МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
(государственный университет)

ФАКУЛЬТЕТ УПРАВЛЕНИЯ И ПРИКЛАДНОЙ МАТЕМАТИКИ

Базовая организация - ИПМ им.М.В.Келдыша РАН

Кафедра «Математическое моделирование и прикладная математика»
специализация «Управление динамическими системами»

Квалификационная работа на соискание степени магистра
по направлению 03.04.01 «Прикладные математика и физика»
магистерская программа «Управление динамическими системами»

Исследование алгоритмов управления макетом микроспутника с нежесткими элементами конструкции на аэродинамическом столе

Студент группы 272в
Шачков Марк Олегович

Научный руководитель:
к. ф.-м. н., доц.
Иванов Данил Сергеевич

Москва, 2018

Оглавление

Аннотация

В работе приведено описание процесса разработки комплекса программного обеспечения для постановки экспериментов на аэродинамическом столе. Описаны инструменты и технологии использованные в процессе проектирования и разработки, а также описана архитектура разработанного программного комплекса. В качестве демонстрации процесса взаимодействия с комплексом приведены примеры экспериментов с макетами микроспутников: определение характеристик двигателей, удержание макета в точке, определение возмущений действующих на макет, движение макета по траектории.

Введение

Процесс решения прикладной задачи управления динамической системой состоит из двух больших этапов - математическое моделирование и непосредственная техническая реализация. Как показывает практика, на стыке этих этапов лежит область в которой часто возникают проблемы, связанные с недостаточной точностью моделирования вызванной нехваткой ресурсов (как человеческих, так и материальных). Решением такого рода проблем является более тесная интеграция процессов моделирования, разработки и технической реализации. Одним из способов является создание имитационных стендовых комплексов, позволяющих, используя применяемые в конечном продукте аппаратные и программные решения, воссоздать (в некотором приближении) динамику управляемой системы.

Примером описанного выше подхода является применение в задачах исследования и разработки алгоритмов управления положением и ориентацией малых космических аппаратов стендовых комплексов типа аэростол. Механизм работы таких стендов заключается в обеспечении движения макетов по поверхности ровного стола практически без сопротивления за счёт создания воздушной подушки между поверхностью стола и основанием макета. Такая подушка может быть создана как при помощи постоянного потока воздуха через перфорированную поверхность стола, так и поддувом воздуха под основание макета из баллонов со сжатым газом расположенных непосредственно на макете.

Установленный в ИПМ им.М.В.Келдыша стенд КОСМОС (Рисунки 1, 2) был одним из первых созданных в России частной космической компанией ООО “Спутникс” (аналогичной конструкции стенд установлен в образовательном центре “Сириус”). Этот стенд состоит из двух перфорированных металлических плит, составляющих поверхность стола; опор, обеспечивающих выравнивание плит относительно горизонта; промышленного вентилятора, обеспечивающего подачу воздуха в системы проводящих каналов. Конструкция стенда не требует применения системы подачи воздуха

Рис. 1 — Стенд КОСМОС.

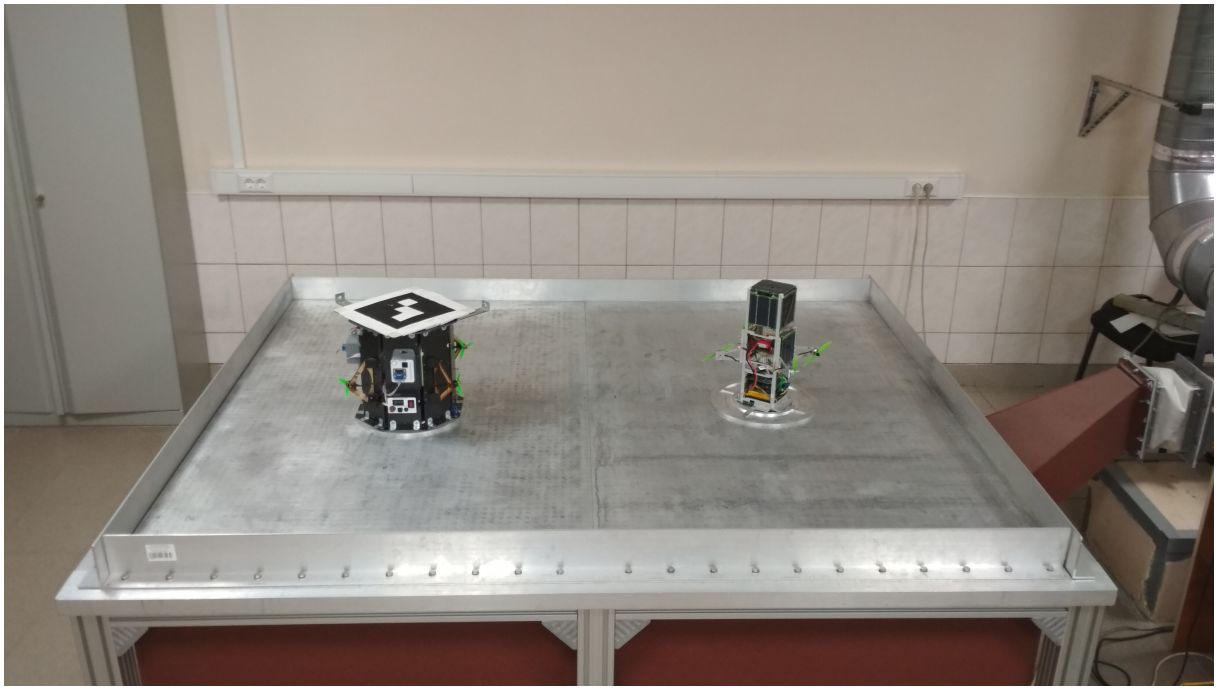

Рис. 2 — Стенд КОСМОС.

устанавливаемой непосредственно на борту макетов, тем самым исключая потребность в постоянном наполнении или смене баллонов с воздухом. Более подробное описание конструкции стенда, а также исчерпывающий обзор аналогов можно найти в работе [1]

В качестве основных направлений применения описанного стендового комплекса можно выделить следующие:

- структурная и параметрическая идентификация математических моделей,
- постановка и решение задач управления, синтез управляющих законов,
- построение и реализация наблюдателей для оценки вектора состояния системы,
- разработка и тестирование программной реализации алгоритмов управления.

Из приведённого списка становится понятно, что использование стенда КОСМОС практически невозможно, без программного комплекса осуществляющего сбор информации о состоянии стенда и обеспечивающего управление макетами микроспутников.

Глава 1. Программный комплекс для проведения экспериментов на аэродинамическом столе

Для того, чтобы лучше понять мотивацию к разработке описываемого программного комплекса необходимо представить типичный сценарий того, как исследователь взаимодействовал со стендом КОСМОС в его оригинальной поставке:

1. Определяется цель эксперимента. В качестве примеров можно привести: исследование характеристик стенда, идентификация математической модели движения макета, изучение качественных характеристик того или иного алгоритма управления.

2. Если цель подразумевает активное управление движением макетов (используя бортовые актуаторы), то разрабатывается соответствующее программное обеспечение.

