ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Факультет Информационных технологий и программирования

Направление (специальность) Прикладная математика и информатика

Квалификация (степень) Бакалавр

Специализация

Кафедра Компьютерных технологий Группа 4538

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

Метод построения управляющих конечных автоматов с учетом непрерывных выходных воздействий на основе обучающих примеров

Автор квалификационной работы Бужинский И.П.

Руководитель Царев Ф.Н.

Консультанты:

а) По экономике и организации производства

б) По безопасности жизнедеятельности и экологии

в)

К защите допустить

Зав. кафедрой Васильев В.Н.

“___” июня 2013 г.

Санкт-Петербург, 2013 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ **5**

ГЛАВА 1. УПРАВЛЕНИЕ ОБЪЕКТАМИ СО СЛОЖНЫМ ПОВЕДЕНИЕМ **7**

1.1. Теория автоматического управления 7

1.2. Управление, основанное на поведениях 8

1.3. Иерархические системы управления 9

1.4. Методы машинного обучения 11

1.5. Автоматизированное построение управляющих автоматов 14

1.6. Постановка задачи 17

1.6.1. Процесс управления *18*

1.6.2. Тесты *19*

Выводы по главе 1 20

ГЛАВА 2. АВТОМАТЫ С НЕПРЕРЫВНЫМИ ВОЗДЕЙСТВИЯМИ **21**

2.1. Ранее предложенный способ представления 21

2.2. Предлагаемый способ представления 22

Выводы по главе 2 26

ГЛАВА 3. МЕТОД ПОСТРОЕНИЯ АВТОМАТОВ **27**

3.1. Функции приспособленности 28

3.2. Расстановка выходных воздействий 30

3.3. Поисковая оптимизация 33

3.3.1. Генетический алгоритм *34*

3.3.1. Муравьиный алгоритм *34*

Выводы по главе 3 36

ГЛАВА 4. ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ **37**

4.1. Управление моделью самолета 37
4.1.1. Сравнение производительности генетического и муравьиного алгоритмов 39
4.1.2. Оценка эффективности использования масок значимости вещественных переменных 40
4.1.3. Сравнение метода с прототипом 42
4.2. Управление моделью вертолета 47
Выводы по главе 4 49
ЗАКЛЮЧЕНИЕ 50
СПИСОК ЛИТЕРАТУРЫ 51
ПРИЛОЖЕНИЕ А. ПРИМЕР ПОСТРОЕННОГО АВТОМАТА 54

ВВЕДЕНИЕ

Автоматное программирование – парадигма, в рамках которой программы проектируются в виде систем автоматизированных объектов управления [1]. Автоматизированный объект управления состоит из объекта управления и системы управления, которая может представлять собой один или несколько управляющих конечных автоматов. Автомат взаимодействует с объектом управления по тактам, на каждом из которых автомат получает событие, меняет свое текущее состояние и генерирует выходное воздействие. Формально, управляющим конечным автоматом называется шестерка $(S, s_0, E, A, \delta, \lambda)$, где S – конечное множество состояний, $s_0 \in S$ – начальное состояние, E – множество входных событий, A – множество выходных воздействий, $\delta: S \times E \to S$ – функция переходов, λ – функция выходов ($\lambda: S \times E \to A$ для автоматов Мили и $\lambda: S \to A$ для автоматов Мура).

Автоматный подход позволяет задать формальное и понятное описание поведения системы управления и предоставляет возможность верификации программ [2]. Одна из областей, для которых целесообразно применение автоматного программирования – управление *объектами со сложным поведением*, то есть объектами, которые могут вырабатывать различные выходные воздействия в ответ на одинаковые входные воздействия. Для многих объектов со сложным поведением ручное построение автоматов затруднительно, поэтому разумно автоматизировать этот процесс. Такую возможность предоставляет поисковая оптимизация, в частности – генетическое программирование [3, 4].

Можно выделить два подхода, применяемых для разработки систем управления. Подход, основанный на *моделировании*, предлагает оперировать с математической моделью объекта управления и применяется в автоматической теории управления [5]. С другой стороны, системы управления можно строить на основе *примеров их поведения* – в частности, такой подход используется в машинном обучении [6]. Оба подхода могут комбинироваться, как, например, в [7, 8]. В случае автоматизированного построения конечных автоматов моделирование предполагает симуляцию поведения автомата в некоторой среде, что при сложности модели объекта управления может занимать продолжительное

время. Использование обучающих примеров, или тестов, приводит к меньшим временным затратам.

Ранее был предложен метод [9], позволяющий строить по тестам автоматы, выходные воздействия которых могут быть не только дискретными, но и непрерывными (вещественными). Время построения автоматов изначально составляло несколько часов, но было позже сокращено [10] до нескольких десятков минут. Однако качество автоматов, построенных с помощью этого метода, не всегда было достаточным: не все построенные автоматы справлялись с поставленной в работах [9, 10] задачей – выполнением фигуры пилотажа при управлении беспилотным самолетом. Кроме того, были обнаружены элементы полета, построить автоматы для выполнения которых с помощью метода, предложенного в [9], не удалось.

Цель настоящей работы – разработка метода построения автоматов, устраняющего недостатки метода [9]. Во-первых, на элементах полета, к которым метод [9] применим, ожидается получить возможность строить автоматы с поведением, лучше соответствующим тестам. Во-вторых, планируется расширить область применения метода [9] на новые элементы полета.

Ключевая идея, которая позволит достигнуть цели настоящей работы – новый способ представления автоматов, выходные воздействия которых могут быть непрерывными. Автоматы будут использовать для формирования выходных воздействий произвольные вещественные переменные, вычисляемые по входным воздействиям автомата. Кроме того, рассматривается модификация функции приспособленности, больше подходящая для построения автоматов, которые должны мгновенно реагировать на изменения входных воздействий. Для поиска автоматов используются генетический и муравьиный алгоритмы.

ГЛАВА 1. УПРАВЛЕНИЕ ОБЪЕКТАМИ СО СЛОЖНЫМ ПОВЕДЕНИЕМ

Напомним, что под *объектами со сложным поведением* понимаются объекты, которые могут демонстрировать различное поведение при одинаковых входных воздействиях. Примерами таких объектов являются роботы и беспилотные летательные аппараты (БПЛА). В настоящей главе приведен краткий обзор, затрагивающий различные методы построения систем управления, в том числе предлагающие использовать управляющие конечные автоматы. В конце главы описана решаемая в настоящей работе задача.

1.1. Теория автоматического управления

Теория автоматического управления [5] предлагает множество подходов для создания и анализа систем управления. Ключевым элементом теории управления является *математическая модель* управляемой системы. Наиболее простым и законченным является математический аппарат для линейных моделей. Существует два основных способа описания линейных систем.

1.1.1. Модели пространства состояний

Для объекта управления задается *вектор состояний* x – набор переменных, полный в том смысле, что при известном значении состояния $x(t)$ и при заданном законе управления $u(t)$ выходное воздействие системы $y(t')$ может быть вычислено для всех $t' > t$. В рамках моделей пространства состояний дифференциальные уравнения, описывающие поведение линейной системы, выглядят следующим образом:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = Ax(t) + Bu(t);$$

$$y(t) = Cx(t) + Du(t),$$

где *A*, *B*, *C*, *D* – матрицы, согласующиеся по размерности с векторами x и y.

1.1.2. Модели входа-выхода

В моделях входа-выхода рассматриваются системы линейных дифференциальных уравнений произвольного порядка, непосредственно связывающие вход системы $u(t)$ с ее выходом $y(t)$:

$$\frac{\mathrm{d}^n y(t)}{\mathrm{d}t^n} + a_{n-1}\frac{\mathrm{d}^{n-1} y(t)}{\mathrm{d}t^{n-1}} + \cdots + a_0 y(t) = b_{n-1}\frac{\mathrm{d}^{n-1} u(t)}{\mathrm{d}t^{n-1}} + \cdots + b_0 u(t).$$

Оба способа описания обобщаются на случай нелинейных моделей, но задача анализа нелинейных моделей значительно сложнее. В некоторых случаях нелинейную модель объекта управления можно *линеаризовать* в пределах некоторых диапазонов функционирования. Для управления нелинейными объектами также применяются такие техники, как *адаптивное управление* [11], рассматривающее системы, меняющие свои параметры в процессе управления, а также *робастное управление* [11], позволяющее создавать регуляторы, обеспечивающие управление даже при неточности или неизвестности модели объекта.

1.2. Управление, основанное на поведениях

Управление, основанное на поведениях (*behavior-based control*) [12] – одна из методологий, нашедшая широкое применение в робототехнике. В рамках этой методологии управление объекта является результатом работы нескольких режимов – *поведений*, решающих конкретные задачи, стоящие перед объектом управления. Каждое поведение может быть разработано, например, на основе методов теории управления. Достоинством управления, основанного на поведениях, является модульность: система управления может быть построена из отдельных «блоков».

Приведем пример. Рассмотрим задачу управления роботом, который должен достичь заданной цели, не столкнувшись с препятствиями. В этом случае система управления может включать в себя два поведения: для приближения к цели и для удаления от препятствий. Поведения могут работать как параллельно (управляющее воздействие системы является функцией двух аргументов – воздействий различных

поведений), так и попеременно, переключаясь при выполнении некоторых условий. Во втором случае, упрощающем анализ поведения всей системы управления, удобно строить ее как *гибридный автомат* (*hybrid automaton*) [13], представляющий собой конечный автомат, состояния которого соответствуют различным непрерывным режимам управления системой. Пример гибридного автомата приведен на рис. 1.

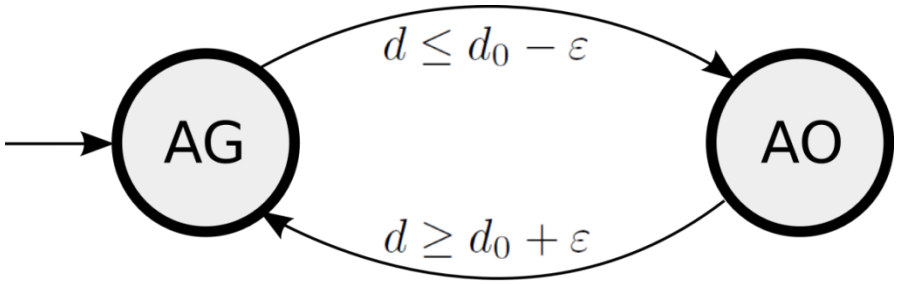

Рис. 1. Пример гибридного автомата для системы управления роботом, решающей задачу приближения к цели (AG – approach goal) при отсутствии столкновений с препятствиями (AO – avoid obstacles). d – текущее расстояние до препятствия, d_0 – пороговое расстояние для переключения режимов.

Одним из недостатков гибридных автоматов является «шероховатость» управления в моменты перехода между режимами. Если рассмотреть автомат на рис. 1, то этот недостаток виден, когда расстояние до препятствия близко к d_0. Решение данной проблемы предложено в работе [14] и заключается во введении дополнительных состояний гибридных автоматов, позволяющим роботам «скользить» вдоль границы в пространстве состояний, рядом с которой должно происходить переключение режимов. Законы управления для таких режимов могут быть автоматически получены из законов управления режимов, которые разделяет граница.

1.3. Иерархические системы управления

Иерархические системы управления состоят из режимов управления, находящихся на различных уровнях абстракции. В то время как нижний уровень системы управления может выполнять простые функции (например, прямолинейное приближение к текущей цели), верхние уровни решают более сложные задачи и

могут осуществлять планирование. Иерархические системы управления можно рассматривать как частный случай систем управления, основанных на поведениях.

Для примера опишем систему управления (рис. 2), используемую в работе [15], которая посвящена решению задачи посадки беспилотного вертолета. Используемый в этой работе вертолет оснащен камерой, направленной перпендикулярно земле, а также системой *GPS*-навигации, гиростабилизатором и сонаром.

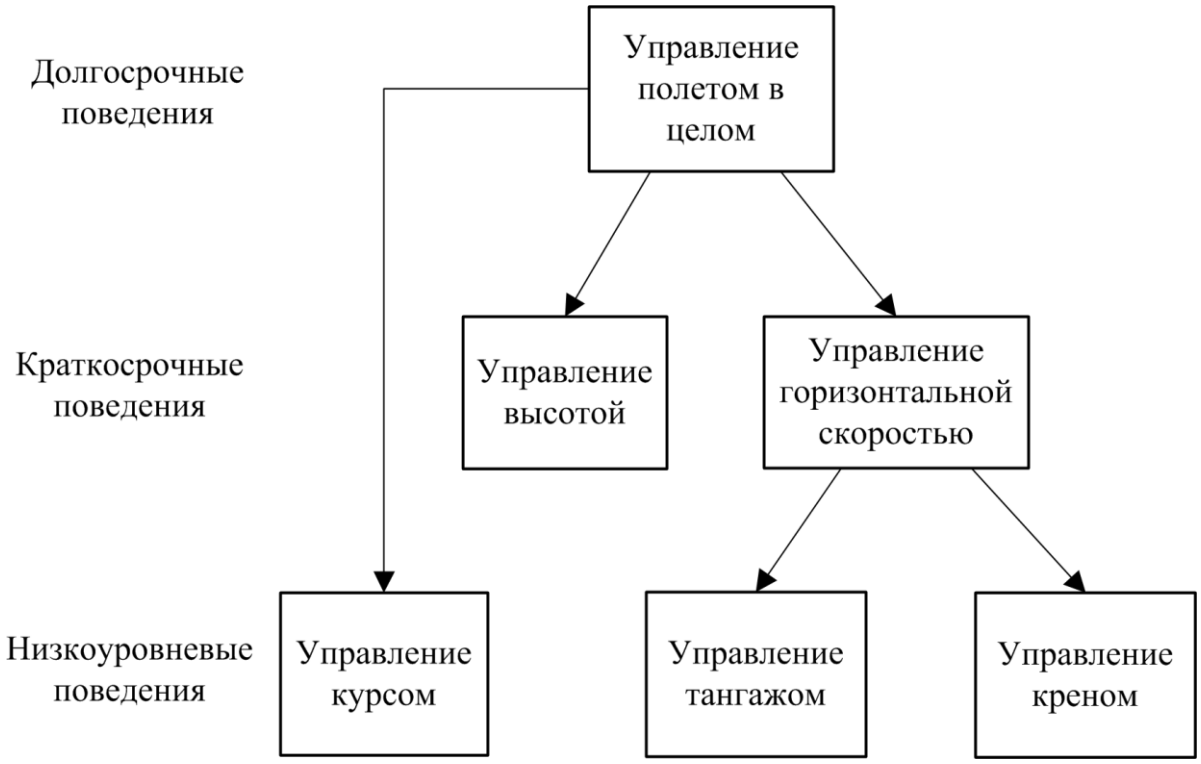

Рис. 2. Упрощенная схема иерархической системы управления беспилотным вертолетом из работы [15].

