования потоков данных (data flow languages). Эти языки также активно развивались от текстовых к широко распространенным сейчас визуальным языкам потоков данных[?]. В случае программирования потоков данных наглядность визуальных языков особо превосходит текстовые, так как сами потоки данных явно отображены на диаграмме. Существуют большие и сложные среды программирования, такие как LabVIEW[9] и Simulink[10], которые предоставляют большой и даже громоздкий набор библиотек для программирования роботов. Подробнее про языки программирования роботов будет сказано в секции II.

Для обучения основам робототехники и кибернетики существует большое количество конструкторов роботов, таких как линейка конструкторов LEGO MINDSTORMS[11], контструктор TRIK[12]. Подавляющее большинство современных языков программирования, используемых для обучения программированию на них, основаны на модели потока управления, а не потока данных. При этом во время их освоения зачастую появляется ощущение неудобства решения на них типовых задач управления роботом. Дополнительная «ступень» в виде простого в освоении потокового языка была бы весьма полезной при переходе от учебных языков программирования к тем, что используются в индустрии. Целью данной работы является создание такого языка программирония конструкторов роботов TRIK.

Остаток работы устроен следующим образом: в секции II дан краткий обзор существующих визуальных языков программирования для роботов и их применени, в секции III обозреваются архитектуры, на которых может быть построена система управлениями робота-

[1]http://www.icra2016.org/ ICRA 2016 в Стокгольме. [Дата обращения 10 марта 2016]

[2]http://www.iros2016.org/ IROS2016. [Дата обращения 10 марта 2016]

[3]https://st.inf.tu-dresden.de/MORSE14/ Model-Driven Robot Software Engineering 2014. [Дата обращения 10 марта 2016]

[4]https://sites.google.com/site/vlhcc2016/ VL/HCC 2016. [Дата обращения 10 марта 2016]

[5]https://sselab.de/lab2/public/wiki/MDKE15/index.php Model-Driven Knowledge Engineering for Improved Software Modularity in Robotics and Automation. [Дата обращения 10 марта 2016]

[6]http://www.legoengineering.com/program/nxt-g/ NXT-G. [Дата обращения 10 марта 2016]

[7]http://www.trikset.com/. TRIK [Дата обращения 10 марта 2016]

[8]http://www.legoengineering.com/program/robolab/. ROBOLAB [Дата обращения 10 марта 2016]

[9]http://www.ni.com/labview/ LabVIEW System Design Software. [Дата обращения 10 марта 2016]

[10]http://www.mathworks.com/products/simulink/ Simulation and Model-Based Design. [Дата обращения 10 марта 2016]

[11]http://www.lego.com/en-us/mindstorms/products LEGO MINDSTORMS EV3. [Дата обращения 10 марта 2016]

[12]http://blog.trikset.com/p/blog-page_6355.html/ Конструктор TRIK. [Дата обращения 10 марта 2016]

ми, в секциях IV и V дается описание языка и его применение к решению задачи управлнения роботом при помощи оператора с автоматическим избеганием столкновений, наконец, в секции VI коротко будут сформулированы итоги работы.

II. Обзор современных языков программирования роботов

Среды программирования роботов можно разделить на три класса: учебные, позволяющие программировать небольших роботов; промышленные, обладающие богатым инструментарием для создания систем управления роботов, различных моделей; академические, которые реализуют какие-либо интересные идеи, однако часто недоступны для скачивания либо их стабильность оставляет желать лучшего.

К учебным, к примеру, можно отнести среду разработки EV3 Software[?] для конструктора Lego Mindstorms EV3, среда разработки NXT-G[13] и конструктор LEGO MINDSTORMS NXT и ROBOLAB[14], конструктор TRIK и среда программирования TRIK Studio[15]. Упомянутые среды программирования позволяют с легкостью решать типовые задачи управления роботом: найти выход из лабиринта, проехать по линии, используя сенсоры; создавая примитивные системы управления достаточные для обучения пользователя основам программирования и управления роботом. Но своей простотой они обязаны слабой гибкости языка, по сути они предоставляют последовательную модель потока управления роботом, описанную наглядными графическими моделями.

В инженерной индустрии весьма популярна среда общего назначения среда LabVIEW компании National Instruments и визуальный потоковый язык программирования G, а также среда для моделирования динамических моделей и различных систем управления Simulink. Данные программные продукты предоставляют пользователю огромный набор моделей и библиотек для создания любых систем управления, испытательных стендов, систем реального времени, используя развитый модельно-ориентированный подход. В частности на LabVIEW возможно программирование небольших роботов. Хотя известны применения LabVIEW в образовательных целях[?], большинство времени в образовательном процессе уходит на изучение самой среды, а не алгоритмов и подходов робототехники. Следует отметить, что эти среды распространяются под коммерческой лицензией.

