**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ,**

**МЕХАНИКИ И ОПТИКИ**

|  |
| --- |
| КТиУ |

Факультет

(название факультета)

|  |
| --- |
| СУиР |

Кафедра

(название кафедры)

Направление подготовки (специальность) Интеллектуальные технологии в робототехнике

**О Т Ч Е Т**

**о \_\_\_\_\_\_\_\_\_учебной\_\_\_\_\_\_\_\_практике**

(наименование практики)

**Тема задания:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ARDrone\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Студент** Зенкин А.М. P3235

(Фамилия И.О.) номер группы

|  |
| --- |
| **Капитонов А.А.** |

**Руководитель практики от организации:**

(Фамилия И.О., должность и место работы)

|  |
| --- |
| **Бойков В.И.** |

**Ответственный за практику от университета:**

(Фамилия И.О., должность)

**Практика пройдена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Подписи членов комиссии**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(подпись)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(подпись)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(подпись)

**Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Санкт-Петербург**

**2016**

**Цели, поставленные мной перед практикой( на период ее проведения) :**

* *Научиться взлетать и управлять дроном*
* *Научиться получать изображения/видео с дрона*
* *Научиться управлять дроном через клавиатуру/джойстик*
* *Связать ARDrone с OpenCV*

**Цели на дальнейшую работу с дроном:**

* *Научиться обрабатывать изображения в OpenCV( распознавание объектов, движение за объектами)*
* *Научиться отслеживать пройденный дроном путь*
* *Научиться вырисовывать в пространстве фигуры дроном( автономно)*

**Оглавление**

1. **Введение**

* ***Что такое AR.Drone***
* ***Немного физики***
* ***Технические характеристики***
* ***Порты и их использование***

1. **Взаимодействие с дроном**

* ***Первый опыт запустить дрона***
* ***ROS и ARDrone***
* ***Установка готовой сборки с ROS’ом на борту***
* ***Установка и запуск библиотеки***
* ***Компьютерное зрение***
* ***Ориентация в пространстве***
* ***Полетам по прямоугольнику***

1. **Вывод**
2. **Список литературы**

***AR.Drone 2.0***

******

**Введение**

**Что такое AR.Drone**

AR.Drone это квадрокоптер, которым можно управлять с любого устройства, имеющего wifi. Дрон оснащен встроенными датчиками и контроллерами, которые стабилизируют транспортное средство во время полета и дают возможность зависнуть квадрокоптеру на месте и ждать дальнейших команд( однако на практике это оказалось не так, дрон не всегда стабилизируется и с изрядной периодичностью его уносит в сторону). Поэтому он стал достаточно популярной исследовательской платформой для компьютерного зрения и робо разведки.

**Немного физики**

Механизм содержит четыре ротора, которые размещены на концах крестовидной рамы. Одна пара крутится по часовой стрелке, другая против( можно увидеть на рисунке справа)

Маневры совершаются путем изменения углов тангажа, крена и рысканья.



тангаж



крен



рысканье



вверх/вниз

|  |  |
| --- | --- |
| Процессор | Модель: OMAP 3630 CPU 32-х битный  Код: Cortex A8 Тактовая частота: 720\* МГц Число ядер/потоков: 1/1 Техпроцесс: 45 нм Дополнительно: OpenGL ES 1.1, OpenGL ES 2.0, OpenVG |
| Память | 1 GB DDR2-RAM( 200 MHz and 128 MB NAND-Flash) |
| Система | Kernel Linux uclibc 2.6.32.9-gbb4d210 using uclibc as c-library |
| Беспроводная связь | WLAN standards b/g/n |
| Навигация | - Inertial Navigation System (INS) 6 степеней свободы, это позволяет выполнять рысканье, тангаж и движение крена - 3-х осный акселерометр( ±50mg) - 3-x осный гироскоп( ±2000°/second) - 3-х осный магнитометр( ±6°)  - датчик давления( ±10Па) для ориентации по высоте( >6м) - дальномер для ориентации по высоте( <6м)  - вертикальная QVGA камера(320х420) для ground speed |
| Вес | 420 г.( уличный корпус)  380 г.( комнатный корпус) |
| Материал | Корпус: полипропилен Лопости: карбон |
| Батарея | 1000 mAh, 11.1 В – в стандартной конфигурации  1500 mAh 11.1 В – можно установить |
| Мотор | 4 мотора 28.500 об/мин (RPM) каждый управляется 8 MIPS AVR CPU контроллером |
| Камера | Фронтальная камера: 720p 30 fps  Вертикальная камера: QVGA 320х420 |
| Интерфейсы | 1 x Разъем питания 1 х USB( для подключения карты памяти или LTE модема) |

**

* Рысканье влево/вправо – квадрокоптер поворачивается влево/вправо
* Тангаж вниз/вверх – квадрокоптер наклоняется вниз/вверх
* Крен влево/вправо – квадрокоптер наклонятся влево/вправо

**Технические характеристики**

**Порты и их использование**

После того, как мы подключим батарейку к дрону, он автоматически загружается и готов к использованию. Квадрокоптер проверит двигатели( прокрутив каждое на пару градусов) и запустит незашифрованную точку доступа wifi с именем ardrone2\_( шесть цифр). После я просканировал порты с помощью программы Nmap и выяснил, что открыто два хорошо известных порта: 21( FTP) и 23 ( Telnet). Остальные порты связаны с операциями дрона( передача информации и тд). Таблица ниже показывает использование всех портов.

