

# Командный тур

Проход морских судов по акватории Арктики играет важную роль в экономике приарктических государств, в т.ч. России, Норвегии, Швеции, Канады и США. Оперативное наблюдение ледовой обстановки не только способствует безопасности судоходства, но и позволяет решать задачи точного планирования отгрузок.



Контроль движения судов и мониторинг ледовой обстановки осуществляется, в том числе, следующими инструментами:

- AIS (Automatic Identification System) - международная система определения положения судов, состоящая из бортовых станций на кораблях, наземных опорных станций и станций космического базирования.
- Данными космической съемки спутниками ДЗЗ, в т.ч.
  - Данные радарной съемки (в т.ч. например канадского RadarSat-2)
  - Обзорные данные с оптических мультиспектральных камер низкого разрешения (в т.ч. например аппаратов Aqua и Terra)
  - Данные с оптических камер (в первую очередь видимого и инфракрасного диапазона) высокого разрешения.

Участникам предлагаются компоненты задачи разработки проекта созвездия спутников ДЗЗ с оптическими и мультиспектральными камерами, предназначенным для мониторинга ледовой обстановки Северного Морского Пути.

## 5.1. День первый

### Задача 5.1.1. (35 баллов)

Сегодня вашей задачей станет предварительная подготовка к большой комплексной задаче. Ниже задачи разделены на блоки и даны небольшие инструкции как их выполнить. Вы можете выполнять задачи любым способом, не обязательно следовать им от и до, если вы знаете решение.

### Спутник

Необходимо решить базовые задачи:

- собрать КА;
- проверить работу его устройств, прежде всего:
  - вывод данных в веб-консоль;
  - исправная работа маховика;
  - исправная работа датчика угловой скорости (ДУС);
  - исправная работа магнитометра.
- заставить стабилизировать космический аппарат.

Рекомендуем использовать для программирования язык C.

В данных задачах вам поможет сайт: <http://www.orbcraft.sputnix.ru/doku.php?id=stabilization>

Исходник кода по стабилизации не возбраняется брать и адаптировать с сайта.

### Система управления полезной нагрузкой

Для получения снимков с камеры рекомендуем использовать Raspberry. Микрокомпьютер уже имеет карту памяти с предустановленной ОС. Также установлен пакет Motion, который имеет возможность транслировать потоковое видео с камеры, а также задавать конфигурации камеры. Однако для работы необязательно использовать именно его.

Для начала подключите Raspberry к ПК и попробуйте получить пару снимков с камеры.

Инструкция по подключению:

1. Сейчас вам понадобится сканер сетевых подключений. Для упрощения и ускорения работы с ним советуем отключить wifi.
  - 1.1. Откройте командную строку через комбинацию клавиш "window + r" и введите в поиске "cmd"
  - 1.2. В командной строке введите команду "ipconfig"
  - 1.3. Найдите строку с упоминанием ip4, здесь будет указан локальный ip, вашего компьютера, он пригодится вам далее.
2. Скачайте, установите и откройте сканер сетевых подключений. Наиболее по-

пулярен "advanced ip no в некоторых случаях куда быстрее работает "10-стрейк сканирование сети". Далее будет приведена инструкция к последней утилите.

- 2.1. Выберите самое левое изображение с лупой - "сканирование сети".
  - 2.2. Выберите "сканирование диапазона".
  - 2.3. В графе "интерфейс" вам нужно указать устройство с ip4 адрес из "ipconfg который вы нашли ранее.
  - 2.4. В графе "диапазон" вам нужно добавить диапазон, включающий ip4 вашего компьютера.  
Например для ip4 169.254.29.10 диапазон будет 169.254.0.0 - 169.254.255.255.  
Затем нужно убрать галочки или удалить все лишние диапазоны.
  - 2.5. В следующем окне вам нужно оставить галочку только на "icmp ping-ге другие убрать, цифры не трогать.
  - 2.6. Запускайте сканирование и ждите появления ip4 raspberrу. Обычно он находится до 40% загрузки. Если нет - обратитесь за помощью.
3. Скачайте, установите и откройте Putty.
- 3.0. Для упрощения перезагрузки программы после ошибки советуем закрепить её в стартовой панели.  
В графе host Name укажите pi<адрес ip4 из сканера>  
connection type: ssh  
Далее нажмите open.
4. Далее должна открыться unix-авая командная строка raspberrу.  
Пароль: raspberrу Это нормально, что при вводе вы не видите пароль.

## *Приготовление к оптике*

Изначально оптическая система будет настраиваться по источнику света. У вас есть необходимые детали для сборки и установки источника:

- корпус с 3д-пазлом
- патрон для лампочки
- вилка для подключения к сети

Необходимо собрать и установить источник на рабочий стол напротив.

## **Система оценки**

В первый день начисления баллов не происходит, так как рабочего командного времени крайне мало. Однако есть чек-лист, который может помочь вам скоординировать свою работу.

Задача	Отметка о выполнении
Спутник	
Спутник собран и соблюдена развесовка (при подвесе нет наклона спутника)	
Данные выводятся в web-консоль	
ДУС работоспособен и способен снимать показания	
Магнитометр работоспособен и способен снимать показания	
Маховик работоспособен	
Спутник осуществляет стабилизацию	
СУПН	
Осуществлено подключение ПК к Raspberry. Есть возможность войти в терминал или графический интерфейс.	

## Решение

### Спутник

#### 1. Проверка работоспособности устройств:

ДУС:

```

1 def control():
2     num = 1
3     print "Reset hyro #%d" % num
4     hyro_request_reset(num)
5     sleep(1)
6     print "Enable hyro #%d" % num
7     hyro_turn_on(num)
8     sleep(2)
9     print "Get RAW data from hyro #%d" % num
10    for i in range(0,10):
11        (ret, x, y, z) = hyro_request_raw(num)
12        if ret==0:
13            print "[%d] (x, y, z) = %d %d %d" % (i+1, x, y, z)
14        else:
15            print "[%d] Fail!" % (i+1)
16        sleep(1)
17    print "Disable hyro #%d" % num
18    hyro_turn_off(num)
19    pass

```

Магнитометр:

```

1 def control():
2     num = 1
3     print "Reset magnetometer #%d" % num
4     magnetometer_request_reset(num)
5     sleep(1)
6     print "Enable magnetometer #%d" % num
7     magnetometer_turn_on(num)
8     print "Get RAW data from magnetometer #%d" % num
9     for i in range(0,10):
10        (ret, x, y, z) = magnetometer_request_raw(num)
11        if ret==0:
12            print "[%d] (x, y, z) = %d %d %d" % (i+1, x, y, z)

```

```

13     else:
14         print "[%d] Fail!" % (i+1)
15         sleep(1)
16     print "Disable magnetometer #%d" % num
17     magnetometer_turn_off(num)
18     pass

```

Маховик:

```

1 def control():
2     num = 1
3     print "Reset motor #%d" % num
4     motor_request_reset(num)
5     sleep(1)
6     #print "Enable hyro #%d" % num
7     motor_turn_on(num)
8     #print "Get RAW data from hyro #%d" % num
9     (err, speed) = motor_set_speed(num, 3000);
10    sleep(15)
11    print(err, speed)
12    motor_set_speed(num, 0);
13    sleep(15);
14    (err, speed) = motor_set_speed(num, -3000);
15    sleep(5)
16    print(err, speed)
17    motor_set_speed(num, 0);
18    sleep(2);
19    motor_turn_off(num);
20    pass

```

## 2. Стабилизация

Пример кода для стабилизации:

```

1 def control():
2     hyro_turn_on(1)
3     motor_turn_on(1)
4     sleep(1)
5     kp=1
6     kd=12
7     motor_speed = 0
8     speed_hyro_old = 0
9     for i in range(300):
10        # снимаем показания с датчика угловой скорости и берём значение
11        # по оси z
12        speed_hyro_new = hyro_request_raw(1)[3]
13        # вычисляем значение скорости на маховик через изменение угловой
14        # скорости и предыдущее значение скорости маховика
15        motor_speed = kp*motor_speed - kd*(speed_hyro_new - speed_hyro_old)
16        #sleep(3)
17        # пишем ограничение скорости на маховик
18        if motor_speed > 4000:
19            motor_speed = 4000
20        if motor_speed < -4000:
21            motor_speed = -4000
22        # устанавливаем вычисленную скорость на маховик,
23        # предварительно округлив её до целого
24        motor_set_speed(1, int(motor_speed_new))
25        sleep(1)
26        # выводим в веб-консоль полученные значения
27        print('Угловая скорость: %d') % speed_hyro_new
28        print('Скорость маховика: %d') % motor_speed

```

```

29     motor_set_speed(1, 0)
30     motor_turn_off(1)
31     hyro_turn_off(1)

```

## СУПН

Первоначально подключение Raspberry к ПК осуществляется через патч-корд. Ниже приведен алгоритм действий как можно подключить Raspberry к ПК с GNU/Linux-системой.

