УДК 007.52 DOI:10.18324/2077-5415-2020-1-53-60

Разработка системы команд для управления роботом-манипулятором

Д.С. Колтыгин[a], И.А. Седельников[b]

Братский государственный университет, ул. Макаренко, 40, Братск, Россия
[a] kds@brstu.ru, [b] Ohtargil@yandex.ru
[a] https://orcid.org/0000-0002-8250-6907, [b] https://orcid.org/0000-0001-9797-9312
Статья поступила 20.01.2020, принята 06.02.2020

Выбор или разработка языка управления роботом-манипулятором являются неотъемлемой частью процесса создания системы управления. Языки различаются тем, на каком уровне они предназначены работать, задачами, которые призваны решать. Некоторые из них являются результатом разработки универсального способа общения человека и робота, другие призваны решать прикладные задачи конкретных типов механизмов в определенных сферах производственной деятельности. Многообразие языков программирования также определяется тем, что многие из них разрабатывались теми или иными производителями для конкретных платформ и сред исполнения. Одной из эффективных является разработка системы команд для существующего или разрабатываемого контроллера. Предлагаемое исследование посвящено разработке языка программирования и системы команд для использования в создании программы управления роботом-манипулятором, включая команды, необходимые для функционирования технологического процесса, а также команды высокоуровневого языка программирования. Как результат разработки алгоритмов работы команд и их программной реализации, предлагается язык команд для управления технологическим процессом. Создана библиотека, реализующая представленные функции для использования при написании программы управления технологическим оборудованием. Как известно, любой технологический процесс на производстве включает в себя множественное выполнение однотипных операций, что, в свою очередь, подразумевает использование циклических структур. Для управления с помощью датчиков и сенсоров, а также построения сложных алгоритмов действий возникает необходимость использования разветвляющих структур. Для увеличения возможностей данных структур применяются дополнительные команды управления, которые могут быть необходимы при выполнении некоторых операций. К ним можно отнести операции перехода к другим командам программы управления, добавление заранее прописанных программ или их частей в указанное место текущей программы управления, а также операторы управления, такие как прерывание, продолжение цикла и приостановка на указанное время. Применение команд, осуществляющих выбор системы координат управления, позволяет быстрее и точнее настроить оборудование для применения в технологическом процессе, что, наряду с прочим, определяет практическую значимость исследования.

Ключевые слова: робот; манипулятор; G-код; система команд; алгоритм; управление.

Development of a command system for controlling a robotic arm

D.S. Koltygin[a], I.A. Sedelnikov[b]

Bratsk State University; 40, Makarenko St., Bratsk, Russia
[a] kds@brstu.ru, [b] Ohtargil@yandex.ru
[a] https://orcid.org/0000-0002-8250-6907, [b] https://orcid.org/0000-0001-9797-9312
Received 20.01.2020, accepted 06.02.2020

The choice or development of a control language for the robot-manipulator is an integral part of the process of creating a control system. Languages differ in the level at which they are intended to work and the tasks that they are called to solve. Some of them are the result of work on the development of a universal way of communication between humans and robots, while others are designed to solve applied problems of specific types of mechanisms in specific areas of human production. The variety of programming languages is also determined by the fact that many of them were developed by various manufacturers for specific platforms and runtimes. One of the effective ways to solve this problem is to develop a command system for an existing or being developed controller. The proposed study is devoted to the development of a programming language and a command system for use in creating a control program for a robot manipulator by including the commands necessary for the functioning of the technological process, as well as commands of a high-level programming language. As a result of the development of team work algorithms and their software implementation, a command language is proposed for process control. A library has been created that implements the functions presented for use when writing a process equipment control program. Any technological process in production includes the multiple execution of the same type of operations, which, in turn, implies the use of cyclic structures. To control using sensors and sensors, as well as to build complex action algorithms, it is necessary to use branching structures. To increase the capabilities of these structures, it is possible to use additional control commands, which may be necessary when performing certain operations. These include operations of switching to other commands of the control program, adding pre-registered programs or parts of them to a specified location in the current control program, as well as control operators such as interrupting, continuing a cycle, and pausing for a specified time. The use of commands that select the control coordinate system allows you to quickly and accurately configure the equipment for use in the process, which, among other things, determines the practical significance of the study.

Keywords: robot; manipulator; G-code; command system; algorithm; control.