|  |  |
| --- | --- |
| Порт | Пояснение |
| 21 (TCP) | FTP сервер, который обрабатывает полученные видео и картинки с дрона |
| 23 (TCP) | Telnet сервер, предоставляющий корневую оболочку |
| 5551 (TCP) | FTP доступ к папке обновлений для того чтобы обновить прошивку |
| 5553 (TCP) | При записи видео с телефона тут будут доступны кадры ( 720p) |
| 5554 (UDP) | NAVDATA: Здесь отправляются текущие данные телеметрии(состояние, скорость, частота вращения ротора) ( 15 cmds/s demomode, 200 cmds/s full/debug mode) cmds/s - это общее число SCSI - команд, передаваемых к системе хранения, включая операции ввода-вывода вывода виртуальных машин, а также сервисные команды (например, SCSI reservations). |
| 5555 (TCP) | Здесь доступен поток видео для клиентов |
| 5556 (UDP) | ATCMD: Дрон контролируется с помощью AT команд. Эти команды периодически посылаются на дрона( 30 cmds/s) |
| 5559 (TCP) | Control port – здесь передаются важные данные, такие как конфигурация |

FTP служба запускается без пароля, поэтому мы имеем прямой доступ к /data/video/ и другим подкатегориям в дроне, что на мой взгляд не является особо безопасным, так как любой человек может подключиться к вашему дрону и смотреть все его данные.

Если подключиться к порту Telnet, то мы попадаем в корневую папку дрона, где лежит вся его операционная система. Следующие интересные вещи были найдены:

**Пусть** **Пояснения**

/bin/check\_update.sh Скрипт с обновлениями

/bin/init\_gpios.sh Инициализация GPIO порта

/bin/mount\_usb.sh Установка USB девайсов

/bin/pairing\_setup.sh Скрипт для работы со смартфоновскими приложениями

/bin/parallel-stream.sh Поток с камеры

/bin/reset\_conﬁg.sh Reset conﬁg.ini

/bin/umount\_usb.sh Отключение USB девайсов

/bin/Wiﬁ\_setup.sh Запуск wifi подключения и других сервисов

/sbin/udevd.sh Запуск udevd с udev\_init загрузчиком

/usr/sbin/loadAR6000.sh Дополнительные настройки wifi

/etc/inetd.conf Сервер FTP (/update and /data/video)

/etc/udhcpd.conf Конфигурации DHCP сервера для wifi

/data/conﬁg.ini Главный файл конфигураци

**Взаимодействие с дроном**

**Первый опыт запустить дрона**

Я начал изучение дрона с того, что приступил к прочтению официального сайта, вскоре нашел раздел для разработчиков, в котором предлагалось скачать SKD, предусмотренную Parrot для создания своих приложений и проектов, что я и сделал. ( [ссылка на официальный сайт](http://developer.parrot.com/products.html)).

Установка:

1. Скачиваем SDK с официального сайта.
2. Создадим папку SDK( без разницы, как называется, мне просто так будет удобно) и соберем наш проект.

$ cd SDK/Examples/Linux/

$ make

1. Результат появится в SDK/Examples/Linux/Build/Release

При сборке на Linux столкнулся со следующими проблемами:

**Ошибка:**

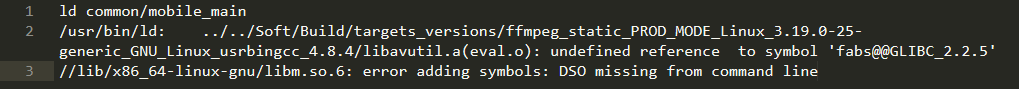


**Проблема:**  
 Не установлен ncurse

**Решение:**

sudo apt-get install libncurses5-dev libncursesw5-dev

**Ошибка:**

****

**Проблема:**  
 У нас использутся библиотека limb, но она пропущена в командной строке

**Решение проблемы:**

1) Переходим в папку с MakfFile

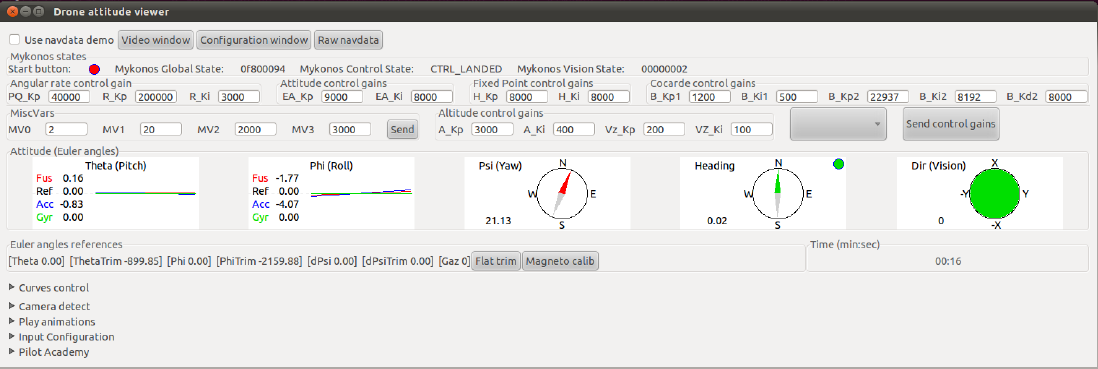
…/ARDrone\_SDK\_2\_0\_1/Examples/Linux/Navigation/Build/

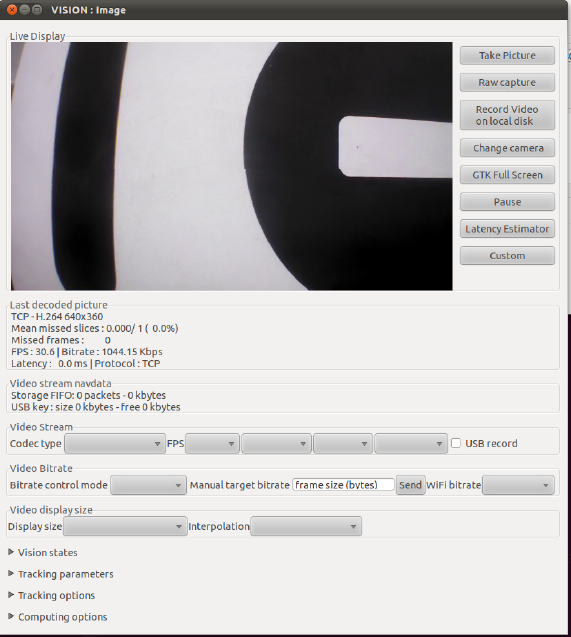
2) Открывваем его текстовым редактором и ищием вот такую строчку( если не ошибаюсь, то она должна быть 131)