1. Можно установить пакет, который поможет осуществить подключение Raspberry к вашему ПК. Например, это может быть dnsmasq. Для этого наберите в терминале:

```
terfire@TerFire:~$ sudo apt-get install dnsmasq-base
```

2. Далее не лишним будет создать ethernet-соединение, позволяющее подключаться устройствам к вашему ПК, используя диспетчер сети.
3. Используя dnsmasq можно определить IP, который имеет Raspberry в локальном подключении:

```
terfire@TerFire:~$ cat /var/lib/misc/dnsmasq.leases
1554302986 b8:27:eb:d8:f7:79 10.42.0.90 raspberrypi 01:b8:27:eb:d8:f7:79
```

IP вашей Raspberry в данном случае - 10.42.0.90

4. Далее подключимся к Raspberry по ssh:

```
terfire@TerFire:~$ ssh pi@10.42.0.90
pi@10.42.0.90's password:
Linux raspberrypi 4.14.98-v7+ #1200 SMP Tue Feb 12 20:27:48 GMT 2019 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Apr  3 14:50:39 2019 from 10.42.0.1
pi@raspberrypi:~$
```

Пароль по умолчанию: raspberry

Если всё прошло успешно имя пользователя сменится на pi@raspberrypi.

Теперь у нас есть доступ к терминалу Raspberry, и мы можем ей управлять.

## 5.2. День второй

Сегодняшний день в большей степени посвящён работе с полезной нагрузкой: необходимо собрать и провести первичную настройку оптической системы, а также научиться получать с неё снимки. Но не стоит забывать и про базовые задачи, которую должна решать спутниковая платформа. В этот день у вас добавляется также задача по моделированию группировки.

## Спутник

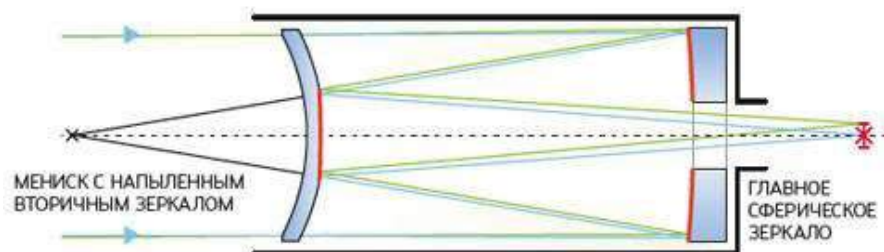
Первоначально доведите до конца задачу по стабилизации. Далее переходите к задаче ориентации. Как соориентировать спутник описано здесь:

<http://www.orbicraft.sputnix.ru/doku.php?id=lesson7>

Цель: иметь работоспособный алгоритм, который потом можно будет быстро подправить при изменении сборки КА.

## Оптика

Необходимо из оптических элементов осуществить сборку оптической системы. В основе оптической системы лежит схема Кассегрена-Максутова, в которой присутствует два оптических элемента (см. рисунок)



Такая схема используется в малых космических аппаратах (МКА), так как система зеркал позволяет значительно увеличить фокусное расстояние при сохранении небольших размеров. Однако возможна также сборка без мениска. В этом случае ваше изображение будет фокусироваться перед главным зеркалом.

Для сборки вам уже даны необходимые детали. Однако для того, чтобы закрепить камеру и иметь возможность получать четкое изображение, необходимо подобрать её фокусное расстояние. Для этого следуйте инструкции.

## Сборка телескопа

1. Посмотреть все детали и проверить что всё на месте. В комплекте должны быть:
  - а) вогнутое зеркало с отверстием посередине
  - б) выпукло-вогнутая линза (мениск) с напылением посередине
  - в) труба серая основная
  - г) труба главного зеркала: чёрная труба и задней стенкой, в задней стенке круглое отверстие по центру и три пары по равнобедренному треугольнику
  - д) опора для главного зеркала - диск с отверстиями почти такими же как в держателе зеркала, кроме трёх маленьких круглых по треугольнику и резьбой, и закрывашка для главного зеркала - кольцо с резьбой которая подходит к резьбе опоры.

- е) труба для мениска - чёрная труба с внутренней резьбой
  - ё) держатель для мениска - кольцо с наружной резьбой
  - ж) линейный переместитель (транслятор) - две чёрные детали которые как по рельсам ходят друг по другу, с хомутом для крепления камеры
2. Вставить мениск в трубу для мениска и накрутить держатель. Мениск должен быть обращён своей выпуклой стороной к главному зеркалу. На мениске есть бленда - светозащитная перегородка которая не пускает паразитный свет. Вставить трубу мениска в основную трубу.
  3. Вставить болты в 3 отверстия опоры главного зеркала, шляпка болта должна находиться внутри стенок опоры. Надеть пружинки на болты. Надеть трубу главного зеркала. Должно получиться так, что 3 пружинки зажаты между двумя деталями. Затянуть детали болтами, но не до упора.
  4. Закрутить три винта в круглые отверстия трубы по треугольнику, чтобы они давили на опору главного зеркала. Этими тремя винтами будет производиться юстировка главного зеркала
  5. Вставить трубу главного зеркала в основную серую трубу
  6. Вставить длинный болт в части линейного переместителя. Прикрутить винтами хомут для камеры. Вставить выступ линейного переместителя в соответствующее отверстие на трубе главного зеркала.

После завершения сборки необходимо завершить получить изображение двух точечных источников. Изображение будет оцениваться программой, написанной на Octave в автоматическом режиме.

### ***Юстировка***

После завершения сборки можно приступить к поиску изображения и юстировке. Перед юстировкой нужно убедиться что мениск сидит в трубе как можно центрированное (т. е. его оптическая ось должна быть по возможности параллельна оси трубы). Это можно сделать только на глаз, и только довольно грубо, но, тем не менее, посмотрите как оно всё ориентировано - чтобы в мениске не было явного перекоса, или с одной стороны он не зашёл и т. д.

Нужно установить телескоп в тёмную комнату, включить источник света и направить телескоп на источник. Первичное наведение можно сделать по тени телескопа. Когда телескоп наведён на источник, тень от трубы круглая, когда не наведён - эллиптическая. Сделайте тень от телескопа круглой и убедитесь, что в центре тени нет круглого светлого пятна.

Дальше нужно попробовать поймать изображение в телескопе. Для этого нужно взять экран (лист белой бумаги) и поставить его за выходным отверстием. Лист надо размещать на таком же расстоянии, на котором в будущем будет находиться камера. Если телескоп наведён, то на листке появится небольшое яркое пятно. Если оно не появляется, то нужно наводиться по мениску. Для наведения по мениску нужно аккуратно посмотреть на линзу телескопа со стороны источника света, не загораживая его. На линзе будет светлое пятно - отражение от главного (большого) зеркала попадает на линзу вместо вторичного (маленького) зеркала. Нужно подвигать телескоп или источник чтобы светлое пятно на линзе попало на вторичное (маленькое) зеркало. После того как пятно попадёт на вторичное зеркало, будет видно что оно



отражается от него. В идеале после отражения от вторичного зеркала свет попадает в отверстие главного зеркала. Если нет, этот свет будет виден по краям отверстия на главном зеркале, нужно подвигать телескоп чтобы он провалился.

После того как изображение попало на белый листочек, его нужно сфокусировать. Для этого нужно найти положение этого белого листочка при котором размер пятна минимален. Если изображение расфокусировано, то будет наблюдаться бублик - из-за того что в центре входного зрачка есть вторичное зеркало. Если изображение сфокусировано, то на листочке появятся тонкие детали источника. Можно поменять увеличение системы изменением положения мениска. Чтобы отодвинуть изображение от главного зеркала, нужно придвинуть мениск ближе к главному зеркалу. Чем дальше будет изображение от телескопа, тем больше увеличение.

Итак, после того как изображение поймано и сфокусировано, нужно вместо листа бумаги поставить камеру на то же самое место. Если быть совсем точным, нужно поставить именно кристалл приёмника веб-камеры в фокус, такой зеленоватый прямоугольничек с то место где находится изображение. Расстояние от этого приёмника до переднего края серебристой оправы вебкамеры составляет 2-3 см, нужно будет сфокусировать. Фокусировка производится движением камеры в хомуте (если его ослабить). Если размаха линейного переместителя не хватает для фокусировки, то нужно подвигать мениском, иными словами, грубая фокусировка производится перемещением мениска.

Если Вы собрали оптическую систему, то можно попробовать найти фокусное расстояние и получить чёткое изображение. Для этого необходимо:

- подключить камеру к ПК или Raspberry;
- установить камеру на переместитель;
- вывести изображение на экран ПК; для этого можете воспользоваться любым ПО, способным работать с веб-камерой, например VLC.