Низкоуровневые поведения отвечают за управление углами крена, тангажа и курса. В их основе лежат пропорциональные регуляторы. Углы, которые эти поведения должны поддерживать, являются результатом работы поведений более высокого уровня. Поведения, решающие кратковременные задачи – в данном случае управление управления горизонтальной скоростью и высотой – находятся на следующем уровне абстракции. Система управления горизонтальной скоростью получает целевую скорость и отправляет необходимые для ее достижения углы крена и тангажа поведениям нижнего уровня. Система управления высотой

разделена на три режима: режимы управления зависанием, скоростью и сонарный режим управления. Последние два режима используются во время посадки. Наконец, верхний уровень системы управления отвечает за выполнение задачи полета в целом и передает параметры полета поведениям более низкого уровня.

Описанная система управления осуществляла посадку беспилотного вертолета в различных погодных условиях. Она оказалась способной выполнить задачу, даже если посадочная площадка перемещалась (при этом посадка происходила после остановки площадки).

1.4. Методы машинного обучения

Машинное обучение [6] – раздел слабого искусственного интеллекта, изучающий алгоритмы, способные к самообучению. В настоящем разделе будет в основном рассматриваться частная задача машинного обучения – задача обучения с учителем. В этой задаче алгоритму доступны обучающие примеры, или тесты, являющиеся парами из входного и выходного воздействий. Выходные воздействия могут быть дискретными – это задача *классификации* – и вещественными – это задача *регрессии*. Алгоритм должен «обобщить» поведение, содержащееся в тестах, то есть с большой вероятностью давать правильные ответы (или хорошие приближения к ним для задачи регрессии) на неизвестные во время обучения входные воздействия.

Одной из моделей, нашедшей широкое применение в машинном обучении, является искусственная нейронная сеть [6]. Частным случаем нейронной сети является многослойный перцептрон (рис. 3), который можно обучать с помощью алгоритма обратного распространения ошибки [6].

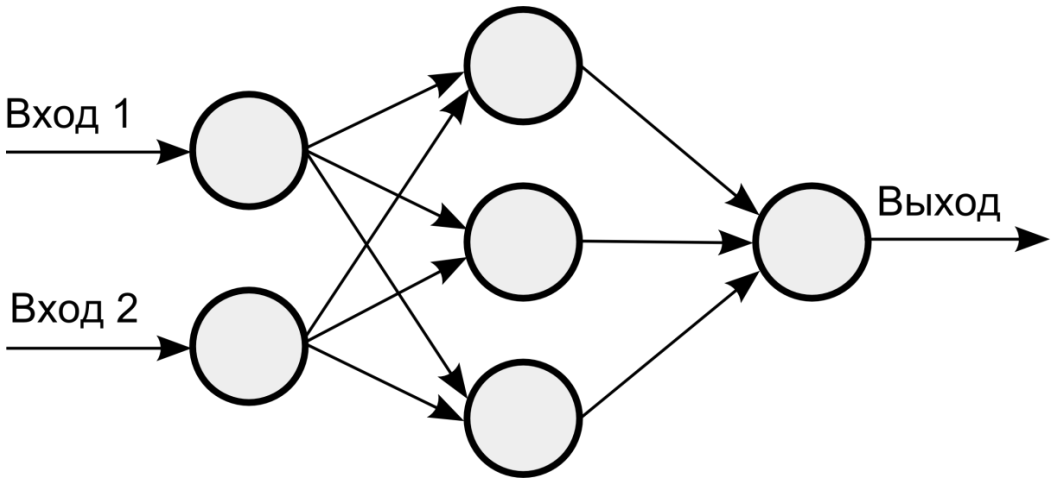

Рис. 3. Многослойный перцептрон. Слева направо расположены входной, скрытый и выходной слои искусственных нейронов

В достаточно ранней работе [16] рассматривается метод обучения многослойного перцептрона, управляющего автономным автомобилем. Предполагается, что автомобиль имеет постоянную скорость и имеет один управляющий параметр – поворот руля. Для решения проблемы используется искусственная нейронная сеть, обучаемая алгоритмом обратного распространения ошибки.

Входными данными для нейронной сети в работе [16] являются снимки дороги, уменьшенные до разрешения 30×32. В качестве входного слоя используется набор из 30×32 нейронов, соответствующих пикселам снимка. Нейронная сеть имеет один скрытый слой, состоящий из пяти нейронов. Выходным воздействием нейронной сети является поворот руля. На выходном слое нейронной сети находятся 30 нейронов, соответствующих его поворотам от крайнего левого до крайнего правого. В качестве выходного воздействия берется центр масс «горба» вокруг выходного нейрона с максимальным уровнем активации.

Обучение нейронной сети предлагается производить непосредственно во время вождения. Во время обучения поддерживается множество из 200 тестов – снимков дороги. На каждой итерации алгоритма обратного распространения ошибки с камеры приходит новый снимок дороги, который вместе с текущим положением руля образует новый тест. Кроме того, дополнительные 14 тестов получаются за счет сдвигов изображения и текущего поворота руля. Использование сдвигов

изображения улучшает качество управления, получающегося в результате обучения. Часть старых тестов заменяется новыми: выбираются десять тестов, на которых ошибка нейронной сети минимальна, и пять случайных тестов.

Многослойный перцептрон – далеко не единственная модель нейронной сети. Значительно отличается от него, например, нейронная сеть обобщенной регрессии (general regression neural network). Такая нейронная сеть была использована в работе [17] для обучения системы управления беспилотным вертолетом на основе примеров управления человека. Сеть хранит в памяти пары ($\boldsymbol{X}$, $\boldsymbol{Y}$), где $\boldsymbol{X}$ – входное воздействие, $\boldsymbol{Y}$ – соответствующее ему выходное воздействие. $\boldsymbol{X}$ и $\boldsymbol{Y}$ могут быть как скалярами, так и векторами. Задача нейронной сети обобщенной регрессии заключается в предсказании наиболее вероятного выходного воздействия $\boldsymbol{Y}$ для заданного входного воздействия $\boldsymbol{X}$ на основе имеющейся информации.

В университете Стэнфорда также проводились исследования [7, 8], посвященные автономному управлению вертолетом. Рассматривалась задача выполнения вертолетом воздушных маневров при наличии записей полетов, осуществленных человеком. Был предложен метод машинного обучения, выделяющий «скрытую» траекторию, которую пилот пытался продемонстрировать, на основе совершенных им полетов, а также генерирующий регулятор, осуществляющий оптимальное управление вертолетом. Подход, предложенный в [7, 8], предлагает рассматривать записанные траектории полетов эксперта как зашумленные наблюдения «скрытой» траектории.

Состояние вертолета в [7, 8] представляется его координатами, ориентацией, скоростью и угловыми скоростями. Каждая траектория экспертных данных является набором векторов вида $y_j^k = \left[s_j^k, u_j^k\right]$, где k – номер траектории, j – момент времени, s_j^k – вектор состояния, u_j^k – вектор значений управляющих параметров вертолета (используются четыре таких параметра). «Скрытая» траектория рассматривается как результат зашумления имеющихся траекторий с учетом возможной переиндексации по времени. Используемые модели основаны на нормальном вероятностном распределении. Для оценки их параметров также используются данные полетов

эксперта (не обязательно описывающие воздушные маневры, которые вертолет должен демонстрировать).

Как уже упоминалось, предлагаемый метод решает две задачи:

1. нахождение «скрытой» траектории;
2. нахождение регулятора, обеспечивающего полет вертолета вдоль найденной траектории.

Первая задача решается поочередным применением *EM*-алгоритма [18] как способа оценки параметров скрытой Марковской модели и метода динамического программирования. Решение второй задачи основано на методах линейно-квадратичной оптимизации [19], являющейся частным случаем задачи обучения с подкреплением. Для нахождения оптимального регулятора также используется динамическое программирование.

1.5. Автоматизированное построение управляющих автоматов

Как уже упоминалось, одним из возможных подходов при создании систем управления является автоматизированное построение управляющих конечных автоматов. Системы управления, основанные на автоматном программировании, можно разрабатывать и вручную. Такой подход, например, был использован в работе [20]. Применение этого подхода может быть связано с большими временными затратами, поэтому привлекательной является возможность автоматизации процесса проектирования управляющих систем. Примером использования генетического алгоритма [3, 4] для решения задачи управления сравнительно простым объектом – муравьем на разделенном на клетки торе – служит работа [21].

В работе [22] с помощью генетического алгоритма производится поиск автомата управления моделью самолета. Управление осуществляется на высоком уровне абстракции: автомат может использовать встроенный в самолет автопилот. Для представления функций перехода и выхода используются *сокращенные таблицы*, предполагающие различные их значения только при разных значениях *значимых* предикатов (предикаты описывают состояние самолета). Каждому

состоянию автомата сопоставлена булева маска, определяющая значимость предикатов и являющаяся частью особи оптимизационного алгоритма. Пример маски значимости предикатов приведен в табл. 1, а пример сокращенной таблицы для такой маски – в табл. 2.

Табл. 1. Пример маски значимости предикатов (число значимых предикатов равно двум)

p_1	$\boldsymbol{p_2}$	$\boldsymbol{p_3}$	p_4	p_5	p_6
0	**1**	**1**	0	0	0

Табл. 2. Пример сокращенной таблицы. δ – значение функции переходов, z_1 и z_2 – выходные воздействия на переходах.

p_2	p_3	δ	z_1	z_2
0	0	2	0	1
0	1	1	0	1
1	0	3	1	1
1	1	4	0	0

Функция приспособленности (ФП), использованная в работе [22], была основана на моделировании поведения автомата в авиасимуляторе, что делала процесс ее вычисления длительным (около пяти минут), из-за чего время работы генетического алгоритма достигло месяца. Тем не менее, результатом работы алгоритма явился автомат, имеющий десять состояний и проводящий самолет по заданному маршруту.

Другая ФП, основанная на тестах, используется в работе [9]. В этой работе также рассматривается задача построения автомата, управляющего моделью беспилотного самолета, но задача автомата теперь состоит в выполнении самолетом некоторой фигуры пилотажа – например, мертвой петли. Самолет имеет органы управления двух типов – дискретные (стартер, магнето), положения которых задаются целыми числами из конечного множества, и непрерывные (руль направления, руль высоты, элероны), положение которых задается числами из некоторого отрезка вещественной оси.

Предложенная в работе [9] ФП является мерой сходства выходных воздействий, записанных в тестах, и выходных воздействий, выработанных

автоматом, запущенным на последовательностях входных воздействий из тестов. Для построения автоматов, как и в работе [22], используется генетический алгоритм. Особью алгоритма является *каркас* автомата – автомат, выходные воздействия которого не определены. Выходные воздействия на каркасах автоматов расставляются автоматически с помощью алгоритма [9], максимизирующего ФП на заданном каркасе. Похожая идея расстановки выходных воздействий использовалась в более ранней работе [23], посвященной построению автоматов с дискретными воздействиями по тестам. Схематично процесс расстановки выходных воздействий показан на рис. 4.

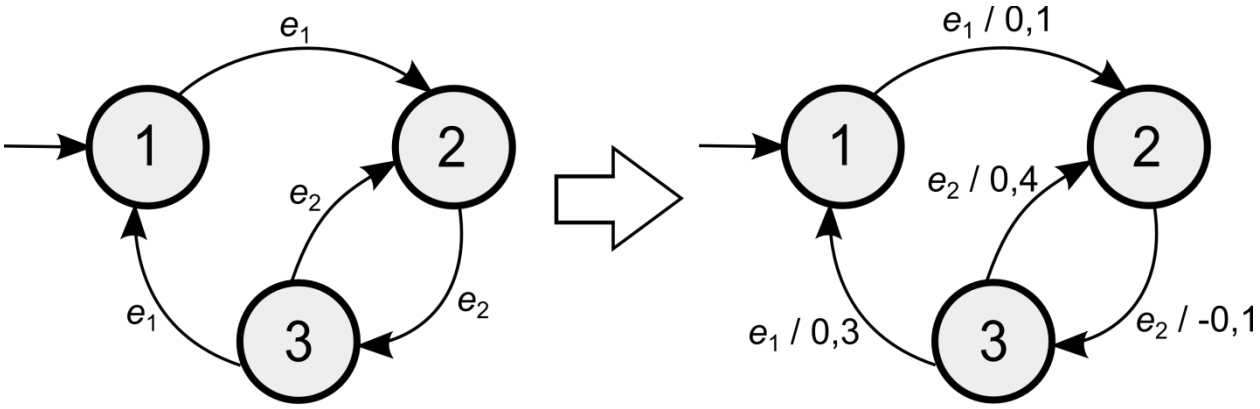

Рис. 4. Процесс расстановки выходных воздействий на каркасе автомата (случай одного непрерывного выходного воздействия, e_1, e_2 – события автомата).

Подход, предложенный в работе [9], **будет усовершенствован в настоящей работе**. Отметим, что в случае необходимости построения автомата, решающего более общую задачу (например, управление самолетом на протяжении всего рейса), систему управления можно сделать иерархической. В этом случае вводится автомат *верхнего уровня*, каждое состояние которого соответствует некоторому режиму работы объекта управления. Для каждого режима полета автомат можно построить на основе подхода из работы [9].

Метод автоматизированного построения автоматов верхнего уровня был разработан в работе [24]. Входными данными для этого метода являются тесты, описывающие полный цикл управления объектом, а также автоматы для каждого из режимов управления (в случае модели самолета это взлет, приземление, посадка и т. д.). Каждый тест разделен на области, соответствующие разным режимам управления. Метод решает три задачи:

1. определение используемых режимов в каждом тесте;
2. сопоставление областей теста и состояний автомата верхнего уровня;
3. определение переходов автомата верхнего уровня.

Первая задача решается путем выбора для каждой области теста того автомата нижнего уровня, который ей больше всего соответствует. Выбирается автомат с минимальным значением *функции расстояния*, которая вычисляется посредством сравнения заранее записанного поведения автомата с областью теста. Вторая задача в работе [24] решается тривиально, поскольку предполагается существование взаимно однозначное соответствие между состояниями автомата верхнего уровня и режимами управления. Переходы автомата верхнего уровня, формируемые при решении третьей задачи, определяются сменами режимов в тестах. Условия переходов должны обеспечивать корректную смену режимов и ищутся в форме $[\neg]p_1 \wedge \ldots \wedge [\neg]p_k$, где $p_1, \ldots, p_k$ – предикаты, а квадратные скобки означают необязательность их содержимого.

Построение автомата верхнего уровня занимало порядка двух минут Разработанный в работе [24] метод позволил, как и в работе [22], создать автомат, управляющий самолетом в течение всего рейса, но при существенно меньших временных затратах.