3) Добавляем вконце -lm (те libm, которой нам и не хватает для счастья)

GENERIC\_LIBS+=-liw -lpc\_ardrone -lgthread-2.0 -lgtk-x11-2.0 -lrt -lxml2 -ludev -lswscale -lSDL -lm  
**Похожие ошибки:**  
Могут возникнуть похожие ошибки, метод решения такой же, смотрим на библиотеку, которой не хватает, заменяем lib на l, все остально сохраняем и добавляем в Makefile(обратите внимание, добавляем в тот Makefile, где выдает ошибку)  
  
У меня возникли проблемы со сборкой demo\_video  
( переходим в папку с MakfFile …/ARDrone\_SDK\_2\_0\_1/Examples/Linux/video\_demo/Build  
открываем Makefile, ищем строчку ENERIC\_LIBS+= … и добавляем библиотеки, в мое случе не хватало следующих: **demo\_video -lgdk-x11-2.0 -lgobject-2.0 )**

После проделанных манипуляций я все-таки смог запустить SDK, она представляет их себя довольно приятный интерфейс, в котором отсутствует управление c клавиатуры, что было достаточно неудобно, можно было установить только управление джойстиком( и то это у меня не получилось). Также отображается состояние дрона, есть вкладка с видео и конфигурациями.





Проведя не одну пару часов за разбором этого SDK, я решил, что надо искать другой подход к дрону, так как времени у меня было немного, а разбор всего, что было написано, занял бы достаточно долгий период времени. Поэтому я начал гуглить дальше и нашел отличную библиотеку, которая работает под ROS’ом.

**Что такое ROS (Robot Operating System)**

Собственно, до практики я не имел понятия, что такое ROS как с ним взаимодействовать. Поэтому я потратил отд ельный период времени на изучение его части, которая была необходима для работы с библиотекой и написания простых скриптов для ARDrone.

ROS является основой программного обеспечения, которое позволяет разрабатывать робототехнические приложения, которые контролируют и взаимодействуют с роботами. ( прочитать больше можно тут <http://www.ros.org> )

**ROS и ARDrone**

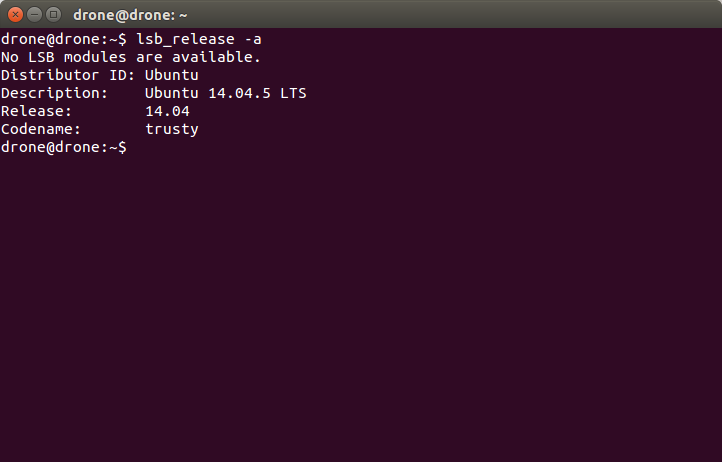
Как я уже говорил выше, я нашел просто замечательную библиотеку под ARDrone - ardrone-autonomy.( [ссылка на обзор](http://ardrone-autonomy.readthedocs.io/en/latest/installation.html)). Итак, дальше я опишу порядок установки ROS и этой чудной вещицы:

Установим и настроим ROS( далее опишу, как скачать и настроить уже готовую систему)

Ссылка на туториал( [тык](http://wiki.ros.org/indigo/Installation/Ubuntu))

1) **Настроим репозитории Ubuntu:**

**2)** Настроим компьютер, чтобы он мог принять программное обеспечение от packages.ros.org(sourse.list). Будем устанавливать ROS Indigo( добавлем сервера ros, чтобы менеджер знал откуда брать пакеты). Откроем командную строку (Ctrl+alt+T) и введём следующее( $ обозначенет строчку, вводить не надо):

**Внимание!!!**  
ROS Indigo поддерживает только Saucy (13.10) и Trusty (14.04)

Чтобы узнать версию ubuntu, введите следующую команду:

$ lsb\_release -a

**3) Настройка ключей**

*$ sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net --recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116*

**4) Обновляем пакеты**$ sudo apt-get update

**5) Команда установки ROS (рекомендованная конфигурация Desktop-Full)**$ sudo apt-get install ros-indigo-desktop-full

**6) Инициализируем rosdep:**

$ sudo rosdep init

$ rosdep update

**7) Установка окружающей среды:**

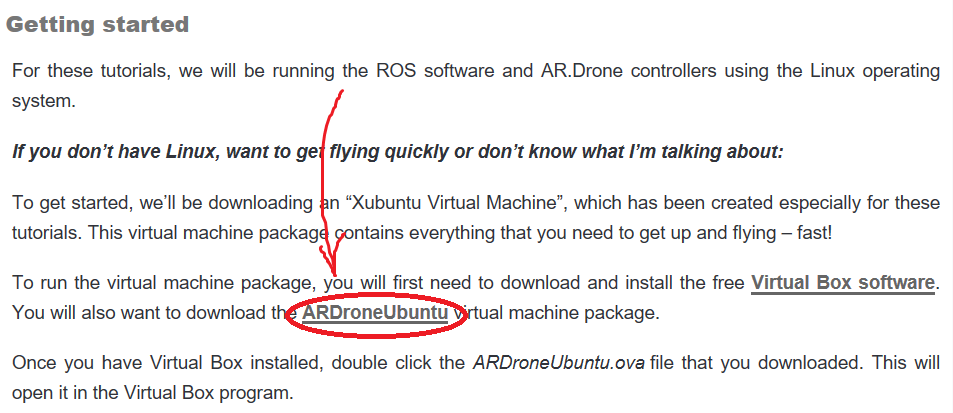
$ echo "source /opt/ros/indigo/setup.bash" >> ~/.bashrc

