Технические материалы

ОБЗОР МОДУЛЯ ТЕХНИЧЕСКОГО ЗРЕНИЯ

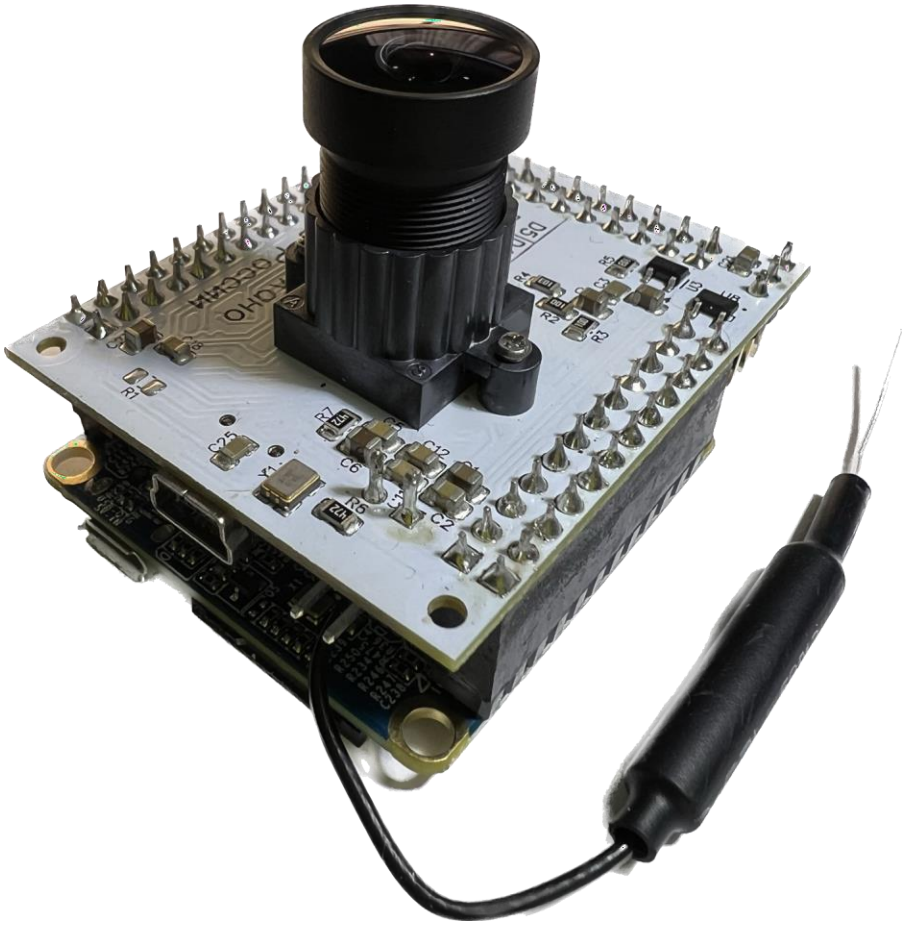


Рис. 1.1 Модуль технического зрения

Элементы модуля технического зрения.

1. Объектив со стандартной резьбой M12

2. Светодиодный индикатор, который отображает состояние модуля

3. Разъемы TTL, используемые для подключения модуля к Dynamixel-совместимому устройству

4. Разъемы подключения к другим устройствам: питания и интерфейсам UART, I2C, SPI

5. Разъем USB, который служит для подключения модуля к ПК

6. Кнопка, используемая для перезагрузки телекамеры

7. Отладочные выводы UART – используются для сервисного обслуживания модуля.