Введение. Одним из важнейших этапов после определения набора системных операций, требуемых для написания управляющих программ, является выбор либо разработка способа написания программы управления.

Существует достаточно много языков программирования роботизированных систем, вот только некоторые из них: AL, AML, MELFA Basic 4, Movemaster Command, IRL, KRL, RAPID, Vplus, FROB, RPL, RPS, RCCL, Saphira, Colbert. Конечно, все эти языки различаются тем, на каком уровне они предназначены работать. Так, некоторые предназначены для «общения» с внешним миром, например, с оператором, другие — для реализации логики алгоритма на машинном уровне (языки низкого уровня). Также есть отличие в задачах, которые призваны решать эти языки. Возможно, некоторые из них, более «теоретические», являются результатом разработки универсального способа общения человека и робота, в то время как другие призваны решать более прикладные задачи конкретных типов механизмов в конкретных сферах производственной деятельности. Кроме того, многообразие языков программирования определяется тем, что многие из них разрабатывались для конкретных платформ и сред исполнения конкретными производителями.

В соответствии с решаемыми задачами управления выделяют четыре уровня языка программирования:

1. Низший уровень — используется для управления исполнительными приводами в виде точных значений линейного или углового перемещения отдельных звеньев интеллектуальной системы.

2. Уровень манипулятора — позволяет осуществлять общее управление всей системой, позиционируя рабочий орган робота в координатном пространстве.

3.Уровень операций — служит для формирования рабочей программы путем указания последовательности необходимых действий для достижения конкретного результата.

4. Высший уровень — задание программы без детализации указывает, что надо сделать.

Разработка используемого языка программирования требует детальной оценки преимуществ и недостатков существующих вариантов.

В связи с тем, что для языков программирования РТК не существует единого стандарта, производители используют собственные разработки, имеющие значительные отличия.

Чтобы выбрать наиболее эффективный подход к описанию программы управления разрабатываемой системы управления манипуляционным роботом, рассмотрим особенности программирования робототехнологических комплексов (РТК) ведущих производителей [1; 2]

Существует три основных подхода к описанию синтаксиса языка программирования для РТК:

1. Использование команд с параметрами для задания траекторий и точек перемещения (ABB, Yaskawa, Motoman).

2. Использование структурированного представления данных (KUKA).

3. Использование кодовых команд (Motorola).

Рассмотрим их достоинства и недостатки.

Использование команд с параметрами позволяет записать в одной строке кода всю необходимую информацию о перемещениях, включая тип интерполяции траектории и требуемые перемещения, однако при этом теряется гибкость в задании параметров и усложняется их использование в циклических операциях, где требуется изменение отдельных параметров.

Использование структур данных для задания параметров движения делает управляющую программу более удобной для реализации сложных алгоритмов и доступа к отдельным элементам описания траектории. Недостатком данного подхода является увеличение объема кода и усложнение компилятора.

Наконец, использование кодовых программ позволяет написать программу, обладающую наименьшим размером и наиболее простую для компиляции, но при этом такая программа будет наименее читаемой.

Для управление роботом производителем выбран язык программирования стандарта G-code (RS-274), команд которого недостаточно для полноценной реализации функций.

Основная система команд G-код. *G-код* — условное именование языка программирования устройств с числовым программным управлением (ЧПУ). Был создан компанией Electronic Industries Alliance в начале 1960-х. Окончательная доработка была одобрена в феврале 1980 г. как стандарт RS274D. Комитет ISO утвердил G-код как стандарт ISO 6983-1:2009, Госкомитет по стандартам СССР — как ГОСТ 20999-83 [3; 4].

Сводная таблица кодов. Основные (называемые в стандарте подготовительными) команды языка начинаются с буквы **G**:

- Перемещение рабочих органов оборудования с заданной скоростью (линейное и круговое).
- Выполнение типовых последовательностей (таких, как обработка отверстий и резьба).
- Управление параметрами инструмента, системами координат и рабочих плоскостей.

Таблица 1. Подготовительные (основные) команды

Коды	Описание
G00-G03	Позиционирование инструмента
G17-G19	Переключение рабочих плоскостей (XY, ZX, YZ)
G20-G21	Не стандартизовано
G40-G44	Компенсация размера различных частей инструмента (длина, диаметр)
G53-G59	Переключение систем координат
G80-G85	Циклы сверления, растачивания, нарезания резьбы
G90-G91	Переключение систем координат (абсолютная, относительная)

Технологические команды языка начинаются с буквы M и включают такие действия, как:

- Сменить инструмент.
- Включить/выключить шпиндель.
- Включить/выключить охлаждение.
- Работа с подпрограммами.