## *СУПН*

Первоначально убедитесь в том, что Raspberry подключается к вашему ПК.

После того, как мы научились подключаться к Raspberry пора попробовать получить данные с камеры.

1. Подключите камеру к Raspberry.
2. Можно использовать motion для того, чтобы проверить, получает ли ПК данные от камеры. Как это сделать вы можете увидеть, например, здесь: <http://academicfox.com/raspberry-pi-usb-web-kamera-potokovoe-vydeo-strym/>

Более подробно о функциях motion, а также о снимках (snapshots), можно прочитать здесь

[https://motion-project.github.io/motion\\_config.html#snapshot\\_filename](https://motion-project.github.io/motion_config.html#snapshot_filename)

После получения потокового видео в конфигурациях motion необходимо:

- задать разрешение 640 × 480
- убрать текстовые подписи с кадров

Вы можете использовать графический интерфейс или терминал для управления вашей Raspberry.

## *Проектирование группировки*

Разработка проекта созвездия в происходит в симуляторе «Орбита» и включает задачи:

1. Осуществить обзорную съемку всей зоны интереса с наименьшим временным диапазоном и минимальной стоимостью группировки
2. Осуществить съемку максимальной доли зоны интереса с наилучшим пространственным разрешением и минимальной стоимостью группировки

Сегодня и завтра доступна первая задача: съёмка указанной зоны с низким разрешением мультиспектральной камеры. Вашей команде доступна задача по ссылке: <https://nti.orbitagame.ru/>

Используя до 5 аппаратов, обеспечьте полное покрытие указанной зоны съемкой аппаратами с мультиспектральными камерами в ИК-диапазоне с ПР (пространственным разрешением) не хуже заданного.

Зона съемки: Область, ограниченная  $68^\circ$  и  $78^\circ$  северной широты,  $32^\circ$  и  $74^\circ$  восточной долготы, включающая в себя возможные маршруты судов между г. Мурманском и терминалом на м. Каменный (Обская губа).

Определение узлов зоны: Зона разбивается по широте с севера на юг на полосы шириной 10 км, остаток отбрасывается. Дальше каждая полоса разбивается по долготы с запада на восток на ячейки  $10 \times 10$  км, остаток отбрасывается. Назовем “узлом” центр каждой такой ячейки. Узел считается снятым, если след аппарата в надире прошел от него не далее, чем на расстоянии половины ширины полосы съемки. “Вес” узла =  $1/N$ , где  $N$  - количество узлов в зоне интереса.

Подзадачи:

- Снять не менее одного раза в сутки указанную зону с ПР не более 4000 м
- Снять не менее одного раза в сутки указанную зону с ПР не более 500 м
- Снять не менее одного раза в сутки указанную зону с ПР не более 250 м
- Снять указанную зону от 2 до 6 раз в сутки с ПР не более 4000 м
- Снять указанную зону от 2 до 6 раз в сутки с ПР не более 500 м
- Снять указанную зону от 2 до 6 раз в сутки с ПР не более 250 м

Старт работы КА на орбите начинается с 01 июня 2019 года с 00:00 по UTC. Первые три часа пролёта не учитываются.

## *Критерии оценивания*

- Первая в сутки съемка узла с разрешением не более 4000 м: 1 балл  $\times$  (вес узла)
- Первая в сутки съемка узла с разрешением не более 500 м: 3 балла  $\times$  (вес узла)
- Первая в сутки съемка узла с разрешением не более 250 м: 6 баллов  $\times$  (вес узла)
- Съемка узла от двух до шести раз в сутки с разрешением не более 4000 м: 2 балла  $\times$  (вес узла).

- Каждая последующая съемка узла от двух до шести раз в сутки с разрешением не более 500 м: 2 балла × (вес узла).
- Каждая последующая съемка узла от двух до шести раз в сутки с разрешением не более 250 м: 2 балла × (вес узла).

*Общие критерии оценки:*

- Проход над точкой менее, чем через 60 минут после предыдущего, не засчитывается.
- При условной стоимости группировки  $S$  свыше 10 единиц результат домножается на коэффициент  $10/S$ .
- При снижении аппарата на высоту 300 км и менее баллы перестают начисляться.
- Итоговая сумма баллов за задачу округляется до одного знака после запятой.
- Оценка за задачу равна сумме оценок за подзадачи.

Максимальное количество баллов: 40

Зона разбивается по широте с севера на юг на полосы 10 км, остаток отбрасывается. Дальше каждая полоса разбивается по долготе с запада на восток на ячейки  $10 \times 10$  км, остаток отбрасывается. В центре каждой ячейки - узел. Процент покрытия зоны определяется как процент снятых узлов от их общего числа. Узел считается снятым, если след аппарата прошел от нее не далее, чем на расстоянии половины ширины полосы съемки.

Параметры задачи:

	Обзорная съемка
Зона интереса	Область, ограниченная $68^\circ$ и $78^\circ$ северной широты, $32^\circ$ и $74^\circ$ восточной долготы, включающая в себя возможные маршруты судов между г. Мурманском и терминалом на м. Каменный (Обская губа)
Шаг сетки в зоне интереса	10 км
Требуемая частота съемки каждой точки в зоне интереса	Минимальная не менее 1 раза в сутки, рекомендуемая не менее 1 раза в 4 часа
Максимальное число аппаратов	5
Начало моделирования	01.06.2019 00:00 <sup>1</sup>
Период начисления баллов	с 01.06.2019 03:00 включительно по 02.06.2019 03:00 не включительно (1 сутки).

Фактическая зона маршрутов танкеров:



### Система оценки

Задача	Макс. балл	Балл команды
<b>Спутник</b>		
Сборка выполнена корректно, устройства в работоспособном состоянии	5	
Стабилизация: спутник в состоянии погасить начальную закрутку и продержаться в стабильном состоянии 40 секунд	10	
Ориентация: спутник способен повернуться на заданный угол и удерживать своё положение не менее 40 секунд	15	
<b>Оптика</b>		
Осуществлена сборка оптической системы	5	
Проведена юстировка системы	5	
С камеры выводится сфокусированное изображение	10	
<b>СУПН</b>		
Raspberry подключается к ПК	5	
Raspberry получает видеопоток с камеры	5	

## День второй, часть 2

### Интеграция оптики и СУПН

Получите фото с Raspberry и передайте его на свой ПК.

Интеграция СУПН и спутника

Подключите Raspberry к бортовой сети спутника. Это можно сделать несколькими способами:

1. Настройте обмен данными между Arduino и Raspberry. Например, можно использовать Serial, чтобы получать данные на Raspberry с Arduino. О том, как это можно сделать примерно описано здесь: <https://arduinoplus.ru/podkluchenie-raspberry-arduino/>  
Подключите к сети спутника Arduino, используя инструкцию: [http://www.orbcraft.sputnix.ru/doku.php?id=arduino\\_module\\_base\\_lesson](http://www.orbcraft.sputnix.ru/doku.php?id=arduino_module_base_lesson)  
Далее, используя шилд, подсоедините через Arduino свою Raspberry к сети спутника.
2. Попробуйте напрямую подключить свою Raspberry к БКУ любым способом.

### *Сохранение результатов*

Используя параметры входа зайдите в свое хранилище на сервере

Логин: RS№, где № - выданный вам порядковый номер.

Пароль: Password1

Адрес сервера в сети RS: 192.168.1.2

Под конец дня необходимо выгрузить на сервер:

- фото собранного спутника
- фото собранной оптической системы
- фото с камеры с двумя источниками тока
- код стабилизации
- код ориентации

### **Система оценки**

Задача	Макс. балл	Балл команды
<b>СУПН</b>		
Фото сохраняется на ваш ПК в автоматизированном режиме	10	
<b>Интеграция</b>		
Настроен обмен данными между Raspberry и БКУ	15	

Если задачи выполнены, можно приступать к общему монтажу всей системы. Наиболее сложной частью является монтаж оптической системы к корпусу спутника.

