САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

Кафедра Системного Программирования

Самойлов Антон Сергеевич

Стабилизация квадрокоптера по крену и тангажу под управлением системы Linux

Курсовая работа

Научный руководитель:

Дыдычкин Д. А.

Санкт-Петербург
2013

Оглавление

Введение

Квадрокоптер — это летательный аппарат с 4 роторами, расположенными на концах двух, обычно перпендикулярных, осей. Роторы квадрокоптера, расположенные на разных осях вращаются в противоположных направлениях для компенсации крутящего момента (рис. 1).

Существуют также модификации с бóльшим или меньшим количеством винтов: 3, 6, 8, 10, 12. Общее название для этих летательных аппаратов — мультикоптеры или мультироторы. Однако, несмотря на конструктивные отличия, большинство решаемых задач не зависят от количества винтов.

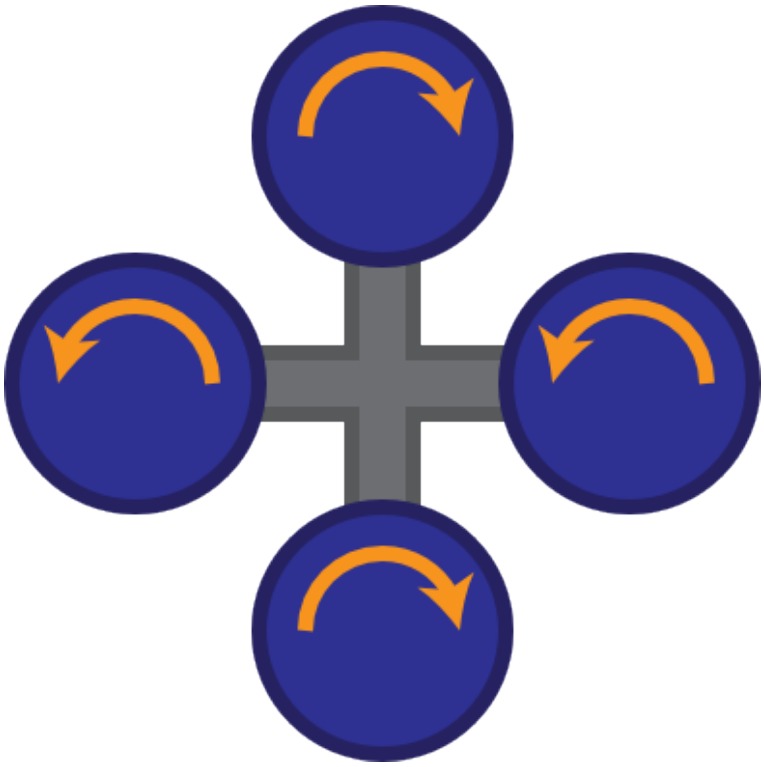

Рис. 1: Схема расположения и направлений вращения винтов квадрокоптера

Так как квадрокоптер в силу своей конструкции не обладает устойчивостью к внешним воздействиям, при полете возникает задача стабилилизации квадрокоптера по трем углам относительно его центра: крену, тангажу и рысканию (рис. 2). Однако, скорости реакции человека не достаточно для эффективной стабилизации, поэтому на практике используются системы автоматической стабилизации на основе показаний датчиков,

установленных на квадрокоптере, таких как акселерометры и гироскопы.