$ source ~/.bashrc

**8) Установка rosinstal:**

$ sudo apt-get install python-rosinstall  
  
Всё, на этом установка ros'a закончена. Также есть способ сразу установить готовую систему с росом, но там очень старая версия, так что я не рекомендую это делать

**Установка готовой сборки с ROS’ом на борту**

Скачиваем и устанавливаем виртуальную систему [тык](http://robohub.org/up-and-flying-with-the-ar-drone-and-ros-getting-started/)

Правда версия ROS’a у этой штуки очень древняя, поэтому не рекомендую ей пользоваться.

**Установка и запуск библиотеки**

Установить библиотеку очень просто, используя туториал, ссылку на который я помести выше. Для запуска вводим в терминал следующие строчки:  


С помощью данного драйвера мы получаем полный доступ ко всем функциям дрона. После многократных полетов, если их можно было назвать таковыми, я решил, что для отладки все-таки необходимо установить какой-нибудь симулятор, мой выбор пал на пакет tum\_simulator, основанный на Gazebo. ( [Ссылка на GitHub](https://github.com/dougvk/tum_simulator))

Как я уже говорил, пакет использует симулятор gazebo и состоит из:

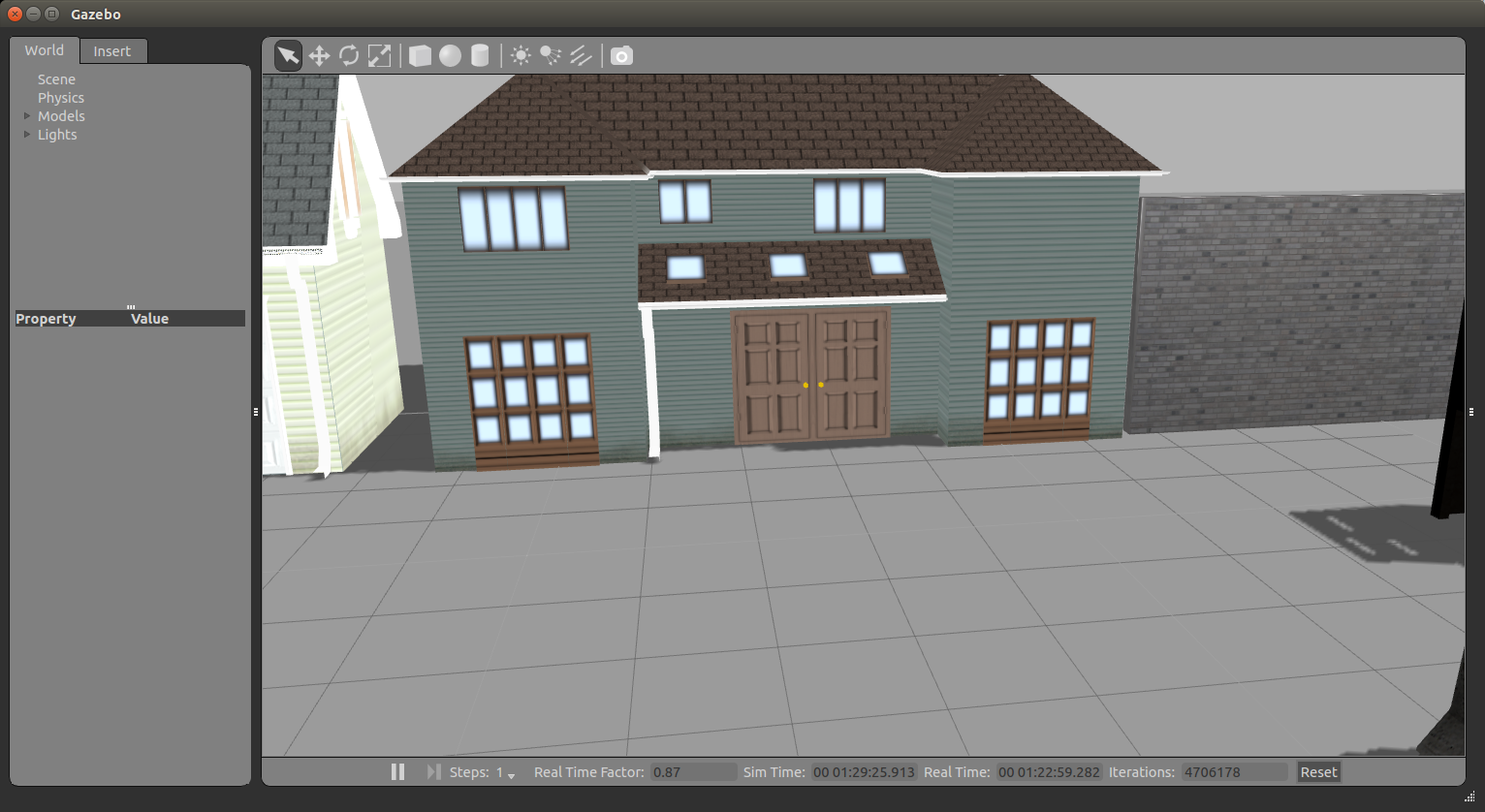
1) cvg\_sim\_gazebo содержит модели, в частности модель квадракоптера и его окружающую среду

2) cvg\_sim\_gazebo\_plugins содержит плагины для управлением робота, получения с него данных( /ardrone/navdata)