Страна-изготовитель	Россия
Матрица	OV7670
Разрешение видеопотока	640x480-2048x1536 пикселей
Кадровая частота	До 60 кадров в секунду
CPU	Allwinner H3, 4 ядра, 1.2GHz
Объем ОЗУ	512MB
Объем eMMC	8GB
Тип оптики	Стандартная на держатель M12
Углы обзора объективов по горизонтали	60.5 deg
Углы обзора объективов по вертикали	45 deg
Потребляемая мощность	500mW
Напряжение питания, Выводы, Глубина цвета	1.7-3.3V, 5-12V, 48, 8bit
Поддержка интерфейсов	TTL, SPI, USBV2, I2C, UART SWIM, SWD, Wi-Fi
Поддерживаемые уровни напряжения портов ввода-вывода	3.3В, 5В, 12В
Максимальное качество видеопотока на пк	Разрешение 2048x1536 пикселей при частоте до 15 кадров в секунду
ОС	Ubuntu
Поддерживаемые ОС	Windows
Габариты (ШхВхГ)	40x49x35.5

,

Универсальный вычислительный модуль

Помимо программируемого контроллера с периферийной материнской платой, в комплектацию образовательного набора также входит компактный контроллер универсальныйвычислительный модуль(далееУВМ). Данный контроллер предназначен для применения в проектах Интернета Вещей, в связи с чем он обладает компактными габаритами и необходимыми интерфейсами для подключения к сети. Этот универсальный вычислительный модуль, рассчитанный на взаимодействие с Dynamixel-совместимыми устройствами, представляет собой электронную плату с размещенным на ней вычислительным микроконтроллером Atmega2560, чипом беспроводной связи по интерфейсам Wi-Fi и Bluetooth, набором цифровых и аналоговых линий передачи связи, а также портами для подключения Dynamixel-совместимых устройств (рис. 3.1).

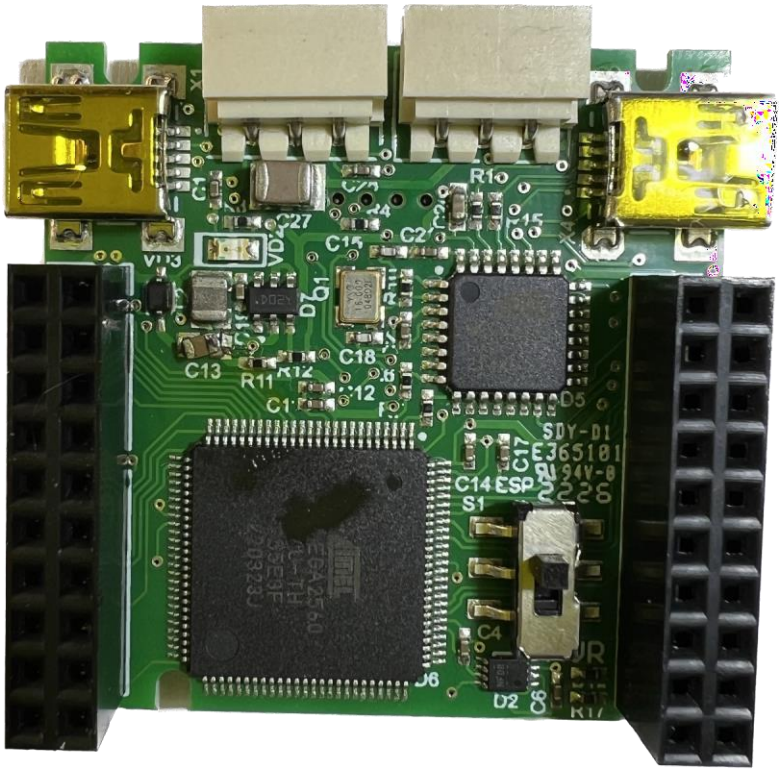


Рис. 3.1. Программируемый контроллер УВМ

Технические характеристики контроллера УВМ

Габариты – 40х40 мм

Номинальное напряжение питания – 12В

Флеш-память – 256Кб (из которых 8Кб используются для загрузчика)

ОЗУ – 8Кб

Энергонезависимая память – 4Кб

Тактовая частота - 16МГц

Количество 3-х пиновых портов Dynamixel – 2 шт.

Количество портов USB 2.0 – 2 шт.

Количество линий ввода-вывода – 40 шт.

В том числе: Линия «земля» – 1 шт.

Линия +3.3В – 1 шт.

Линия +5В – 1 шт.

Линия +12В – 1 шт.

Интерфейс SPI – 1 шт.

Интерфейс I2C – 1 шт.

Интерфейс UART – 1 шт.

Цифровые линии ввода-вывода – 12 шт.