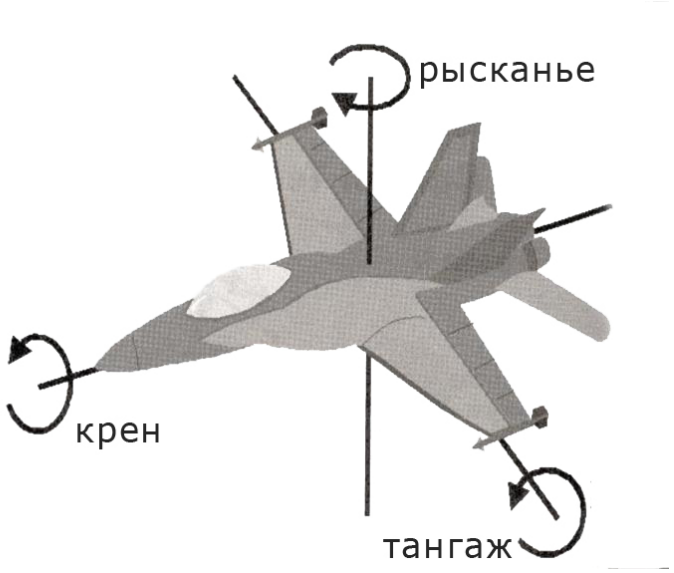

Рис. 2: Углы летательного аппарата относительно его центра

Так как угол рыскания не критичен для удержания квадрокоптера в воздухе, основной задачей данной работы являлась стабилизация квадрокоптера по крену и тангажу. Система стабилизации может быть легко доработана для стабилизации по рысканию, при условии добавления магнитометра (компаса).

Работа выполнялась на основе платы робоконструктора ТРИК [5]. Кибернетический конструктор ТРИК представляет из себя плату управления с комплектом деталей для сборки различных роботов. На плате представлены процессор на архитектуре ARM, набор элементов управления и аналоговых и силовых выходов. Несмотря на то, что квадрокоптер не входит в число стандартных роботов, которых можно собрать из базовых деталей, плата хорошо подходит для реализации алгоритмов стабилизации и управления.

Плата конструктора ТРИК работает под управлением операционной системы на базе ядра Linux. Подобные системы имеют большое количество преимуществ, как то:

- Открытость. Большинство систем на базе ядра Linux разрабатываются под открытыми лицензиями, что влечет за собой высокое качество кода и низкое количество ошибок.

- Портируемость. Дистрибутивы операционных систем на ядре Linux доступны для большого количества различных платформ, в том числе с архитектурами отличными от x86.

- Производительность. Большинство Linux дистрибутивов поддерживают множество компилируемых языков программирования, которые отличаются высокой производительностью по сравнению с интерпретируемыми языками и языками компилируемыми в байт-код виртуальной машины. Это особенно актуально на процессорах со сравнительно невысокой вычислительной мощностью.

- Поддержка устройств. Для операционных систем на базе ядра Linux написано большое количество драйверов устройств, что позволяет с легкостью подключать новое оборудование.

- Сообщество. Многочисленное сообщество разработчиков, а так же большое количество материалов в Сети позволяют быстро находить и устранять возникающие проблемы.

1. Постановка задачи

В рамках данной работы были поставлены следущие задачи:

- Разработать программу управления квадрокоптером. Программа должна быть способна изменять скорости вращения моторов, читать показания датчиков, установленных на плате управления и предоставлять интерфейс для управления летательным аппаратом.
- Изучить существующие алгоритмы для восстановления значений углов квадрокоптера относительно центра (крен, тангаж) по показаниям датчиков. Реализовать один из алгоритмов.
- Изучить подходы и алгоритмы для автоматической стабилизации летательного аппарата по углам относительно центра. Реализовать алгоритм стабилизации.
- Разработать систему удаленного мониторинга состояния квадрокоптера, управления и сбора телеметрических данных.

2. Описание доступных датчиков

Для лучшего понимания опишем принципы работы двух доступных датчиков — акселерометра и гироскопа.