Другой пример промышленной системы — Microsoft Robotics Developer Studio (MSRDS)[?], бесплатная для академического использования, удобную для программирования распределенных робототехнических систем в терминах потоков данных, компоненты которых представляются веб-сервисами. MSRDS официально поддерживает множество робототехнических платформ, среди которых есть LEGO NXT[?], однако автономный режим не поддерживается. MSRDS обладает возможностью поддержки пользовательских робототехнических платформ, однако сама среда не поддерживается с 2014 года.

В данной области ведется множество исследовательских работ, к примеру, в диссертации[?] описан визуальный язык программирования в терминах расширенных машин Мура, [?],[?] описывают визуальный язык в терминах потоков данных с привязкой к языку программирования *occam-π* и инструментарию Transterpreter, а также применение к управлению «роем» роботов. Работа[?] описывает визуальный язык и среду для обучения новичков программированию роботов, среда предоставляет удобный интерфейс, однако полученная технология обладает слишком «слабым» функционалом и требует значительной доработки, а также недоступна для скачивания.

Исходя из результатов обзора ясно, что ни одна из упомянутых сред не подходит в том виде, в котором она есть для программирования TRIK в терминах потоков данных. При этом самым близким к желаемому результату является TRIK Studio, так как это единственная среда в открытом доступе с легко масштабируемой архитектурой, обладающей возможностью расширения новым визуальным языком программирования роботов и переиспользования кодовой базы таких «рутинных» операций, как взаимодействие с роботом.

III. Архитектура Брукса

С выходом статьи Родни Брукса[?], описывающей возможность декомпозировать задачу управления роботом горизонтально на уровни ответственности, стало возможно проще решать следующие задачи управления: задача управления роботом, избегающим столкновений, с пульта; задача управления с пульта балансирующим сегвеем, избегающим столкновений с препятствиями. Языки потоков данных удачно подходят под эту модель, что подтверждают неоднократные публикации на эту тему: [?], [?], [?], [?]. В этой модели, система управления представлена как набор уровней ответственности, которые отвечают за различные модели поведения робота и строятся последовательно. При этом по очевидным причинам упрощается масштабируемость системы и увеличивается ее отказоустойчивость.

У архитектуры Брукса есть множество альтернатив, к примеру, «Колония» Джонотана Коннеля, архитектура «выбора-действия» Патти Мэйса, «схема двигателя» Рональда Аркина[?], модель управления для системы Телеробота[?]. Однако среди упомянутых подхо-

[13]http://www.legoengineering.com/program/nxt-g/ NXT-G. [Дата обращения 10 марта 2016]

[14]http://www.legoengineering.com/program/robolab/ ROBOLAB [Дата обращения 10 марта 2016]

[15]http://www.trikset.com/ TRIK [Дата обращения 10 марта 2016]

дов, подход Брукса является наиболее популярным, поэтому основополагающей концепцией разрабатывамого языка было решено сделать поддержку именно его.

IV. Описание языка

Уровень развитие модельно-ориентированного(DSM) подхода[?], позволяют быстро создавать достаточно сложные визуальные языки программирования за дни или даже часы. Среда программирования TRIK Studio является примером системы, созданной с применением DSM-подхода на базе платформы QReal[?][?]. Основываясь на промышленном опыте разработчиков TRIK Studio, было решено создавать потоковый язык программирования роботов на базе платформы QReal. Первый прототип языка был создан в течение недели. На момент написания статьи язык и средства его поддержки находятся в активной разработке.

Опишем некоторые особенности разрабатываемой технологии.

- Архитектура Брукса легко выражается средствами языка.
- Диаграммы поведения роботов, так же как и в TRIK Studio, имеют возможность быть проинтерпретированными на двумерной имитационной модели робота.
- В ближайших планах стоит генерация диаграмм потоков данных в текстовые языки, используемые для программирования TRIK — в первую очередь, JavaScript, F#[?] и Kotlin.

Элементы языка

Элементы языка можно разделить по назначению на несколько групп.