Аналоговые линии ввода – 16 шт.

Индикаторы – 1 шт (5В).

Переключатель ESP-УВМ – 1 шт.

Беспроводные интерфейсы – Wi-Fi 802.11 b/g/n/d/e/i/k/r (802.11n до 150 Мбит/с), Bluetooth V4.2 BR/EDR и BLE.

Органы управления (кнопки) – 3 шт.

Основные элементы контроллера выглядят следующим образом:

1. MiniUSB порт для подключения к ПК и программирования основного контроллера Atmega2560. Данный порт может быть использован для подачи питания на модуль питания 5В.

2. MiniUSB порт для подключения к ПК и программирования модуля беспроводной связи Bluetooth\Wi-Fi. Данный порт может использоваться для подачи питания на модуль питания 5В.

3. 3-х пиновые DXL-порты. Используются для подключения контроллера в цепь в качестве Dynamixel-совместимого устройства. Контроллер УВМ может использоваться как в качестве ведущее Dynamixel-совместимого устройства (с его помощью можно управлять подключенными к нему сервоприводами Dynamixel), так и как ведомое (модуль может выступать в качестве Dynamixel-совместимого устройства и обмениваться данными с внешним контроллером через протокол Dynamixel). Более того, эти порты могут применяться для подачи питания на модуль питания 12В. Наличие внешнего источника питания 12В обязательно при использовании контроллера в качестве Dynamixel-совместимого устройства.

4. Индикационный светодиод линии 5В.

5. Переключатель интерфейса беспроводного Wi-Fi\Bluetooth модуля. В положении ESP (верхнее положение) интерфейс модуля беспроводной связи подключен к miniUSB порту для его программирования и настройки. В положении УВМ (нижнее положение) интерфейс модуля беспроводной связи подключен к микроконтроллеру Atmega2560, что обеспечивает обмен данными с ним посредством Serial2 Atmega2560.

6. Гнезда с выводами для подключения внешних устройств и плат расширения.

7. Программируемый светодиод. Подключен к линии PB7 (D13) микроконтроллера Atmega2560.

8. Кнопка перезагрузки микроконтроллера Atmega2560.

9. Кнопка перезагрузки модуля беспроводной связи.

10. Кнопка ввода в режим загрузчика модуля беспроводной связи.

Для работы с этим контроллером в среде программирования Arduino IDE в качестве платы необходимо выбирать Arduino Mega 2560. Данный модуль является ее аналогом.

Для расширения функционала контроллера УВМ существует ряд модулей расширения.

Интернет вещей и ИИ

ИНТЕРНЕТ ВЕЩЕЙ

Использование модуля в качестве BT-устройства

Модуль ESP-WROOM-32 может работать как в режиме Wi-Fi, так и в режиме Bluetooth 4 (он же Bluetooth LE). Для активации требуемого режима работы необходимо загрузить определенный программный код в модуль с помощью среды разработки Arduino IDE. Для этого необходимо выполнить подготовку модуля к работе по ранее рассмотренной схеме.

В качестве основного примера для настройки работы модуля в качестве Bluetooth устройства будет использоваться стандартный пример BLE_uart, находящийся по умолчанию после установки программной поддержки модуля в среде Arduino IDE в меню «Файл - Примеры - ESP32 BLE Arduino - BLE_uart». Данный пример полноценно демонстрирует процесс создания BLE устройства и организацию процесса обмена данными через UART - соединение между модулем и внешним устройством, в роли которого выступает основной микроконтроллер - atmega2560.