1.6. Постановка задачи

Теперь формально опишем решаемую в настоящей работе задачу. Задан набор тестов, на основе которых необходимо сгенерировать автомат, способный управлять объектом со сложным поведением. У объекта есть набор управляющих параметров, значения которых автомат может изменять. В настоящей работе будем считать, что все управляющие параметры характеризуются числами из некоторых отрезков вещественной оси. Тесты являются примерами желаемого поведения объекта управления и могут быть записаны человеком.

В качестве объекта управления в настоящей работе будет рассматриваться модель летательного аппарата – самолета (как и в работах [9, 22]) или вертолета. Желаемым поведением этих объектов является выполнение некоторого элемента

полета или некоторой фигуры пилотажа. Для моделирования используется авиасимулятор, необходимыми требованиями к которому состоят в возможности записи во время полета параметров летательного аппарата (скорости, угла крена и т. п.) и положений его органов управления, которые являются управляющими параметрами. В качестве такого симулятора выбран *FlightGear* [25].

1.6.1. Процесс управления

Опишем, как происходит взаимодействие автомата и объекта управления. *Кортежем входных воздействий* автомата назовем упорядоченный набор из P вещественных чисел, описывающих состояние объекта управления. Применительно к задаче управления моделью летательного аппарата кортеж входных воздействий состоит из параметров полета – высоты, скорости, углов курса, тангажа, крена и т. д.

Объект управления обладает *управляющими параметрами* – дискретными, то есть имеющими конечную область значений, и непрерывными (вещественными) – параметрами, значения которых ограничены некоторыми отрезками вещественной оси. В терминах летательного аппарата управляющие параметры соответствуют положениям его органов управления, которые также могут быть дискретными (стартер, который может быть включен или выключен) или непрерывными (положение руля высоты, варьирующееся от крайнего левого до крайнего правого, может быть задано числами из отрезка $[-1,1]$).

Далее будем считать, что все управляющие параметры непрерывны. Обозначим нижнюю и верхнюю границы отрезка возможных значений j-го управляющего параметра как $c_j^{\min}$ и $c_j^{\max}$ соответственно ($j = 1..C$, где C – число управляющих параметров).

Кортежем выходных воздействий автомата назовем упорядоченный набор $(o_1, \dots, o_C)$ из C вещественных чисел, такой, что $c_j^{\min} \le o_j \le c_j^{\max}$ $(j = 1..C)$. Кортеж выходных воздействий является «снимком» значений всех управляющих параметров в некоторый момент времени. Множество выходных воздействий автомата будем считать образованным всеми возможными кортежами выходных воздействий:

$$A = \left[c_1^{\min}, c_1^{\max}\right] \times \ldots \times \left[c_C^{\min}, c_C^{\max}\right].$$

Рассматриваемые автоматы являются синхронными – их такты происходят через равные промежутки времени (в настоящей работе – 0,1 с). По кортежам входных воздействий в начале каждого такта вычисляются значения *переменных* автомата (о переменных будет сказано подробней в главе 2). Значения переменных подаются на вход автомату, который определяет свое следующее состояние и выходное воздействие – изменение управляющих параметров объекта управления. Схема взаимодействия автомата и объекта управления приведена на рис. 5.

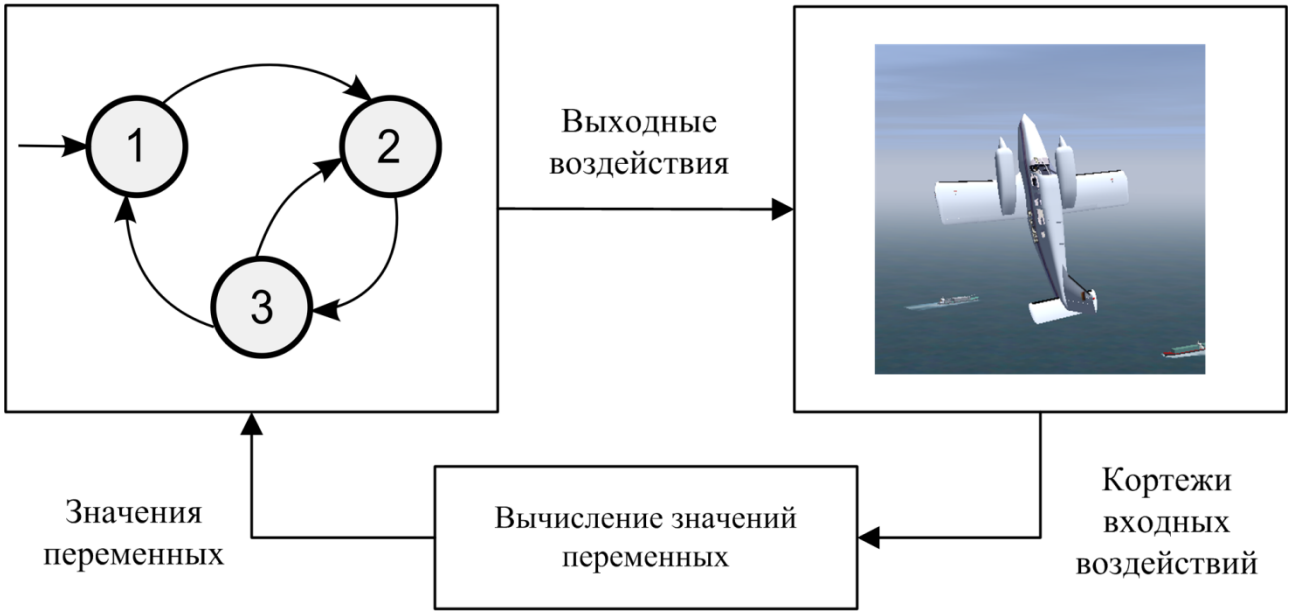

Рис. 5. Процесс взаимодействия автомата и объекта управления

1.6.2. Тесты

Формализуем понятие теста, или обучающего примера. Обозначим число моментов времени, записанное в -м тесте как $\mathrm{len}[i]$ ($i = 1..N$, где N – число обучающих примеров). Далее это число будем называть *длиной* -го теста. Каждый тест состоит из двух частей. Первая из них – последовательность кортежей входных воздействий $\mathrm{in}[i]$, состоящая из вещественных чисел $\mathrm{in}[i][t][j]$, где $t = 1..\mathrm{len}[i]$ – момент дискретизированного времени, а $j = 1..P$ – j-е входное воздействие в кортеже. Вторая часть – последовательность кортежей выходных воздействий $\mathrm{out}[i]$, состоящая из вещественных чисел $\mathrm{out}[i][t][j]$: здесь t снова определяет момент времени, а $j = 1..C$ – это номер управляющего параметра. Разности между

соседними моментами времени, записанными в тесте, равны интервалу между тактами автомата. Пример теста приведен в табл. 3.

Табл. 3. Пример теста ($P = 4, C = 3, \text{len}[i] = 235$)

Значения	**Описание**	***t* = 1**	**…**	***t* = 10**	**…**	***t* = 235**
$\text{in}[i][t][1]$	Угол тангажа (°)	3,078	…	3,544	…	2,412
$\text{in}[i][t][2]$	Угол крена (°)	−0,076	…	0,351	…	1,759
$\text{in}[i][t][3]$	Угол курса (°)	198,03	…	198,11	…	205,64
$\text{in}[i][t][4]$	Скорость (узлы)	251,42	…	252,29	…	289,40
$\text{out}[i][t][1]$	Положение элеронов (число от −1 до 1)	0,000	…	0,032	…	−0,003
$\text{out}[i][t][2]$	Положение руля направления (число от −1 до 1)	0,000	…	0,016	…	−0,001
$\text{out}[i][t][3]$	Положение руля высоты (число от −1 до 1)	−0,035	…	−0,039	…	−0,011

Выводы по главе 1

1. Рассмотрены основные подходы к решению задачи управления объектом со сложным поведением, в том числе основанные на использовании конечных автоматов.
2. Рассмотрен также подход, который будет усовершенствован в настоящей работе.
3. Сформулирована задача, решаемая в настоящей работе.

ГЛАВА 2. АВТОМАТЫ С НЕПРЕРЫВНЫМИ ВОЗДЕЙСТВИЯМИ

Использование входных и выходных воздействий, заданных вещественными числами, требует конкретизации описанной во введении классической модели управляющего автомата. В настоящей главе будут описаны два способа представления автоматов, способных интерпретировать вещественные входные воздействия и управлять вещественными параметрами – один из способов был предложен ранее в [9], другой способ предлагается в настоящей работе.

Для начала рассмотрим понятие *переменной* автомата. Переменная – это величина, значение которой на каждом такте работы автомата вычисляется по кортежам входных воздействий. В общем случае значение переменной может зависеть и от кортежей входных воздействий, выработанных объектом управления на предыдущих тактах (можно считать, что вычислитель значений переменных, приведенный на рис. 5, обладает памятью). Переменные бывают булевыми – это *предикаты* – или вещественными. Примерами предикатов являются утверждения «вертикальная скорость положительна» или «угол тангажа больше 5°». Примерами вещественных переменных являются сами элементы кортежей входных воздействий (скорость, высота), их степени, скорости их изменения и т. д.

Первый способ, предложенный в [9], использует только предикаты. Второй способ предлагается в настоящей работе и использует как предикаты, так и вещественные переменные.

2.1. Ранее предложенный способ представления

Пусть $p_1, \dots, p_m$ – набор предикатов. Входные события для автомата в предложенном в [9] способе реализации имеют вид p_i и $\neg p_i$ для каждого предиката p_i. Для каждого события в каждом состоянии автомата может находиться переход, для которого заданы конечное состояние и кортеж выходных воздействий (более точно – кортеж *изменений* выходных воздействий).

Такт работы автомата состоит из нескольких (не более m) переходов: -й по очередности переход происходит по событию, соответствующему значению

предиката p_i, при этом если переход отсутствует, то автомат сохраняет свое состояние. Пусть на некотором такте выполнились переходы, которым сопоставлены кортежи выходных воздействий $u_1, \dots, u_r$, тогда изменение кортежа выходных воздействий u всего такта вычисляется следующим образом:

$$\Delta u = \sum_{i=1}^{r} u_i.$$

Пример автомата с описанным способом представления приведен на рис. 6.

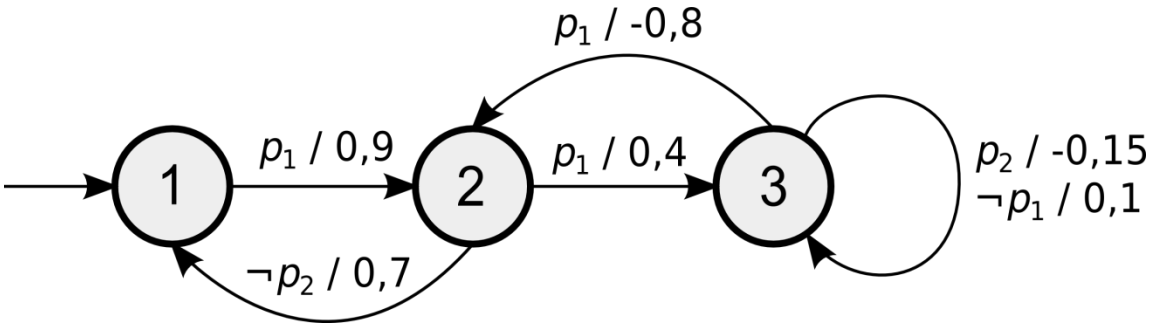

Рис. 6. Пример автомата с ранее предложенным способом представления

Одним из недостатков данного способа представления является возрастание числа хранимых и совершаемых за такт переходов автомата с числом предикатов *m*, результатом чего является ненаглядность функции переходов при больших *m*. Поскольку на каждом такте автомат сменяет свое состояние несколько раз, его состояния сложны для интерпретации. Другой недостаток рассмотренного способа – невозможность полноценного учета вещественных данных, находящихся в кортежах входных воздействий (предикаты являются лишь их дискретными «срезом»). Так, данный способ не позволяет реализовать законы управления, используемые в ПИД-регуляторах. Предлагаемый в настоящей работе способ представления автоматов не обладает этими недостатками.

2.2. Предлагаемый способ представления

Рассмотрим другой способ представления автоматов. Для представления **функции переходов** будем использовать ранее разработанный метод сокращенных таблиц [22]. Пример сокращенной таблицы уже был приведен в табл. 1 и табл. 2. В настоящей работе предполагается, что число значимых предикатов одинаково в

каждом состоянии автомата и заранее задано. С одной стороны, такое решение упрощает реализацию генетических операций (мутации и кроссовера) с сокращенными таблицами. С другой стороны, в случае переменного числа предикатов поисковая оптимизация будет отдавать предпочтение автоматам с б*о*льшим их числом, что может привести к проблеме *переобучения* и увеличить время работы алгоритма поисковой оптимизации. Для построения автоматов по тестам, записанным в рамках настоящей работы, оказалось достаточно лишь одного или двух значимых предикатов в состоянии.

В качестве альтернативы сокращенным таблицам могут быть использованы деревья решений [26]. Выбор сокращенных таблиц в настоящей работе обусловлен тем, что их реализация, а также реализация операций с ними проще.

Ключевым компонентом для представления **функции выходов** являются вещественные переменные. Для каждого управляющего параметра $j = 1..C$ задается набор $\{v_{j,i}\}_{i=1}^{n_j}$ из n_j вещественных переменных. Наборы переменных для различных j могут, вообще говоря, пересекаться. Наглядным примером пересечения является использование тождественной единицы как переменной для каждого управляющего параметра. Использование вещественных переменных позволяет более точно контролировать параметры объекта управления, поскольку возможные зависимости между входными и выходными воздействиями автоматов становятся не ограниченными дискретностью предикатов.

В каждом состоянии автомата хранится маска значимости вещественных переменных (пример приведен в табл. 4): только помеченные единицей в маске переменные влияют на формирование соответствующих им выходных воздействий. Использование масок значимости переменных позволяет уменьшить время работы процедуры расстановки выходных воздействий, которая будет описана в главе 3.

Табл. 4. Пример маски значимости вещественных переменных для одного из состояний автомата ($C = 2, n_1 = n_2 = 3$).

Управляющий параметр	Переменная	Значение маски
1	$v_{1,1}$	1
	$v_{1,2}$	1
	$v_{1,3}$	0
2	$v_{2,1}$	0
	$v_{2,2}$	1
	$v_{2,3}$	1

Каким же образом можно использовать вещественные переменные для выработки выходных воздействий? Отметим, что в процессе поиска автомата, поведение которого соответствует тестам, выходные воздействия можно расставлять автоматически на каркасе автомата, как это сделано в [9]. В этой работе каждому переходу сопоставляется кортеж изменений выходных воздействий, при этом сами кортежи определяются из условия максимизации ФП посредством решения систем линейных уравнений. Далее мы будем рассматривать только модели выработки выходных воздействий, допускающие оптимизацию их параметров таким способом (в [9] этими параметрами были сами значения кортежей).

