В таблице указаны баллы, полученные роботами после 100 раундов игры, в скобках указано отношение баллов, полученных построенным роботом, к общей сумме баллов.

Робот `sample.Walls` имеет более сложное поведение – двигаясь вдоль стен, робот направляет пушку в сторону поля и стреляет в цель. С помощью генетического программирования было получено решение с отношением баллов (к общей сумме) 0,62. С помощью *Q*-обучения за 500 раундов был построен робот, побеждающий `sample.Walls` с отношением полученных баллов 0,71.

Заключение

В работе рассмотрена задача построения управляющего автомата для робота-агента на примере игры «Robocode». Для решения задачи предложен метод, основанный на двухэтапном построении управляющего автомата с помощью обучения с подкреплением. Предложенный метод успешно справляется с задачей построения автомата, управляющего танком в игре «Robocode».

Сравнение результатов применения *Q*-обучения с результатами, полученными методом генетического программирования, показывает, что на обоих этапах результаты работы методов не сильно отличаются друг от друга. При наведении пушки *Q*-обучение показало лучший результат, в то время как при преследовании цели лучший результат показывает метод генетического программирования. В задаче построения автомата против робота `sample.Fire` лучший результат показало генетическое программирование, а при построении автомата против робота `sample.Walls` – *Q*-обучение.

Полученные данные позволяют утверждать, что методы обучения с подкреплением применимы к построению управляющих автоматов для роботов-танков и не уступают в эффективности генетическому программированию.

Литература

1. Поликарпова Н.И., Шалыто А.А. Автоматное программирование. – СПб: Питер, 2009 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/books/_book.pdf, своб.
2. Царев Ф.Н. Разработка метода совместного применения генетического программирования и конечных автоматов. Бакалаврская работа. – СПбГУ ИТМО, 2007 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/papers/_2010_03_03_tsarev.pdf, своб.
3. Mitchell M. An Introduction to Genetic Algorithms. – The MIT Press, 1996.
4. Бедный Ю.Д. Применение генетических алгоритмов для генерации автоматов при построении модели максимального правдоподобия и в задачах управления. Магистерская диссертация. – СПбГУ ИТМО, 2008 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/papers/bednij/, своб.
5. Соколов Д.О. Применение двухэтапного генетического программирования для построения автомата, управляющего моделью танка в игре «Robocode». Бакалаврская работа. – СПбГУ ИТМО, 2009.
6. Sutton R.S., Barto A.G. Reinforcement Learning. An Introduction. – The MIT Press, 1998.
7. Хайкин С. Нейронные сети. Полный курс. – М.: Вильямс, 2006.

Чернявский Илья Игоревич – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, chernyavsky@rain.ifmo.ru

УДК 004.4'242

АВТОМАТИЧЕСКИЙ СИНТЕЗ СИСТЕМЫ УПРАВЛЕНИЯ МОБИЛЬНЫМ РОБОТОМ ДЛЯ РЕШЕНИЯ ЗАДАЧИ «КЕГЕЛЬРИНГ»

С.А. Алексеев, А.И. Калиниченко, В.О. Клебан, А.А. Шалыто

Приводится пример автоматического синтеза системы управления мобильным роботом для решения задачи «Кегельринг». Автоматический синтез системы проводится с использованием генетического алгоритма, при помощи которого определяется структура управляющего автомата.

Ключевые слова: автоматное программирование, генетические алгоритмы, автоматический синтез систем управления.

Введение

Для построения систем управления мобильными роботами целесообразно использовать технологию автоматного программирования [1–3], в которой, в частности, предлагается строить программу как систему автоматизированных объектов управления, в которых управляющая программа представляет собой систему автоматов, взаимодействующих между собой за счет вложенности и вызываемости. Использование автоматного подхода при создании подобных систем обладает рядом достоинств, таких как возможность повышения уровня автоматизации процесса верификации, документируемость, упрощение внесения изменений и т.д.

На практике встречаются задачи, для которых известно, что они могут быть решены при помощи конечных автоматов, но эвристически построить для них автомат чрезвычайно сложно [4]. Для решения подобных задач могут быть использованы методы автоматического синтеза программ, одним из которых является генетическое программирование.

Эффективность применения генетического программирования для синтеза автоматов продемонстрирована в работах [5–8], но, к сожалению, ни в одной из них не проверялась работа синтезированных систем управления на реальных объектах.

В данной работе приводится пример автоматического синтеза системы управления роботом для задачи «Кегельринг» и его проверка на реальном мобильном роботе.