Параметры команд задаются буквами латинского алфавита.

Таблица 2. Параметры команд

Код	Описание
X	Координата точки траектории по оси X
Y	Координата точки траектории по оси Y
Z	Координата точки траектории по оси Z
P	Параметр команды
F	Скорость рабочей подачи. Для фрезерных станков это дюймы в минуту (IPM) или миллиметры в минуту (mm/min), для токарных станков — дюймы за оборот (IPR) или миллиметры за оборот (mm/rev).
S	Частота вращения шпинделя
R	Параметр стандартного цикла или радиус дуги (расширение стандарта)
D	Параметр коррекции выбранного инструмента
L	Число вызовов подпрограммы
I	Параметр дуги при круговой интерполяции. Инкрементальное расстояние от начальной точки до центра дуги по оси X
J	Параметр дуги при круговой интерполяции. Инкрементальное расстояние от начальной точки до центра дуги по оси Y
K	Параметр дуги при круговой интерполяции. Инкрементальное расстояние дуги по оси Z

Команды, реализованные разработчиком оборудования ООО «Уральские станки» (Челябинск). Компанией «Уральские станки» разработана программа (рис. 1) для управления роботом-манипулятором DELTA, общий вид которого представлен на рис. 2.

В учебном пособии [5], которое является руководством пользователя для данного робота, представлены следующие команды.

Таблица 3. Команды, реализованные разработчиком оборудования

Код	Описание
Перемещение манипулятора	
G01 X… Y… Z… F…	Выполняется одновременное перемещение по указанным координатам в заданную точку со скоростью F (если указана). Если параметр F не указан — перемещение будет выполнено с заданной ранее скоростью. X, Y, Z — оси управления роботом, параметр каждой оси 00000 пятизначное число
Команды управление схватом	
M11 P	Сжатие схвата на заданное количество шагов
M10 P	Разжатие схвата на заданное количество шагов
Если параметр P не задан — будет использовано значение по умолчанию	
Команды управления положением робота	
RHOME	Координаты X, Y, Z будут возвращены в исходное положение. Схват будет разжат
HOME	Процедура выхода в ноль с калибровкой. Выходит в ноль по заранее заложенной программе по всем осям. В установленной последовательности на производстве
HMX, HMZ, HMY, HMV, HMW	Выход в ноль по соответствующим осям
MOVL0000/ MOVR0000	Команда движения схвата без абсолютных координат. Хранится и используется только относительное значение. Такое значение индицируется также в информационной строке
MOVV	Команда абсолютных величин перемещения. Координата хранится в счетчике и выводится в информационной строке вместе со всеми координатами. При выходе в ноль обнуляется командой HMV, при общем выходе в ноль (одной командой HOME) также обнуляется
MOVW	Команда абсолютных величин перемещения. Координата хранится в счетчике и выводится в информационной строке вместе со всеми координатами. При выходе в ноль командой HMWW обнуляется, при общем выходе в ноль (одной командой HOME) также обнуляется
Системные команды	
ACSR	Процедура управления коэффициента разгона. Величина должна лежать в пределах от 1 до 8. Чем меньше цифра, тем разгон длиннее. По умолчанию величина равна 5
SPDR	Длина площадки разгона. Чем она больше, тем больше величина минимального разгона. В шагах по умолчанию установлена в районе 500
CDR	При получении команды контроллер выводит текущие координаты в стандартном виде

Как видно, помимо отсутствия команд, требуемых для автоматизации производственных процессов, отсутствует возможность реализовать требуемые функции внутри программы, например, с помощью меню или кнопок.

Из описанного выше можно сделать вывод, что существующего набора команд и программных средств недостаточно для создания полноценного автоматизированного производства. Как основная система команд, так и команды, реализованные разработчиком оборудования, не позволяют выполнять циклические и условные операции, что является основой автоматического производства. Отсутствует возможность использования датчиков, что в купе с неточно реализованной моделью затрудняет использование робота-манипулятора. Более подробно о всех недостатках РТК указано в статьях [5; 6].

Разработка системы команд. Для управления роботом могут применяться несколько систем координат. Для этого в язык программирования включен набор средств, предназначенных для преобразования между координатными системами.