Одной из идей использования вещественных переменных является использование зависящих от них кортежей выходных воздействий, записанных на переходах. Пусть для управляющих параметров $u_1, \dots, u_C$ и заданы переменные $\{v_{1,i}\}_{i=1}^{n_1}, \dots, \{v_{C,i}\}_{i=1}^{n_C}$. Для фиксированного параметра u_j каждому переходу автомата можно сопоставить переменное выходное воздействие $r_{j,0} + r_{j,1}v_{j,1} + \dots + r_{j,n_j}v_{j,n_j}$ с автоматически подбираемыми коэффициентами $r_{j,0}, \dots, r_{j,n_j}$. Слагаемое $r_{j,0}$ можно опустить, предположив, что одна из переменных – тождественная единица. Таким образом, на каждом такте работы автомата значение параметра u_j будет изменяться на линейную комбинацию текущих значений вещественных переменных, записанную на переходе.

Однако если разместить выходные воздействия не на переходах, а в состояниях, таким образом превратив автомат Мили в автомат Мура, то можно сделать состояния автомата более простыми для интерпретации. В этом случае каждое состояние автомата определяет закон управления, выполняющийся на протяжении сохранения автоматом этого состояния, а автомат представляет собой систему переключающихся регуляторов – то есть, является гибридной системой, близкой по своим свойствам к упомянутым в главе 1 гибридным автоматам [13]. В случае определения выходных воздействий состояниями автомата закон управления параметром u_j для состояния s выглядит следующим образом:

$$\Delta u_j = \sum_{i=1}^{n_j} r_{s,j,i} v_{j,i}. \quad (1)$$

Здесь Δu_j – изменение u_j за такт работы автомата, а $r_{s,j,i}$ – коэффициенты, вычисляемые алгоритмом расстановки выходных воздействий. Возможность автоматической расстановки этих коэффициентов с помощью решения систем линейных уравнений будет рассмотрена в главе 3.

Пример автомата с предлагаемым способом представления приведен на 0. В каждом состоянии автомата значимыми являются один предикат и две переменные, при этом всего заданы три предиката p_1, p_2, p_3 и три переменные v_1, v_2, v_3 для единственного управляющего параметра u (в названиях переменных индекс управляющего параметра опущен).

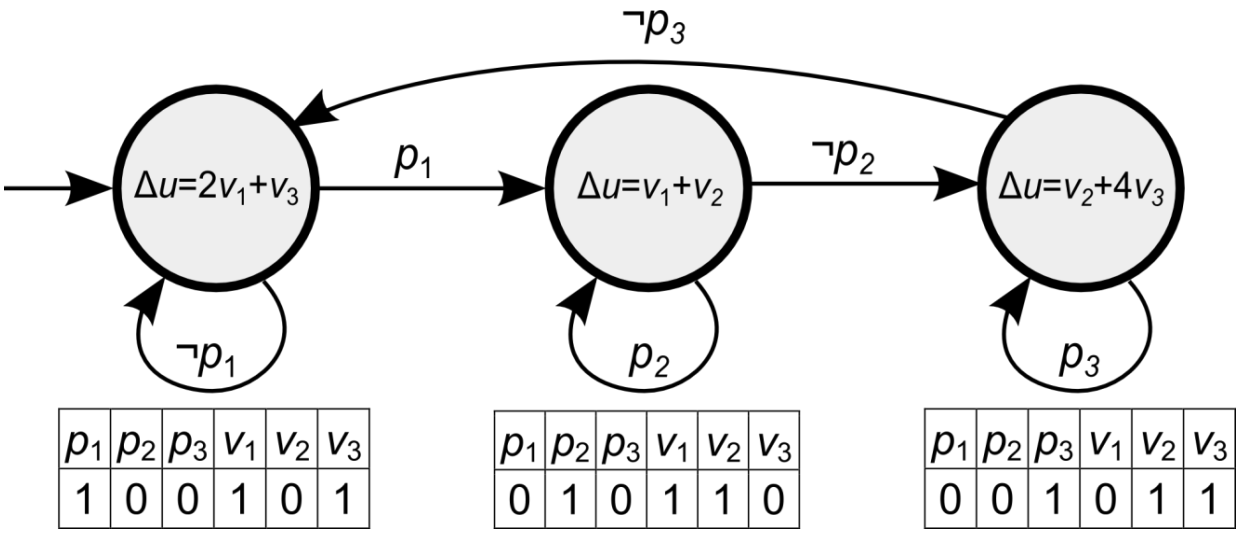

Рис. 7. Пример автомата с предлагаемым способом представления. Внутри состояний приведены законы управления параметром *u*, внизу – маски значимости переменных и предикатов, соответствующие различным состояниям. В каждом состоянии значимыми являются один предикат и две вещественные переменные

Выводы по главе 2

1. Рассмотрен ранее разработанный способ представления автоматов с непрерывными выходными воздействиями. Этот способ обладает рядом недостатков.
2. Предложен способ представления автоматов, которые используют для выработки выходных воздействий вещественные переменные. Он разработан с целью устранения недостатков предыдущего способа.

ГЛАВА 3. МЕТОД ПОСТРОЕНИЯ АВТОМАТОВ

В настоящем разделе описывается предлагаемый метод построения управляющих автоматов. Этот метод является улучшением метода, предложенного в [9], и позволяет строить автоматы, использующие для выработки выходных воздействий вещественные переменные.

Общая схема метода, которая не отличается от схемы метода [9], приведена на рис. 8. Входными данными для метода является набор тестов (тесты определены в главе 1). Построение автоматов осуществляется на основе поисковой оптимизации (например, с помощью генетического алгоритма или разновидности муравьиного алгоритма [27], которые используются в настоящей работе). Используется ФП, сравнивающая поведение автомата с тестами (в настоящей работе исследовано применение двух таких функций). Особью алгоритма поисковой оптимизации является каркас автомата – автомат с неопределенными выходными воздействиями. Использование каркаса в качестве особи алгоритма уменьшает пространство поиска и делает его конечным. Выходные воздействия определяются автоматически для каждого сгенерированного в процессе поисковой оптимизации каркаса. Далее отдельно рассматривается каждая часть метода и ее реализация в настоящей работе.

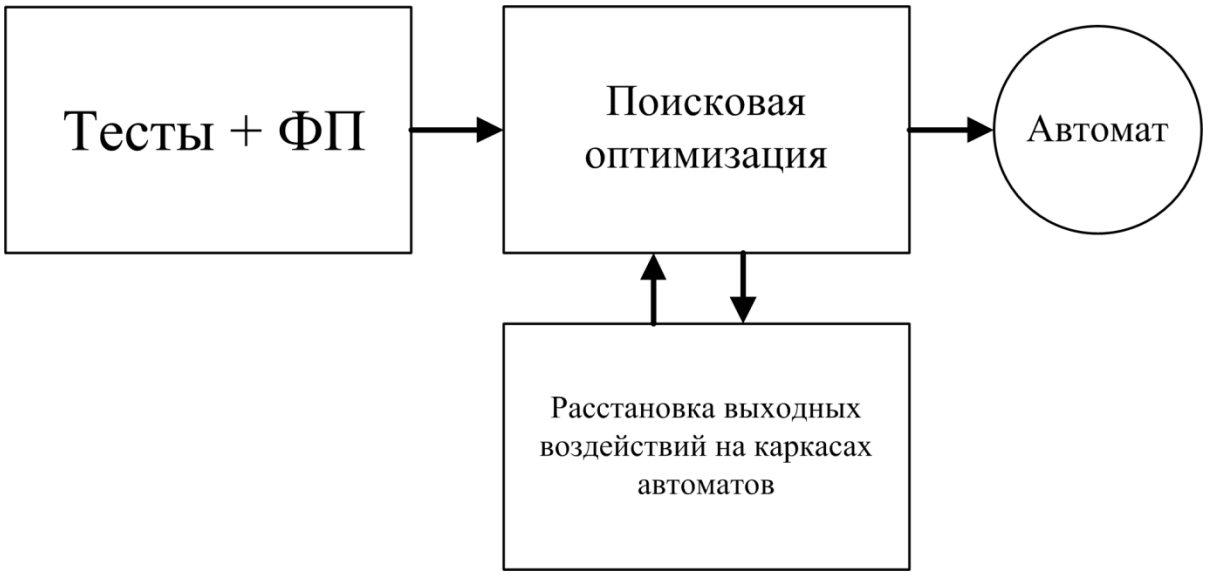

Рис. 8. Общая схема метода построения автоматов по тестам

3.1. Функции приспособленности

В настоящей работе используются две ФП, основанные на сходстве поведения, которое демонстрирует автомат, принимая на вход кортежи входных воздействий из тестов, и «эталонного» поведения, записанного в тех же тестах. Одна из этих функций близка к используемой в [9], а другая является ее модификацией. Кроме того, использование предлагаемого способа представления автоматов делает осмысленным добавление в ФП дополнительного слагаемого, о котором будет написано ниже.

Фиксируем некоторый управляющий автомат. Пусть $\mathrm{ans}[i]$ – последовательность кортежей выходных воздействий, выработанных автоматом, которому последовательно передаются кортежи входных воздействий из *i*-го теста (то есть, автомат запускается на входных воздействиях теста как конечный преобразователь). Эта последовательность состоит из чисел $\mathrm{ans}[i][t][j]$, где *t*, как и раньше, означает момент времени, а j – номер управляющего параметра.

Будем рассматривать расстояние между последовательностями $\mathrm{ans}[i]$ и $\mathrm{out}[i]$ в качестве штрафа для автомата:

$$\rho(\mathrm{ans}[i], \mathrm{out}[i]) = \sqrt{\frac{1}{\mathrm{len}[i]} \sum_{t=1}^{\mathrm{len}[i]} \frac{1}{C} \sum_{j=1}^{C} \left(\frac{\mathrm{ans}[i][t][j] - \mathrm{out}[i][t][j]}{c_j^{\max} - c_j^{\min}} \right)^2} \qquad (2)$$

Это расстояние можно интерпретировать как среднеквадратическое отклонение значений управляющих параметров, заданных последовательностями $\mathrm{out}[i]$ и $\mathrm{ans}[i]$. Пример проекции этих последовательностей на один управляющий параметр (положение руля высоты самолета) приведен на рис. 9.

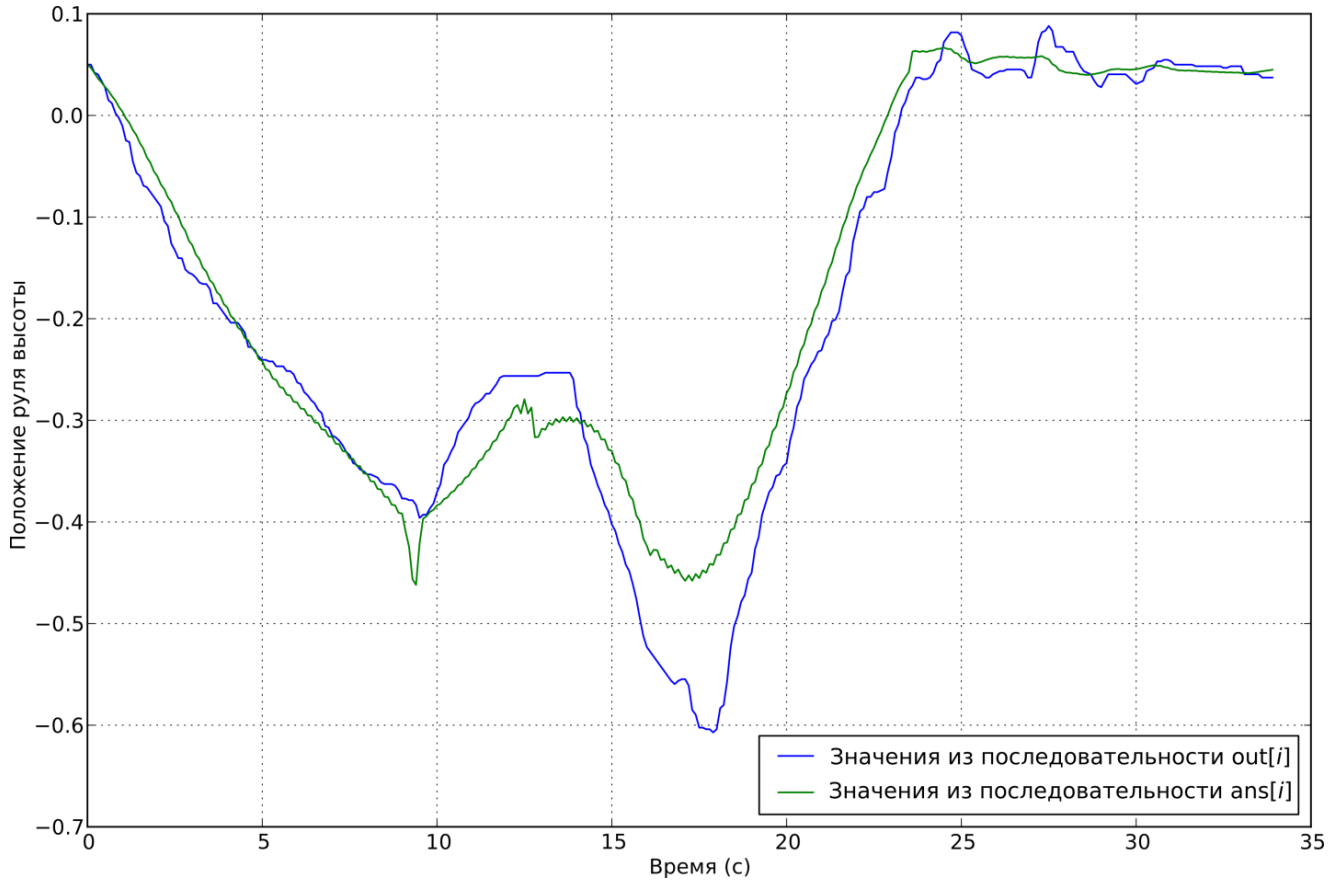

Рис. 9. Пример проекции последовательностей $\mathrm{out}[i]$ и $\mathrm{ans}[i]$ на один управляющий параметр

Одним из потенциальных достоинств предлагаемого способа представления автоматов является наглядность его состояний. Ее трудно получить, если состояния автомата часто меняются, поэтому будем учитывать число смен состояния автомата при запуске его на тестах как дополнительный штраф. Пусть τ – число смен состояния, которое автомат может совершить на протяжении выполнения элемента полета без штрафа, а τ_i – число смен состояния, выполненное автоматов при прохождении i-го теста. В качестве штрафа в настоящей работе используется следующая величина:

$$P_\tau = K\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\max(\tau_i - \tau, 0))^2}.$$

Здесь K – эмпирически подбираемый положительный коэффициент. В настоящей работе он брался в диапазоне от 0,00015 до 0,0003.