- *Элемент потока данных.* Связь, реализующая поток для передачи данных.
- *Блоки управления приводами.* Блоки принимающие числовые значения, отвечающие за подачу импульсов на приводы (силовые моторы, сервомоторы и т.д.).
- *Блоки считывания данных с сенсоров.* Блоки, генерирующие соответствующие значения, которые подлежат дальнейшей обработке.
- *Блоки синхронизации и фильтрации.* Позволяют временно блокировать передачу данных, устанавливать количество наборов пропускаемых данных и время между отправками.
- *Блоки поддержки устройств ввода.* Позволяют считывать данные с устройств операторского контроля (в данный момент поддержан только джойстик, в планах — компьютерная мышь и клавиатура).
- *Блоки рисования на экране.* Отвечают за векторное и растровое рисование на экране.
- *Блоки видеозрения.* Предоставляют доступ ко всем возможностям видеозрения контроллера TRIK.
- *Блоки взаимодействия между роботами.* Отвечают за групповую координацию роботов.
- *Блок текстового программирования.* Блок, позволяющий произвести обработку данных, приходящих в блок на текстовом языке (статически типизируемом диалекте Lua, поддержка которого унаследованна от кодовой базы TRIK Studio). Чаще всего такой блок будет использоваться для задания математических операций над данными.
- *Блоки управляющих конструкций.* В данную категорию входят блоки, отвечающие за циклы, условные развилки, множественный выбор, распараллеливания, подавления, ингибиции и блока «подпрограмма» для переиспользования кода. Последние четыре блока обеспечивают поддержку архитектуры Брукса в языке.

Ограничения парадигмы поточного программирования

Классическая модель программирования потоков данных и блоков, генерирующих и принимающих данные, предполагает, что каждый блок будет работать в отдельном потоке или на отдельном процессоре. Поэтому технологическим вызовом является учет мобильности платформ типа TRIK и LEGO NXT, так как ограниченные вычислительные мощности в режиме автономной работы делают не только желательным, а скорее необходимым максимальное сокращение нитей исполнения в генерируемом коде.

Поэтому модель управления потоками данных в языке была реализована псевдопараллелизмом на основе переходящих между блоками токенов управления и системы событий. Это позволило программировать роботов в стиле потоков данных, но при этом не терять в производительности при выполнении сгенерированного по модели кода на роботе.

V. Пример решения задачи

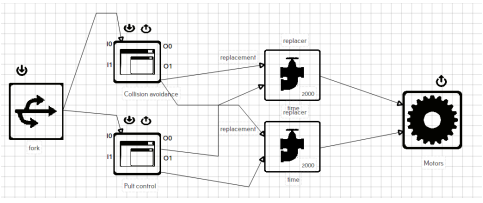

Рис. 1: Программа управления роботом.

Рассмотрим задачу управления движением робота с помощью пульта, при условии, что робот не позволит нам столкнуться лбом с припятствием. Имеем двухколесного робота для обнаружения припятствий оборудованного спереди датчиком расстояния.

Очевидно, что можно разбить задачу на два уровня поведения. Первый будет ответственен за избегание столкновений: если робот близок к столкновению, то он должен отъехать назад, вне зависимости от того,

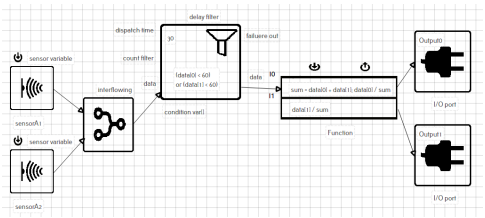

Рис. 2: Уровень избегания столкновений.

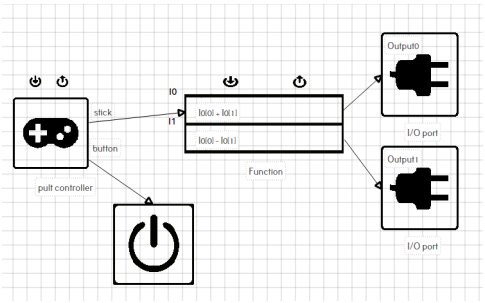

Рис. 3: Уровень управления с пульта.

что нажимает пользователь на пульте. Второй будет отвечать за обслуживание запросов пользователя.

Рассмотрим первый уровень: бла-бла-бла Рассмотрим второй уровень: бла-бла-бла Оба уровня вместе.

VI. Заключение

На момент написания статьи был реализован прототип технологии програмирования роботов TRIK в терминах потоков данных. Система предоставляет возможность интерпретации диаграмм на двумерной имитационной модели робота и (в ближайшем будущем) генерации кода из диаграмм в текстовые языки, используемые для программирования TRIK (JavaScript, F# и Kotlin). Также технология предоставляет поддержку архитектуры Брукса на уровне языка, что продемонстрировано в статье на примере операторского управления роботом с автоматическим избеганием столковений.

Полученная система может рассматриваться как платформа для дальнейших научных исследований. К примеру, интересной представляется автоматическая генерации метамодели языка по спецификациям ПО промежуточного уровня на роботе (к примеру, ROS[16]). Другим направлением возможной работы является описание строгой семантики языка для применения различных формальных методов анализа программ, выраженных в нем.

[16]http://www.ros.org ROS [Дата обращения 10 марта 2016]