Наиболее удобным с точки зрения пользователя является задание координат в виде структур, включающих элементы как рабочей, так и собственной координатной системы манипуляционного робота.

Данное представление позволяет использовать в одной управляющей программе координаты различных систем координат без необходимости пользователю следить за текущим режимом.

Для решения этой проблемы в состав средств языка были введены команды, указывающие требуемую конфигурацию робота в пространстве. Так как типы манипуляторов могут различаться по своей конфигурации, данные команды имеют не жестко заданный набор аргументов, а учитывают особенности кинематической структуры робота. Таким образом, значение их параметров определяется не описанием языка, а конкретной реализацией системы управления роботом, предназначенной для использования с манипулятором.

Предлагается ввести три команды, определяющие системы координат, используемые при функционировании управляющей программы.

Для создания сложных программ управления язык программирования поддерживает набор средств алгоритмического программирования [8], позволяющих оптимизировать процесс разработки и выполнения программы. По аналогии с языками высокого уровня к ним относятся подпрограммы, циклы, ветвления.

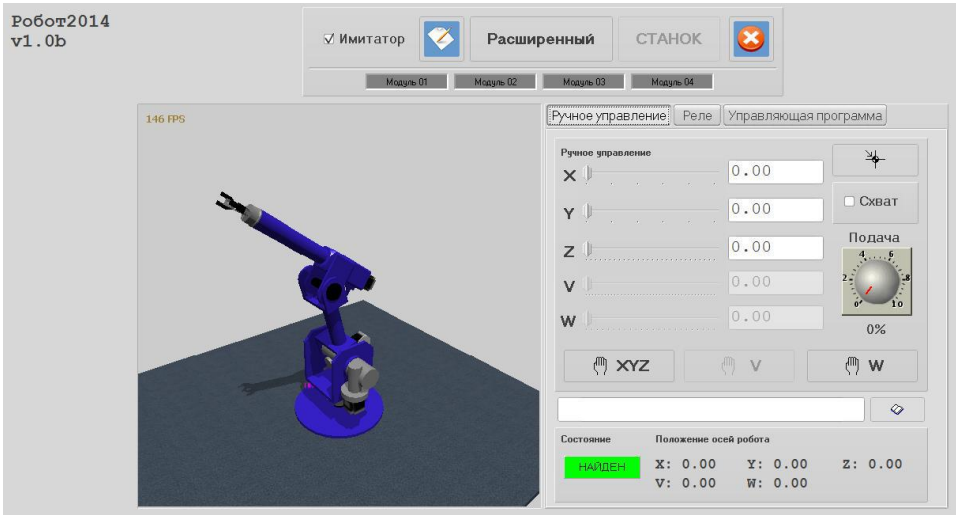

Рис. 1. Интерфейс программы Robot 2014 *v.1.0b*

Подпрограммы предназначены для выделения отдельных блоков кода для многократного их использования в теле программы. Они могут применяться как для простого выполнения команд, так и для проведения вычислений либо иных операций, приводящих к получению результата (проверка значения системных переменных, чтение состояния системных устройств).

Рис. 2. Общий вид манипулятора DELTA

Кроме самих циклов в языке предусмотрены команды для их досрочного прерывания, приостановки и продолжения (BREAK, PAUSE и CONTINUE соответственно).

Команды ветвления предназначены для организации сложных программ, управляемых условными операторами. В коде программы они могут быть представлены в виде блоков условий IF–THEN–ELSE, сопровождающихся наборами команд, предназначенных для отработки при выполнении (невыполнении) условия.

Для удобства возможно создавать программу управления из отдельных модулей, прописанных заранее программ или частей кода. С помощью таких модулей удобно реализовывать команды перехода к стационарным точкам или часто встречающиеся группы операций. Реализация в языке представлена командой PR: «адрес программы».

В табл. 4 содержится описание всех разработанных и реализованных команд.