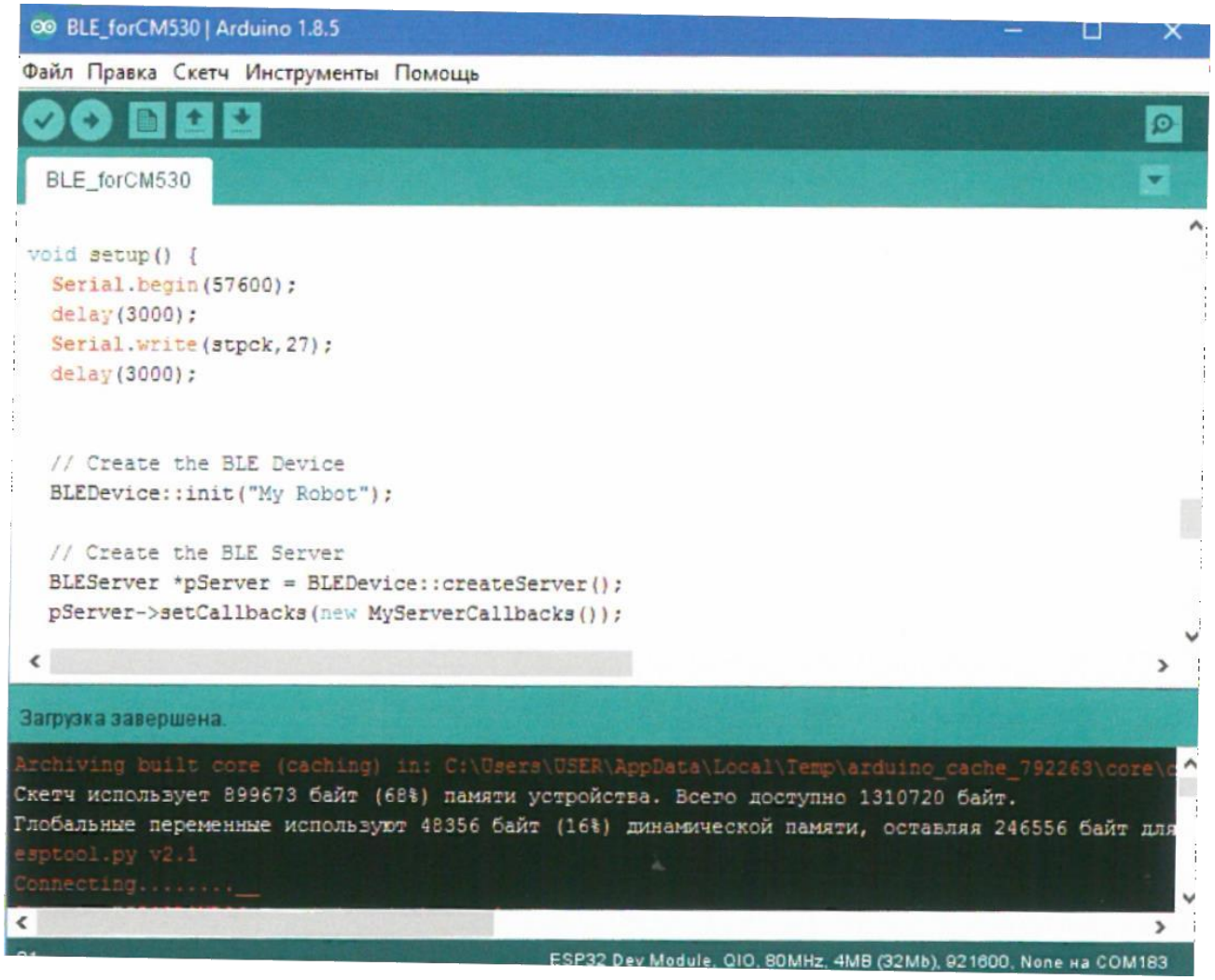
Наименование устройства задается через строку:

```
BLEDevice::init("UART Service");
```

где вместо UART Service необходимо указать требуемое наименование, например, «My Robot»:

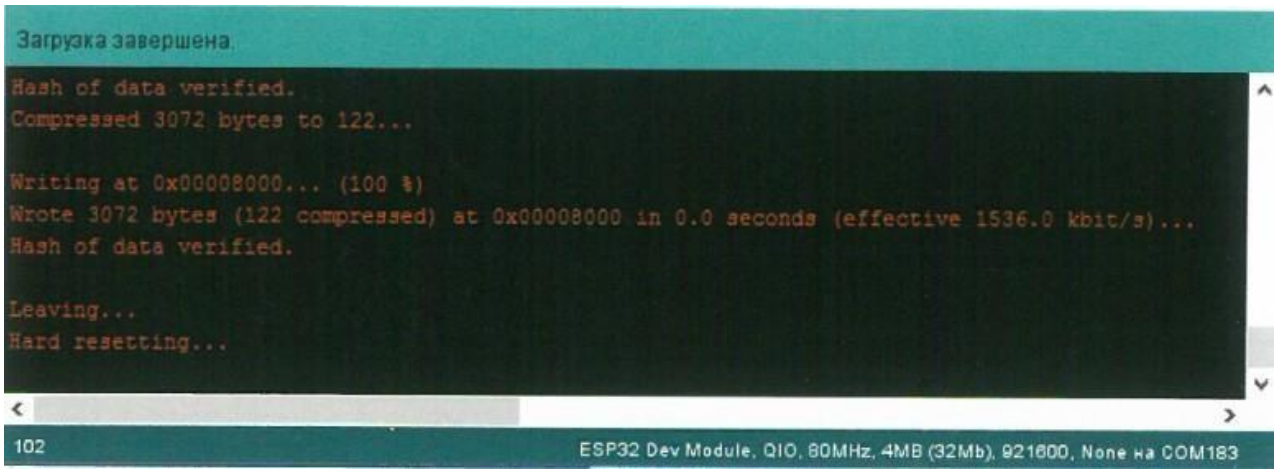
```
BLEDevice::init("My Robot");
```

Для загрузки данного примера в модуль, необходимо подключить его к компьютеру через указанный USB порт, переместить переключатель в положение USB, подать питание на контроллер и нажать в среде Arduino IDE кнопку загрузки скетча. А по окончанию компиляции, когда в терминал будет выведена строка Connecting , зажать на 2-3 секунды обе кнопки бутлоадера, затем отпустить Reset и потом отпустить Boot.



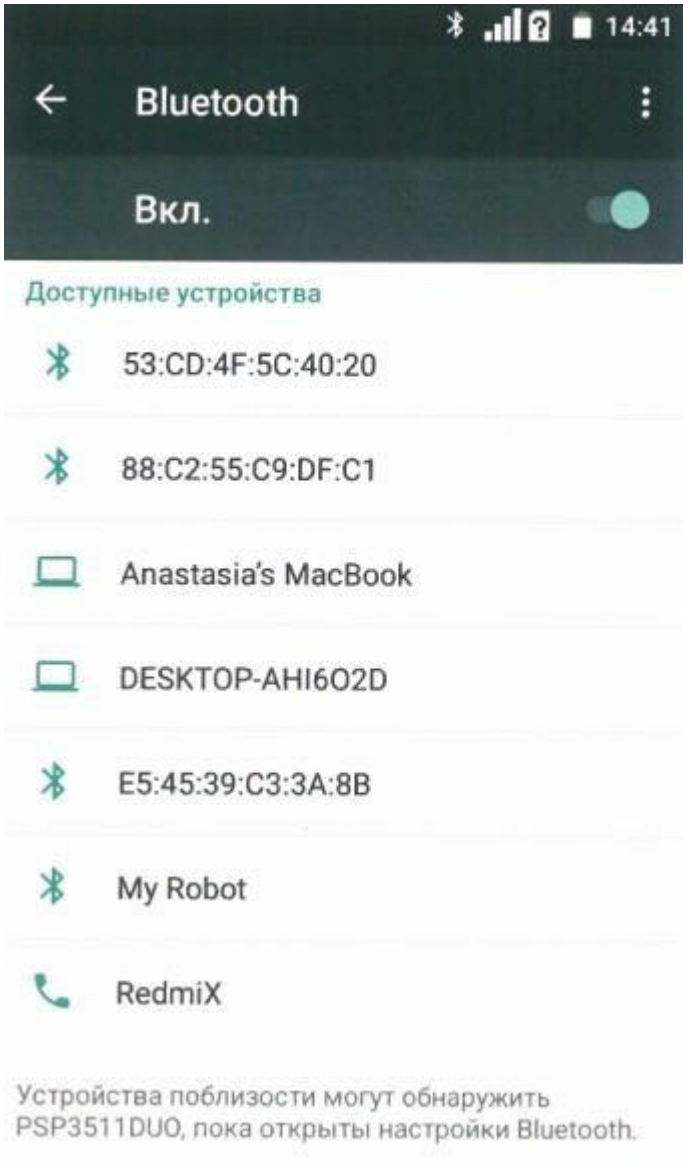
Начало загрузки прошивки в ESP 32

После окончания загрузки в терминал будет выведена информация о том, что загрузка завершена, и появится сообщение о необходимости аппаратной перезагрузки.