3. В случае необходимости, обеспечивается система независимых (внешних) измерений, позволяющая определить и записать положение и ориентацию макетов во время проведения эксперимента.

4. Если того требует эксперимент, необходимо обеспечить передачу координат макетов непосредственно на борт для их учёта в алгоритме управления.

Описанный выше подход весьма неэффективен, так как существенная часть времени уходит на написание программного кода, обеспечивающего технические аспекты проведения эксперимента. Так, например, система независимых измерений может быть реализованна в виде отдельного модуля, что позволит исследователю пропустить пункты 3 и 4 в приведённом выше сценарии. Первая версия такой системы, подробное описание которой можно найти в работе [2], была ранее разработана непосредственно в ИПМ им.М.В.Келдыша.

Для оптимизации описанного выше сценария, необходимо отметить факт того, что меняющейся от эксперимента к эксперименту частью является лишь реализация алгоритмов управления движением, что говорит о возможности создания программного интерфейса, позволяющего исследователю абстрагироваться от технических аспектов проведения эксперимента и существенно сэкономить время отводимое на пункт 2.

В этой главе приведено описание разработанного программного интерфейса и модулей обеспечивающих его функционирование. Отдельно описан модуль обеспечивающий работу системы независимых измерений.

1.1. Характеристики стенда КОСМОС

Прежде чем перейти непосредственно к описанию программного комплекса, необходимо рассказать о характеристиках компонент стенда так как они во многом определяют требования к программному комплексу. Первоначально, стенд КОСМОС, установленный в ИПМ им.М.В.Келдыша, содержал всего один макет, характеристики которого приведены в таблице 1.1.

Аппаратная платформа	Raspberry Pi 2 Model B
Программная платформа	GNU/Linux (Raspbian)
Сенсоры	Акселерометр, датчик угловой скорости, магнетометр, видеокамера
Актуаторы	Маховик
Способ подключения сенсоров и актуаторов	Шина RS-485
Средства связи	Wi-Fi модуль

Таблица 1.1 — Характеристики макета первой версии.

Из приведённых выше характеристик становится понятно, что при помощи установленных на борту актуаторов управление возможно лишь ориентацией макета. Для того, чтобы получить возможность решать исследовательские задачи предполагающие управление положением макета, на борт, в горизонтальной плоскости, были установлены 4 винтомоторные группы (на базе бесколлекторных двигателей постоянного тока). Со схемой

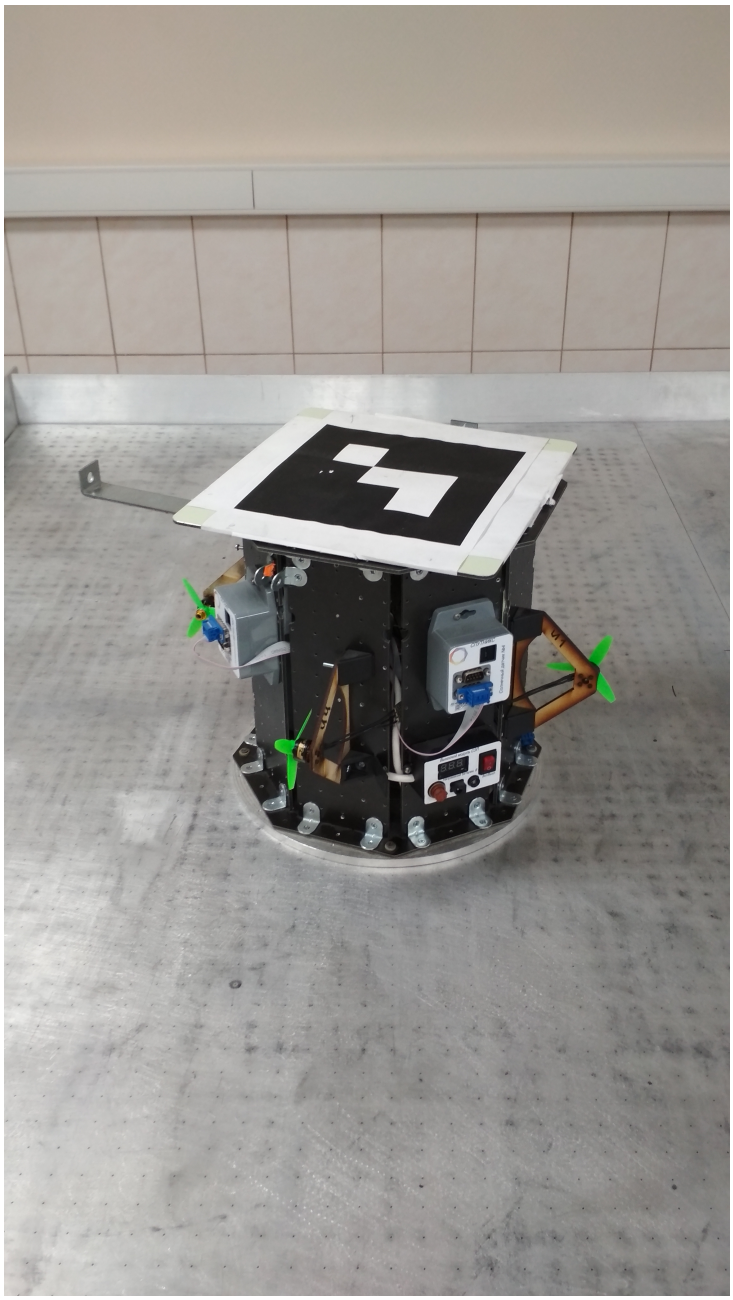

Рис. 1.1 — Макет первой модификации.

установки ВМГ можно ознакомиться на рисунке 1.2, стрелочками обозначены направления и точки приложения управляющих сил, создаваемых каждой из ВМГ.

Для управления двигателями были установленны электронные контроллеры скорости (ESC), принимающие командный шим сигнал. Так как используемый в качестве бортового компьютера Raspberry Pi 2 Model B содержит лишь два аппаратных канала для генерации шим сигнала, то необхо-

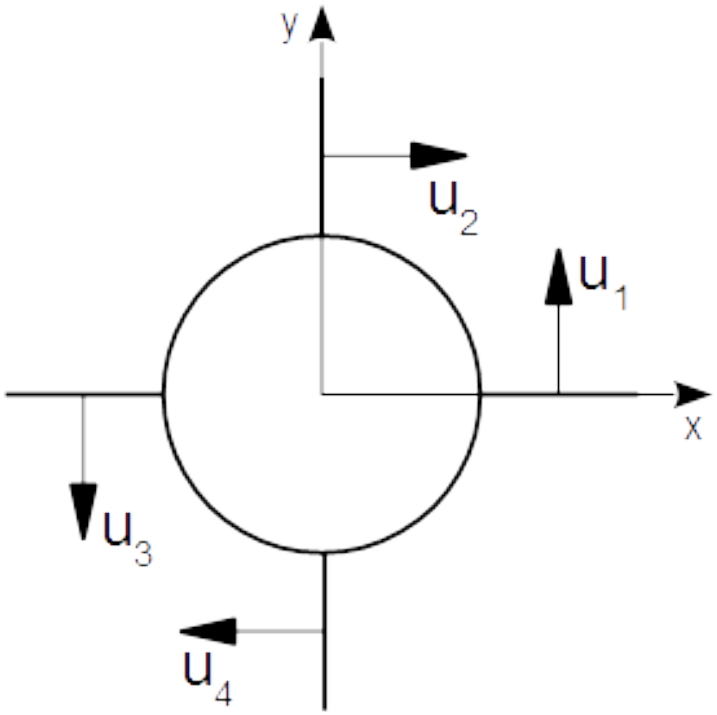

Рис. 1.2 — Точки приложения и направления управляющих сил.