Таблица 4. Разработанные команды

Код	Описание
Управление системой координат	
STEP	Определяет систему координат, основанную на количестве шагов двигателя робота
ROBOT	Определяет систему координат, основанную на координатах робота
DEKART	Определяет систему координат, основанную на декартовых координатах
Команды организации циклической структуры	
LOOP	Безусловный цикл (цикл без условия выхода). Тело цикла начинается с последующей строки

Код	Описание
END_LOOP	Окончание безусловного цикла
FOR	Параметрический цикл (цикл с изменением значения переменной). Тело цикла начинается с последующей строки
END_FOR	Окончание параметрического цикла
WHILE	Цикл с предусловием (цикл с предварительным условием). Тело цикла начинается с последующей строки
END_WHILE	Окончание параметрического цикла
Команды управления циклом	
BREAK	Прерывание цикла
PAUSE: «время»	Приостановка цикла на заданное время
CONTINUE	Пропускает все оставшиеся в теле цикла действия и переходит к следующей итерации
Команды организации условной структуры	
IF:	Условный оператор
THEN:	Оператор выполнения условия
ELSE:	Оператор невыполнения условия (не обязательно)
END_IF	Окончание условной конструкции
Операторы перехода	
GO: «метка»	Переход на «метка»
«метка»:	Определитель перехода
Команды установления границ	
BEGIN	Начало программы
FINISH	Конец программы
DO	Начало подпрограммы
END_DO	Конец подпрограммы
Системные команды	
//	Нечитаемый комментарий
PR: «адрес программы»	Вставка преднаписанной программы (аналогично командам библиотеки)

Алгоритмы работы команд. Ниже представлены схемы функционирования основных алгоритмических операций. На рис. 3–5 изображены алгоритмы циклической структуры — работы цикла с предусловием (рис. 3), безусловного цикла (рис. 4) и цикла с параметром (рис. 5). Данные структуры не только позволяют реализовать циклические операции, которые ранее отсутствовали, но и выбрать наиболее удобный вариант в зависимости от производственной задачи и заданных условий.

На рис. 6 представлен алгоритм работы команды перехода по меткам. Данная операция требуется при навигации внутри управляющей программы, например, при работе с датчиками, в циклических и условных конструкциях.

На рис. 7 показан алгоритм работы условного оператора. Данная операция является одной из основных в процессе автоматизации, она служит для проверки данных, датчиков и т. д.

На основе разработанных команд создана библиотека, позволяющая расширить функциональные возможности языка и повысить его уровень. Данная библиотека и команды используются в программе управления роботом-манипулятором DELTA, на которую получено авторское свидетельство Роспатента № RU 2019614518 [9], но также может быть использована и при создании других РТК.