Акселерометр (трехосевой) — прибор, измеряющий проекцию ускорения на три перпендикулярные друг другу оси. То есть, если представить показания акселерометра в виде трехмерного вектора, этот вектор будет представлять сумму векторов собственного ускорения объекта и ускорения, обратного ускорению свободного падения (1).

$$a_{accel} = a_{real} - g \tag{1}$$

Акселерометр имеет довольно высокий уровень шума сигнала, кроме того очень чувствителен к вибрации. Поэтому для извлечения полезной иформации используются алгоритмы фильтрации сигнала. Предположим входной сигнал в момент времени t_n имеет значение x_n, тогда после фильтрации сигнал принимает значение y_n. В рамках работы были реализованы следующие фильтры:

- Скользящее среднее. Формула выходного сигнала для скользщего среднего с размером окна k имеет вид (2). Иными словами, это усреднение последних k значений сигнала.

$$MovingAvg_k(x_n) = y_n = \sum_{i=n-k+1}^{n} \frac{x_i}{k} \tag{2}$$

- Гауссовский фильтр. Этот фильтр — один из простейших рекурсивных фильтров. Рекурсивные фильтры — фильтры, для которых выходной сигнал зависит не только от входного, но и от предыдущих значений выходного сигнала. В работе использовались две вариации фильтра Гаусса — (3) и (4).

$$Gauss_3(x_n) = y_n = \frac{x_n + 2y_{n-1} + y_{n-2}}{4} \tag{3}$$

$$Gauss_4(x_n) = y_n = \frac{x_n + 3y_{n-1} + 3y_{n-2} + y_{n-3}}{8} \tag{4}$$

- Упрощенный фильтр Калмана. Еще один пример рекурсивного фильтра, задающийся формулой (5), где K - параметрический коэффициент.

$$Kalman_K(x_n) = y_n = Kx_n + (1 - K)y_{n-1} \tag{5}$$

- Различные комбинации вышеописанных фильтров, как то $Kalman_K(Gauss_3(x_n))$, $Gauss_4(Kalman_K(x_n)))$, и т.д.

Гироскоп (трехосевой) — прибор, измеряющий проекцию угловой скорости на три перпендикулярные друг другу оси. Гироскоп отличается достаточно точным сигналом (нет необходимости фильтровать значения сигнала). Однако, так как для задачи стабилизации наибольший интерес представляют углы, необходимо интегрировать показания гироскопа по времени. В связи с тем, что точное интегрирование дискретного сигнала невозможно, вычисленный угол со временем начинает расходиться с реальным. Для коррекции показаний гироскопа по углам крена и тангажа используется акселерометр, для коррекции рыскания — магнитометр.

3. Восстановление углов

Прежде всего стоит отметить, что акселерометр не способен отследить равномерное вращение в плоскости, перпендикулярной вектору $\vec{g}$.

Если считать собственное ускорение квадрокоптера малым по сравнению с ускорением свободного падения $\vec{g}$, можно определить углы крена и тангажа только по показаниям акселерометра. Однако практические исследования показывают, что собственный шум акселерометра, в сочетании с вибрацией от вращения винтов и собственными ускорениями квадрокоптра не позволяет определять углы с необходимой для стабилизации точностью. Если использовать фильтрацию достаточной силы для сглаживания сигнала, то появляется задержка показаний датчика, препятствующая своевременному реагированию на изменение положения летательного аппарата. Кроме того, при сильном сглаживании теряется информация о сравнительно небольших изменениях угла.

Таким образом, для решения задачи стабилизации в рамках работы использовалось сочетание акселерометра и гироскопа.

Для представления поворотов удобно использовать кватернионы. Обозначим за $\Lambda_g(t)$ кватернион поворота, вычисленный в момент времени t по показаниям гироскопа, за $\Lambda_a(t)$ — кватернион поворота, вычисленный в тот же момент по показаниям акселерометра. Для вычисления этих значений используются формулы, приведенные в [6]. Для коррекции ошибки интегрирования показаний гироскопа, берется их взвешенная сумма (6).

$$\Lambda(t) = \alpha\Lambda_a(t) + (1-\alpha)\Lambda_g(t) \tag{6}$$

Здесь α — коэффициент коррекции, который обычно не превышает 0.01, в данном случае α был выбран равным 0.005.