### *Решение*

#### *Спутник*

1. Калибровка магнитометра

2. Используя программу Magneto, необходимо определить калибровочные коэффициенты для магнитометра. После их определения необходимо будет задать функцию калибровки магнитометра:

```

1 def mag_calibrated(magx,magy,magz):
2     magx_cal = 1.06*(magx + -7.49) + -0.01*
3         (magy + -23.59) + 0.07*(magz + -108.24)
4     magy_cal = -0.01*(magx + -7.49) + 1.11*
5         (magy + -23.59) + 0.09*(magz + -108.24)
6     magz_cal = 0.07*(magx + -7.49) + 0.09*
7         (magy + -23.59) + 1.00*(magz + -108.24)
8     return magx_cal, magy_cal, magz_cal

```

Далее для корректного вычисления угла необходимо добавить небольшой пересчёт угла:

```

1 def angle_transformation(alpha, alpha_goal):
2     if alpha<=(alpha_goal - 180):
3         alpha = alpha + 360
4     elif alpha>(alpha_goal +180):
5         alpha = alpha - 360
6     return alpha

```

3. Код ориентации:

```

1 import math
2
3 kd = 200
4 # К-т дифференциальной обратной связи. Если угловая
5 # скорость спутника положительна, то спутник надо
6 # раскручивать по часовой стрелки, т.е. маховик надо разгонять
7 #по часовой стрелке.
8 kp = -50
9 # К-т пропорциональной обратной связи. Если текущий угол
10 # больше целевого, то спутник надо вращать против часовой стрелки,
11 # соответственно маховик надо разгонять против часовой стрелки.
12 time_step = 0.05 # Временной шаг работы
13
14 mtr_num = 1 # Номер маховика
15 hyr_num = 1 # Номер ДУС
16 mag_num = 1 # Номер магнитометра
17
18 def mag_calibrated(magx,magy,magz):
19     magx_cal = 1.06*(magx + -7.49) + -0.01*(magy + -23.59) +
20         0.07*(magz + -108.24)
21     magy_cal = -0.01*(magx + -7.49) + 1.11*(magy + -23.59) +
22         0.09*(magz + -108.24)
23     magz_cal = 0.07*(magx + -7.49) + 0.09*(magy + -23.59) +
24         1.00*(magz + -108.24)
25     return magx_cal, magy_cal, magz_cal
26
27 def angle_transformation(alpha, alpha_goal):
28     if alpha<=(alpha_goal - 180):
29         alpha = alpha + 360
30     elif alpha>(alpha_goal +180):
31         alpha = alpha - 360
32     return alpha
33
34 def motor_new_speed_PD(mtr_speed, alpha, alpha_goal, omega, omega_goal):
35     mtr_new_speed = int(mtr_speed + kp*(alpha-alpha_goal) +

```

```

36         kd*(omega-omega_goal))
37     return mtr_new_speed
38     # return 0
39
40 def initialize_all(): # Функция инициализации всех систем
41     print "Enable motor №", mtr_num
42     motor_turn_on(mtr_num)
43     sleep(1)
44
45     print "Enable angular velocity sensor №", hyr_num
46     hyro_turn_on(hyr_num) # Включаем ДУС
47     sleep(1)
48
49     print "Enable magnetometer", mag_num
50     magnetometer_turn_on(mag_num)
51     sleep(1)
52
53 def switch_off_all():
54     print "Finishing..."
55     print "Disable angular velocity sensor №", hyr_num
56     hyro_turn_off(hyr_num) # Выключаем ДУС
57     print "Disable magnetometer", mag_num
58     magnetometer_turn_off(mag_num)
59     motor_set_speed(mtr_num, 0)
60     sleep (1)
61     motor_turn_off(mtr_num)
62     print "Finish program"
63
64 def control():
65     initialize_all()
66     mtr_state = 0 # Инициализируем статус маховика
67     hyro_state = 0 # Инициализируем статус ДУС
68     mag_state = 0 # Инициализируем статус магнитометра
69     alpha_goal = 0 # Целевой угол
70     omega_goal = 0 # Целевая угловая скорость
71     mag_alpha = 0
72
73     for i in range(500):
74         # опрос датчиков и маховика
75         mag_state, magx_raw, magy_raw, magz_raw =
76             magnetometer_request_raw(mag_num)
77         hyro_state, gx_raw, gy_raw, gz_raw =
78             hyro_request_raw(hyr_num)
79         mtr_state, mtr_speed = motor_request_speed(mtr_num)
80
81         if not mag_state: # если магнитометр вернул код ошибки 0,
82             # т.е. ошибки нет
83             magx_cal, magy_cal, magz_cal =
84                 mag_calibrated(magx_raw,magy_raw,magz_raw)
85             magy_cal = - magy_cal
86             mag_alpha = math.atan2(magy_cal, magx_cal)/math.pi*180
87             print "magx_cal =", magx_cal, "magy_cal =", magy_cal,
88                 "magz_cal =", magz_cal
89             # Вывод откалиброванных значений магнитометра
90             print "mag_alpha atan2= ", mag_alpha
91         elif mag_state == 1:
92             print "Fail because of access error, check the connection"
93         elif mag_state == 2:
94             print "Fail because of interface error, check your code"
95

```

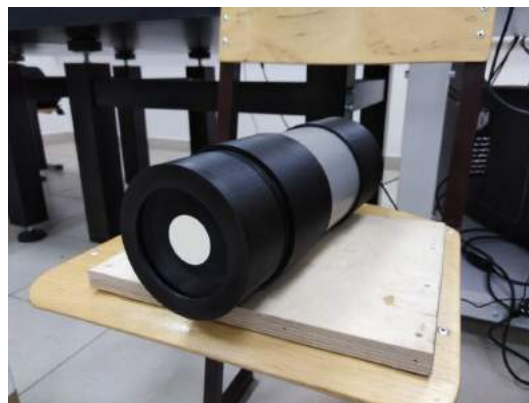
```

96         if not hyro_state:
97             # если ДУС вернул код ошибки 0, т.е. ошибки нет
98             gx_degs = gx_raw * 0.00875
99             gy_degs = gy_raw * 0.00875
100            gz_degs = gz_raw * 0.00875
101            omega = gz_degs
102            # если ДУС установлен осью z вверх, то угловая скорость
103            # спутника совпадает с показаниями ДУС по оси z
104            print "gx_degs =", gx_degs, "gy_degs =", gy_degs,
105                  "gz_degs =", gz_degs # Выводим данные
106        elif hyro_state == 1: # если датчик вернул сообщение об ошибке 1
107            print "Fail because of access error, check the connection"
108        elif hyro_state == 2: # если датчик вернул сообщение об ошибке 2
109            print "Fail because of interface error, check your code"
110
111        if not mtr_state:      # если маховик вернул код ошибки 0, т.е.
112                               #ошибки нет
113            print "Motor_speed: ", mtr_speed
114            mtr_new_speed = motor_new_speed_PD(mtr_speed,mag_alpha,
115                                                alpha_goal,gz_degs,omega_goal)
116            s# установка новой скорости маховика
117            motor_set_speed(mtr_num, mtr_new_speed)
118            sleep(time_step)
119        switch_off_all()

```



## Оптика

После сборки оптическая система выглядит следующим образом:



После первичной юстировки можно получить достаточно чёткие изображения с камеры, что невозможно изначально, так как у камеры отсутствует линза и автофокус.



Оригинальная картинка, выставленная на экране смартфона:	Картинка с камеры, полученная на расстоянии 4 метра:
	

## СУПН

Используя пакет `motion` можно получить изображение с камеры. Для удобства настройки оптической системы для начала наладим потоковое видео с камеры.

1. Подключаемся к Raspberry
2. Заходим в файл с конфигурациями `motion`:

```
pi@raspberrypi:~ $ sudo nano /etc/motion/motion.conf
```

Далее настраиваем вывод потокового видео, задавая следующие настройки:

```
start_motion_daemon = yes
```

```
...
```

```
stream_localhost off
```

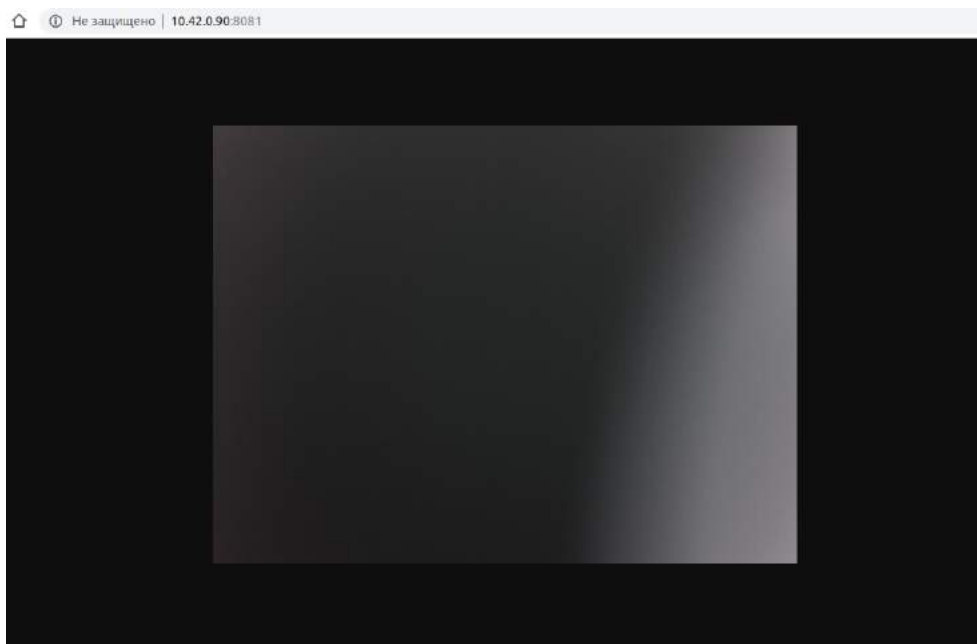
```
...
```

```
webcontrol_localhost off
```

3. Далее, закрыв файл, включаем трансляцию:

```
pi@raspberrypi:~ $ sudo service motion start
```

Если всё успешно, то в вашем браузере по адресу `http://10.42.0.90:8081/` (где 10.42.0.90 - это IP самой Raspberry) начнётся трансляция видео.