Объединим штрафы $\rho(\text{ans}[i], \text{out}[i])$, рассчитанные на различных тестах, и штраф P_τ в функцию приспособленности f (неявным аргументом функции является каркас автомата):

$$f = 1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N}\rho^2(\text{ans}[i], \text{out}[i])} - P_\tau. \quad (3)$$

Другая рассматриваемая в настоящей работе функция f_Δ учитывает различия не между значениями управляющих параметров, а между изменениями их значений на соседних тактах работы автомата. Пусть $\Delta\text{ans}[i][1][j] = 0$ и $\Delta\text{ans}[i][t][j] = \text{ans}[i][t][j] - \text{ans}[i][t-1][j]$ при $t > 1$. Аналогичным образом определим последовательность $\Delta\text{out}[i][t][j]$. Функция f_Δ задается следующим выражением:

$$f_\Delta = 1 - \sqrt{\frac{1}{N}\sum_{i=1}^{N}\rho^2(\Delta\text{ans}[i], \Delta\text{out}[i])} - P_\tau,$$

Как будет показано в результате экспериментального исследования, использование f_Δ предпочтительней, если автомат должен моментально реагировать на изменения параметров объекта управления. Отметим, что при построении автоматов с ранее предложенным способом представления, которое будет также осуществляться в экспериментальном исследовании, штраф P_τ не используется.

3.2. Расстановка выходных воздействий

Выходные воздействия автомата могут быть расставлены на каркасе автомата, так чтобы ФП (f или f_Δ) достигла максимума для заданного каркаса. Во многом такая возможность обусловлена видом функции приспособленности, благодаря которому задача максимизации сводится к задаче решения нескольких систем линейных уравнений. В [9] описан алгоритм расстановки непрерывных выходных воздействий для автоматов, представленных в виде, описанном в разделе 2.1. Данный алгоритм запускается для каждого сгенерированного в процессе работы алгоритма поисковой оптимизации автомата. Ниже приведена модификация этого

алгоритма, адаптированная для предложенного в разделе 2.2 способа представления автоматов.

Фиксируем некоторый каркас автомата. Далее мы будем рассматривать задачу максимизации функции f. Задача максимизации f_Δ решается так же с точностью до изменения последовательностей $\mathrm{ans}[i]$ и $\mathrm{out}[i]$ на последовательности $\Delta\mathrm{ans}[i]$ и $\Delta\mathrm{out}[i]$ соответственно.

Для начала заметим, что штраф P_τ является константой, поэтому задача упрощается до следующей:

$$\sum_{i=1}^{N} \rho^2(\mathrm{ans}[i], \mathrm{out}[i]) \to \min.$$

Раскрывая ρ по определению, получаем:

$$\sum_{i=1}^{N} \frac{1}{\mathrm{len}[i]} \sum_{t=1}^{\mathrm{len}[i]} \sum_{j=1}^{C} \left(\frac{\mathrm{ans}[i][t][j] - \mathrm{out}[i][t][j]}{c_j^{\max} - c_j^{\min}} \right)^2 \to \min.$$

Далее заметим, что для каждого управляющего параметра задача может решаться независимо. Фиксируем индекс параметра j_0. Получаем:

$$g_{j_0} := \sum_{i=1}^{N} \frac{1}{\mathrm{len}[i]} \sum_{t=1}^{\mathrm{len}[i]} (\mathrm{ans}[i][t][j_0] - \mathrm{out}[i][t][j_0])^2 \to \min. \qquad (4)$$

Теперь необходимо выразить g_{j_0} через оптимизируемые параметры $r_{s,j,i}$ из выражения (1). Вспомним, что некоторые из этих чисел заведомо должны быть нулями в соответствии с масками значимости вещественных переменных. Кроме того, в настоящий момент нас интересуют только числа $r_{s,j_0,i}$ для фиксированного j_0. Пусть $\hat{r}_1, \ldots, \hat{r}_M$ – это в точности те параметры, которые необходимо оптимизировать. Каждое число $\hat{r}_l$ $(1 \le l \le M)$ соответствует некоторому состоянию автомата $\hat{s}_l \in S$ и некоторой переменной $\hat{v}_l$. Вспомним выражение (1) и предположим, автомат запущен на i-м тесте, а t – текущий момент времени. Обозначим за $\beta_{i,t,l}$ числа, которые будут умножены на $\hat{r}_l$ и добавлены к Δu_j на текущем такте. Если состояние $\hat{s}_l$ является текущим, то $\beta_{i,t,l} = \hat{v}_l$, иначе $\beta_{i,t,l} = 0$.

После принятия таких обозначений выражение (1) может быть применено к работе управляющего автомата на тесте:

$$\text{ans}[i][t][j] = \text{ans}[i][t-1][j] + \sum_{l=1}^{M} \beta_{i,t,l}\hat{r}_l\,, t > 1. \quad (5)$$

Естественно также считать, что выполняется утверждение $\text{ans}[i][1][j] = \text{out}[i][1][j]$. Теперь приведем (5) к замкнутой форме:

$$\text{ans}[i][t][j] = \text{out}[i][1][j] + \sum_{l=1}^{M} \alpha_{i,t,l}\hat{r}_l\,; \quad (6)$$

$$\alpha_{i,t,l} = \sum_{t'=2}^{t} \beta_{i,t',l}. \quad (7)$$

Подставляя (6) в (4) и приравнивая к нулю частные производные g_{j_0} по различным параметрам $\hat{r}_{l_0}$, получаем:

$$2\sum_{i=1}^{N} \frac{1}{\text{len}[i]} \sum_{t=1}^{\text{len}[i]} \alpha_{i,t,l_0} \left(\text{out}[i][1][j_0] + \sum_{l=1}^{M} \alpha_{i,t,l}\hat{r}_l - \text{out}[i][t][j_0] \right) = 0.$$

Это выражение, в свою очередь, переписывается в более удобном виде:

$$\sum_{l=1}^{M} \left(\sum_{i=1}^{\text{N}} \frac{1}{\text{len}[i]} \sum_{t=1}^{\text{len}[i]} \alpha_{i,t,l_0}\,\alpha_{i,t,l} \right) \hat{r}_l = \sum_{i=1}^{N} \frac{1}{\text{len}[i]} \sum_{t=1}^{\text{len}[i]} \alpha_{i,t,l_0}\,(\text{out}[i][t][j_0] - \text{out}[i][1][j_0]). \quad (8)$$

Теперь становится видно, что выражения такого вида для различных l_0 образуют систему линейных алгебраических уравнений. Система может быть решена методом Гаусса за время $O(M^3)$. Тем не менее, нахождение ее матрицы займет $O\left(M^2 \sum_{i=1}^{N} \text{len}[i]\right)$ времени. Решение подобной системы должно быть проведено для каждого управляющего параметра. Для различных управляющих параметров j_0 числа $\hat{r}_1, \ldots, \hat{r}_M$ будут соответствовать различным оптимизируемым параметрам.

Обобщая содержимое раздела, запишем сам алгоритм расстановки выходных воздействий:

Для каждого управляющего параметра $j_0 = 1..C$:

1. Выделить поднабор $\hat{r}_1, \dots, \hat{r}_M$ заведомо ненулевых чисел из набора $\{r_{s,j_0,i}\}_{s,i}$ $(s \in S, i = 1..n_j)$.
2. Вычислить все $\{\alpha_{i,t,l}\}_{i,t,l} (i = 1..N, t = 1..\text{len}[i], l = 1..M)$ согласно (7).
3. Найти матрицу системы уравнений для параметра j_0 согласно (8).
4. Решить получившуюся систему методом Гаусса, тем самым найдя значения $\hat{r}_1, \dots, \hat{r}_M$.

Время, требуемое для расстановки выходных воздействий, можно уменьшить, если для некоторых органов управления использовать одни и те же вещественные переменные и одну и ту же маску их значимости. Так, в настоящей работе для построения автоматов, управляющих моделью самолета, использовалось две маски: одна для руля высоты, другая для элеронов и руля направления. В этом случае системы уравнений для элеронов и руля направления отличаются лишь правыми частыми, поэтому нахождение их левых частей и их решение (наиболее трудоемкую часть итерации алгоритма) можно осуществлять совместно. В варианте [9] настоящего алгоритма левые части систем для всех управляющих параметров совпадали по причине отсутствия вещественных переменных.

3.3. Поисковая оптимизация

В настоящей работе для генерации автоматов при помощи поисковой оптимизации исследовалось применение генетического алгоритма и муравьиного алгоритма [28, 29], предложенного в [27]. В [10] показано, что при использовании муравьиного алгоритма производительность метода построения автоматов может быть увеличена. В настоящей работе упомянутые алгоритмы используются для построения автоматов с предлагаемым способом представления. Сравнение их эффективности на задаче, поставленной в настоящей работе, приведено в главе 4.

Для каждого каркаса автомата A с предлагаемым в настоящей работе способом представления рассмотрим множество его «соседей» $N(A)$. В это множество входят автоматы, полученные из A:

- изменением стартового состояния s.
- изменением одного из значений функции переходов в некотором состоянии каркаса A;
- перестановкой двух различных значений в одной из масок значимости (предикатов или вещественных переменных) в некотором состоянии каркаса A.

3.3.1. Генетический алгоритм

Реализованный генетический алгоритм использует турнирный метод отбора. Оператор мутации, примененный к каркасу A, равновероятно выбирает один из элементов множества $N(A)$. Используется оператор однородного кроссовера, при этом каждая маска значимости рассматривается как отдельный ген (изменения внутри масок не происходят). Размер поколения равен 300, при этом 10 % особей являются элитными. В случае стагнации в течение 2000 вычислений ФП применяется «большая» мутация: половина особей в текущем поколении заменяется на случайно сгенерированные.

3.3.1. Муравьиный алгоритм

Опишем разновидность муравьиного алгоритма, предложенную в [28] для построения конечных автоматов. Пространство поиска в алгоритме представляется в виде ориентированного графа G, вершинами которого являются автоматы (особи алгоритма, элементы пространства поиска), а ребрами – *мутации* между автоматами (в настоящей работе вместо автоматов используются их каркасы). Граф строится постепенно, начиная с единственной вершины – сгенерированного случайным образом каркаса. Каждому ребру графа соответствует некоторое значение *феромона*, которое изменяется муравьями в процессе работы алгоритма.

Работа алгоритма состоит из последовательности итераций, в начале каждой из которых в некоторые вершины графа G помещаются N_{ants} муравьев, которые далее перемещаются по графу в поисках особей с высоким значением ФП. Размещение муравьев происходит случайным образом на вершинах пути, который проделал муравей, нашедший лучшую особь из рассмотренных муравьями за все

итерации алгоритма. Небольшой фрагмент графа G и пример участка пути муравья в нем приведены на рис. 10.

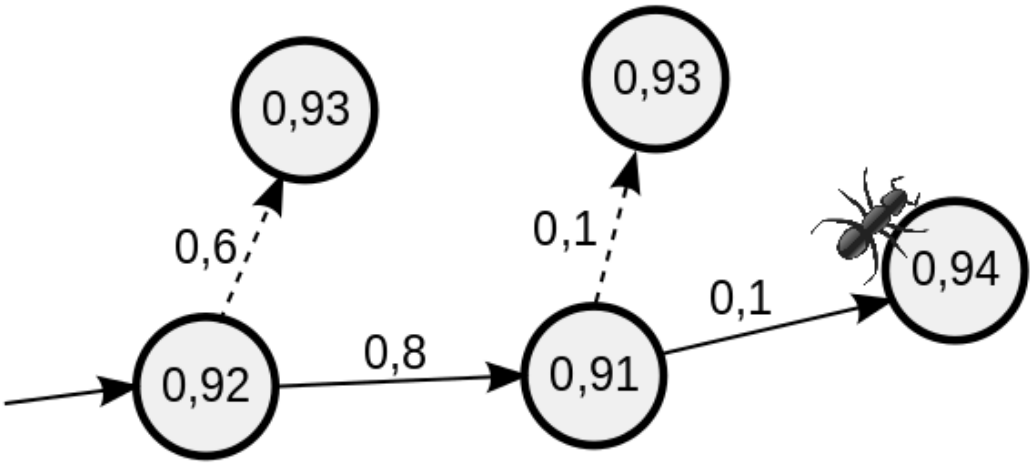

Рис. 10. Часть графа G. Внутри вершин показаны ФП каркасов, на ребрах – значения феромона. Путь муравья обозначен сплошными стрелками, остальные ребра графа – пунктирными

Путь муравья формируется следующим образом. Пусть муравей находится в вершине графа, соответствующей некоторой особи A. С вероятностью p_{new} путем мутаций текущей особи (выбирается случайный элемент из $N(A)$) муравей добавляет в граф N_{mut} новых ребер и перемещается по тому из них, которое ведет к особи с наибольшим значением ФП. В противном случае, муравей выбирает ребро для перемещения исходя из значений феромона. Подробности обновления и использования феромона муравьями приведены в [28], но в настоящей работе использование феромона не приводит к заметному изменению производительности алгоритма. Если в течение N_{stag} перемещений муравья максимальное найденное им значение ФП не увеличивается, то его путь завершается. Используемые в настоящей работе параметры муравьиного алгоритма приведены в табл. 7.

Табл. 5. Параметры муравьиного алгоритма при построении автоматов с различным способом представления

Способ представления автомата	N_{ants}	N_{stag}	N_{mut}	p_{new}
Ранее разработанный	4	40	35	0,25
Предлагаемый	2	30	40	0,25

Выводы по главе 3

1. Предложен метод построения автоматов со способом представления, описанным в разделе 2.2.
2. Рассмотрена возможность использования различных функций приспособленности и различных алгоритмов поисковой оптимизации.
3. Обоснована возможность автоматического определения выходных воздействий для каркасов автоматов. Приведена процедура для их определения.

ГЛАВА 4. ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ

Настоящий раздел содержит результаты экспериментальных исследований предлагаемого метода построения автоматов. В первой части раздела метод применяется для управления моделью самолета. Производится сравнение эффективности использования муравьиного и генетического алгоритмов, обосновывается применение масок значимости вещественных переменных, а также производится сравнение предлагаемого метода с его прототипом [9]. Во второй части раздела метод применяется для построения автоматов, управляющих моделью вертолета.

Все использованные в экспериментальном исследовании наборы тестов были записаны в авиасимуляторе *FlightGear*. Там же запускались построенные автоматы. Для построения автоматов использовался персональный компьютер с четырехъядерным процессором *Intel Core* 2 *Quad Q*9400. Метод построения автоматов был реализован на языке программирования *Java*.