Постановка задачи

Цель робота в задаче «Кегельринг» состоит в выталкивании за пределы ринга расположенные в нем кегли за минимально возможное время. Ринг представляет собой площадку круглой формы диаметром один метр, которая ограничена темной контрастной полосой. При выполнении задания робот не должен выходить за пределы ринга. Порядок расстановки кеглей определен следующим образом:

- перед началом состязания на ринге расставляют восемь кеглей;
- робот устанавливается в центр ринга;
- методом жеребьевки из ринга убирают четыре кегли, что позволяет внести в задание элемент случайности.

Целью данной работы является автоматический синтез системы управления мобильным роботом для решения данной задачи.

Устройство робота

Робот представляет собой трехколесное шасси. В качестве движителя используются два электродвигателя постоянного тока, которые образуют дифференциальный привод, позволяющий роботу осуществлять «танковый» разворот на месте (рис. 1). На роботе установлены два типа датчиков: инфракрасный дальномер, предназначенный для детектирования кеглей, и датчик линии, предназначенный для сигнализации о том, что робот пересек ограничительную линию.

Выходными воздействиями для робота являются: вращение по часовой стрелке (z_0); вращение против часовой стрелки (z_1); движение вперед (z_2); движение назад (z_3); останов (z_4).

Входными воздействиями являются: наличие/отсутствие линии под роботом (x_0); перед роботом нет кегли (x_1); перед роботом есть кегля на большом расстоянии (x_2); перед роботом есть кегля на среднем расстоянии (x_3); перед роботом есть кегля на малом расстоянии (x_4); кегля на очень малом расстоянии (робот толкает кеглю) (x_5).

В процессе автоматического синтеза системы управления используются два робота, соответствующие по характеристикам друг другу: реальный и виртуальный.

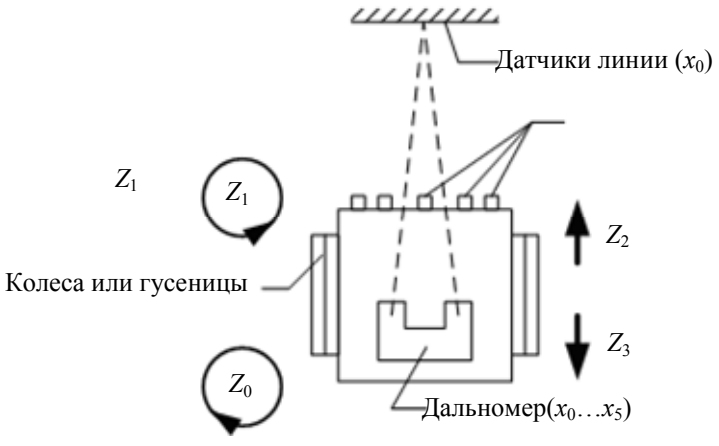

Рис. 1. Устройство мобильного робота

Схема работы генетического алгоритма

Для осуществления автоматического синтеза системы управления роботом разработан программно-аппаратный комплекс (рис. 2), который включает в себя симулятор, предназначенный для моделиро-

вания поведения реального робота, и приложение, реализующее генетический алгоритм (в дальнейшем просто генетический алгоритм).

Симулятор работает следующим образом: для каждой поданной ему на вход особи вычисляется функция приспособленности, которая используется при отборе особей в генетическом алгоритме. При работе симулятора желательно обеспечить наиболее точное соответствие виртуальной и реальной сред. Это необходимо для того чтобы синтезированная при помощи симулятора система управления оставалась работоспособной в реальной среде.

В качестве симулятора в работе использована программная среда Webots. Данная среда позволяет создавать виртуальные модели роботов, окружение для них, а также программировать физическое поведение и выполнять моделирование.

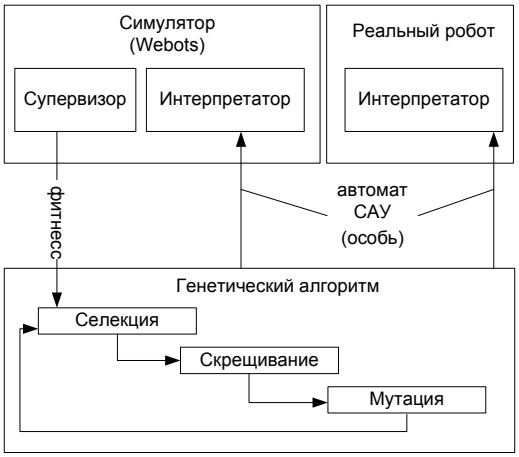

Рис. 2. Структурная схема разработанного комплекса

Среда эмуляции Webots позволяет оснастить виртуального робота моделями двигателей, сервоприводов и датчиков. Специально для решения поставленной задачи спроектирована виртуальная модель реального робота, а также виртуальный ринг с кеглями.