3) message\_to\_tf работа с tf пакетом

4) cvg\_sim\_msgs содержит сообщения для симулятора

Вот, что из себя представляет следующий интерфейс:

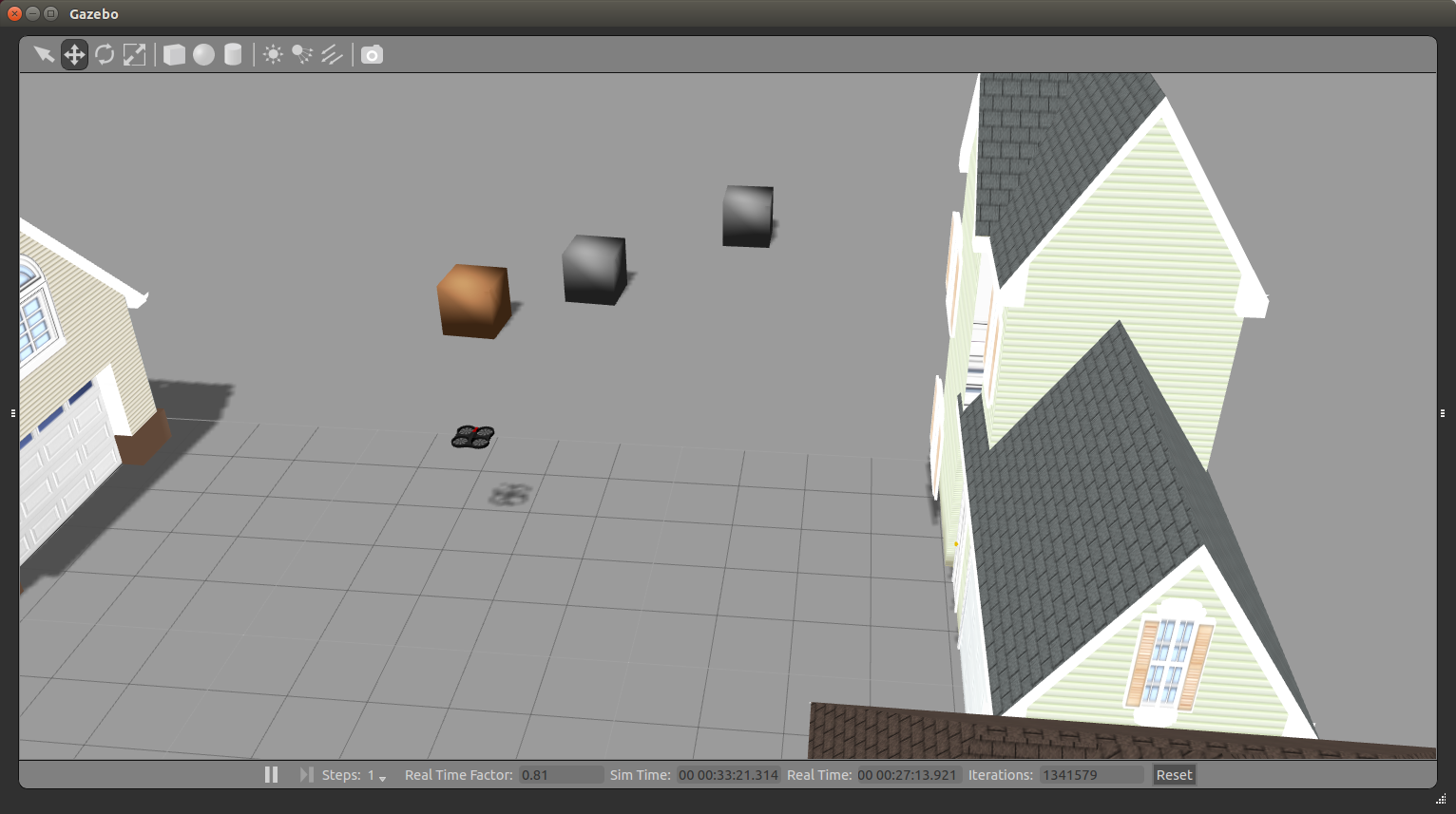


Дальше я разобрался в ROS’е и в том, как он устроен. Конечно, у меня не было времени, чтобы весь его изучить, но какие-то азы для работы с библиотекой я освоил. Дальше я приступил к рассмотрению команд, которые можно отправлять дрону, и информации, которую дрон шлет на порты. Началась прорисовываться более ясная картина того, как работать с дроном и получать какой-то “отклик” от него. Также мне удалось подключить джойстик и управлять дроном непосредственно им. Было весело, но надо двигаться дальше, а именно к компьютерному зрению.