димо было решить вопрос программной реализации, с учетом ограничений как процессора, так и ОС. В качестве решения была выбрана библиотека pigpio [14]. Реализация эффективной генерации шим сигнала в этой библиотеке основана на использовании DMA (direct memory access):

1. В оперативной памяти формируется специальным образом подготовленный буфер.

2. Настраивается канал переодической посылки содержимого этого буфера (через DMA) на адрес отображённого в память физического порта процессора.

3. Для изменения скваженности шим сигнала меняется содержимое буфера.

Учтя недоработку в первой модификации макета, компания “Спутникс” разработала вторую модификацию, исполненную в виде 3U кубсата. Вторая модификация собрана на базе модульной системы применяемой компанией для создания кубсатов предназначенных непосредственно для

полётов в космос. Характеристики второй модификации приведены в таблице 1.2.

Аппаратная платформа	Raspberry Pi Compute Module 3
Программная платформа	GNU/Linux (Raspbian)
Сенсоры	Акселерометр, датчик угловой скорости, магнетометр
Актуаторы	4 ВМГ
Способ подключения сенсоров и актуаторов	Шина CAN
Средства связи	Wi-Fi модуль и УКВ радио-модуль

Таблица 1.2 — Характеристики макета второй версии.

Основным отличием от первой доработанной модификации является способ подключения сенсоров и актуаторов - CAN шина. Таким образом, любая из ранее разработанных программ управления движением макета первой модификации требует учёта аппаратных изменений. Решение этой проблемы в общем виде будет приведено ниже.

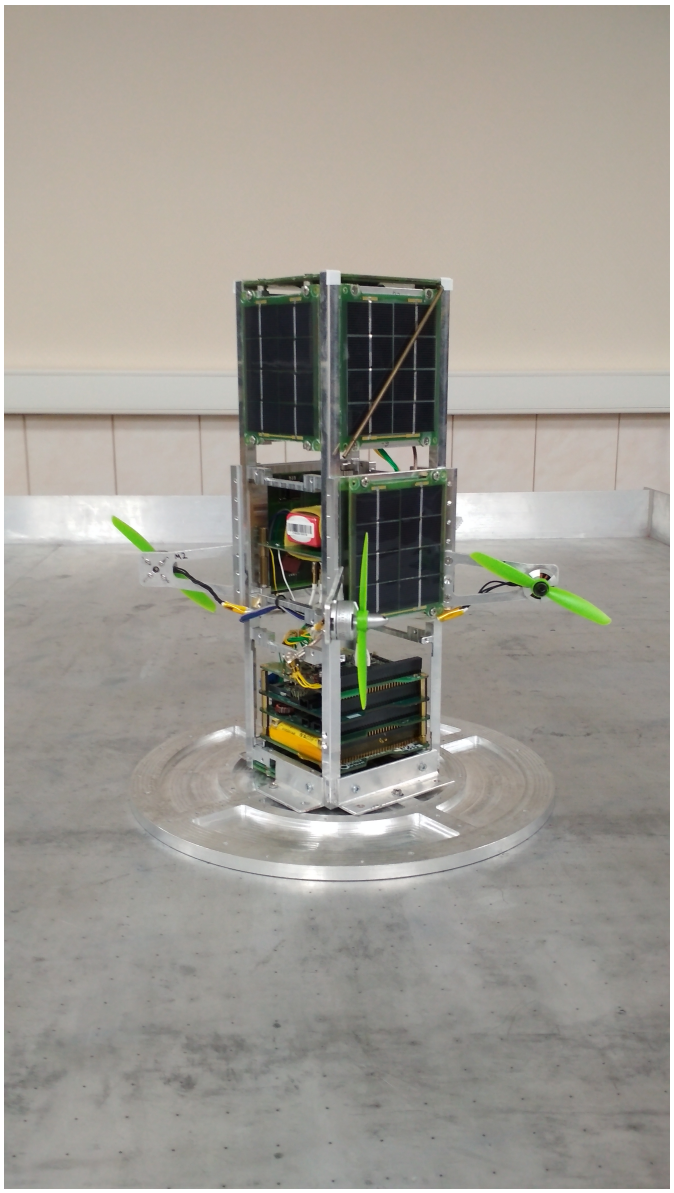

Рис. 1.3 — Макет второй модификации.

1.2. Структура программного комплекса и используемые инструменты

Исходя из структуры стенда, описанной ранее в тексте логичным является декомпозиция программного комплекса на два модуля:

- Модуль sat - программа исполняемая на борту макетов, отвечающая за работу функции управления движением.
- Модуль station - программа исполняемая на ПК, являющимся терминалом управления ходом эксперимента, отвечающая за работу си-

стемы определения положения макетов и обеспечивающая передачу, индикацию и сохранение состояния эксперимента.

Более подробное описание модулей будет приведено ниже. Здесь же стоит отметить используемые технологии и инструменты.

В качестве основного языка для разработки был выбран C++11 (ISO/IEC 14882:2011), как актуальная и наиболее поддерживаемая версия языка C++. Выбор C++ обусловлен с одной стороны, необходимостью низкоуровневой работы с компонентами ОС в модуле sat, с другой стороны, программным интерфейсом библиотек и фреймворков используемых в модуле station.

В качестве среды разработки была использована IDE Clion, в связи с ее кроссплатформеностью и использованием удобной проектной модели на основе CMake. Отдельно стоит отметить весьма пригодившийся функционал удалённой отладки на базе клиент-серверного отладчика GDB. Сборка модуля sat производилась при помощи кросс-компиляции тулчейном для платформы Raspberry Pi [15].

1.3. Подсистема определения положения макетов

Отдельно необходимо описать подсистему определения положений макетов. Развивая результаты работы [2], было принято решение использовать подход основанный на анализе видеоизображения получаемого с камеры, установленной над поверхностью стола, с целью поиска, выделения и обработки реперных точек жёстко связанных с макетом. Определив координаты этих точек в плоскости кадра, зная их расположение на макете, а также зная положение камеры относительно поверхности стола (определённое в результате решения задачи Perspective-n-Point, постановку и один из методов можно найти в [4]), можно определить положения и ориентации макетов в системе координат связанной с плоскостью стола.