Рис. 3. Алгоритм работы цикла с предусловием

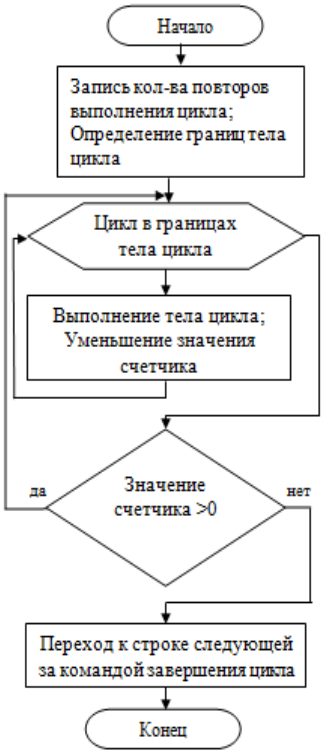

Рис. 4. Алгоритм работы безусловного цикла

Рис. 5. Алгоритм работы цикла с параметром

Рис. 6. Алгоритм работы команды перехода по меткам

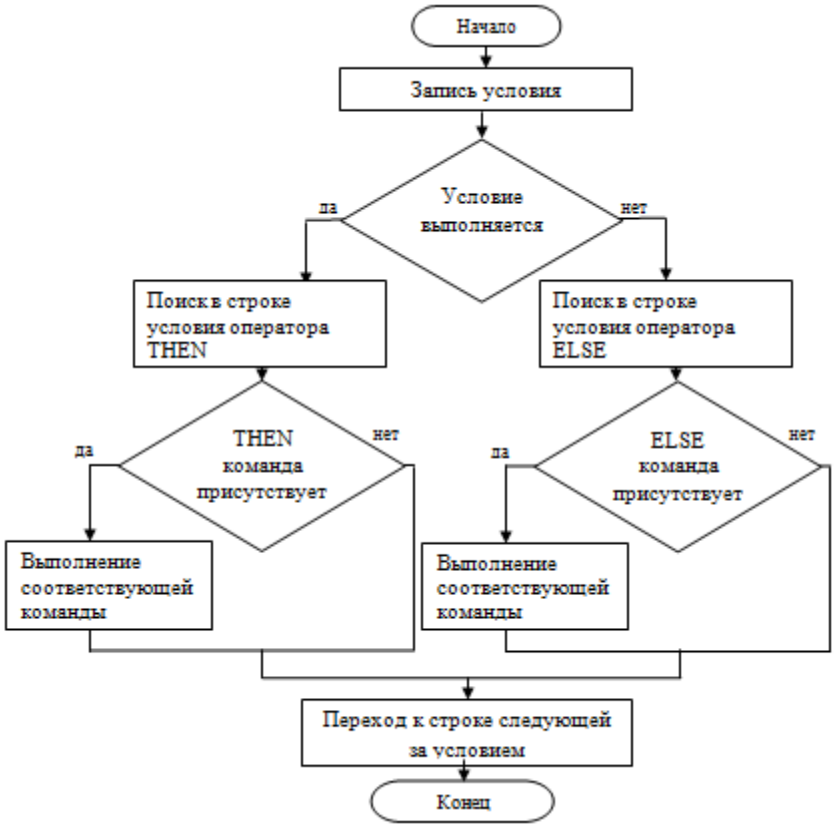

Рис. 7. Алгоритм работы условного оператора

Примеры реализации разработанных команд.

```
BEGIN //начало программы
// указываем систему координат, в которой будут выполняться операции
ROBOT
// безусловный цикл заполнения контейнера деталями выполняется 5 раз
@1: //метка перехода
LOOP 5
G01 X03900 Y01200 Z09450 F100
M11
G01 X01600 Y02200 Z09450 F100
M10
IF(f=1) // проверка наполненности контейнера
```

```
THEN // условие выполняется
BREAK //прерывание цикла наполнения
END_LOOP
//проверка наполненности контейнера
IF(f=1) // проверка условия «f – состояние датчика»
THEN // условие выполняется
WHILE(f=1)
G01 X03900 Y01200 Z09450 F100
M10
G01 X01600 Y02200 Z09450 F100
PAUSE:12 // пауза 12 сек на время сверления детали
G01 X03900 Y01200 Z09450 F100
M11
END_WHILE
ELSE // условие не выполняется
GO: @1 // переход на метку «@1»
END_IF
PR: C:/1.txt // запуск подпрограммы «1.txt» (вставляет в текущее место код, написанный в файле)
FINISH // конец программы
```

Содержимое «1.txt»

```
FOR(0;i<5;i+2)
RHOME
G01 X03900 Y01200 Z09450 F100
G01 X01600 Y02200 Z09450 F100
END_FOR
```

Заключение. Разработана система команд, позволяющая составлять программу управления не только для роботов, но и для станков с ЧПУ. Создана библиотека, реализующая представленные функции, для использования при написании программы управления технологическим оборудованием.

Любой технологический процесс на производстве включает в себя множественное выполнение однотипных операций, что, в свою очередь, подразумевает использование циклических структур. Для управления с помощью датчиков и сенсоров, а также построения сложных алгоритмов действий возникает необходимость использования разветвляющих структур. Для увеличения возможностей данных структур возможно применение дополнительных команд управления, которые могут быть необходимы для выполнения некоторых операций. К ним можно отнести операции перехода к другим командам программы управления, добавление заранее прописанных программ или их частей в указанное место текущей программы управления, а также операторы управления, такие как прерывание, продолжение цикла и приостановка на указанное время.

Применение команд, осуществляющих выбор системы координат управления, позволяет быстрее и точнее настроить оборудование для применения в технологическом процессе. Дальнейшая доработка языка может предполагать добавление необходимых команд и инструкций, необходимость в которых возникает в ходе использования языка.