Чтобы получить углы по имеющемуся кватерниону, используются формулы (7).

$$\begin{aligned}
\psi &= arctg\left(\frac{1-2\Lambda_y^2-2\Lambda_z^2}{2\Lambda_y\Lambda_w-2\Lambda_x\Lambda_z}\right)\\
\gamma &= arcsin(2\Lambda_x\Lambda_y+2\Lambda_z\Lambda_w)\\
\theta &= arctg\left(\frac{1-2\Lambda_x^2-2\Lambda_z^2}{2\Lambda_x\Lambda_w-2\Lambda_y\Lambda_z}\right)
\end{aligned} \tag{7}$$

Здесь Λ_w, Λ_x, Λ_y и Λ_z — четыре компоненты кватерниона. Кроме того в северном полюсе, где $\Lambda_x\Lambda_y + \Lambda_z\Lambda_w \approx \frac{1}{2}$ формулы (7) принимают вид (8), а в южном полюсе, где $\Lambda_x\Lambda_y + \Lambda_z\Lambda_w \approx -\frac{1}{2}$, формулы (7) принимают вид (9).

$$\begin{aligned} \psi &= 2arctg\left(\frac{\Lambda_w}{\Lambda_x}\right) \\ \gamma &= \frac{\pi}{2} \\ \theta &= 0 \end{aligned} \tag{8}$$

$$\begin{aligned} \psi &= -2arctg\left(\frac{\Lambda_w}{\Lambda_x}\right) \\ \gamma &= -\frac{\pi}{2} \\ \theta &= 0 \end{aligned} \tag{9}$$

4. Система стабилизации

После того, как программой управления были получены углы крена и тангажа, необходимо применить поправку к мощностям двигателей, для устранения возможного отклонения. Для этой цели обычно применяется ПИД или ПД регулятор.

ПИД (пропорционально-интегрально-дифференциальный) регулятор — алгоритм, который на основе отклонения от величины, в которой необходимо стабилизироваться, выдает поправку на соответствующие моторы. Если считать отклонение от необходимой величины в момент времени t равным $e(t)$, то формула (10) выражает необходимую поправку. K_P, K_I и K_D — пропорциональный, интегральный и дифферееенциальный коэффициенты соответственно.

$$u(t) = P + I + D = K_P\, e(t) + K_I \int_0^t e(\tau)\, d\tau + K_D \frac{d}{dt} e(t) \tag{10}$$

В случае ПД (пропорционально-дифференциального) регулятора компонента I обнуляется, и формула (10) принимает вид (11).

$$u(t) = P + D = K_P\, e(t) + K_D \frac{d}{dt} e(t) \tag{11}$$

Акселерометр и гироскоп располагаются на плате так, чтобы их оси были сонаправлены собственным осям квадрокоптера. Таким образом, восстановленные углы представляют отклонения от положения стабильности. Так как задачи стабилизации по крену и тангажу можно считать независимыми, удобно представлять углы в виде двухмерного (или трехмерного в случае 3 углов) вектора, и все опреации по вычислению поправки проводить в векторном виде.

Чтобы квадрокоптер не терял высоту, сумма тяг винтов должна сохраняться при произвольном наклоне. Рассмотрим ситуацию с одной осью. Если обозначить в момент времени t суммарную мощность моторов на оси за $M(t)$, а за $P_1(t)$ и $P_2(t)$ обозначить мощности, которые необходимо выдать на два мотора оси соответственно, расчетная

формула принимает вид (12).

$$\begin{aligned} P_1(t) &= \sqrt{\frac{M^2(t) + u(t)}{2}} \\ P_2(t) &= \sqrt{\frac{M^2(t) - u(t)}{2}} \end{aligned} \tag{12}$$

Квадратный корень возникает из физического закона, согласно которому подъемная сила винта пропорциональна квадрату угловой скорости вращения. Можно проверить, что при таком выборе мощностей подъемная сила сохраняется — (13).

$$P_1^2(t) + P_2^2(t) = \frac{M^2(t) + u(t)}{2} + \frac{M^2(t) - u(t)}{2} = M^2(t) \tag{13}$$

5. Реализация программы управления

Так как процессор на плате ТРИК имеет архитектуру ARM, и сравнительно небольшую вычислительную мощность, программу управления нецелесообразно компилировать на самой плате. Но проект, скомпилированный для архитектуры x86 не может корректно работать на процессоре с архитектурой ARM. Чтобы обойти эти ограничения используется техника кросскомпиляции — сборки проекта под архитектуру, отличную от той, на которой сборка производится. Для сборки проекта использовалась среда для кросскомпиляции OpenEmbedded [2].