Так как камера не имеет автофокуса и линзы, изображение будет расфокусированным.

4. Ваши фото сохраняются по адресу, заданному в файле `motion.conf` строкой `target_dir`. По умолчанию это `home/pi/motion`. Вы можете изменить директорию на более удобную. Задав параметры `motion.conf` можно сделать так, что снимок будет делаться по команде:

```
pi@raspberrypi:~/Test1 $ scp photo.jpg terfire@192.168.88.156:\ОПТИ/Test1
```

И сохраняться в указанную вами директорию.

5. Перед отладкой автоматизированной передачи получаемого камерой снимка на ваш ПК рекомендуется наладить беспроводное соединение между вашим ПК и Raspberry. Самый простой способ - открыв файл `wpa_supplicant.conf` и прописав там логин и пароль от сети, в которой находится ваш ПК.
6. Самый простой способ передать файл - использовать команду `scp`.
- 6.1. Необходимо открыть для передачи `ssh`-порт на вашем ПК. Можно использовать `uwf`.
- 6.2. После открытия порта будет работать команд `scp`:

```
pi@raspberrypi:~/Test1 $ scp photo.jpg terfire@192.168.88.156:\ОПТИ/Test1
```

По указанной директории вы увидите свой файл.

7. Теперь необходимо автоматизировать процесс. Можно написать небольшой `python`-скрипт, который будет осуществлять указанные действия самостоятельно.

### *Проектирование группировки*

Оптимального решения можно достичь, используя два спутника. Для спутника, размещаемого на орбите, можно выбрать один из трёх типов камеры:

Тип и диапазон	Диапазон съемки	Ширина полосы при размещении на высоте 500 км, в км	Пространственное разрешение на высоте 500 км, в м	Условная стоимость аппарата с камерой этого типа
LR1 / ИК	МС/ИК	1 250	600	5
LR2 / ИК	МС/ИК	313	150	5
LR3 / ИК	МС/ИК	156	75	7.5

Задача заключается прежде всего в том, чтобы подобрать орбиту таким образом, чтобы она покрыла максимум зоны с допустимым разрешением. Разрешение, в свою очередь ограничивает высоту орбиты.

*Пример решения:*

Известно, что необходимо заснять зону, ограниченную  $68^\circ$  и  $78^\circ$  северной широты,  $32^\circ$  и  $74^\circ$  восточной долготы. Отсюда делаем вывод, что наклонение орбиты не должно быть меньше  $78^\circ$ .

Очевидно, что чем больше высота, тем хуже разрешение, но тем больше ширина полосы.

Рассмотрим случай, когда один спутник находится на низкой орбите. Чем меньше высота орбиты, тем чаще спутник будет делать обороты вокруг Земли. При низких орбитах стоит рассматривать камеру LR1, которая позволит снимать с приемлемым разрешением и достаточно большой шириной захвата.

The image shows a web browser window with the address bar displaying "localhost:3000/#". The page content is as follows:

**Спутник 2**

Наклонение [°]

Большая полуось [км]

Эксцентриситет

Долгота восходящего узла [°]

Аргумент перицентра [°]

Аномалия

**Аномалия: Истинная аномалия**

Истинная аномалия [°]

Камера

Размер пикселя [мм]

Фокусное расстояние [мм]

Размеры сенсора [мм]

Стоимость [баллы]

Описание решения

Решение

Спутники

Спутник 1

Наклонение [°]

Большая полуось [км]

Эксцентриситет

Долгота восходящего узла [°]

Аргумент перицентра [°]

Аномалия

Аномалия: Истинная аномалия

Истинная аномалия [°]

Камера

Размер пикселя [мм]

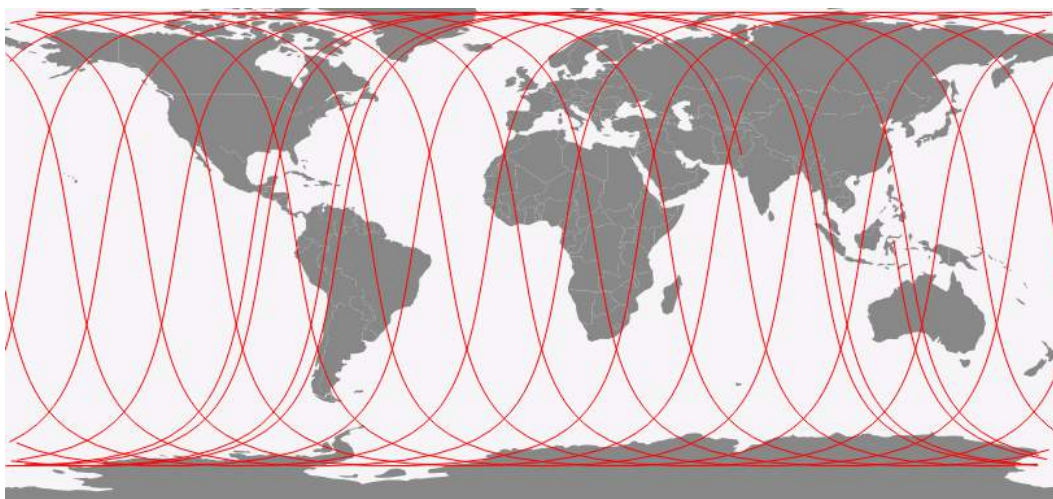
Фокусное расстояние [мм]

Размеры сенсора [мм]

Стоимость [баллы]

При подобногo рода орбитах получаем проход над интересующей нас зоной:





## Интеграция

Самым простым способом является использование стандартной библиотеки спутника, использующейся для работы с подключаемой к его сети Arduino:

```
arduino_send(0, 1, None, 100)
```

0 - номер Arduino;

1 - идентификатор сообщения;

None - передаваемые данные;

100 - ожидание ответа в мс.

Схема взаимодействия строится следующим образом:

1. В БКУ на исполнение запускается программа, содержащая код:

```
1 def control():
2     arduino_send(0, 1, None, 100)
```

2. На ардуино скетч ждёт команду от БКУ и выводит данные в Serial:

```
1 #include <OrbcraftBus.h>
2 // подключаем библиотеку для работы с конструктором ОрбиКрафт
3
4 /*
5  * Объявим переменную msg как тип данных Message
6  * Message - представляет собой структуру, содержащую
7  * идентификаторы и данные передаваемого сообщения
8  */
9 Message msg;
10
11 /*
12  * Объявим переменную bus как тип данных OrbcraftBus
13  * OrbcraftBus - представляет собой класс, описывающий взаимодействие
14  * Arduino и шины конструктора Orbcraft
15  */
16 OrbcraftBus bus;
17
18 // Объявим переменную msgSize, в которую будет записываться размер
19 // принятого сообщения
20 uint16_t msgSize = 0;
21
22 void setup() {
23     Serial.begin(9600); // задаем скорость обмена информацией по Serial
```

```

24 }
25
26 void loop() {
27   msgSize = bus.takeMessage(msg); // пробуем прочитать сообщение с помощью метода
28                                   // takeMessage
29   Serial.println(msgSize);
30   delay(1000);
31 }
32
33 void serialEvent2(){
34   bus.serialEventProcess();
35 }

```

3. На Raspberry пишем python-скрипт, который читает Serial:

```

1 import serial
2 import time
3
4 ser=serial.Serial("/dev/ttyACM0",9600)
5 ser.baudrate=9600
6
7 read_ser=ser.readline()
8 print(read_ser)

```

В итоге в терминале после запуска созданного вами файла с расширением .py, содержащего код, представленный выше, будет выводиться цифровое значение.

### 5.3. День третий

Пришло время интегрировать спутник, полезную нагрузку и СУПН. Следуйте следующему алгоритму действий:

1. Настройте беспроводную передачу данных с Raspberry
2. Соедините оптику и камеру
3. Соедините Raspberry и спутник
4. Присоедините к корпусу спутника оптическую систему
5. Допишите алгоритм ориентации так, чтобы при повороте на заданный градус спутник передавал фото на ваш ПК

#### *Настройка беспроводной сети Raspberry*

Для настройки беспроводного подключения можно использовать несколько вариантов:

1. Использовать беспроводную сеть, которую раздает БКУ вашего аппарата. В этом случае в вашей сети будет 3 устройства - БКУ, ЦУП и СУПН.
2. Этот способ наиболее предпочтителен, так как все устройства находятся в одной подсети.
3. Использовать сеть RS для беспроводного подключения СУПН. Этот способ не автономен и требует постоянного присутствия внешней сети. Также, многие компьютеры не способны одновременно подключиться к нескольким беспроводным сетям. По запросу мы можем фиксировать автоматически выдаваем-

мый адрес устройства на роутере. Этот способ удобен для настройки СУПН, однако после настройки конфигурации попытайтесь перейти к способу №1.