Задача поиска реперных точек сильно упрощается в случае использование специально помещённых на макеты высококонтрастных меток. Библиотека ArUco ([5], [6], [13]) решает задачи как генерации таких меток (имея основной целью минимизацию вероятностей ошибок первого и вто-

рого рода при поиске таких меток в кадре), так и определения пиксельных координат угловых точек меток (пример изображения метки можно увидеть на рисунке 1.4). Именно эта библиотека была использована в составе программного комплекса для решения задачи определения положений и ориентаций макетов.

Для получения кадров видеопотока наиболее эффективным инструментом (в смысле простоты использования и кроссплатформености) является библиотека OpenCV [7]. Ее использование было обусловлено ещё и тем, что библиотека ArUco использует матричные типы данных из состава библиотеки OpenCV.

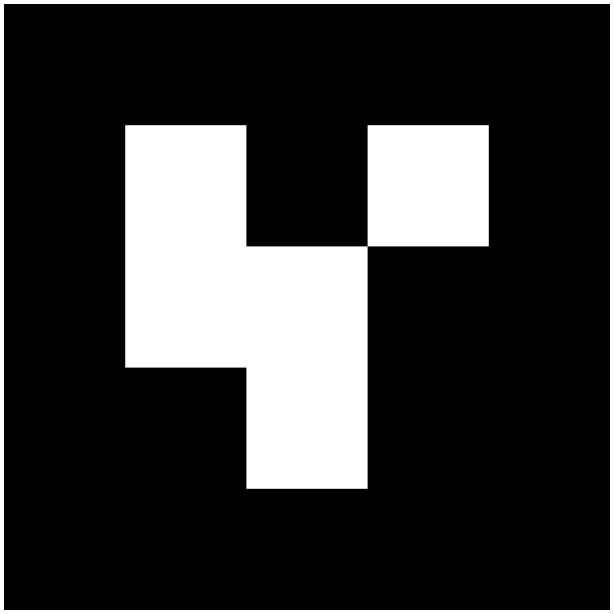

Рис. 1.4 — Метка сгенерированная библиотекой ArUco.

1.4. О передаче сообщений между макетами и станцией управления

Так как в качестве модулей связи в макетах выступают Wi-Fi модемы, то оптимальным (но не единственным) способом организации обмена информацией между макетами и станцией управления является подключение всех макетов и станции в роли клиентов к общей точке доступа (маршрутизатору). Отметим плюсы и минусы такого подхода:

+ Относительные простота интеграции и использования.

+ Возможность использования классических сетевых стеков операционных систем.

– Физические ограничения на максимальное количество эффективно обслуживаемых клиентов одной Wi-Fi сети.

– Необходимость введения системы адресации независимой от используемого оборудования.

Первый из указанных минусов решаем исключительно использованием более современных стандартов Wi-Fi и применения оборудования поддерживающего одновременную многоканальную передачу и прием (MIMO). Второй из указанных минусов может быть нивелирован либо явным заданием сетевых адресов устройств на стороне маршрутизатора (статические правила DHCP), либо использованием протокола mDNS и присвоением каждому из устройств уникального имени (sat1.local, sat2.local и т.п.). В работе был сделан выбор в пользу первого решения, однако второе решение не потребует большой доработки программного обеспечения и обеспечит большую гибкость.

Существует два типа данных, передаваемых в процессе работы стенда:

1. Данные с измеренными положениями и ориентациями макетов.

2. Сообщения, содержащие команды (или сигналы) изменения состояния и результаты работы различных алгоритмов.

Первый тип данных периодически генерируется на станции управления и имеет время актуальности равное периоду измерений. Таким образом, оптимальным способом доставки таких данных является протокол UDP, так как он обеспечивает минимальные издержки за счёт отсутствия гарантии доставки, что не критично для быстро инвалидируемых данных.

Второй тип данных не терпит потерь, что обуславливает выбор в пользу стека протоколов TCP/IP. Первая реализация системы передачи сообщений опиралась на использование сырых вызовов posix функций send/recv

(или их программных обёрток), однако столкнувшись со сложностями в разработке диссектора сетевого потока, выделяющего отдельные сообщения из набора байт передаваемых через TCP соединение, было принято решение использовать кроссплатформенную библиотеку асинхронной передачи сообщений nanomsg-NG [16](потомка библиотеки ZeroMQ во втором поколении).

1.5. Описание модуля sat

Модуль sat, выполняемый на макете, работает в режиме демона, стартуя при запуске ОС. Общая схема потока управления представлена на диаграмме 1.5

Основу модуля sat составляют два класса: Station и ControlHL. Класс Station является абстракцией инкапсулирующей логику обмена данными со станцией. Объект этого класса, создаваемый в единственном экземпляре, отвечает за установку и поддержание соединения со станцией, путем создания nng сокета типа pair0 и вызова функции nng_listen. Класс ControlHL выполняет роль подсистемы обеспечивающей работу пользовательской функции управления движением макета. Объект этого класса, создаваемый в единственном экземпляре, инициализирует подсистему взаимодействия с сенсорами и актуаторами макета. Методы этого класса позволяют запустить функцию управления движением в отдельном потоке. Для работы с потоками используется реализация стандарта POSIX Threads - библиотека pthreads.

Одним из компонентов системы, требующим реализации пользователя программного комплекса, является функция управления движением. Сигнатура этой функции (как и типы аргументов) описана в заголовочном файле предоставляемом исследователю. Эта функция должна принимать в качестве аргументов два указателя на объекты типов StationData и ControlBackend.

Класс StationData инкапсулирует данные о положении и ориентации макетов, полученные от станции управления, а также строковый поток и бестиповый буфер для передачи отладочной, информационной и прочей