4.1. Управление моделью самолета

Метод был применен для построения автоматов управления моделью самолета (использовались модели гражданского самолета *Piper PA*34-200*T* и истребителя *Gloster Meteor*). Были рассмотрены задачи выполнения трех фигур пилотажа, для каждой из которых был записан набор тестов:

- «мертвая петля» – 33 теста;
- «бочка» (разворот вокруг продольной оси на 360°) – 28 тестов;
- разворот в горизонтальной плоскости на 180° (далее – разворот) – 19 тестов.

Примеры траекторий, сделанных при записи набора тестов для разворота, приведены на рис. 11.

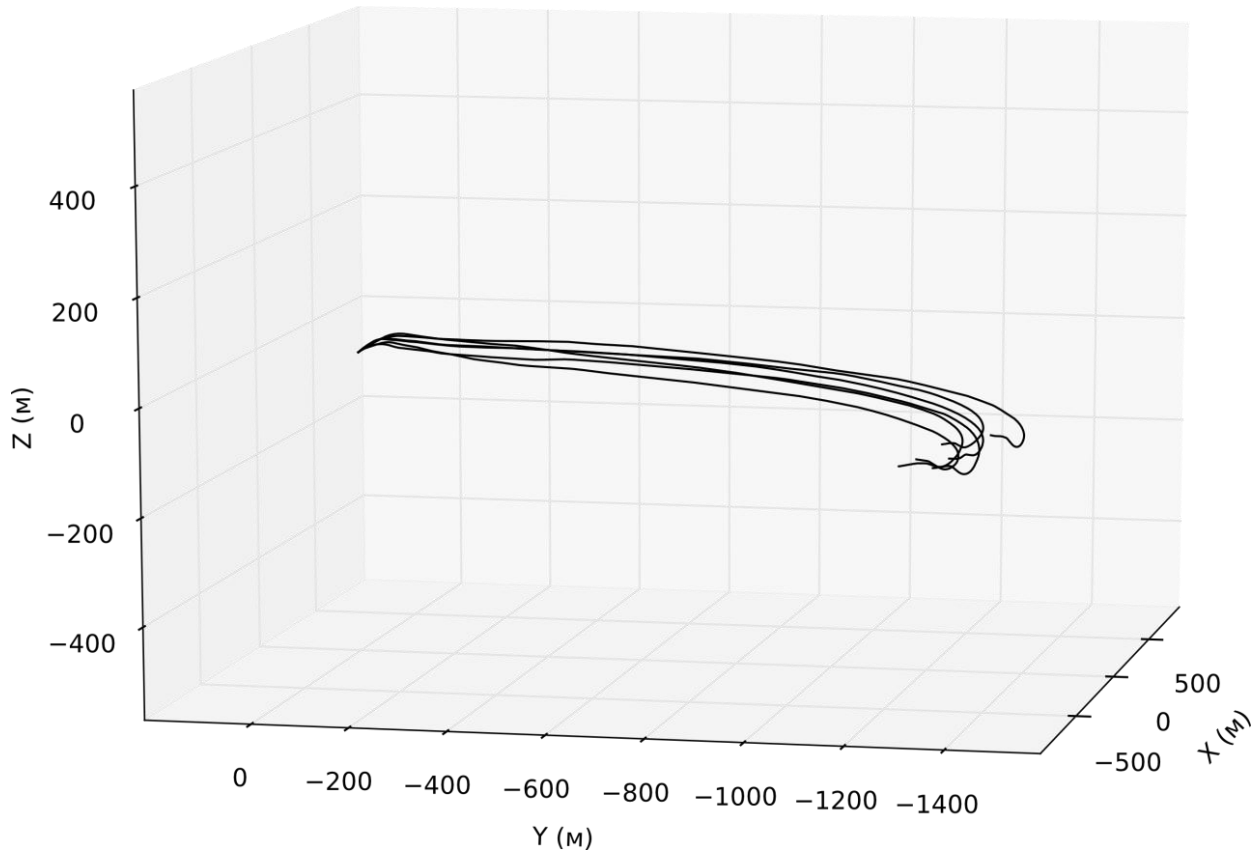

Рис. 11. Примеры траекторий, сделанных самолетом при записи тестов для разворота

Для каждой фигуры пилотажа были подобраны наборы предикатов и вещественных переменных, необходимые для работы метода. Их подбор осуществлялся итеративно: если качество выполнения фигуры пилотажа построенными автоматами являлось неудовлетворительным, набор переменных корректировался.

Напомним, что в настоящей работе исследуется применение двух ФП: f и f_Δ. При построении автоматов оказалось, что использование функции f позволяет построить автоматы для всех трех фигур пилотажа, а использование f_Δ применимо только к развороту, но увеличивает качество его выполнения автоматами. Далее будут рассмотрены четыре **конфигурации** запусков метода:

- $(\text{МП}, f)$ – «мертвая петля» при использовании f;
- $(\text{Б}, f)$ – «бочка» при использовании f;
- $(\text{Р}, f)$ – разворот при использовании f;
- $(\text{Р}, f_\Delta)$ – разворот при использовании f_Δ.

4.1.1. Сравнение производительности генетического и муравьиного алгоритмов

В качестве алгоритмов поисковой оптимизации в настоящей работе используются генетический и муравьиный алгоритмы. Генетический алгоритм использовался в прототипе предлагаемого метода построения автоматов [9]. Муравьиный алгоритм улучшил производительность [10] этого метода, однако при построении автоматов с предложенным способом представления сравнение производительности этих алгоритмов может дать другие результаты.

Было проведено по 50 запусков двух алгоритмов при числе состояний автомата от трех до пяти на всех четырех конфигурациях. Критерием остановки запуска была стагнация в течение 5000 вычислений ФП. В табл. 6 и табл. 7 приведено сравнение производительности алгоритмов. Измерялись среднее число вычислений ФП в запуске и среднее значение ФП, достигнутое алгоритмами по окончании запусков. Как видно их таблиц, муравьиный алгоритм позволяет получать бо́льшие значения ФП за меньшее время. Различие в производительности алгоритмов не является критическим, однако по результатам их сравнения далее в настоящей работе будет использоваться муравьиный алгоритм.

Табл. 6. Результаты запусков генетического алгоритма на различных конфигурациях

Параметр сравнения	**Число состояний**	(МП, f)	(Б, f)	(Р, f)	(Р, f_Δ)
Среднее значение ФП	3	0,9854	0,9851	0,9892	0,99748
	4	0,9865	0,9859	0,9898	0,99750
	5	0,9871	0,9865	0,9900	0,99750
Среднее число вычислений ФП	3	12300	12300	8800	8300
	4	15900	17000	11200	10100
	5	18600	20700	10900	10600

Табл. 7. Результаты запусков муравьиного алгоритма на различных конфигурациях

Параметр сравнения	**Число состояний**	(МП, f)	(Б, f)	(Р, f)	(Р, f_Δ)
Среднее значение ФП	3	0,9855	0,9851	0,9893	0,99747
	4	0,9866	0,9862	0,9899	0,99750
	5	0,9873	0,9865	0,9901	0,99750
Среднее число вычислений ФП	3	9300	9600	8200	8100
	4	12300	11600	8900	9000
	5	14200	13000	10800	9600

4.1.2. Оценка эффективности использования масок значимости вещественных переменных

Основная цель, преследуемая использованием масок значимости вещественных переменных – возможность масштабирования метода построения автоматов на случай, когда задано большое их число. В этом случае можно ожидать, что в каждом состоянии автомата для приемлемого качества управления достаточно использовать лишь небольшую часть переменных. Как будет показано ниже, отказ от использования масок значимости переменных не приводит к существенному изменению качества автоматов и в то же время замедляет метод их построения. Отказ от их использования равносилен использованию масок, состоящих из одних единиц, при этом для каждого управляющего параметра по-прежнему задаются свои наборы переменных.

В проведенном исследовании число вещественных переменных было равно порядка 7–8, при этом использование масок значимости (порядка 4–5 значимых переменных) ускоряло вычисления ФП примерно в два раза. Причина этому – асимптотика алгоритма расстановки выходных воздействий, задаваемая выражением $O\left(M^2 \sum_{i=1}^{N} \mathrm{len}[i] + M^3\right)$, где M – число оптимизируемых параметров для некоторого органа управления. Это число растет линейно при увеличении числа значимых переменных, что приводит к квадратичному росту времени выполнения наиболее трудоемкой части алгоритма, связанной с нахождением матриц систем линейных уравнений. Заметим также, что, поскольку маски значимости являются

частью особи алгоритма поисковой оптимизации, отказ от их использования уменьшает пространство поиска, но на рассмотренных наборах тестах уменьшение среднего числа вычислений ФП за счет этого оказалось мало.

Перейдем к оценке качества построенных автоматов. Сгенерированные автоматы оценивались в авиасимуляторе *FlightGear* с использованием следующих критериев. Каждый оцениваемый автомат запускался пять раз в условиях, аналогичным условиям записи тестов. Использовались два параметра оценки, равные средним отклонениям углов крена и тангажа от записанных в тестах. Более формально, по множеству троек $\{(j,i,t) \mid j = 1..5, i = 1..N, t = 1..\mathrm{len}[i]\}$ усреднялось значение $|\alpha_{j,t}^{\mathrm{run}} - \alpha_{i,t}^{\mathrm{test}}|$, где $\alpha_{j,t}^{\mathrm{run}}$ – значение угла на -том запуске автомата в момент времени t, а $\alpha_{i,t}^{\mathrm{test}}$ – значение угла в тот же момент времени в i-м тесте. Исключение составила оценка отклонения угла крена для «мертвой петли»: вместо $\alpha_{i,t}^{\mathrm{test}}$ использовался ноль как более осмысленное «эталонное» значение. Используемые критерии качества не учитывают разницу в относительной скорости выполнения фигур пилотажа. В используемых наборах тестов эта разница была мала, поэтому было принято решение не дорабатывать критерии для ее учета. Средние значения рассмотренных критериев на самих наборах тестов приведены в табл. 8.

Табл. 8. Средние значения критериев качества поведения автоматов в моделировании для различных наборов тестов

Критерий качества	**«Мертвая петля»**	**«Бочка»**	**Разворот**
Среднее отклонение крена	1,535	17,882	2,084
Среднее отклонение тангажа	8,100	2,279	0,579

Табл. 9 и табл. 10 показывают значения критериев качества автоматов, построенных с использованием и без использования масок значимости переменных. Для оценки качества автоматов в моделировании отбирались автоматы с максимальными значениями ФП из десяти лучших по достигнутому значению ФП запусков муравьиного алгоритма.

Табл. 9. Значение критериев качества для автоматов, построенных с использованием масок значимости вещественных переменных

Критерий качества	**Число состояний**	(МП, f)	(Б, f)	(Р, f)	(Р, f_Δ)
Среднее отклонение крена	3	1,8378	17,9345	3,2831	1,9772
	4	2,1673	15,0424	3,183	2,1055
	5	5,0243	15,6605	3,3624	1,833
Среднее отклонение тангажа	3	10,8957	3,822	1,5512	0,4966
	4	13,8881	2,2089	1,6847	0,4546
	5	20,2396	2,6068	0,8981	0,4644

Табл. 10. Значение критериев качества для автоматов, построенных без использования масок значимости вещественных переменных

Критерий качества	**Число состояний**	(МП, f)	(Б, f)	(Р, f)	(Р, f_Δ)
Среднее отклонение крена	3	1,3024	34,5983	3,0956	2,0921
	4	2,2978	15,2318	3,1988	1,9323
	5	3,8047	18,505	5,0787	1,7654
Среднее отклонение тангажа	3	8,5337	10,0767	1,2824	0,4928
	4	12,2546	2,0312	1,6795	0,4624
	5	16,3688	3,4073	1,7203	0,4686

Однозначной выгоды по качеству автоматов от использования или неиспользования масок значимости по данным таблиц не видно. На наборе тестов «мертвая петля» их использование снижает качество автоматов. В то же время, на наборе тестов «бочка» эффект противоположный, что может означать переобучение в связи с большим числом параметров, оптимизируемых алгоритмом расстановки выходных воздействий. Далее в настоящей работе маски значимости будут использоваться, поскольку они ускоряют метод построения автоматов.

4.1.3. Сравнение метода с прототипом

Предложенный метод построения автоматов и его прототип [9] различаются типом генерируемых автоматов и реализацией процедуры расстановки выходных

воздействий, при этом их общая схема (см. рис. 8) совпадает, что позволяет легко провести их сравнение.

Для построения автоматов с ранее разработанным способом представления, описанном в разделе 2.1 (далее – способ 2.1), были итеративно подобраны наборы предикатов, как и для построения автоматов с предложенным способом представления (далее – способ 2.2). Ранее в табл. 7 были приведены результаты множественных запусков муравьиного алгоритма, осуществлявшего поиск автоматов со способом представления 2.2. В табл. 11 приведены результаты запусков муравьиного алгоритма при построении автоматов со способом представления 2.1 (каждое значение в таблице, как и ранее, приведено усредненным по 50-ти запускам).

Табл. 11. Результаты запусков муравьиного алгоритма при построении автоматов со способом представления 2.1

Параметр сравнения	**Число состояний**	(МП, f)	(Б, f)	(Р, f)	(Р, f_Δ)
Среднее значение ФП	3	0,9814	0,9832	0,9891	0,99731
	4	0,9834	0,9854	0,9900	0,99734
	5	0,9838	0,9856	0,9900	0,99734
Среднее число вычислений ФП	3	11000	12600	9400	10400
	4	10400	12700	12400	10700
	5	11800	10500	11200	11100

Сравним данные табл. 7 и табл. 11. Как видно из таблиц, использование способа представления 2.2 позволяет получить автоматы с более высоким значением ФП. Обобщая результаты по числу вычислений ФП в запусках, можно сказать, что оно варьируется от 9000 до 14000 и слабо зависит от способа представления автоматов. Время построения автоматов занимало порядка пяти-десяти минут при использовании способа представления 2.1 и порядка **десяти-двадцати минут** при использовании способа 2.2. Причина возникновения этой разницы – разное время вычисления ФП для автоматов с разными способами представления. В среднем двукратное по сравнению со способом представления 2.1 время вычисления ФП для

автоматов со способом 2.2 является недостатком предлагаемого в настоящей работе метода представления автоматов, однако этот недостаток некритичен в силу малости абсолютного времени, требуемого методу построения автоматов.

Перейдем к сравнению качества автоматов, построенных с помощью предлагаемого метода и метода [9]. В табл. 12 приведены данные, аналогичные данным из табл. 9, но рассчитанные для автоматов со способом представления 2.1.