4. Вы создаете собственную локальную подсеть с помощью компьютера или мобильного устройства. Этот способ так-же не автономен. Кроме того, многие компьютеры не способны одновременно подключаться к сети БКУ и отдельной сети, в которой будет ваша СУПН. Этот способ возможно использовать как промежуточное решение, перейдя в последствии к способу №1.
5. СУПН может раздавать собственную беспроводную сеть, к которой подключится ЦУП. Этот способ возможен, однако не рекомендуется в виду нестабильной работы. Также, вы вероятно не сможете одновременно управлять БКУ и СУПН из ЦУП.

### *Интеграция оптики и СУПН*

Получите фото с Raspberry, и передайте его на свой ПК.

Ваша задача сохранять изображения, полученные камерой с системой оптических элементов в памяти СУПН, и далее передавать их на компьютер с которого осуществляется управление БКУ (ЦУП). При сохранении файлов используйте уникальный идентификатор вашей команды (номер или буквенно-цифровой шифр) а также уникальный идентификатор снимка (дату и время или буквенно-цифровой шифр в описанном вами формате).

Полученные изображения должны быть отправлены в реальном времени на управляющий БКУ компьютер, с использованием стандартных сетевых протоколов (HTTP, UDP, FTP, SFTP, и других)

### *Интеграция СУПН и спутника*

Подключите Raspberry к бортовой сети спутника. Ваша задача - делать снимки камерой, подключенной к модулю СУПН по команде от управляющей программы БКУ. Также, при сохранении снимков в названии файлов нужно использовать уникальный идентификатор, получаемый от БКУ.

Эту задачу возможно сделать несколькими способами:

*Способ 1.* Настройте обмен данными между спутником и Raspberry через Arduino: Например, можно использовать подключение Arduino к сети спутника через переходной шилд, а подключение Arduino к Raspberry осуществить через USB.

Подключите к сети спутника Arduino, используя инструкцию:  
[http://www.orbcraft.sputnix.ru/doku.php?id=arduino\\_module\\_base\\_lesson](http://www.orbcraft.sputnix.ru/doku.php?id=arduino_module_base_lesson).

При подключении Arduino и Raspberry использовать Serial на Arduino и ttyUSB0 со стороны Raspberry, чтобы получать данные на Raspberry с Arduino. О том, как это можно сделать примерно описано здесь: <https://arduinoplus.ru/podkluchenie-raspberry-arduino/>

*Способ 2.* Попробуйте напрямую подключить свою Raspberry к БКУ используя шнур конвертор шины rs485 - USB (как правило ttyUSB0 со стороны Raspberry) с дополнительным питанием для Raspberry от шины спутника. В этом случае разбор протокола обмена данными в бортовой шине аппарата ложится на Raspberry. Для разбора протокола шины спутника используйте примеры и исходные коды библиотек с сай-



та: [http://www.orbicraft.sputnix.ru/doku.php?id=arduino\\_module\\_base\\_lesson](http://www.orbicraft.sputnix.ru/doku.php?id=arduino_module_base_lesson)

### *Полная интеграция*

Полная интеграция включает в себя сборку компонентов в единую систему и выстраивание циклограммы работы космического аппарата. Необходимо:

- прикрепить оптическую систему к КА
- включить в бортовую систему КА СУПН, при этом камера должна быть как элементом оптической системы, так и управляться Raspberry
- выстроить работу КА

#### Монтаж оптической системы

Необходимо прикрепить собранную оптическую систему к спутнику. Для этого у вас есть крепления для трубы (пара хомутов), которые можно прикрепить к спутнику. При креплении оптики к вашему спутнику учтите тонкости весовой балансировки.

Учтите, что есть возможность изготовить детали путём лазерной резки и 3д-печати. Уточняйте у организаторов по поводу того, как это сделать.

#### Циклограмма работы КА

1. Уже в полной сборке ваш спутник должен уметь ориентироваться на заданный угол. Для этого вам необходимо будет поправить уже имеющийся у вас код ориентации.
2. В вашей программе управления спутником после выполнения ориентации СУПН должна получать снимок. Для этого вам необходимо доработать обмен данными БКУ и Raspberry, а также интегрировать сюда ваше решение по автоматизированному получению снимка.
3. После того, как СУПН сделает снимок, она должна передать его на ЦУП. В роли ЦУПа выступает ваш ПК.

Итогом выполнения программы управления вашим устройств должен стать снимок объекта, на который наводится ваш КА. Снимок должен быть передан на ЦУП.

### *Проектирование группировки*

Разработка проекта созвездия в происходит в симуляторе «Орбита» и включает задачи:

1. Осуществить обзорную съемку всей зоны интереса с наименьшим временным диапазоном и минимальной стоимостью группировки
2. Осуществить съемку максимальной доли зоны интереса с наилучшим пространственным разрешением и минимальной стоимостью группировки

Сегодня становится доступна вторая задача в симуляторе. Отличие от предыдущей в том, что необходимо снять узлы прохода с максимально хорошим разрешением. Покрытие зоны при этом ограничено набором зон интереса, координаты которых указаны в условии.

Используя до 10 аппаратов с камерами для детального наблюдения, обеспечьте максимальное покрытие указанной зоны с наилучшим качеством по наименьшей цене при съемке в видимом диапазоне.

Зона съемки: области (широта-широта и долгота-долгота):

- 68-69, 73-74
- 70-71, 73-74
- 73-74, 70-72
- 71-72, 64-65
- 70-71, 57-59
- 70-71, 48-49
- 69-70, 43-44
- 69-70, 37-38
- 68-69, 33-35

Определение “узлов” зоны: Зона разбивается по широте с севера на юг на полосы шириной 1 км, остаток отбрасывается. Далее каждая полоса разбивается по долготу с запада на восток на ячейки  $1 \times 1$  км, остаток отбрасывается. Назовем “узлом” центр каждой такой ячейки. Узел считается снятым, если след аппарата в надире прошел от него не далее, чем на расстоянии половины ширины полосы съемки. “Вес” узла  $= 1/N$ , где  $N$  - количество узлов в зоне интереса.

Подзадачи:

- Снять не менее одного раза в сутки указанную зону с ПР не более 20 м.
- Снять не менее одного раза в сутки указанную зону с ПР не более 10 м.
- Снять не менее одного раза в сутки указанную зону с ПР не более 3 м.
- Снять указанную зону от 2 до 6 раз за время симуляции с ПР не более 20 м.
- Старт работы КА на орбите начинается с 01 июня 2019 года с 00:00 по UTC. Первые три часа полёта не учитываются.

Примечание: Период начисления баллов с 01.06.2019 03:00 по 03.06.2019 03:00 по времени симуляции.

### ***Критерии оценивания***

- Первая за двое суток съемка узла с разрешением съемки не более 20 м: 10 баллов  $\times$  (вес узла).
- Первая за двое суток съемка узла с разрешением съемки не более 10 м: 20 баллов  $\times$  (вес узла).
- Первая за двое суток съемка узла с разрешением съемки не более 3 м: 30 баллов  $\times$  (вес узла).
- Съемка узла от двух до шести раз за время симуляции с разрешением не более 20 м: 4 балла  $\times$  (вес узла) (за каждое).

*Общие критерии оценки:*

- Проход над точкой менее, чем через 60 минут после предыдущего, не засчитывается.
- При условной стоимости группировки  $S$  свыше 10 единиц результат домножается на коэффициент  $10/S$ .
- При снижении аппарата на высоту 300 км и менее баллы перестают начисляться.
- Итоговая сумма баллов за задачу округляется до одного знака после запятой.
- Оценка за задачу равна сумме оценок за подзадачи.

Максимальное количество баллов: 80.

Балл за первую задачу идет в зачет второго дня. Балл за вторую - в зачет этого дня. Балл, полученный за решение задачи в симуляторе суммируется с баллами, полученными вами по листу критериев оценки.

### *Сохранение результатов*

Крайне важно сохранить результаты работы дня на сервере в папке day01.

0. Нам важно понимать, как именно вы решаете задачу, поэтому записывайте команды, которые используете при работе с raspberri в отдельный файл.
- 0.a) Простой вариант. В любом виде и порядке записывайте команды в обычный doc или txt файл.
- 0.б) Создайте bash-файл по каждому заданию. Например:

```
sudo touch algo.sh
vim algo.sh
```

Далее вам откроется окно текстового редактора vim. для удобства открывайте его в отдельном окне терминала.