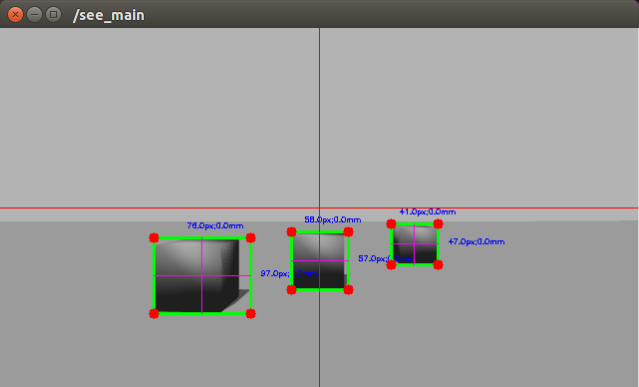
**Компьютерное зрение**

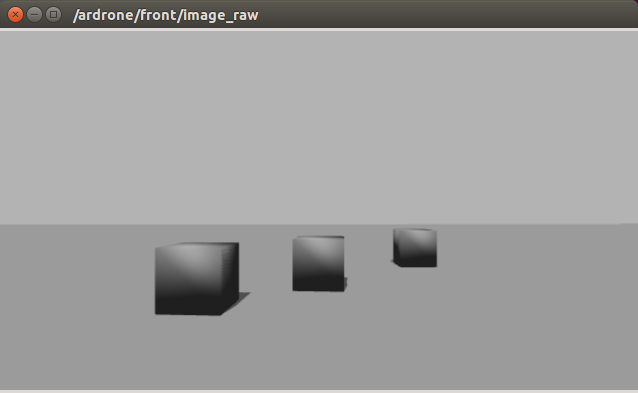
Изначально я поставил себе простую цель – связать дрон с OpenCV, так как это уже половина дела. Для получения такого результата я воспользовался пакетом cvision, который позаимствовал у коллеги с кафедры, поэтому не могу дать ссылку на GitHub или описание на ros.org. Сначала я решил все опробовать в симуляторе gazebo. Я взял уже имеющуюся карту и добавил к ней три куба, которые находились на разном расстоянии от квадрокоптера. Дальше моей программе необходимо было распознать эти три куба и вывести размер в пикселях ( программа была позаимствована у коллег с кафедры). Первое, с чем я столкнулся, это то, что при появлении большого количества контуров, программа перегружалась и видео очень сильно запаздывало, что никуда не годится, так как дрону нужна практически моментальная обратная связь. Пришел к тому, что надо понижать качество изображения и лучше всего концентрироваться на одном объекте, чтобы посторонние не занимали времени на отрисовку. Ниже я привел результаты работы алгоритма и положение дрона на карте:

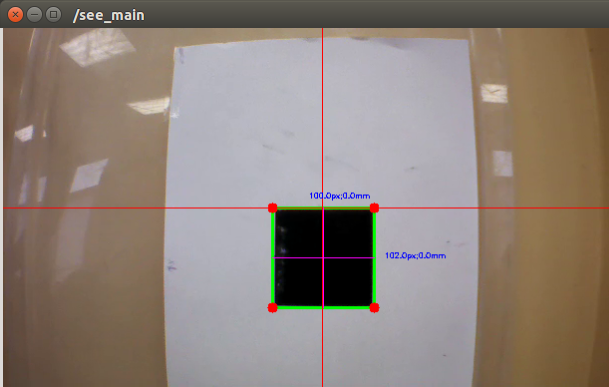
Вид на всю картину:



Собственно, работа алгоритма представлена на фронтальной камере дрона. Для того, чтобы не было помех я сделал так, чтобы на фоне них не было никаких других элементов. Вторая проблема, с которой я столкнулся, это то, что не всегда удается определить контуры кубы, иногда получается так, что два куба определяются, как один, или же граница куба получатся много больше самого куба, но это все недоработки программы, которые поправляются, но, если использовать один четкий контур, то проблем не возникает. Это я покажу немного позже уже на реальном квадрокоптере.

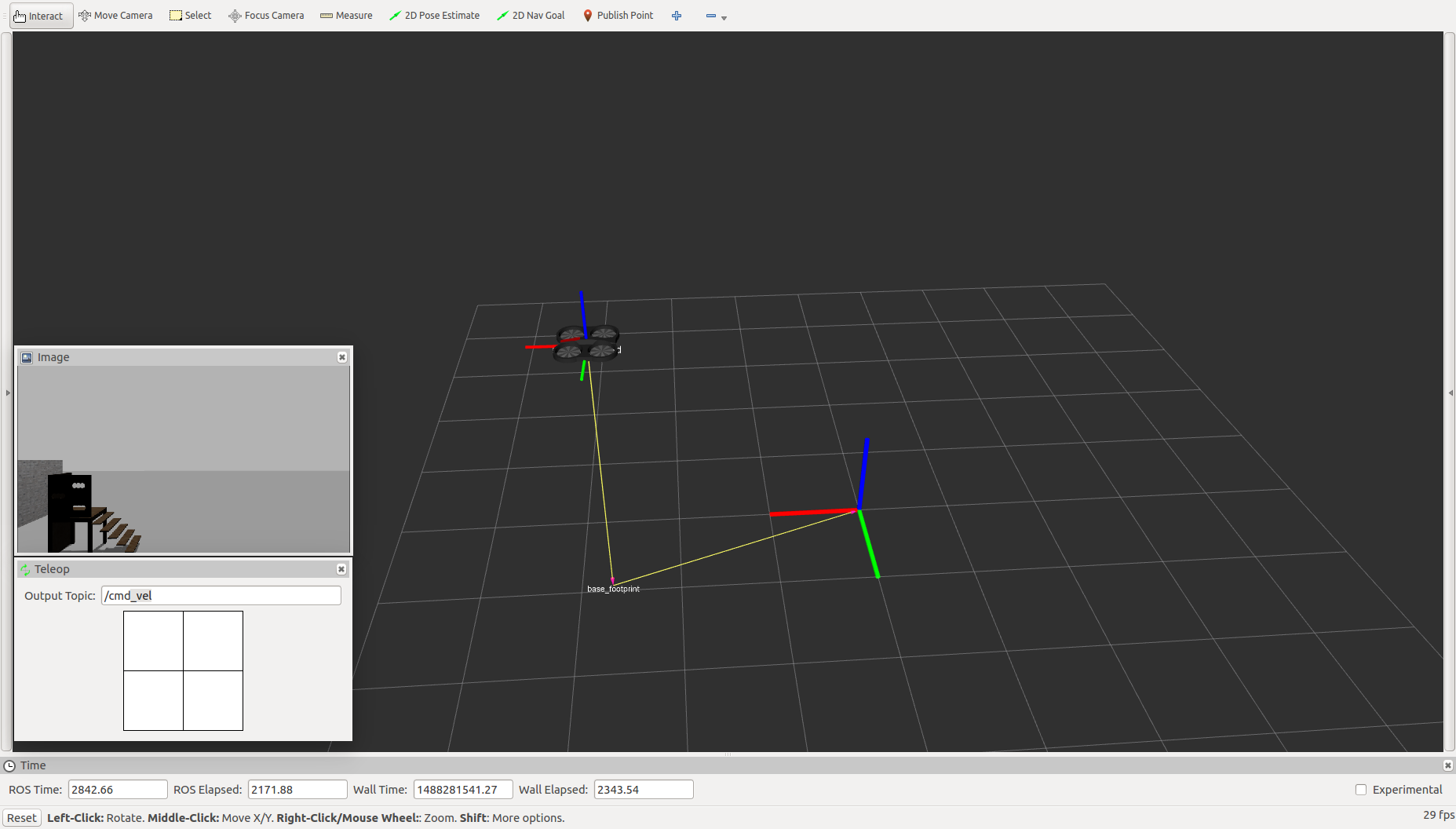
 То, что видит дрон до обработки: То, что видит дрон после обработки:

  
Не дело играть только с симулятором, поэтому я решил провести испытание на реальном дроне, которые, честно сказать, меня порадовали.

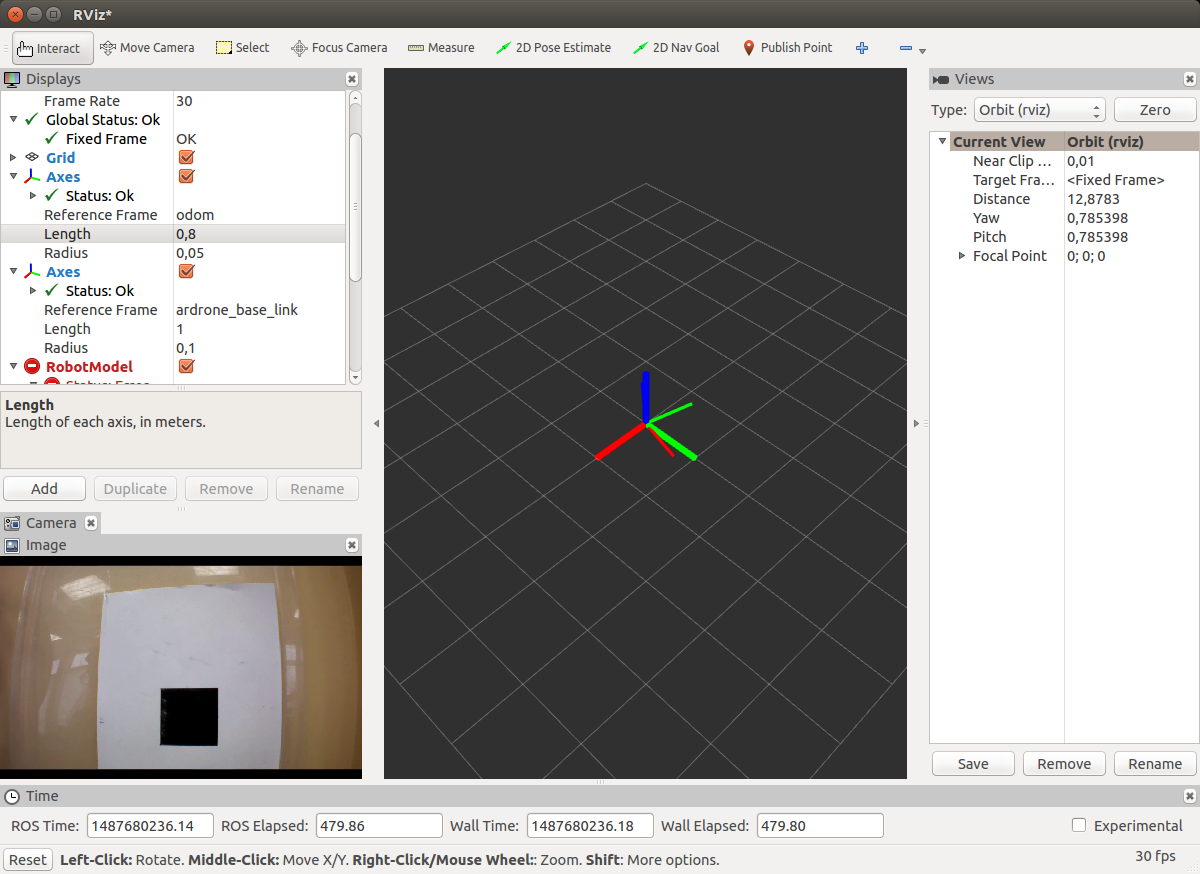
До обработки: После обработки:

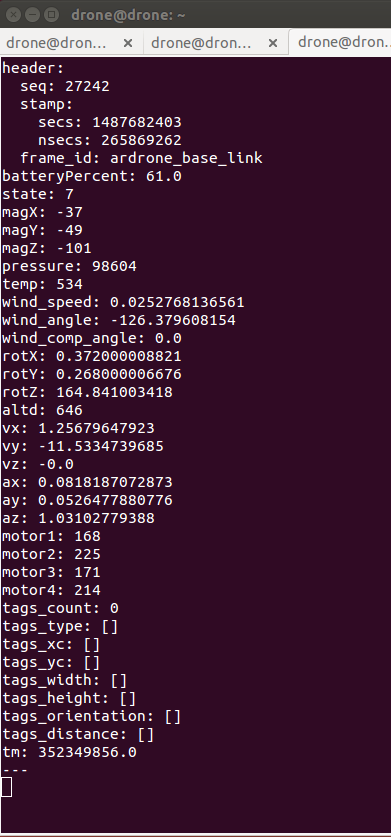
Программа находит контур квадрата и его цент, дальше находит цент самой камеры. Это необходимо, чтобы в дальнейшим квадрокоптер смог “следить” за объектом. Алгоритм будет следующий, если центр камеры не совпадает с центом квадрата, то с помощь PD регулятора дрон будет выравнивать свое положение в пространстве.