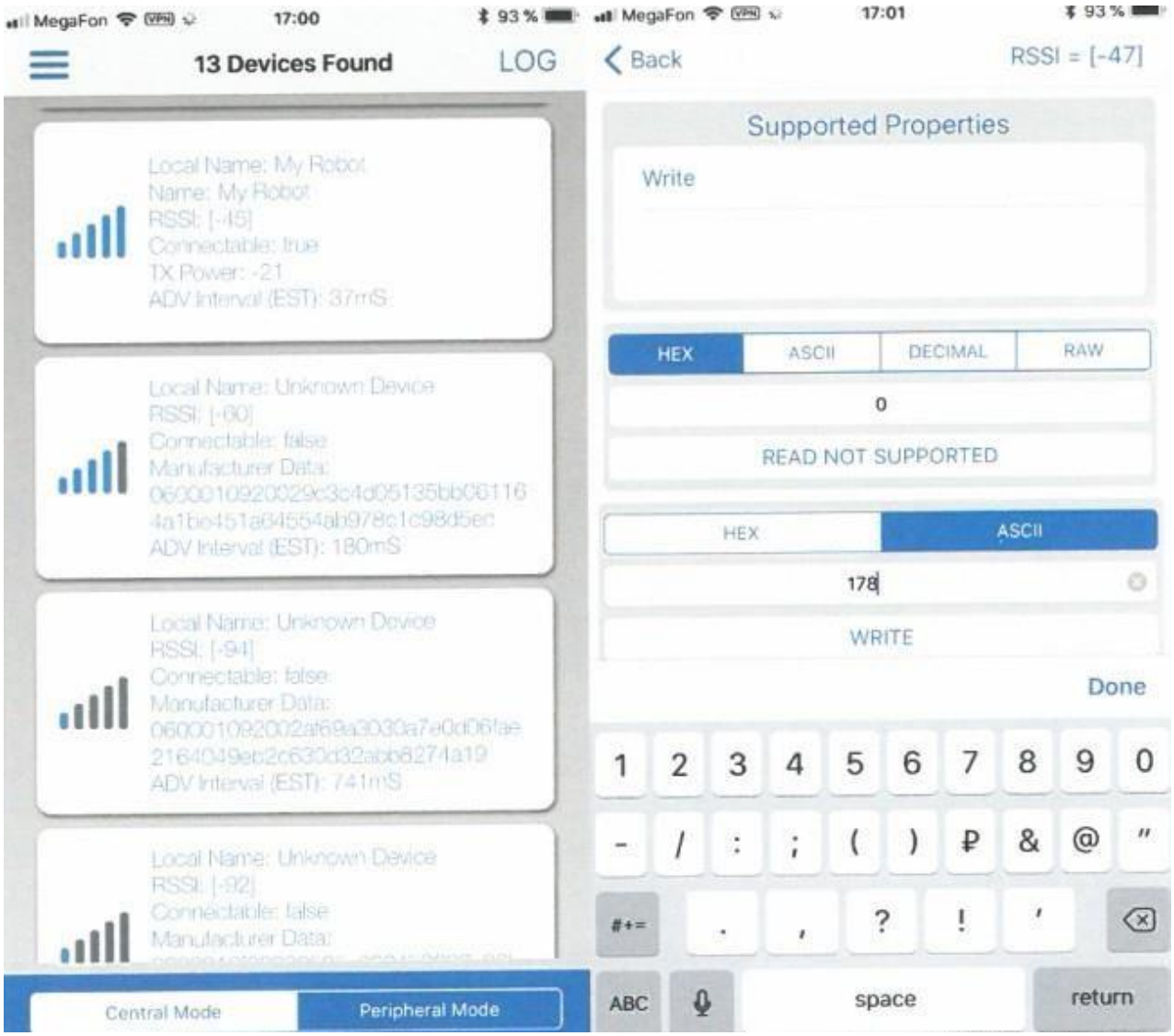
Окончание загрузки прошивки в модуль

Для выполнения такой перезагрузки необходимо выключить и включить заново питание контроллера. После этого в окружении появится Bluetooth - устройство My Robot.