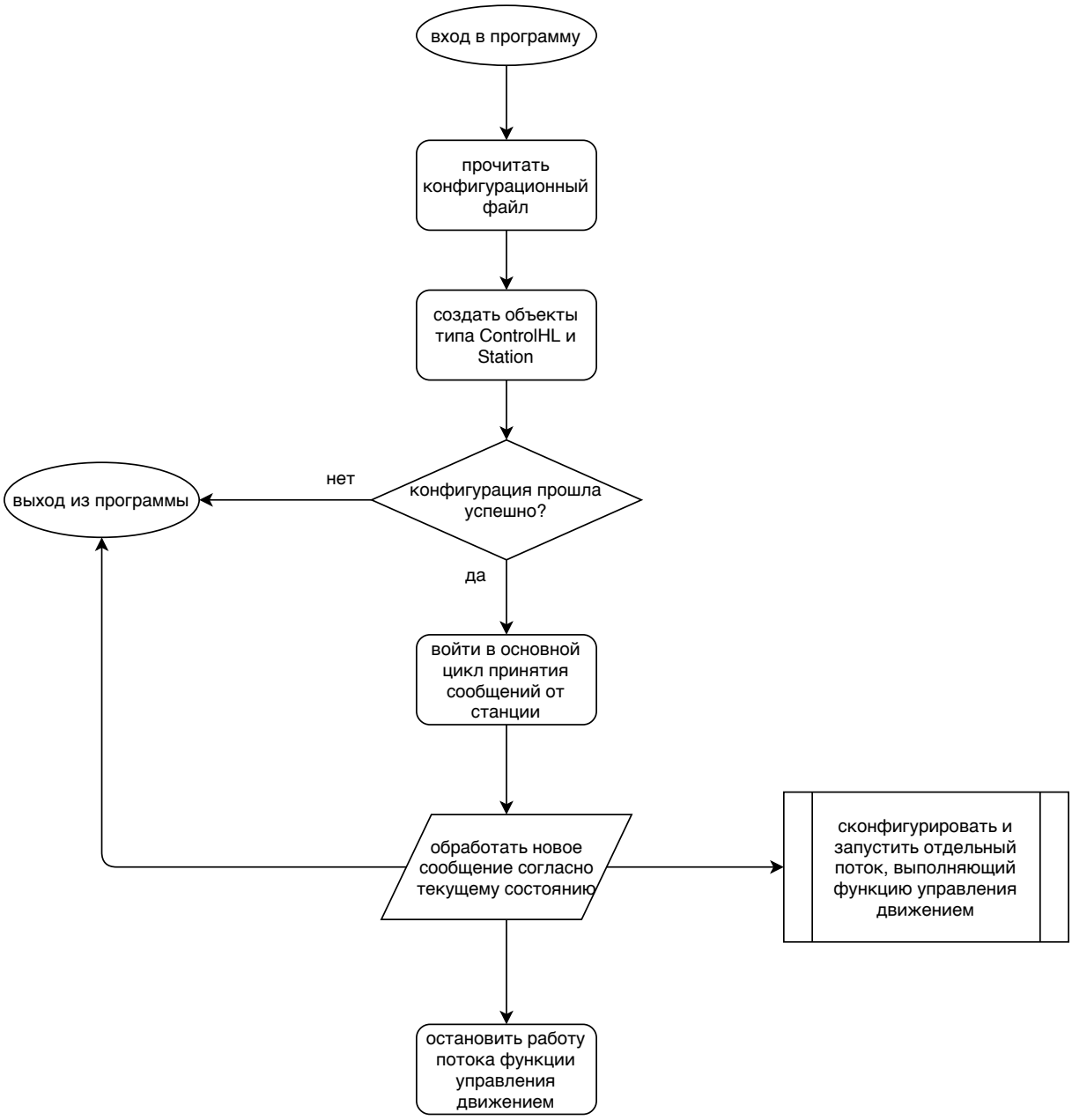

Рис. 1.5 — Диаграмма потока выполнения модуля sat.

информации на станцию управления. Поля этого класса обрабатываются каждый раз непосредственно перед вызовом функции управления движением, что обеспечивает предоставление пользовательской функции актуальной информации о состоянии макета.

Класс ControlBackend является абстрактным, т.е. содержит только виртуальные методы. Этот класс служит базовым, для классов реализующих логику инициализации и работы с сенсорами и актуаторами макета. В текущей версии программного комплекса включены две потомка класса

ControlBackend: ControlBackendPigpio и ControlBackendUnican. Эти классы используются в первой и второй версии макетов соответственно. Накладные расходы на вызовы виртуальных методов, при подобной схеме использования, практически отсутствуют, так как указатель передаваемый в пользовательскую функцию не меняет своё значение на всем протяжении работы программы, что позволяет предсказателю переходов процессора эффективно нагружать вычислительный конвейер.

Такая архитектура позволяет не останавливая работу демона, приняв сообщение от станции содержащее исходный текст функции управления движением, скомпилировать эту функцию в динамическую библиотеку и загрузить в память процесса указатель на актуальную реализацию используя функции dlopen() и dlsym().

В процессе проведения различных экспериментов была разработана библиотека стандартных функций управления движением, которая состоит из следующих примитивов:

- Вывод отладочной информации.
- Удержание макетом положения и ориентации.
- Движение макета по окружности.
- Идентификация характеристик ВМГ.
- Определение возмущений действующих на макет.

Эти функции могут быть использованы для отладки и настройки очередной версии макета (или всего стенда), путём указания в конфигурации эксперимента соответствующего параметра.

1.6. Описание модуля station

Одной из целей поставленных перед разработкой модуля станции управления экспериментом была кросслпатформенность программного обеспечения. Эта цель была достигнута (работа модуля проверена на операционных системах Mircosoft Windows 10 и GNU/Linux Ubuntu 16.04) бла-

годаря применению фреймворка Qt [17] для работы с графическим интерфейсом и многопоточностью. В силу объектно-ориентированной парадигмы языка C++, а так-же по причине использования абстракции асинхронных сигналов из состава Qt (модель signal-slot), привести единую диаграмму потока выполнения модуля station представляется проблематичным, но в качестве иллюстрации общей логики работы модуля приведён рисунок 1.6.

В качестве строительных блоков были использованы классы Sat и Experiment, являющиеся абстракциями над состоянием и логикой взаимодействия с макетами и ходом эксперимента соответственно.

Объекты класса Sat сопоставлены используемым в текущем эксперименте макетам, хранят в себе актуальную информацию о состоянии макетов, а также отвечают за поддержание соединения и коммуникацию с макетами. Также, в составе методов класса Sat, реализован фильтр Калмана, отвечающий за оценку скоростей макетов по данным измерений полученных из подсистемы определения положения макетов.

Объект класса Experiment, создаваемый в единственном экземпляре, отвечает за чтение конфигурационного файла эксперимента, выделение в отдельные потоки выполнения функций определения положений макетов и обмена сообщениями с макетами, а также осуществляет логирование текущего состояния эксперимента.

Одной из нереализованных частей функционала модуля station является графический конфигуратор эксперимента. Такой элемент интерфейса позволил бы как упростить процесс создания файла конфигурации, так и наглядно отобразить его состояние. Изображение окна графического интерфейса модуля station приведено на рисунке 1.7.