Виртуальная модель робота оснащена двумя двигателями, дальномером и датчиком линии, эквивалентными тем, которые установлены на реальном роботе.

Для обеспечения работы виртуальной модели на языке программирования Java был разработан интерпретатор автоматов управления (построенных при помощи генетического алгоритма).

Кроме того, был разработан контроллер-супервизор, который отслеживает перемещения модели робота и кеглей, и на основе этих данных вычисляет значение функции приспособленности.

В процессе работы генетический алгоритм обеспечивает процесс эволюции особей, которые в данном случае представлены в виде управляющих автоматов. Вначале алгоритм генерирует случайную начальную популяцию (набор случайных автоматов). Особи этой популяции подаются на вход симулятора Webots (рис. 2), который рассчитывает значение функции приспособленности для каждой из особей и предоставляет эти данные генетическому алгоритму. На следующем шаге генетический алгоритм осуществляет селекцию, скрещивание и мутацию в соответствии с полученными значениями функции приспособленности.

Любую из полученных в популяции особей, при необходимости, можно передать на вход интерпретатору реального робота для проверки функциональности особи в реальных условиях (рис. 3).

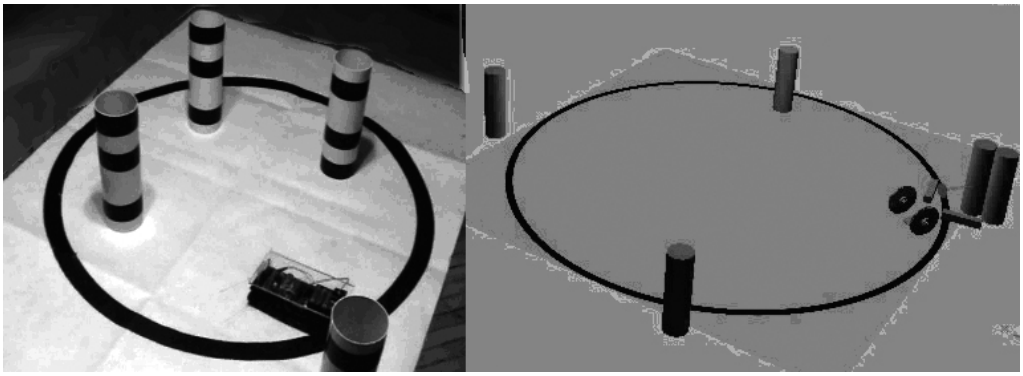

Рис. 3. Реальный робот и его компьютерная модель во время тестирования

Способ кодирования хромосом

Для применения генетического алгоритма к решению задач автоматического синтеза систем управления мобильными роботами необходима модель хромосомы. В рамках рассматриваемой задачи хромосома представляет собой закодированный специальным образом конечный автомат Мили, при помощи которого осуществляется управление роботом.

Опишем способ кодирования для конечного автомата Мили из N состояний, в каждом из которых возможно L входных воздействий. Первый байт строки – номер начального состояния (его значение меньше N). Каждое состояние кодируется следующим образом: первый байт – номер выходного воздействия (ехать вперед/назад, поворачивать), выдаваемого на исполнительные механизмы реального, либо виртуального робота, затем L байт – номера состояний, в которые автомат переходит в зависимости от входных воздействий. Таким образом, в начале байт-строки идет один байт – номер начального состояния, затем $N \times L$ байт – описание состояний.

Полученная модель хромосомы является одной из самых простых, что позволяет применять простейшие операторы скрещивания и мутации.

Операторы скрещивания и мутации

Как отмечено выше, генетический алгоритм осуществляет процесс эволюции особей, похожий на биологическую эволюцию. В ходе этого процесса применяются два основных инструмента – селекция и скрещивание. Для скрещивания особей применяется специальный оператор скрещивания. Его задача – создание особей нового поколения на основе особей предыдущего поколения.