В качестве платформы для реализации программы управления были выбраны язык C++ с фреймворком Qt [1]. Данный выбор имеет несколько преимуществ:

- Сборки C++ и Qt существуют для многих платформ и архитектур, в том числе и для ARM.
- C++ хорошо подходит для систем реального времени, обеспечивая высокую производительность.
- C++ компилируется в машинный код, что также положительно сказывается на производительности.
- C++ является достаточно высокоуровневым языком, что способствует высокой скорости и удобству разработки.
- Qt предоставляет все необходимые классы для работы с файлами, сетью и математическими объектами, такими как кватернионы.
- Qt сигналы и слоты — механизм передачи сообщений между методами, идеально подходит для обработки нерегулярных по времени событий, таких как чтение значений датчиков или исполнение управляющих команд, приходящих по протоколу TCP.
- C++ и Qt имеют обширное сообщество и множество материалов доступно в Сети. Это способствует быстрому решению возникающих проблем.

На рис. 3 проиллюстрированна диаграмма классов проекта. Как можно видеть, за обработку сигналов с датчиков и вычисление необходимой поправки на моторы отвечает класс FlightControl, за распределение мощности по моторам оси — CopterAxis, а непосредственно управляют мощностью конкретных моторов объекты класса CopterMotor. За прием управляющих сигналов по протоколу TCP отвечает класс CopterCtrl.