Файл нужно начать со строки

```
#!/bash
```

Попробуйте записывать в этот файл команды, которые вы использовали для решения задания. Например при подключении к raspberri с компьютера на линуксе можно создать такой файл: `#!/bash`

```
ifconfig
```

```
sudo nmap -sn 10.42.0.0/24
```

```
sudo ssh pi@10.42.0.55
```

Также, стоит сохранить финальные скетчи программ для Arduino и программы выполняемые на Raspberry. Сопроводите их минимальным описанием ожидаемой логики работы, в текстовом файле readme.txt

1. Загрузите фото собранного спутника
2. Загрузите описание алгоритма работы с Raspberry, касающегося получения и передачи фото на ПК
3. Загрузите финальный рабочий код работы спутника
4. Не забудьте также загрузить финальный снимок, полученный с КА

### **Система оценки**

Задача	Макс. балл	Балл команды
Спутник		
Стабилизация: спутник в состоянии погасить начальную закрутку и продержаться в стабильном состоянии 40 секунд	5	
Ориентация: спутник способен повернуться на заданный угол и удерживать своё положение не менее 40 секунд	10	
СУПН		
Настроено беспроводное соединение	5	
Фото сохраняется на ваш ПК в автоматизированном режиме	10	
Интеграция СУПН и спутника		
Настроен обмен данными между Raspberry и БКУ	15	
По сигналу от БКУ Raspberry делает фото и передаёт его на ПК	25	
Полная интеграция		
Осуществлена полная сборка компонентов, компоненты запитаны и находятся в работоспособном состоянии	15	
Решение комплексной задачи: <ul style="list-style-type: none"> <li>● спутник ориентируется с оптической системой на заданный угол</li> <li>● после ориентации спутник делает снимок</li> <li>● снимок передаётся на ПК</li> <li>● полученный снимок не является размытым</li> </ul>	50	

## Решение

### Интеграция СУПН и спутника

Интеграция заключается в адаптации написанных ранее решений и сборке устройств в единую сеть.

#### 1. Код для БКУ:

```

1 import math
2
3 kd = 200
4 # К-т дифференциальной обратной связи. Если угловая скорость
5 # спутника положительна, то спутник надо раскручивать по
6 # часовой стрелки, т.е. маховик надо разгонять по часовой стрелке.
```

```

7  kp = -50
8  # K-т пропорциональной обратной связи. Если текущий угол
9  # больше целевого, то спутник надо вращать против часовой стрелки,
10 # соответственно маховик надо разгонять против часовой стрелки.
11 time_step = 0.05          # Временной шаг работы
12
13 mtr_num = 1                # Номер маховика
14 hyr_num = 1                # Номер ДУС
15 mag_num = 1                # Номер магнитометра
16
17 def mag_calibrated(magx,magy,magz):
18     magx_cal = 1.06*(magx + -7.49) + -0.01*(magy + -23.59) +
19             0.07*(magz + -108.24)
20     magy_cal = -0.01*(magx + -7.49) + 1.11*(magy + -23.59) +
21             0.09*(magz + -108.24)
22     magz_cal = 0.07*(magx + -7.49) + 0.09*(magy + -23.59) +
23             1.00*(magz + -108.24)
24     return magx_cal, magy_cal, magz_cal
25
26 def angle_transformation(alpha, alpha_goal):
27     if alpha<=(alpha_goal - 180):
28         alpha = alpha + 360
29     elif alpha>(alpha_goal +180):
30         alpha = alpha - 360
31     return alpha
32
33 def motor_new_speed_PD(mtr_speed, alpha, alpha_goal,
34                       omega, omega_goal):
35     mtr_new_speed = int(mtr_speed + kp*(alpha-alpha_goal) +
36                       kd*(omega-omega_goal))
37     return mtr_new_speed
38     # return 0
39
40 def initialize_all(): # Функция инициализации всех систем
41     print "Enable motor №", mtr_num
42     motor_turn_on(mtr_num)
43     sleep(1)
44
45     print "Enable angular velocity sensor №", hyr_num
46     hyro_turn_on(hyr_num) # Включаем ДУС
47     sleep(1)
48
49     print "Enable magnetometer", mag_num
50     magnetometer_turn_on(mag_num)
51     sleep(1)
52
53 def switch_off_all():
54     print "Finishing..."
55     print "Disable angular velocity sensor №", hyr_num
56     hyro_turn_off(hyr_num) # Выключаем ДУС
57     print "Disable magnetometer", mag_num
58     magnetometer_turn_off(mag_num)
59     motor_set_speed(mtr_num, 0)
60     sleep (1)
61     motor_turn_off(mtr_num)
62     print "Finish program"
63
64 def control():
65     initialize_all()
66     mtr_state = 0          # Инициализируем статус маховика

```

```

67     hyro_state = 0           # Инициализируем статус ДУС
68     mag_state = 0           # Инициализируем статус магнитометра
69     alpha_goal = 0          # Целевой угол
70     omega_goal = 0          # Целевая угловая скорость
71     mag_alpha = 0
72
73     while true:
74         # опрос датчиков и маховика
75         mag_state, magx_raw, magy_raw, magz_raw =
76             magnetometer_request_raw(mag_num)
77         hyro_state, gx_raw, gy_raw, gz_raw =
78             hyro_request_raw(hyr_num)
79         mtr_state, mtr_speed = motor_request_speed(mtr_num)
80
81         if not mag_state: # если магнитометр вернул код ошибки 0,
82                             # т.е. ошибки нет
83             magx_cal, magy_cal, magz_cal =
84                 mag_calibrated(magx_raw, magy_raw, magz_raw)
85             magy_cal = - magy_cal
86             mag_alpha = math.atan2(magy_cal, magx_cal)/math.pi*180
87             print "magx_cal =", magx_cal, "magy_cal =", magy_cal,
88                   "magz_cal =", magz_cal
89                 # Вывод откалиброванных значений магнитометра
90             print "mag_alpha atan2= ", mag_alpha
91         elif mag_state == 1:
92             print "Fail because of access error, check the connection"
93         elif mag_state == 2:
94             print "Fail because of interface error, check your code"
95
96         if not hyro_state: # если ДУС вернул код ошибки 0,
97                             # т.е. ошибки нет
98             gx_degs = gx_raw * 0.00875
99             gy_degs = gy_raw * 0.00875
100            gz_degs = gz_raw * 0.00875
101            omega = gz_degs
102            # если ДУС установлен осью z вверх, то угловая скорость
103            # спутника совпадает с показаниями ДУС по оси z
104            print "gx_degs =", gx_degs, "gy_degs =", gy_degs, "gz_degs =",
105                  gz_degs # Выводим данные
106        elif hyro_state == 1: # если датчик вернул сообщение об ошибке 1
107            print "Fail because of access error, check the connection"
108        elif hyro_state == 2: # если датчик вернул сообщение об ошибке 2
109            print "Fail because of interface error, check your code"
110
111        if not mtr_state:      # если маховик вернул код ошибки 0,
112                                # т.е. ошибки нет
113            print "Motor_speed: ", mtr_speed
114            mtr_new_speed = motor_new_speed_PD(mtr_speed, mag_alpha,
115                                                alpha_goal, gz_degs, omega_goal)
116            # установка новой скорости маховика
117            motor_set_speed(mtr_num, mtr_new_speed)
118            if math.fabs(alpha - alpha_goal) < 0.5:
119                arduino_send(0, 1, None, 100)
120                sleep(time_step)
121                switch_off_all()

```

2. Код для Arduino можно оставить таким же, как и в предыдущем пункте.

3. Пример кода для Raspberry:

```
1 #!/usr/bin/env python
```

```
2
3 import serial
4 import time
5 import os
6
7 ser=serial.Serial("/dev/ttyACM0",9600)
8 ser.baudrate=9600
9
10 read_ser=ser.readline()
11 print(read_ser)
12 if read_ser!='0':
13     while True:
14         os.system ("sudo curl -s -o /dev/null http://192.168.88.67:\
15             8080/0/action/snapshot")
16         time.sleep(0.5)
17         os.system ("sudo scp lastsnap.jpg terfire@192.168.88.156:\ОНТИ/Test1")
```

Связка из трёх устройств будет работать, если одновременно запустить на исполнение код БКУ и код на Raspberry.

### *Полная интеграция*

На данном этапе остаётся собрать все устройства в единую сеть, осуществить крепление оптической системы к спутнику и произвести юстировку.