В качестве оператора скрещивания используется одноточечный кроссовер. Генерируется случайное натуральное число C, величина которого меньше длины хромосомы. Затем обе родительские хромосомы делятся на две части, первая из которых содержит первые C генов, а вторая – оставшиеся. Две дочерние особи получаются путем составления нового генотипа на основе частей генотипа обоих родителей.

Рассмотрим пример применения операции одноточечного кроссовера к хромосомам A(0 110 201) и B(1 201 101). Длина каждой из хромосом равна семи. Допустим, что число C равно четырем. Тогда первая дочерняя хромосома будет состоять из четырех первых байт хромосомы A и трех последних байт хромосомы B. Вторая дочерняя хромосома будет состоять из четырех первых байт хромосомы B и трех последних хромосомы A. Части, которыми обмениваются конечные автоматы, выделены пунктирными линиями.

Для внесения многообразия в популяцию применяется оператор мутации, который осуществляет случайные изменения в генотипе случайных особей. При синтезе рассматриваемой системы управления применяется оператор мутации, который изменяет один случайно выбранный байт в хромосомах пяти случайных особей. Выбор именно пяти особей обусловлен тем, что с такими параметрами алгоритм показывал лучшую сходимость.

Функция приспособленности

Для эффективной работы генетического алгоритма необходимо выбрать соответствующую функцию приспособленности – отображение из множества хромосом в множество численных значений их приспособленности. Для рассматриваемой задачи эта функция зависит от того, покидал ли робот ринг, а также следующих параметров:

- число кеглей, которые в какой-либо момент времени находились за пределами ринга;
- число кеглей, которые не покинули ринг;
- расстояние, пройденное роботом;
- время, за которое робот решил задачу.

Особенностью функции приспособленности, использованной в данной задаче, является то, что способ ее вычисления зависит от текущего этапа развития робота. Поясним это. В первом поколении, когда хромосомы строятся случайным образом, робот редко может вытолкнуть хотя бы одну кеглю. Таким образом, если оставить для расчета функции приспособленности только такие параметры, как время, число кеглей и условие, что робот не должен покидать ринг, то большее значение вычисляемой функции часто будут иметь роботы, которые не предпринимают никаких действий и, как следствие, не покидают ринг.

Получается, что на первом этапе выращивания целесообразно использовать функцию приспособленности, включающую в себя «бонус» за пройденное расстояние. Тогда получат преимущество роботы, предпринимающие активные действия, например, использующие датчик линии, ограничивающей ринг, или перемещающиеся внутри ринга. Для вычисления функции приспособленности была использована процедура, приведенная в листинге.

После того, как хромосомы эволюционируют (рис. 4) и роботы научатся убирать кегли с ринга, целесообразно присваивать большее значение функции приспособленности тем роботам, которые способны решить изначально поставленную задачу: не выезжать за пределы ринга и не возвращать кегли на ринг (такое случается, если кегля «цепляется» за робота). Фактически происходит поэтапное обучение робота.

Листинг. Вычисление функции приспособленности

```
fitness = 0;       //Значение функции приспособленности
current_time = 0; //Текущее время модели
max_time = const; //Максимальное время выполнения задания
distance = 0;      //Путь, проделанный роботом
while current_time < max_time do
      if кегли_убраны and робот_не_покидал_ринг then
            fitness = max_time – current_time;
            return fitness;
      else
            fitness = fitness + distance;
            if кегля_покинула_ринг then
                  fitness = fitness + 1;
            if кегля_вернулась_в_ринг then
                  fitness = fitness – 0.5;
return fitness;
```

Рис. 4. Динамика роста функции приспособленности

Селекция

В качестве алгоритма селекции был применен отбор усечением. Эта стратегия использует отсортированное поколение, из которого выбирается наиболее приспособленная половина особей. Затем среди отобранных особей случайным образом выбираются пары, которые участвуют в скрещивании. Кроме того, применяется элитизм, при котором несколько наиболее приспособленных особей попадают в новое поколение без изменений.

Без использования этого приема поколения будут вырождаться, так как при скрещивании двух хорошо приспособленных особей имеется большая вероятность получить полностью неработоспособного потомка.

Калибровка модели и реального робота

Одной из серьезных проблем при использовании генетических алгоритмов является несоответствие между моделью задачи, используемой при синтезе системы управления, и реальным миром.

Автомат, отлично решающий поставленную задачу в эмуляторе, может полностью не справляться с ней в реальной среде. При разработке модели даже для такой простой задачи, как «Кегельринг», невозможно учесть все внешние факторы, влияющие на работу алгоритма: погрешности датчиков, особенности работы электродвигателей, материалы предметов, покрытие ринга и многое другое. Робот, выращенный на одной модели, должен демонстрировать работоспособность и на другой, немного отличающейся от исходной, это означает, что полученная стратегия управления должна быть как можно более робастной.