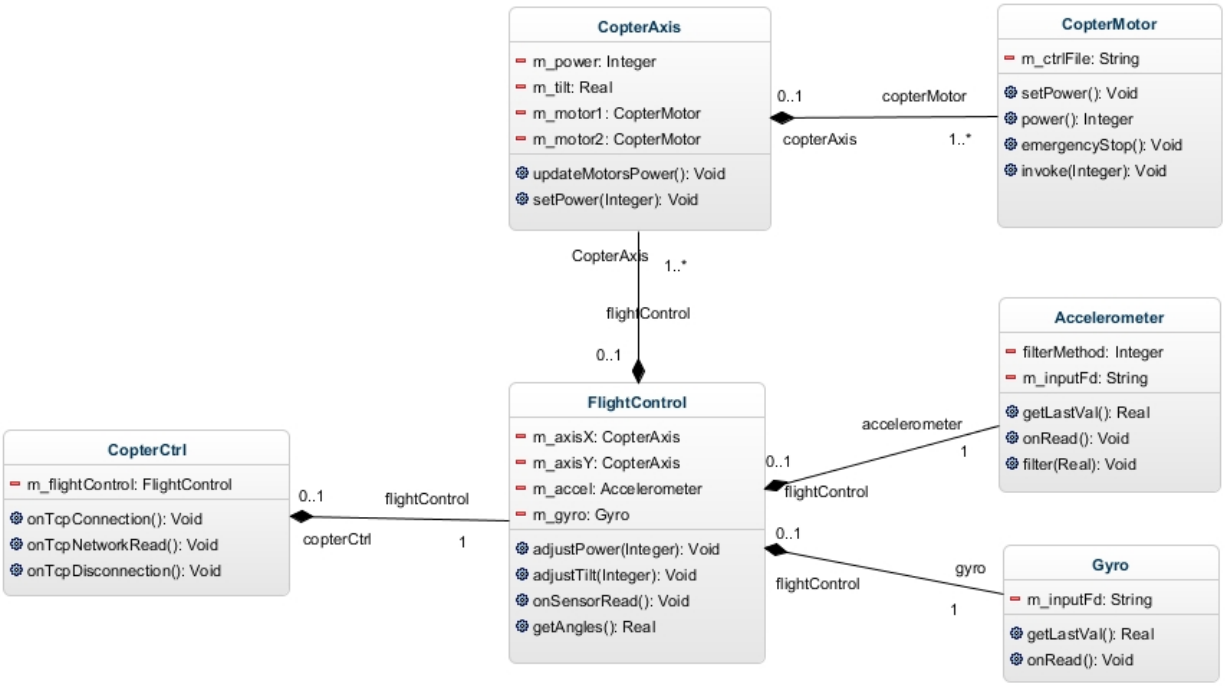

Рис. 3: Диаграмма классов программы управления квадрокоптером

Код проекта открыт и доступен для просмотра и загрузки на сервисе GitHub [7].

6. Удаленное управление и сбор телеметрии

Так как при полете невозможно получить физический доступ к квадрокоптеру, была разработана система для удаленного мониторинга и управления летательным аппаратом.

Для реализации системы удаленного управления были выбраны язык C++ и фреймворк Qt. Использование одинаковых средств для систем стабилизации и удаленного управления позволяет унифицировать интерфейсы доступа и избежать трудностей разработки на разных языках. Кроме того, Qt предоставляет кроссплатформенность, что гарантирует возможность запуска программы управления и мониторинга на всех популярных операционных системах.

Клиентская часть представляет из себя кроссплатформенное desktop-приложение с графическим интерфейсом (рис. 4).