### *Проектирование группировки*

Даны области съёмки:

1. 68-69, 73-74
2. 70-71, 73-74
3. 73-74, 70-72
4. 71-72, 64-65
5. 70-71, 57-59
6. 70-71, 48-49
7. 69-70, 43-44
8. 69-70, 37-38
9. 68-69, 33-35

И камеры с соответствующими характеристиками:

Тип и диапазон	Диапазон съемки	Ширина полосы при размещении на высоте 500 км, в км	Пространственное разрешение на высоте 500 км, в м	Условная стоимость аппарата с камерой этого типа
HR1 / ВД	ВД	44	8	2
HR2 / ВД	ВД	23	4	2
HR3 / ВД	ВД	16	3	3
HR4 / ВД	ВД	88	8	4
HR5 / ВД	ВД	47	4	4
HR6 / ВД	ВД	32	3	6

*Пример решения:*

The screenshot shows a web browser window displaying a configuration page for a satellite named "Спутник 5". The page contains several input fields and buttons for setting orbital and camera parameters.

**Спутник 5**

Наклонение [°]: 74

Большая полуось [км]: 6740

Эксцентриситет: 0

Долгота восходящего узла [°]: 287

Аргумент перицентра [°]: 0

Аномалия: **Истинная аномалия**

Аномалия: Истинная аномалия

Истинная аномалия [°]: 0

Камера: **LR2**

Размер пикселя [мм]: 240

Фокусное расстояние [мм]: 800

Размеры сенсора [мм]: 500 500

Стоимость [баллы]: 5

Описание решения: \_\_\_\_\_



The image shows a web browser window displaying a configuration page for satellite parameters and camera settings. The browser's address bar shows 'localhost:3000/#'. The page is titled 'Спутник 4' (Satellite 4).

**Спутник 4**

Наклонение [°]

Большая полуось [км]

Эксцентриситет

Долгота восходящего узла [°]

Аргумент перигея [°]

Аномалия

Аномалия: Истинная аномалия

Истинная аномалия [°]

Камера

Размер пикселя [мм]

Фокусное расстояние [мм]

Размеры сенсора [мм]

Стоимость [баллы]

Спутник 5

Спутник 3

Наклонение [°]

Большая полуось [см]

Эксцентриситет

Долгота восходящего узла [°]

Аргумент перицентра [°]

Аномалия

Аномалия: Истинная аномалия

Истинная аномалия [°]

Камера

Размер пикселя [мм]

Фокусное расстояние [мм]

Размеры сенсора [мм]

Стоимость [баллы]

Спутник 4

Спутник 5

The image shows a web browser window with the address bar displaying "localhost:3000/#". The page content is organized into sections for satellite and camera parameters.

**Спутник 2**

- Наклонение  $(^\circ)$ : 74
- Большая полуось (км): 6740
- Эксцентриситет: 0
- Долгота восходящего узла  $(^\circ)$ : 317
- Аргумент перицентра  $(^\circ)$ : 0

**Аномалия**

Истинная аномалия

Аномалия: Истинная аномалия

Истинная аномалия  $(^\circ)$ : 0

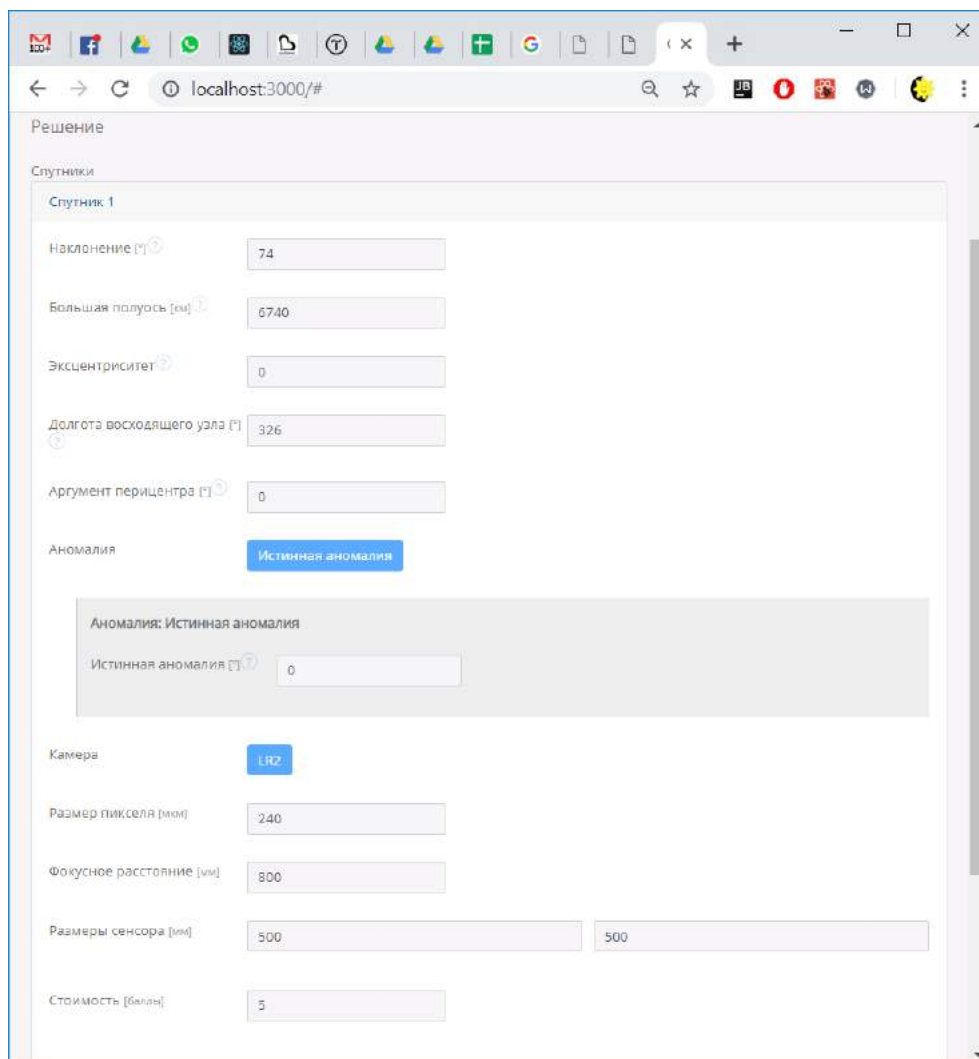
**Камера**

LR2

- Размер пикселя [мм]: 240
- Фокусное расстояние [мм]: 800
- Размеры сенсора [мм]: 500 500
- Стоимость [баллы]: 5

Спутник 3

Спутник 4



## 5.4. День четвертый

Сегодня последний и самый важный день. Вам нужно будет откалибровать свои устройства в подготовленном помещении. Делать это нужно аккуратно, проверив все моменты, так как на саму защиту у вас будет очень мало времени (около 15 минут).

Как будет выглядеть финальное задание:

0. В затемненном коридоре будет стоять специально укрепленный подвес, на расстоянии 4-5 метров от него - экран со снимком поверхности Земли из космоса в высоком разрешении.
1. После подвеса спутника вы настраиваете оптику;
2. Далее подходите к жюри со своими ПК для демонстрации запуска программы управления;
3. Необходимо открыть папку, в которую сохраняются фото;
4. Запустить работу устройств для получения снимков (как это было нужно сделать в последнем задании этого дня), дождаться выполнения программы;
5. Скинуть лучший снимок на флешку жюри
6. Оперативно снять спутник и отнести его в 308.

Требования к снимку:

- разрешение  $640 \times 480$
- максимально детализированное изображение

### Система оценки

Задача	Макс. балл	Балл команды
<b>Интеграция СУПН и спутника</b>		
Настроен обмен данными между Raspberry и БКУ	10	
По сигналу от БКУ Raspberry делает фото и передаёт его на ПК	20	
<b>Полная интеграция</b>		
Осуществлена полная сборка компонентов, компоненты запитаны и находятся в работоспособном состоянии	15	
Решение комплексной задачи: <ul style="list-style-type: none"> <li>• спутник ориентируется с оптической системой на заданный угол</li> <li>• после ориентации спутник делает снимок</li> <li>• снимок передаётся на ПК</li> <li>• полученный снимок не является размытым</li> </ul>	50	

Критерии оценки для комиссии жюри.

Осуществление съёмки. Беспроводная связь. 20 баллов.		Точная ориентация аппарата. 20 баллов.		Автоматизация съёмки. 30 баллов.			Качество снимка. 30 баллов.
На контрольном ноутбуке появилось по крайней мере одно изображение.	Изображения появились без использования проводной связи	Среди полученных изображений есть те, что соответствуют снимаемой области.	Спутник был подвешен и не зафиксирован в пространстве, но сохранил стабильное положение до конца съёмки.	Изображения появились без отдельной команды человека	На контрольном ноутбуке непрерывно, без вмешательства человека, продолжают появляться изображения (по крайней мере 2)	Получено более одного изображения и среди нет таких, что не соответствуют снимаемой области	Полученный снимок не смазан, угадываются очертания отдельных объектов. (отложенная оценка по сравнению снимков)
10	10	10	10	10	10	10	30

Ниже представлено изображение с зеркалом, по которому оценивалось качество съёмки.