Литература

1. Красильникъянц Е.В., Варков А.А., Тютиков В.В. Программное обеспечение системы управления IntNCR манипуляционным роботом // Мехатроника. Автоматизация. Управление. 2012. № 3. С. 31–36.
2. Стандарт МЭК 6-1163/3. Программируемые контроллеры. Ч. 3: Языки программирования. М.: Стандартинформ, 2016.
3. ГОСТ 24836-81. Устройства программного управления промышленными роботами. Методы кодирования и программирования. М., 1981.
4. ГОСТ 27696-88. Промышленные роботы. Интерфейсы. Технические требования. М., 1989.
5. Робот PASCAL DELTA 1-3X-USB+. Сферическая система координат. Челябинск: ООО «Уральские станки», 2015. 47 с.
6. Колтыгин Д.С., Седельников И.А. Методика разработки программы управления роботом для робота-манипулятора Delta // Науч. вестник Новосиб. гос. техн. ун-та. 2018. № 1 (70). С. 103–116.
7. Колтыгин Д.С., Седельников И.А., Программа управления роботом- манипулятором DELTA V.1.0 // Труды Братского государственного университета. Сер. Естественные и инженерные науки. 2017. Т. 1. С. 100–103.
8. Бройнль Т. Встраиваемые робототехнические системы: проектирование и применение мобильных роботов со встроенными системами управления. М.: ИКИ, 2012. 520 с.
9. Седельников И.А., Колтыгин Д.С. ROBOT DELTA V. 1.0: программа для ЭВМ RU 2019614518. Св. о рег. RU 2019614518; заявка № 2019612975 от 19.03.2019.
10. Тютиков В.В., Красильникъянц Е.В., Варков А.А. Компоненты программного обеспечения манипуляционного робота // Вестн. ИГЭУ. Вып. 4. 2011. С. 40–43.
11. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. М.: Мир, 1978. 612 с.
12. Дал У., Дейкстра Э., Хоор К. Структурное программирование. М.: Мир, 1975. 248 с.
13. Елисеев С., Свинин М., Смелягин А. Математическое и программное обеспечение в исследованиях манипуляционных систем. М.: Наука, Сиб. отд-ние, 1992. 294 с.
14. Колтыгин Д.С., Седельников И.А., Программа управления роботом- манипулятором DELTA V.1.0 // Труды Братского государственного университета. Сер. Естественные и инженерные науки. 2017. Т. 1. С. 100–103.
15. De Luca A. Programming supervision and control architectures: тез. докл. конф. // Robotics. Sapienza Universita DI Roma, 2014.
16. Industry 4.0. The Future of Productivity and Growth in Manufacturing Industries. Germany: BCG, 2015. С. 4.
17. Bösl D. Hello Industry 4.0: glossary // KUKA GmbH, Augsburg, 2017. P. 9–12.
18. The High-Tech Strategy. Germany: GTAI, 2014. P. 12–13.
19. Иванов Д.С. Новый индустриальный уклад основан на компетенциях // Умное производство. 2017. № 1. С. 65–71.
20. A new age of industrial production. The Internet of Things, Services and People. ABB, 2015, С. 7.
21. Лопота А. Тенденции развития робототехнических систем // Умное производство. 2016. № 2. С. 83–88.
22. Зимин Г.А., Мордвинов Д.А. Образовательный визуальный потоковый язык для программирования роботов // Труды ИСП РАН, 2016. Т. 28, Вып. 2. С. 45–62.
23. Banyasad O. A Visual Programming Environment for Autonomous Robots. 2000.
24. Simpson J., Jacobsen C.L., Jadud M.C. Mobile robot control. Communicating Process Architectures, 2006. P. 225.
25. Simpson J., Jacobsen C.L. (2008, September). Visual Process-Oriented Programming for Robotics. In CPA. 2008. P. 365–380.
26. Posso J.C., Sampson A.T., Simpson J., Timmis J. Process-Oriented Subsumption Architectures in Swarm Robotic Systems. In CPA. 2011. P. 303–316.

27. Diprose J.P., MacDonald, B.A., & Hosking, J.G. (2011, September). Ruru: A spatial and interactive visual programming language for novice robot programming. In Visual Languages and Human-Centric Computing (VL/HCC), 2011 IEEE Symposium on (p. 25–32). IEEE.
28. Johnston W.M., Hanna J.R., & Millar R.J. (2004). Advances in dataflow programming languages. ACM Computing Surveys (CSUR). 2004. № 36 (1). P. 1–34.
29. Kuzenkova A. Deripaska T. Bryksin Y. Litvinov V. Polyakov. QReal DSM Platform: An Environment for Creation of Specific Visual IDEs. Proceedings of 8th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2013), SCITEPRESS, 2013. P. 251–257.