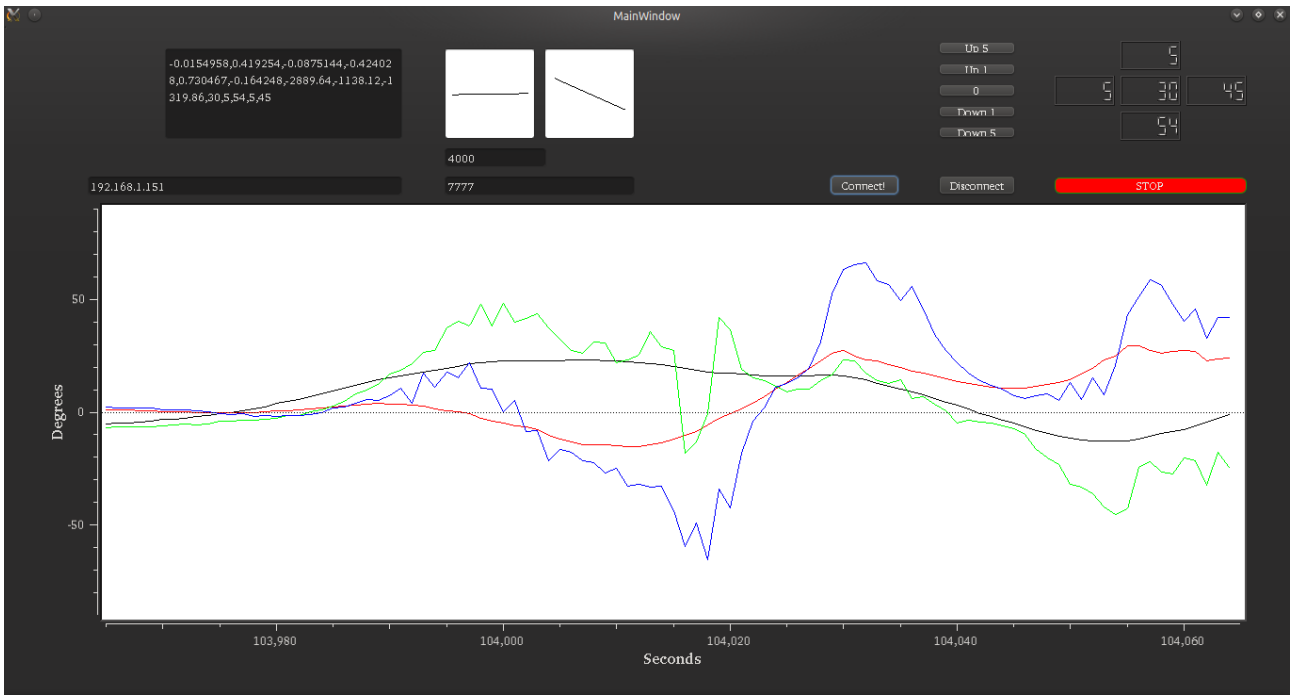

Рис. 4: Пользовательский интерфейс программы управления квадрокоптером

На иллюстрации можно видеть диаграммы наклонов квадрокоптера по двум осям, кнопки аварийной остановки и управления общей мощностью, отображение общей мощности и мощности каждого мотора, интерфейс удаленного подключения и графики показаний акселерометра и восстановленных углов. Графики обновляются в реальном

времени, и легко настраиваются для отображения необходимой информации. В качестве инструмента отображения графиков была выбрана библиотека Qwt [4], так как она предоставляет наиболее полный и простой в использовании набор элементов для вывода графиков.

При отсутствии приложения, доступ к квадрокоптеру можно получить, путем присоединения к порту управления напрямую, например с помощью программы telnet, входящей в стандартный набор программ Unix-подбных операционных систем, или с помощью программы netcat, входящей в стандартный набор программ на операционных системах семейства Windows. Кроме того, простой интерфейс позволяет с легкостью создавать приложения управления квадрокоптером с различных платформ, например мобильного телефона под управлением операционной системы Android или среды для графического программирования роботов QReal:Robots [3].

Со стороны квадрокоптера при старте приложения открываются два порта для TCP соединений: один для управляющих комманд и откликов на управление, второй для посылки телеметрических данных. Так как использованный класс QTcpSocket наследуется от абстрактного класса QAbstractSocket, не представляет сложностей быстрая смена протокола взаимодействия, например на UDP.

Код проекта открыт и доступен для просмотра и загрузки на сервисе GitHub [8].

Заключение

В процессе работы были получены следующие результаты:

- Реализована программа управления квадрокоптером, способная взаимодействовать с моторами и установленными на плате датчиками.
- Реализован алгоритм восстановления углов крена и тангажа на основе показаний гироскопа и акселерометра.
- Реализована система стабилизации квадрокоптера по углам крена и тангажа.
- Получена уверенная стабилизация по одной оси. При закрепленной второй оси, когда у квадрокоптера остается только 1 степень свободы, стабилизация происходит из всех допустимых углов отклонения.
- Получена ограниченная стабилизация на тестовом стенде: квадрокоптер шарнирно закрепляется в центре конструкции. Летательный аппарат возвращается в положение стабильности из сравнительно небольших углов отклонения.

Список литературы

[1] Blanchette Jasmin, Summerfield Mark. C++ GUI Programming with Qt 4. — Prentice Hall PTR, 2008. — ISBN: 0132354160.

[2] OpenEmbedded, the build framework for embedded Linux. — 2013. — http://www.openembedded.org/wiki/Main_Page.

[3] QReal:Robots. — 2013. — http://robots.qreal.ru/.

[4] Qwt - Qt Widgets for Technical Applications. — 2013. — http://qwt.sourceforge.net/.

[5] Кибернетический конструктор ТРИК. — 2013. — http://www.trikset.com/.

[6] Матвеев В.В., Распопов В.Я. Основы построения бесплатформенных инерциальных навигационных систем / Под ред. В.Я. Распопова. — СПб. : ГНЦ РФ ОАО "Концерн "ЦНИИ "Электроприбор", 2009. — С. 118–157.

[7] Реализация системы стабилизации квадрокоптера. — 2013. — http://git.io/TEDsxQ.

[8] Реализация системы удаленного управления квадрокоптером. — 2013. — http://git.io/xwdxLA.