Табл. 12. Значение критериев качества для автоматов со способом представления 2.1

Критерий качества	**Число состояний**	(МП, f)	(Б, f)	(Р, f)	(Р, f_Δ)
Среднее отклонение крена	3	4,2647	20,9963	48,6818	20,0503
	4	5,1842	21,6019	54,1725	22,2518
	5	4,468	25,1473	55,0445	27,1923
Среднее отклонение тангажа	3	15,2574	4,6238	9,4707	4,4263
	4	18,0996	4,4626	8,8672	3,6383
	5	17,5642	4,6499	7,0524	4,6956

Сравним данные, приведенные в табл. 9 и табл. 12. Во-первых, автоматы со способом представления 2.2, построенные для выполнения «мертвой петли» и «бочки», лучше управляют креном и тангажом. Различие в критериях качества еще сильнее проявляется на развороте. Причина столь заметной разницы в том, что задача подбора предикатов для выполнения разворота при использовании способа представления 2.1 не была решена полноценно: почти все автоматы оказались неспособны выполнить разворот и совершали ошибки в различные моменты его выполнения, оказываясь неспособными закончить этот элемент полета. Также отметим, что использование функции приспособленности f_Δ при построении автоматов для разворота позволило достичь качества его выполнения, превосходящее качество записанных тестов (см. табл. 8).

Приведенные в табл. 9 и табл. 12 результаты позволяют сделать вывод о том, что предложенный в настоящей работе способ представления автоматов превзошел ранее разработанный по качеству. Как показало сравнение поведений автоматов с

различными способами представления на тестах, способ представления 2.2 обеспечивает более «плавное» управление параметрами: графики их изменения на тестах напоминают гладкие функции (на рис. 9 зеленая кривая является примером такого графика). Кроме того, более высокие значения ФП говорят о том, что использование вещественных переменных позволяет точнее определять зависимости между входными и выходными воздействиями автомата, чем при использовании только предикатов.

Отдельно рассмотрим поведение автоматов на элементе полета «разворот». Данный элемент полета отличается от остальных тем, что автомат должен моментально реагировать на изменения входных воздействий. В тестах, описывающих разворот, самолет пытался держать постоянный угол крена (около 60º) и корректировал его воздействиями на органы управления. Синяя кривая на рис. 12 показывает пример управления рулем высоты в одном из тестов, описывающих разворот.

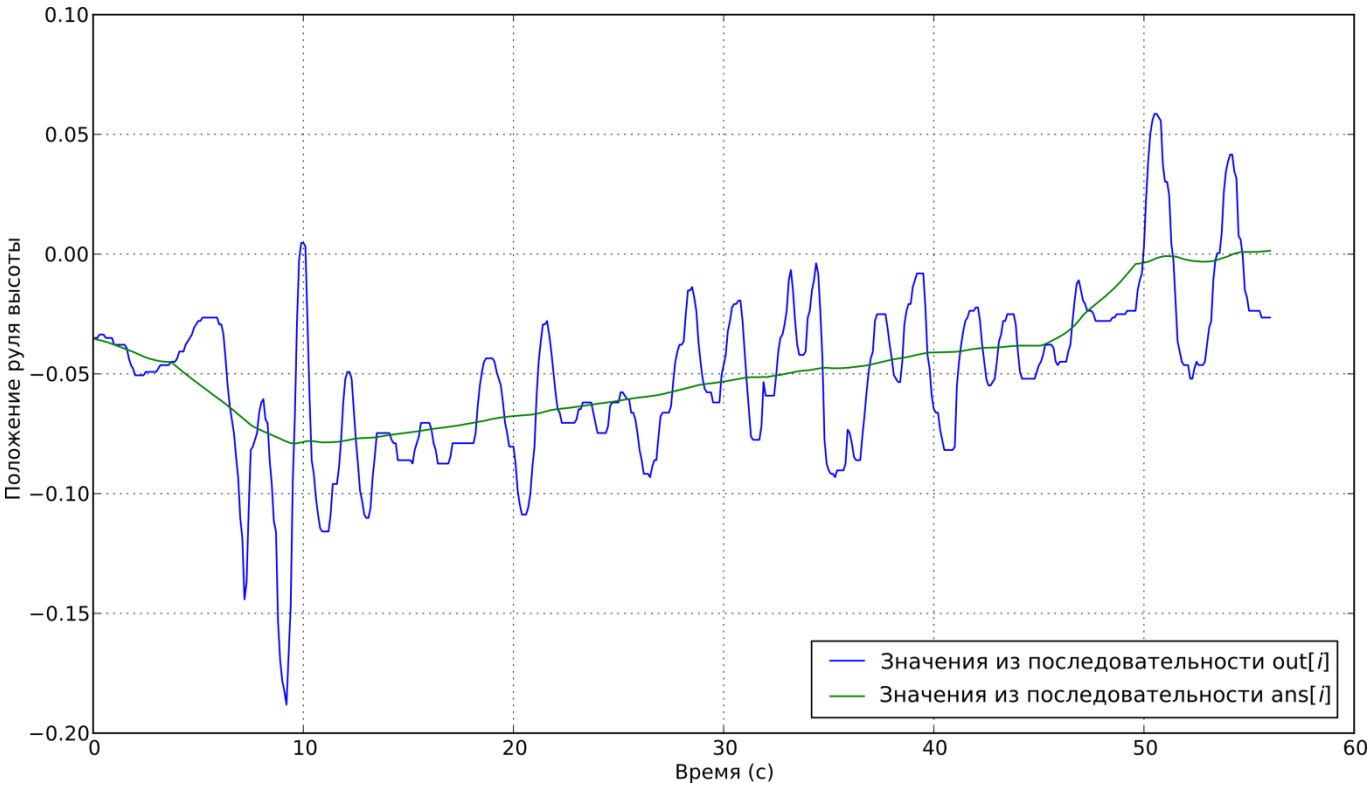

Рис. 12. Пример выходной последовательности теста, описывающего разворот, и автомата при запуске на этом тесте, построенного с помощью функции f

На этом элементе полета работоспособными оказались только автоматы со способом представления 2.2, при этом использование для построения автоматов функции f_Δ увеличивает их качество. Автоматы, построенные с использованием f, хорошо соответствовали тестам по абсолютному положению органов управления, но ценой этому была слабая реакция на изменение входных переменных. На рис. 12 зеленая кривая является примером поведения автомата, построенного для разворота с использованием f. Но даже при использовании функции f_Δ автоматы со способом представления 2.1 не могли выполнить разворот: отклонение углов крена и тангажа от оптимальных не вызывало выходных воздействий, достаточных для их нормализации. Выходные воздействия автоматов со способом представления 2.2 хоть и были слабыми, но линейно возрастали с ростом невязки углов, что предотвращало ее неограниченный рост.

Один из автоматов, построенных для выполнения разворота предложенным методом, приведен в приложении А. На рис. 13 показан снимок экрана авиасимулятора *FlightGear* с аналогичным автоматом, выполняющий этот элемент полета.

Рис. 13. Выполнение разворота истребителем *Gloster Meteor* под управлением одного из построенных автоматов с предложенным способом представления (снимок экрана)

Видеозаписи выполнений различных фигур пилотажа автоматами, построенными в ходе экспериментального исследования предложенного метода, доступны по следующим web-адресам:

- «мертвая петля»: http://youtu.be/vnJ_-gMLjx8;
- «бочка»: http://youtu.be/2GxzcPx0XNw;
- разворот: http://youtu.be/n9q5FmCYs6M.

Учет числа смен состояния, сделанных автоматом при работе на тестах, в ФП, а также использование автоматов Мура позволили сделать состояния автоматов наглядными. На видеозаписях разные состояния автоматов соответствуют выполнению различных частей фигур пилотажа. В то же время смена состояний автоматов, построенных методом [9], выглядит хаотичной.

4.2. Управление моделью вертолета

Предложенный в настоящей работе метод построения автоматов был также опробован на задаче управления моделью вертолета *Robinson R*44 *Raven II*. Для записи тестов и запуска автоматов также использовался авиасимулятор *FlightGear*.

Были записаны два набора тестов. Первый набор состоял из четырех тестов длиной около шести минут и описывал подъем вертолета на небольшую высоту (около 5 м) и зависание над землей (будем называть эту задачу зависанием). Второй набор состоял из 19 тестов длиной около 100 с и описывал подъем, полет вертолета вдоль взлетно-посадочной полосы на фиксированное расстояние и приземление (будем называть эту задачу полетом). Тесты записывались при отсутствии ветра, поскольку с аккуратным ручным управлением вертолетом при его наличии справиться не удалось.

Для построения автоматов использовалась ФП f_Δ. Функция f не подошла для построения автоматов управления вертолетом по тем же причинам, по которым она плохо подошла для разворота самолета. Детальное сравнение предложенного метода с методом [9] не проводилось, однако визуально качество автоматов, построенных с помощью метода [9], было ниже.

Рассмотрим подробнее процесс построения автоматов с помощью предложенного метода. Для задачи зависания строился автомат с двумя состояниями при использовании всего одного предиката («прошло 25 секунд»), а маски значимости вещественных переменных не использовались. За счет этого пространство поиска становилось малым (32 особи) и процесс оптимизации полностью перебирал его за несколько секунд. Состояния построенного автомата соответствовали взлету и зависанию соответственно. Для задачи полета автомат строился с более реалистичными параметрами: три состояния и девять предикатов. Время построения автомата составляло около минуты. Как и ожидалось, состояния построенного автомата соответствовали взлету, полету вперед и посадке. На рис. 14 приведен вертолет, выполняющий полет под управлением автомата.

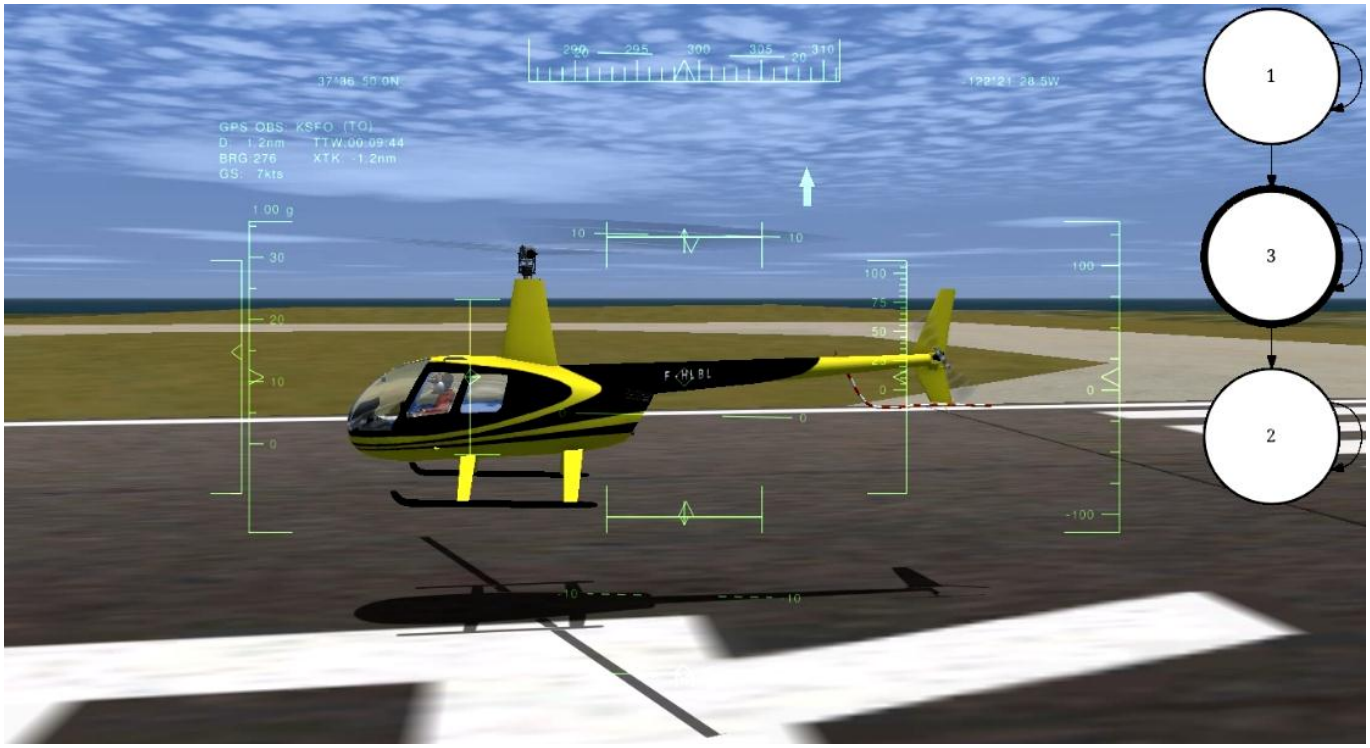

Рис. 14. Модель вертолета выполняет полет под управлением автомата (снимок экрана)

Качество выполнения некоторых элементов задач (таких как подъем и посадка) автоматами не является высоким. Предположительно, причина этому – недостаточно качественное их выполнение в тестах и малое число самих тестов. Примеры видеозаписей работы некоторых из построенных автоматов доступны по следующим web-адресам:

- зависание: http://youtu.be/iFa6FOpNdSI;
- полет: http://youtu.be/vPIaLgzhJgk.

Выводы по главе 4

1. Исследована эффективность использования для решения задачи генетического и муравьиного алгоритмов. Муравьиный алгоритм показал более высокую производительность.
2. Обоснована целесообразность использования масок значимости вещественных переменных.
3. Предложенный метод построения автоматов опробован на нескольких наборах тестов, описывающих выполнения моделями самолета и вертолета различных элементов полета.
4. Проведено сравнение метода с его прототипом, которое показало эффективность предложенного метода.

ЗАКЛЮЧЕНИЕ

В настоящей работе описан метод построения управляющих конечных автоматов по обучающим примерам, улучшающий качество ранее разработанного метода [9]. Метод позволяет строить автоматы, использующие для выработки выходных воздействий вещественные переменные. Такие автоматы являются гибридными системами: наборами непрерывных регуляторов, переключающихся при выполнении охранных условий. Алгоритм автоматизированной расстановки выходных воздействий, предложенный в [9], адаптирован для работы с вещественными переменными.

В работе рассматриваются две функции приспособленности. Одна из них ранее не использовалась при автоматизированном построении управляющих автоматов и в некоторых случаях позволяет улучшить качество метода.

Предложенный метод опробован на пяти наборах тестов, описывающих выполнение различных элементов полета моделями летательных аппаратов – самолета и вертолета. Результаты экспериментальных исследований говорят о том, что метод превосходит свой прототип по качеству.

В работе не был рассмотрен случай наличия у объекта управления дискретных выходных воздействий, так как выполнение рассмотренных элементов полета их не требовало. Такие воздействия возможно учитывать, используя для их хранения сокращенные таблицы, как и для функции переходов. Для их автоматизированной расстановки на каркасе можно использовать алгоритм, предложенный в [9].

По теме настоящей выпускной квалификационной работы сделаны доклады на II Всероссийском конгрессе молодых ученых (НИУ ИТМО), на конференции СПИСОК 2013 (СПбГУ), а также принят стендовый доклад на конференцию GECCO 2013.