References

1. Krasil'nik"yanc E.V., Varkov A.A., Tyutikov V.V. IntNCR control system software for manipulating robots // Mechatronics, automation, control. 2012. № 3. P. 31–36.
2. Standart MEK 6-1163/3. Programmable controller. Part 3: programming Languages. M.: Standartinform, 2016.
3. GOST 24836-81. Software control devices for industrial robots. Coding and programming methods. M., 1981.
4. GOST 27696-88. Industrial robot. Interfaces. Specifications. M., 1989.
5. Robot PASCAL DELTA 1-3X-USB+. Spherical coordinate system. CHelyabinsk: OOO «Ural'skie stanki», 2015. 47 p.
6. Koltygin D.S., Sedel'nikov I.A. Methodology for developing a robot control program for the Delta robot manipulator // Scientific Bulletin of NSTU. 2018. № 1 (70). P. 103–116.
7. Koltygin D.S., Sedel'nikov I.A. DELTA V. 1.0 robot manipulator control program // Trudy Bratskogo gosudarstvennogo universiteta. Ser. Estestvennye i inzhenernye nauki. 2017. V. 1. P. 100–103.
8. Brojnl' T. Embedded robotic systems: design and application of mobile robots with embedded control systems. M.: IKI, 2012. 520 p.
9. Sedel'nikov I.A., Koltygin D.S. ROBOT DELTA V. 1.0: computer program RU 2019614518. Sv. o reg. RU 2019614518; zayavka № 2019612975 ot 19.03.2019.
10. Tyutikov V.V., Krasil'nik"yanc E.V., Varkov A.A. Components of the manipulation robot software // Vestn. IGEU. Vyp. 4. 2011. P. 40–43.
11. Aho A., Ul'man Dzh. Theory of parsing, translation, and compilation. M.: Mir, 1978. 612 p.
12. Dal U., Dejkstra E., Hoor K. Structural programming. M.: Mir, 1975. 248 p.
13. Eliseev S., Svinin M., Smelyagin A. Mathematical and software support of the research manipulation systems. M.: Nauka, Sib. otd-nie, 1992. 294 p.
14. Koltygin D.S., Sedel'nikov I.A., DELTA V. 1.0 robot manipulator control program // Trudy Bratskogo gosudarstvennogo universiteta. Ser. Estestvennye i inzhenernye nauki. 2017. V. 1. P. 100–103.
15. De Luca A. Programming supervision and control architectures: tez. dokl. konf. // Robotics. Sapienza Universita DI Roma, 2014.
16. Industry 4.0. The Future of Productivity and Growth in Manufacturing Industries. Germany: BCG, 2015. P. 4.
17. Bösl D. Hello Industry 4.0: glossary // KUKA GmbH, Augsburg, 2017. P. 9–12.
18. The High-Tech Strategy. Germany: GTAI, 2014. P. 12-13.
19. Ivanov D.S. The new industrial structure is based on competencies // Intelligent manufacturing Journal. 2017. № 1. P. 65–71.
20. A new age of industrial production. The Internet of Things, Services and People. ABB, 2015, P. 7.
21. Lopota A. Trends in the development of robotic systems // Intelligent manufacturing Journal. 2016. № 2. P. 83–88.
22. Zimin G.A., Mordvinov D.A. Educational visual streaming language for robot programming // Trudy ISP RAN, 2016. V. 28, Vyp. 2. P. 45–62.
23. Banyasad O. A Visual Programming Environment for Autonomous Robots. 2000.
24. Simpson J., Jacobsen C.L., Jadud M.C. Mobile robot control. Communicating Process Architectures, 2006. P. 225.
25. Simpson J., Jacobsen C.L. (2008, September). Visual Process-Oriented Programming for Robotics. In CPA. 2008. P. 365–380.
26. Posso J.C., Sampson A.T., Simpson J., Timmis J. Process-Oriented Subsumption Architectures in Swarm Robotic Systems. In CPA. 2011. P. 303–316).
27. Diprose J.P., MacDonald, B.A., & Hosking, J.G. (2011, September). Ruru: A spatial and interactive visual programming language for novice robot programming. In Visual Languages and Human-Centric Computing (VL/HCC), 2011 IEEE Symposium on (p. 25–32). IEEE.
28. Johnston W.M., Hanna J.R., & Millar R.J. (2004). Advances in dataflow programming languages. ACM Computing Surveys (CSUR). 2004. № 36 (1). P. 1–34.
29. Kuzenkova A. Deripaska T. Bryksin Y. Litvinov V. Polyakov. QReal DSM Platform: An Environment for Creation of Specific Visual IDEs. - Proceedings of 8th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2013), SCITEPRESS, 2013. P. 251–257.