Рассмотрим возможные методы решения данной проблемы.

- Обучение робота на различных моделях. Недостатком данного метода является увеличение пространства поиска, что негативно сказывается на времени работы алгоритма.
- Обучение робота с зашумленными входами. Данный метод не увеличивает пространство поиска, но при этом адаптирует робота к выходу в реальный мир.
- Изменение функции приспособленности в сторону меньшей зависимости от параметров среды.

Наряду с перечисленными методами авторами предлагается использовать метод, суть которого заключается в изменении типа выходных воздействий синтезируемой системы управления. Поясним на примере.

Сначала при попытках обучить систему управления использовалось выходное воздействие «поворот», которое предполагало поворот робота на угол в 10°. В результате полученные автоматы использовали данное свойство, выстраивая выходные воздействия в цепочку и получая при этом поворот на необходимый угол. К сожалению, после переноса полученного автомата на реального робота оказалось, что система управления неработоспособна, так как осуществляемый по таймеру поворот происходит на разный угол в зависимости от условий. Избежать данного эффекта удалось, изменив тип выхода на «постоянное вращение». Вместо поворота на определенный угол робот начинал вращаться в заданном направлении. Это позволило добиться большего сходства между реальным роботом и его моделью.

Изменение типа выхода также положительно сказалось и на сходимости алгоритма.

Заключение

В данной работе приводится пример автоматического синтеза системы управления мобильным роботом для решения задачи «Кегельринг». Одним из условий успешного решения задачи служила возможность переноса синтезированной системы управления на реального робота без внесения каких-либо изменений.

Поставленные цели были достигнуты, и реальный робот выполнил поставленную задачу [8].

В ходе выполнения работы обнаружен прием, названный «изменение типа выхода», позволяющий добиться более устойчивой работы синтезированного автомата без увеличения пространства поиска.

Исследование выполнено по Федеральной целевой программе «Научные и научно-педагогические кадры инновационной России на 2009–2013 годы» в рамках государственного контракта П2236 от 11 ноября 2009 года.

Литература

1. Шалыто А.А. Технология автоматного программирования // Труды Всероссийской научной конференции «Методы и средства обработки информации». – М.: МГУ, 2003 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/works/tech_aut_prog/, своб
2. Клебан В.О., Шалыто А.А. Использование автоматного программирования для построения многоуровневых систем управления мобильными роботами // Сборник тезисов 19 Всероссийской научно-технической конференции «Экстремальная робототехника». – СПб: ЦНИИ РТК, 2008. – С. 85–87.
3. Клебан В.О., Шалыто А.А., Парфенов В.Г. Построение системы автоматического управления мобильным роботом на основе автоматного подхода // Научно-технический вестник СПбГУ ИТМО. – 2008. – № 53.
4. Царев Ф.Н., Шалыто А.А. О построении автоматов с минимальным числом состояний для задачи об «Умном муравье» // Сборник докладов X международной конференции по мягким вычислениям и измерениям. – СПбГЭТУ «ЛЭТИ», 2007. – Т. 2. – С. 88–91 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/download/ant_ga_min_number_of_state.pdf, своб.
5. Данилов В.Р. Технология генетического программирования для генерации автоматов управления системами со сложным поведением. Бакалаврская работа. – СПбГУ ИТМО, 2007 [Электронный ресурс]. – Режим доступа: http://is.ifmo.ru/papers/danilov_bachelor/, своб.
6. Поликарпова Н.И., Точилин В.Н. Применение генетического программирования для реализации систем со сложным поведением // Научно-технический вестник СПбГУ ИТМО. – 2007. – № 39.
7. Chambers L. Practical Handbook of Genetic Algorithms. Complex Coding Systems. – CRC Press, 1999. – V. III.
8. Видеоматериалы [Электронный ресурс]. – Режим доступа: http://blog.savethebest.ru/?p=747, своб

Алексеев Сергей Андреевич – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, alex.itmo@gmail.com

Калиниченко Александр Игоревич – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, студент, ittrium@gmail.com

Клебан Виталий Олегович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, аспирант, vk.developer@gmail.com

Шалыто Анатолий Абрамович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, доктор технических наук, профессор, зав. кафедрой, shalyto@mail.ifmo.ru