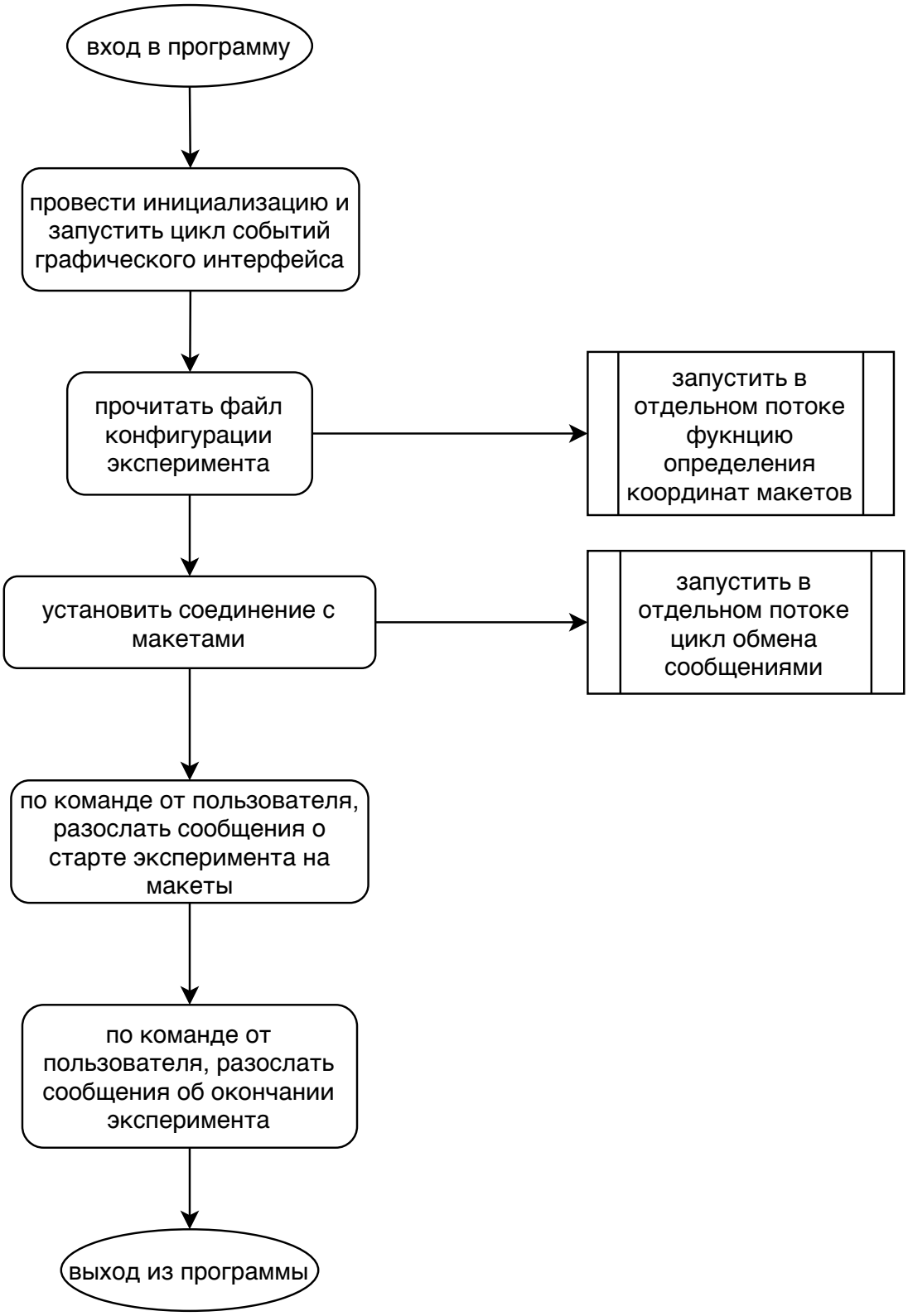

Рис. 1.6 — Диаграмма работы модуля station.

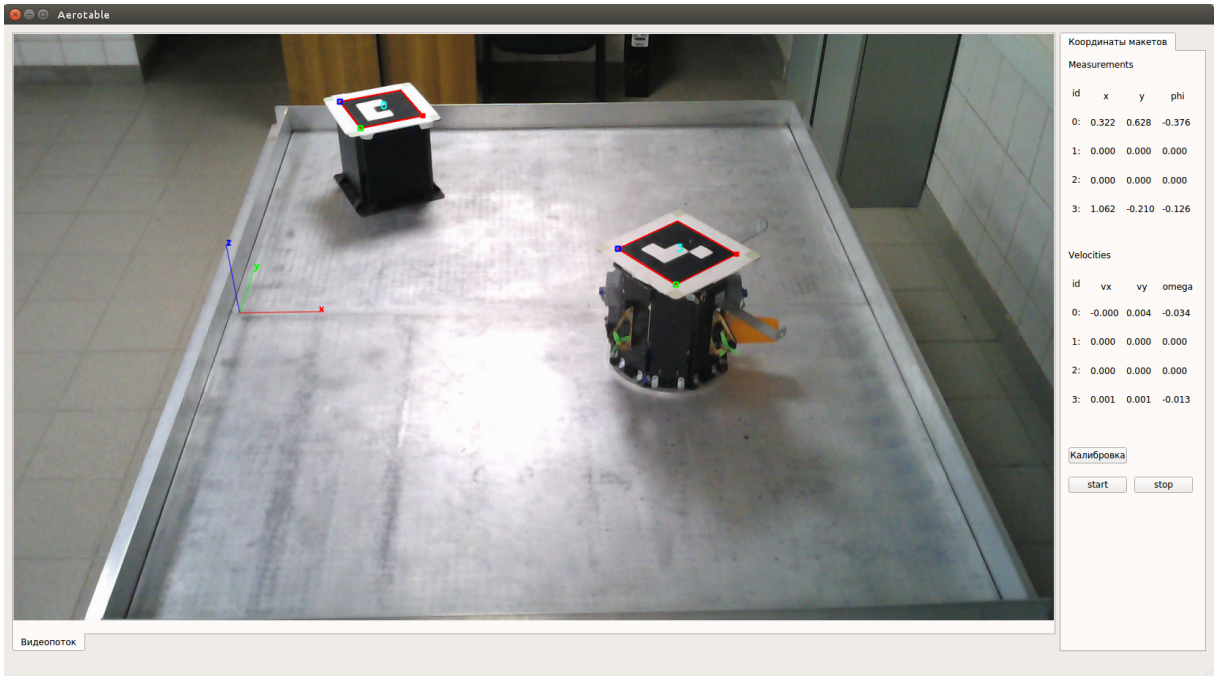

Рис. 1.7 — Графический интерфейс модуля station.

Глава 2. Исследование алгоритмов управления макетом микроспутника

Программный комплекс для взаимодействия со стендом КОСМОС, описанный в предыдущей главе, был разработан в целях упрощения процесса проведения исследований алгоритмов управления движением и ориентацией. В этой главе приведены некоторые из задач, решённых с использованием разработанного программного обеспечения.

2.1. Математическая модель движения макета микроспутника по поверхности аэростола

Одной из первых задач, решаемых в процессе разработки систем управления, является построение математической модели управляемого процесса. Моделированию подлежат как общая динамика системы, так и отдельные ее компоненты, такие как, например, сенсоры и управляющие механизмы.

В качестве основы математической модели движения макета микроспутника по поверхности аэростола были выбраны уравнения Ньютона-Эйлера 2.1, описывающие движение твёрдого тела под действием внешних сил и моментов:

$$\begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} \dot{\vec{v}}^b \\ \dot{\vec{\omega}}^b \end{bmatrix} + \begin{bmatrix} \vec{\omega}^b \times m\vec{v}^b \\ \vec{\omega}^b \times J\vec{\omega}^b \end{bmatrix} = \begin{bmatrix} \vec{F}^b \\ \vec{\tau}^b \end{bmatrix}, \tag{2.1}$$

где вектора $\vec{v}^b \in R^3$ и $\vec{\omega}^b \in R^3$ являются линейной и угловой скоростью центра масс тела в системе координат связанной с телом, матрица I — единичная матрица, m — масса тела, J — матрица тензора инерции, а $\vec{F}^b$ и $\vec{\tau}^b$ являются векторами сил и моментов соответственно.

Заметим, что в рассматриваемом случае реальный размер фазового пространства равен 6, так как движение макета происходит в плоскости и ненулевыми являются лишь первые две и последняя компоненты векторов v^b и ω^b соответственно.