СПИСОК ЛИТЕРАТУРЫ

1. *Поликарпова Н.И., Шалыто А.А.* Автоматное программирование. СПб.: Питер, 2011. – 176 с.
2. *Вельдер С.Э., Лукин М.А., Шалыто А.А., Яминов Б.Р.* Верификация автоматных программ. СПб: Наука, 2011. – 242 с.
3. *Koza J.R.* Genetic programming: on the programming of computers by means of natural selection. Cambridge: MIT Press, 1992.
4. *Курейчик В.М.* Генетические алгоритмы. Состояние. Проблемы. Перспективы // Известия РАН. Теория и системы управления. 1999. № 1. С. 144–160.
5. *Гудвин Г.К., Гребе С.Ф., Сальгадо М.Э.* Проектирование систем управления. М.: Бином. Лаборатория знаний, 2004. – 911 с.
6. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning: Data Mining, Inference, and Prediction. – 2nd ed. – Springer-Verlag, 2009. – 746 p.
7. *Coates A., Abbeel P., Ng A.* Apprenticeship Learning for Helicopter Control // Communications of the ACM, 2009, V. 52, № 7.
8. *Abbeel P., Coates A., Ng A.* Autonomous Helicopter Aerobatics through Apprenticeship Learning // The International Journal of Robotics Research, SAGE Publications, June 2010.
9. *Александров А.В., Казаков С.В., Сергушичев А.А., Царев Ф.Н., Шалыто А.А.* Применение эволюционного программирования на основе обучающих примеров для генерации конечных автоматов, управляющих объектами со сложным поведением // Известия РАН. Теория и системы управления. 2013. № 3, С. 85–100.
10. *Бужинский И.П., Ульянцев В.И.* Применение муравьиных алгоритмов для построения автоматов управления системами со сложным поведением на основе обучающих примеров / Сборник докладов XV Международной конференции по мягким вычислениям и измерениям (SCM 2012). СПб: СПбГЭТУ «ЛЭТИ», 2012. Т. 1, С. 250–253.
11. *Под ред. Н.Д. Егупова.* Методы робастного, нейро-нечеткого и адаптивного управления: Учебник. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. – 744 с.

12. *Mataric M.* Behavior-Based Robotics // MIT Encyclopedia of Cognitive Sciences, MIT Press, April 1999, P. 74–77.
13. *Henzinger T.* The Theory of Hybrid Automata / Proceedings of the Eleventh Annual IEEE Symposium on Logic in Computer Science (LICS), 1996, P. 278–292.
14. *Egerstedt M., Hu X.* A Hybrid Control Approach to Action Coordination for Mobile Robots // Automatica, 2002, V. 38, № 1, P. 125–130.
15. *Saripalli S., Montgomery J., Sukhatme G.* Visually-Guided Landing of an Unmanned Aerial Vehicle // IEEE Transactions on Robotics and Automation, 2003, V. 19, № 3, P. 371–381.
16. *Pomerleau D.* Efficient Training of Artificial Neural Networks for Autonomous Navigation // Neural Computation, 1991, V. 3, № 1, P. 88–97.
17. *Montgomery J.* Learning helicopter control through "Teaching by showing", PhD Thesis, University of South California, 1999.
18. *Dempster A., Laird N., Rubin D.* Maximum likelihood from incomplete data via the EM algorithm // Journal of the Royal Statistical Society, 1977, V. 39, № 1, P. 1–38.
19. *Kwakernaak H., Sivan R.* Linear Optimal Control Systems. Wiley-Interscience, 1972.
20. *Клебан В.О., Шалыто А.А.* Разработка системы управления малоразмерным вертолетом // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики, 2011. № 2 (72). С. 12–16.
21. *Царев Ф.Н., Шалыто А.А.* Применение генетического программирования для генерации автоматов в задаче об «умном муравье» / Труды IV-ой Международной научно-практической конференции «Интегрированные модели и мягкие вычисления в искусственном интеллекте». Том 2. М.: Физматлит. 2007. С. 590–597.
22. *Поликарпова Н.И., Точилин В.Н., Шалыто А.А.* Метод сокращенных таблиц для генерации автоматов с большим числом входных переменных на основе генетического программирования // Известия РАН. Теория и системы управления, 2010. № 2. С. 100–117.

23. *Царев Ф.Н.* Метод построения управляющих конечных автоматов на основе тестовых примеров с помощью генетического программирования. // Информационно-управляющие системы, 2010, № 5, С. 31–36.
24. *Казаков С.В.* Разработка метода построения конечных автоматов верхнего уровня для управления моделью беспилотного самолета на основе обучающих примеров. Бакалаврская работа, СПбГУ ИТМО, 2011.
25. FlightGear [Электронный ресурс]. Режим доступа http://www.flightgear.org/ свободный. Яз. англ. (дата обращения 02.06.13).
26. *Данилов В.Р.* Метод представления автоматов деревьями решений для использования в генетическом программировании // Научно-технический вестник СПбГУ ИТМО, 2008. Выпуск 53. Автоматное программирование. С. 103–108.
27. *Chivilikhin D., Ulyantsev V.* Learning Finite-State Machines with Ant Colony Optimization // Lecture Notes in Computer Science, 2012. V. 7461/2012. P. 268–275.
28. *Dorigo M.* Optimization, Learning and Natural Algorithms. PhD thesis, Politecnico di Milano. Italy. 1992.
29. *Dorigo M., Stützle T.* Ant Colony Optimization. MIT Press, US. 2004.

ПРИЛОЖЕНИЕ А. ПРИМЕР ПОСТРОЕННОГО АВТОМАТА

В настоящем приложении приведен пример одного из автоматов, сгенерированных предлагаемым в работе методом. Автомат построен для фигуры пилотажа «разворот» с использованием ФП f_{Δ}, при этом ее значение для данного автомата равно 0,99742. Число состояний автомата равно трем. Для начала приведем наборы предикатов и вещественных переменных, использованных для построения автомата. Для этого введем следующие обозначения:

- α, β, γ – углы крена, тангажа и курса соответственно в текущий момент времени;
- γ_{change} – расстояние на окружности от угла γ до его значения в начальный момент времени, от $-180°$ до $180°$;
- $\text{lstep}_{a,b}(x) = \begin{cases} 0, & x \le a \\ \frac{x-a}{b-a}, & a \le x \le b \\ 1, & x \ge b \end{cases}$;
- $\text{rstep}_{a,b}(x) = 1 - \text{lstep}_{a,b}(x)$;
- $\text{step}_{a,b,c,d}(x) = \min\left(\text{lstep}_{a,b}(x), \text{rstep}_{c,d}(x)\right)$.

Используемые при построении автомата предикаты приведены в табл. А.1.

Табл. А.1. Использованные при построении автомата предикаты

Предикат	**Условие**	**Предикат**	**Условие**
p_1	$\lvert\alpha\rvert < 3°$	p_8	$\gamma_{\text{change}} < 30°$
p_2	$\lvert\alpha\rvert < 6°$	p_9	$\gamma_{\text{change}} < 60°$
p_3	$\lvert\alpha\rvert < 12°$	p_{10}	$\gamma_{\text{change}} < 120°$
p_4	$\lvert\alpha\rvert < 20°$	p_{11}	$\gamma_{\text{change}} < 150°$
p_5	$\gamma_{\text{change}} < 5°$	p_{12}	$\gamma_{\text{change}} < 160°$
p_6	$\gamma_{\text{change}} < 10°$	p_{13}	$\gamma_{\text{change}} < 170°$
p_7	$\gamma_{\text{change}} < 20°$	p_{14}	$\gamma_{\text{change}} < 175°$

В табл. А.2 приведены использованные вещественные переменные. Для первого и третьего управляющих параметров использовались одинаковые наборы

переменных, чтобы сделать возможным совмещение для них масок значимости (см. конец раздела 4.2).

Табл. А.2. Использованные при построении автомата вещественные переменные

Управляющие параметры (положения органов управления)	**Переменные**
1 – элероны; 3 – руль направления	$v_{1,1} = v_{3,1} = 1$
	$v_{1,2} = v_{3,2} = \alpha$
	$v_{1,3} = v_{3,3} = \alpha - 59{,}5°$
	$v_{1,4} = v_{3,4} = \mathrm{d}\alpha/\mathrm{d}t$
	$v_{1,5} = v_{3,5} = \mathrm{rstep}_{0°,30°}(\gamma)$
	$v_{1,6} = v_{3,6} = \mathrm{lstep}_{150°,180°}(\gamma)$
	$v_{1,7} = v_{3,7} = \mathrm{step}_{0°,30°,150°,180°}(\gamma)$
2 – руль высоты	$v_{2,1} = 1$
	$v_{2,2} = \beta$
	$v_{2,3} = \beta - 2{,}1°$
	$v_{2,4} = \mathrm{d}\beta/\mathrm{d}t$
	$v_{2,5} = \mathrm{lstep}_{0°,30°}(\gamma)$
	$v_{2,6} = \mathrm{rstep}_{150°,180°}(\gamma)$
	$v_{2,7} = \mathrm{step}_{0°,30°,150°,180°}(\gamma)$

В каждом состоянии автомата находится маска значимости предикатов с двумя значимыми предикатами (для состояния 1 фактически оказался значимым только один предикат p_{13}). На рис. А.1 приведен граф переходов полученного автомата. Условия на переходах графа представляют собой результат упрощения булевых формул, заданных масками значимости предикатов.

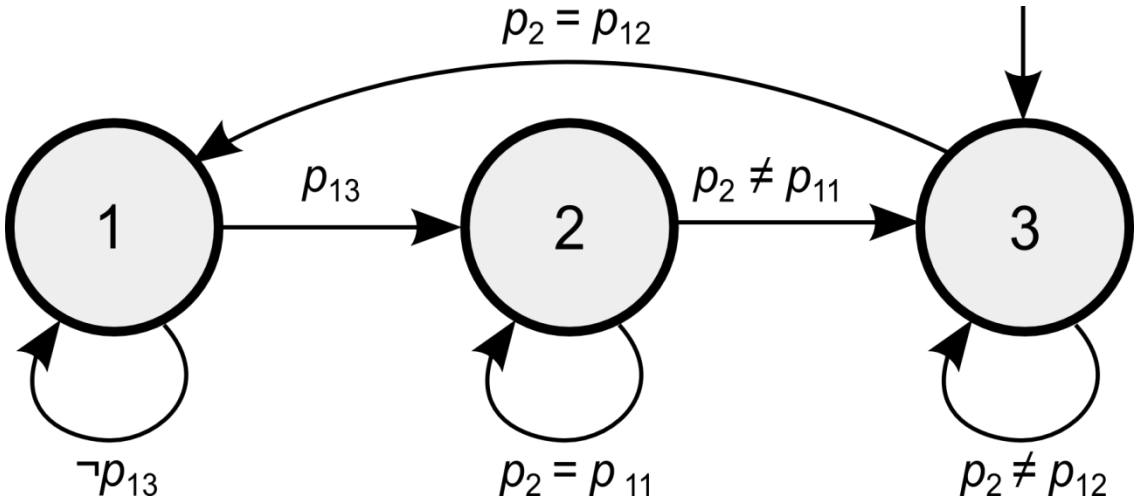

Рис. А.1. Граф переходов построенного автомата

Далее, в табл. А.3 приведены все значимые предикаты и переменные для каждого состояния автомата.

Табл. А.3. Значимые предикаты и вещественные переменные в построенном автомате

Состояние	**Значимые предикаты**	**Значимые вещественные переменные**
1	p_{10}, p_{13}	$v_{1,3}, v_{1,4}, v_{1,5}, v_{1,6}$; $v_{2,1}, v_{2,2}, v_{2,4}, v_{2,5}, v_{2,6}$; $v_{3,3}, v_{3,4}, v_{3,5}, v_{3,6}$
2	p_2, p_{11}	$v_{1,2}, v_{1,3}, v_{1,4}, v_{1,7}$; $v_{2,2}, v_{2,4}, v_{2,5}, v_{2,6}, v_{2,7}$; $v_{1,2}, v_{1,3}, v_{1,4}, v_{1,7}$
3	p_2, p_{12}	$v_{1,2}, v_{1,3}, v_{1,4}, v_{1,7}$; $v_{2,1}, v_{2,2}, v_{2,3}, v_{2,4}, v_{2,7}$; $v_{1,2}, v_{1,3}, v_{1,4}, v_{1,7}$

Наконец, в табл. А.4 приведем найденные алгоритмом расстановки выходных воздействий коэффициенты $r_{s,j,i}$, определяющие функцию выходов автомата. Числа, не приведенные в таблице, заведомо равны нулю из-за ограничений, наложенных масками значимости.

Табл. А.4. Значение коэффициентов $r_{s,j,i}$ в построенном автомате

Состояние s	**Управляющий параметр** j	**Числа** $r_{s,j,i}$
1	1	$r_{1,1,3} = -0{,}000172;\ r_{1,1,4} = -0{,}007213;$ $r_{1,1,5} = -0{,}001151;\ r_{1,1,6} = -0{,}002007.$
	2	$r_{1,2,1} = -0{,}008470;\ r_{1,2,2} = 0{,}004266;$ $r_{1,2,4} = 0{,}018104;\ r_{1,2,5} = 0{,}003241$ $r_{1,2,6} = 0{,}001271.$
	3	$r_{1,3,3} = -0{,}000086;\ r_{1,3,4} = -0{,}003607;$ $r_{1,3,5} = -0{,}000575;\ r_{1,3,6} = -0{,}001003.$
2	1	$r_{2,1,2} = -0{,}000368;\ r_{2,1,3} = -0{,}000073;$ $r_{2,1,4} = -0{,}006240;\ r_{2,1,7} = 0{,}005609.$
	2	$r_{2,2,2} = 0{,}001624;\ r_{2,2,4} = -0{,}001943;$ $r_{2,2,5} = -0{,}005311;\ r_{2,2,6} = -0{,}003180;$ $r_{2,2,7} = 0{,}001603.$
	3	$r_{2,3,2} = -0{,}000184;\ r_{2,3,3} = -0{,}000036;$ $r_{2,3,4} = -0{,}003120;\ r_{2,3,7} = 0{,}002805.$
3	1	$r_{3,1,2} = -0{,}001766;\ r_{3,1,3} = -0{,}000014;$ $r_{3,1,4} = -0{,}020199;\ r_{3,1,7} = 0{,}000111.$
	2	$r_{3,2,1} = 0{,}000000;\ r_{3,2,2} = -0{,}001658;$ $r_{3,2,3} = 0{,}004775;\ r_{3,2,4} = 0{,}012791;$ $r_{3,2,7} = 0{,}014268.$
	3	$r_{3,3,2} = -0{,}000883;\ r_{3,3,3} = -0{,}000007;$ $r_{3,3,4} = -0{,}010100;\ r_{3,3,7} = 0{,}000055.$