**Ориентация в пространстве**

Часть задачи мы выполнили, теперь необходимо получать какие-то данные о том, где сейчас находится наш дрон и в какую сторону сместился относительно точки отсчета. Вернемся опять к нашему симулятору. Для того, чтобы отслеживать перемещение в пространстве, я установил программу rviz.

Только на симуляторе отсутствовал топик /ardrone/odometry, я воспользовался пакетом tf, чтобы построить то, что вы видите выше.

Итак, я решил опробовать приобретенные знания на нашем квадрокоптере. После разбора этой программы я столкнулся с тем, rviz не отслеживает перемещения реального дрона, что поставило меня на некоторое время в тупик. Я смог получить только ориентацию осей с дрона, что показал справа на скриншоте. Даже модель не получилось подгрузить.

После я нашел отличный топик, который отслеживает положение дрона в пространстве на основе датчиков, которые стоят внутри него.

Справа на фотографии вы можете увидеть всё, что можно получить с этого топика. Я еще не успел практически проверить, на сколько эти результаты приближены к реальным, но думаю, что этого будет достаточно для того, чтобы получать адекватный отчет о том, где сейчас дрон находится.

Также еще необходимо сделать отрисовку пути, который прошел квадрокоптер. На данном этапа необходимо поглубже разобраться в программе rviz и пакете tf, чтобы сделать это.



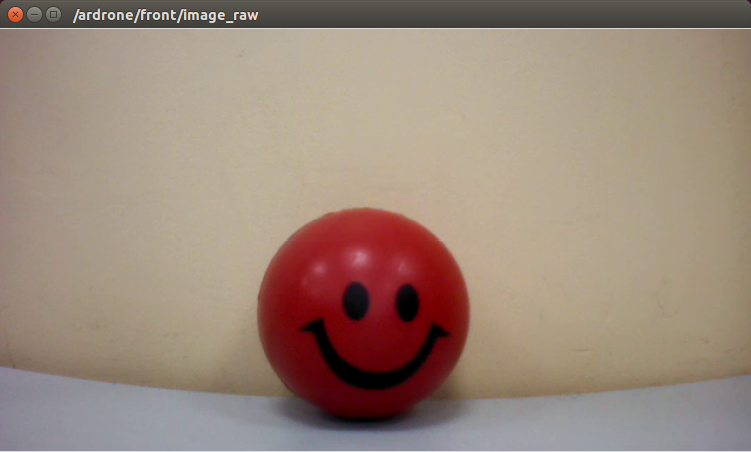
**Полетам по прямоугольнику**

Напоследок я захотел написать примитивный код для полета квадрокоптера по прямоугольнику без обратной связи. Его со временем сильно сносило, поэтому нужно дописать программу и добавить обратную связь. Приме кода, который заставлял летать дрона по некой траектории изображен справа изображен справа.

**Вывод**

За время проведения практики я усовершенствовал множество своих навыков, а также узнал много нового. Начнем с того, что я хоть и на базовом уровне, но научился работать с терминалом Ubuntu. Также узнал, что такое ROS и для чего он нужен, научился работать в нем и писать простые программы. Теперь перейдем непосредственно к теме моей практической работы, а именно к Parrot ARDrone 2.0. Я выполнил все задачи, поставленные мною перед началом практики. Научился управлять им через клавиатуру, консоль, джойстик. Получил видео с фронтальной и нижней камер, связал их с OpenCV, при этом обработав изображения и найдя контуры объектов, в данном случае трех кубов и квадрата на листе. Смог получить полное управление над линейной и угловой скоростями квадрокоптера, что в дальнейшем дает большие возможности в связке с OpenCV. Следующая задача, которую я планирую осуществить – это научить дрон “следовать” за черным квадратом, нарисованным на листе, о чем я, кстати, пару раз говорил в своем отчёте. Для этого мне необходимо лучше разобраться в работе ROS’a, компьютерного зрения и поднять уровень своего программирования, чем я уже и занимаюсь.

У этого проекта множество интересных задач, которые вполне реально реализовать. Начиная от слежения за каким-либо объектом, заканчивая созданием 3D модели местности, в которой находится наш квадрокоптер. Я надеюсь, что в дальнейшим я смогу реализовать хотя бы часть из них.



**Список литературы**

1. Ardrone\_autonomy API Documentation // ros : сайт. 2007. URL: <https://www.kickstarter.com/projects/podolabs/make-any-headphones-wireless-jack-by-podo-labs/description>
2. Мануал по AR.Drone 2.0 // wellbots : 2017 сайт. URL: <https://www.wellbots.com/content/ar.drone_user-guide_uk.pdf>
3. Hacking and securing the AR.Drone 2.0 quadcopter - Investigations for improving the security of a toy // researchgate : сайт. 2008 URL: <https://www.researchgate.net/publication/260420467_Hacking_and_securing_the_ARDrone_20_quadcopter_-_Investigations_for_improving_the_security_of_a_toy>
4. [Up and flying with the AR.Drone and ROS: Handling feedback](http://robohub.org/up-and-flying-with-the-ar-drone-and-ros-handling-feedback/) // robohub : сайт. 2017 URL: <http://robohub.org/tag/parrot-ar-drone-tutorial>
5. ROS driver for Parrot AR-Drone 1.0 and 2.0 quadrocopters // github : сайт. 2017 URL: <https://github.com/AutonomyLab/ardrone_autonomy>
6. Tf Tutorials // ros : сайт. 2007. URL: <http://wiki.ros.org/tf/Tutorials>
7. Rviz // ros : сайт. 2007. URL: <http://wiki.ros.org/rviz>
8. Tum\_simulator on Indigo // github : сайт. 2017 URL: <https://github.com/dougvk/tum_simulator>
9. Up and flying with the AR.Drone and ROS: Getting started // robohub : сайт. 2017 URL: <http://robohub.org/up-and-flying-with-the-ar-drone-and-ros-getting-started>
10. Parrot for developers // developer.parrot : сайт. 2016 URL: <http://developer.parrot.com>