Будем считать, что правая часть системы 2.1 представима в виде суммы:

$$\begin{aligned} \vec{F}^b &= \vec{F}^b_u(\vec{u}) + \vec{F}^b_{ext} \\ \vec{\tau}^b &= \vec{\tau}^b_u(\vec{u}) + \vec{\tau}^b_{ext} \end{aligned} \tag{2.2}$$

где $\vec{F}^b_u(\vec{u})$ и $\vec{\tau}^b_u(\vec{u})$ есть результирующие управляющие сила и момент ($\vec{u} \in [0;1]^4$ - вектор нормализованных управляющих величин, передаваемых в контроллеры скорости двигателей), а $\vec{F}^b_{ext}$ и $\vec{\tau}^b_{ext}$ являются внешними силой и моментом (исследование этих сил приведено ниже по тексту).

Таким образом сформулированная математическая модель позволяет, в предположении об обратимости функций $\vec{F}^b_u(\vec{u})$ и $\vec{\tau}^b_u(\vec{u})$ (и следующей из этого полной управляемости), применять технику линеаризации обратной связи, заключающуюся в выборе закона управления обращающего правую часть системы описывающей процесс и в последующем свободном конструировании желаемой динамики. Однако прежде чем перейти непосредственно к задачам синтеза управляющих законов, необходимо установить функции $\vec{F}^b_u(\vec{u})$ и $\vec{\tau}^b_u(\vec{u})$.

2.2. Идентификация параметров модели

Параметрами используемой математической модели являются инерционные характеристики макета (масса, положение центра масс, тензор инерции), а также функции $\vec{F}^b_u(\vec{u})$ и $\vec{\tau}^b_u(\vec{u})$ описывающие зависимость силы и момента создаваемыми актуаторами установленными на макет и непосредственно управляемыми величинами передаваемыми в контроллеры скорости двигателей.

Будем считать, что каждый из двигателей создаёт силу, направление которой остаётся постоянным. Величина же квадратично зависит от угловой скорости вращения пропеллера являющейся непосредственно регулируемой величиной (динамику бесколлекторного двигателя в рассматриваемой модели можно считать несущественной, в связи с отсутствием необходимости в больших величинах первой производной управляющих сил). Таким образом функции $\vec{F}^b_u(\vec{u})$ и $\vec{\tau}^b_u(\vec{u})$ представимы в следующем виде:

$$\begin{aligned}\vec{F}_u^b(\vec{u}) &= \sum_{i=1}^{4} \vec{f}_i c_i u_i^2 \\ \vec{\tau}_u^b(\vec{u}) &= \sum_{i=1}^{4} \vec{r}_i \times \vec{f}_i c_i u_i^2\end{aligned} \tag{2.3}$$

где $\vec{f}_i$ являются единичными векторами характеризующими направление сил создаваемых в результате вращения вентиляторов, c_i — скалярные коэффициенты зависящие от плотности воздуха, аэродинамических характеристик пропеллера и отношения частоты вращения двигателя к входному сигналу его контроллера, а $\vec{r}_i$ — радиус вектора точек приложения управляющих сил. Таким образом, параметрами модели подлежащими определению являются величины m, J, $\vec{f}_i$, c_i и $\vec{r}_i$.

Идентификация математических моделей типа серый ящик (моделей с известной структурой, но неизвестными параметрами) проводится методом, который можно описать следующими шагами:

1. Провести серию экспериментов с моделируемой системой, заключающихся в подаче различных входных сигналов и записи соответствующих выходных значений.

2. Имея множество последовательностей пар вида $\{\vec{u}(t_i), \vec{y}(t_i)\}_{i=1}^{m}$, где $\vec{u}(t_i)$ и $\vec{y}(t_i)$ являются входом и выходом системы в момент времени t_i, решить задачу поиска параметров модели, позволяющих добиться максимальной схожести результатов моделирования с экспериментом.

Для сбора данных была проведена серия экспериментов, заключающихся в подаче каждому из двигателей постоянного входного сигнала и записи последующего движения макета, используя систему определения положения и ориентации.

Второй шаг используемого метода идентификации заключается в решении следующей проблемы

$$\underset{m,J,\vec{f}_i,c_i,\vec{r}_i}{\text{minimize}} \quad \sum_i \left\| \vec{y}(t_i) - \tilde{\vec{y}}(t_i) \right\|^2 \tag{2.4}$$

где $\tilde{\vec{y}}(t_i)$ является выходом математической модели в момент t_i при входе $\vec{u}(t_i)$.

Этап проведения и сбора экспериментальных данных был выделен в одну из базовых функций управления движением. Этап обработки данных – решение минимизационной задачи, проводится на станции управления экспериментом с использованием библиотеки глобальной оптимизации pagmo [8], [12].

2.3. Определение возмущений действующих на макет

Заметим, что как для решения задачи описанной в предыдущем разделе, так и для решения описанных далее задач синтеза управляющих законов, необходимо знание функций $\vec{F}^b_{ext}$ и $\vec{\tau}^b_{ext}$. При создании стенда КОСМОС предполагалось, что конструкция сможет обеспечить равномерное и прямолинейное движение неуправляемого макета по поверхности стола, однако после введения стенда в эксплуатацию было выяснено, что на макет действуют существенно возмущающие его движение силы и моменты.

Природу возникновения этих сил с достоверной точностью установить представляется весьма проблематичным, поэтому приведём здесь лишь два предположения касательно явлений влияющих на движение макета:

1. Неравномерность скоростей истечения воздуха из отверстий на поверхности стола приводит к отклонению центра давления от центральной оси основания макета, что ведёт (в случае если центр масс не лежит на вертикальной оси проходящей через центр давления) к возникновению силы действующей на макет.

2. Существенные неровности на поверхности стола, ведущие как к уже упомянутой выше неравномерности скоростей воздуха в окрестности основания макета, так и к возможным соприкосновениям основания и поверхности стола.

Оба приведённых выше явления с трудом поддаются моделированию и идентификации, поэтому для уточнения модели движения макета был вы-

бран другой путь - экспериментальное определение внешних сил и моментов. В качестве справедливых были приняты следующие утверждения:

1. Внешние сила и момент являются стационарными - не зависят от времени.

2. Внешние сила и момент являются функциями положения макета на поверхности стола.

Справедливость этих предположения позволяет, задавшись некоторой формой аппроксимации функций и проведя серию измерений, экспериментально определить функции $\vec{F}^b_{ext}$ и $\vec{\tau}^b_{ext}$, с точностью удовлетворяющей потребностям дальнейших исследований.

Эксперимент, используемый в работе для определения функций $\vec{F}^b_{ext}$ и $\vec{\tau}^b_{ext}$, заключается в свободном движении макета по траекториям максимально охватывающим всю поверхность стола. Применив к измеренным при помощи системы определения положения и ориентации макетов фильтр Калмана, компоненты вектора состояния которого включают линейные и угловые ускорения макета, можно определить некоторую оценку значения силы и момента действующего на макет в точках измерений. Результатом проведения эксперимента являются массивы троек $\{x_i, y_i, F^x_i\}$, $\{x_i, y_i, F^y_i\}$ и $\{x_i, y_i, \tau^z_i\}$, где x_i и y_i координаты макета в системе связанной со столом, а F^x_i, F^y_i и τ^z_i – действующие в этих точках на макет силы и момент соответственно.

В качестве аппроксимации функций $\vec{F}^b_{ext}$ и $\vec{\tau}^b_{ext}$ могут быть использованы разнообразные решения: линейная интерполяция, сплайны, разложение в ряды Фурье по различным базисам, аппроксимация при помощи радиально-базисных функций, нейронные сети и другие методы. В работе выбор был сделан в пользу бикубических сплайнов, в связи с относительной простотой и эффективностью алгоритмов позволяющих как определить коэффициенты аппроксимации исходя из имеющихся экспериментальных данных, так и вычислить значение получившейся аппроксимирующей функции в заданной точке.

Для определения узловых точек и коэффициентов бикубических сплайнов был выбран подход описанный в [9] и [10]. Этот подход заключается в решении задачи минимизации следующего функционала:

$$\sum_{i=1}^{K} \|y_i - f(x_i)\|^2 + \lambda \iint \left[\left(\frac{\partial^2 f}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x_1 \partial x_2} \right)^2 + \left(\frac{\partial^2 f}{\partial x_2^2} \right)^2 \right] \mathrm{d}x_1 \, \mathrm{d}x_2. \qquad (2.5)$$

где второй член характеризует величину сглаживания предположительно зашумлённых экспериментальных данных. Выбором параметра λ настраивается необходимая степень сглаживания: нулевое значение соответствует классической интерполяции, тогда как при при увеличении параметра второй член начинает вносить более существенный вклад в общее значение функционала, что ведёт к приближению итоговой поверхности к плоскости. Для решения этой задачи была использована функция surfit из состава библиотеки FITPACK [11] написанной на языке Fortran автором книги [10].

В результате применения описанного подхода экспериментально были получены аппроксимации функций $\vec{F}^b_{ext}$ и $\vec{\tau}^b_{ext}$, которые были использованы как для решения задачи идентификации параметров модели управляющих воздействий, так и в далее описанных задачах управления движением и ориентацией. На рисунке 2.1 приведены карты измеренных величин и направлений ускорений макета на поверхности стола КОСМОС установленного в ИПМ им.М.В.Келдыша.

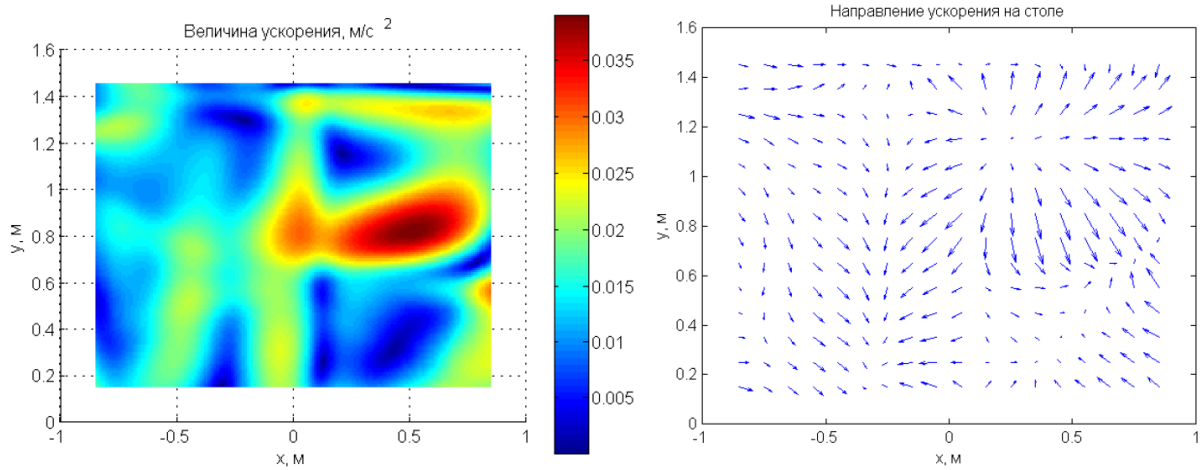

Рис. 2.1 — Величина и направление линейного ускорения.

Заключение

Исследование особенностей реализации алгоритмов управления движением малых космических аппаратов требует использования различного стендового оборудования, в том числе и стендов типа аэростол. К стендам такого типа относится установленный в ИПМ им.М.В.Келдыша стенд КОСМОС. В работе был описан разработанный программный комплекс для проведения экспериментов на стенде КОСМОС. Это программное обеспечение существенно упрощает процесс проведения экспериментов, позволяя исследователю сконцентрироваться на реализации конкретных алгоритмов управления движением.

В качестве примеров использования, были приведены несколько задач, решённых с использованием разработанного комплекса. Дальнейшим направлением развития комплекса видится упрощение процесса конфигурации эксперимента при помощи создания отдельного графического интерфейса для настройки всех параметров.

Литература

[1] Иванов Д. С. и др. Лабораторный стенд для моделирования движения макетов микроспутников //Известия Российской академии наук. Теория и системы управления. – 2018. – №. 1. – С. 117-132.

[2] М. Д. Коптев, Н. Н. Прошунин, Д. С. Иванов, Определение движения макетов системы управления микроспутников на аэродинамическом столе с использованием видеокамеры, Препринты ИПМ им. М. В. Келдыша, 2015, 109, 32 с.

[3] Ivanov D. et al. Determination of disturbances acting on small satellite mock-up on air bearing table //Acta Astronautica. – 2018. – Т. 142. – С. 265-276.

[4] Zheng Y. et al. Revisiting the pnp problem: A fast, general and optimal solution //Proceedings of the IEEE International Conference on Computer Vision. – 2013. – С. 2344-2351.

[5] Garrido-Jurado S. et al. Automatic generation and detection of highly reliable fiducial markers under occlusion //Pattern Recognition. – 2014. – Т. 47. – №. 6. – С. 2280-2292.

[6] Garrido-Jurado S. et al. Generation of fiducial marker dictionaries using mixed integer linear programming //Pattern Recognition. – 2016. – Т. 51. – С. 481-491.

[7] Bradski G., Kaehler A. Learning OpenCV: Computer vision with the OpenCV library. – O'Reilly Media, Inc., 2008.

[8] Francesco Biscani, Dario Izzo, Marcus Märtens. esa/pagmo2: pagmo 2.7 // 2018.

[9] Reinsch C. H. Smoothing by spline functions //Numerische mathematik. – 1967. – Т. 10. – №. 3. – С. 177-183.

[10] Dierckx P. Curve and surface fitting with splines. – Oxford University Press, 1995.

[11] http://www.netlib.org/dierckx/

[12] https://github.com/esa/pagmo2

[13] https://www.uco.es/investiga/grupos/ava/node/26

[14] http://abyz.me.uk/rpi/pigpio/

[15] https://github.com/raspberrypi/tools

[16] https://nanomsg.github.io/nng/

[17] https://www.qt.io/