Появившееся устройство в окружении

Открыв мониторпорта на компьютере и выполнив подключение к данному устройству, можно увидеть передаваемые данные, например, с телефона или планшета. Важно: для передачи данных программа должна поддерживать связь именно с устройством BLE - например, BLU для iOs, позволяющей отправлять какие-либо значения в ASCII нотификации.



Пример отправки данных на BLE устройство

Если перевести переключатель в положение MCU, то все передаваемые данные пойдут в микроконтроллер, где их можно считать стандартными средствами с интерфейса Serial2, используя, например, такой код:

```
void setup() {
 Serial.begin(9600);
 Serial2.begin(115200);
}
void loop() {
```

```
 if (Serial2 .available ()} {
 Serial.write(Serial2.read());
}
}
```

Использование модуля в качестве Wi-Fi устройства

Для активации режима работы в качестве Wi-Fi устройства в модуль также необходимо загрузить определенный программный код, по аналогии с активацией режима работы в качестве Bluetooth устройства. Данный модуль может работать как в качестве WiFi клиента, так и в режиме WiFi точки доступа. В связи с этим, существует большое количество способов применения данного модуля в различных проектах - для рассмотрения данных примеров можно обратится в меню «Файл - Примеры - Примеры для ESP32 Dev Module - WiFi», а также можно обратиться в Интернет - большое количество разработчиков используют в своих проектах модули на основе чипов ESP-32S (ESP-WROOM) и размещают примеры кода в открытом доступе.

Работа в качестве Wi-Fi клиента

В данном режиме работы модулю требуется наличие внешней точки доступа и развернутой Wi-Fi сети для подключения к ней. Рассмотрим пример, в котором будет произведено подключение к существующей сети Wi-Fi и организован обмен данными между модулем и каким-либо еще устройством, подключенным к этой же Wi-Fi сети. В качестве базового примера, на основе которого будет создаваться код, будет использоваться стандартный пример, по умолчанию входящий в библиотеку для работы с модулем и содержащийся в

«Файл - Примеры - Примеры для ESP32 Dev Module - WiFi - SimpleWiFiServer».

В первую очередь, необходимо подключить библиотеку WiFi.h

```
#include <WiFi.h>
```

затем необходимо будет задать ssid и пароль сети, к которой будет выполняться подключение:

```
const char* ssid = «Wifi _for _Robots»;
const char* password = «00000721»;
```

затем выполняется инициализация сервера на 80 порту:

```
WiFiServer server(80);
```

В основной функции setup() проинициализируем соединение по по следовательному порту на скорости 57600 бод

```
Serial.begin(57600);
```

делаем небольшую паузу:

```
delay(10);
```

и инициализируем выполнение подключения к WiFi сети с ранее заданными параметрами:

```
WiFi.begin(ssid, password);
```

после начала подключения выполняем ожидание до тех пор, пока модуль не подключится к заданной сети:

```
while (WiFi.status () != WL_CONNECTED) {
```

```
delay(500);
Serial.print(«.»);
```

при успешном соединении в последовательный интерфейс выводится информация об успешном соединении и выводится присвоенный модулю IP адрес:

```
Serial.println( «»);
Serial.println(«WiFi connected.»);
Serial.println(«IP address: «);
Serial.println(WiFi.localIP());
```

после инициализируем сервер:

```
server.begin();
```

Теперь модуль готов к обмену данными. В функции void loop() добавим ожидание подключения клиента, и в случае подключившегося клиента выполняем чтение данных, поступающих от него. В случае, если клиент отключается, выводится соответствующее сообщение:

```
WiFiClient client = server.available();
if (client) {
Serial.println(«New Client.»);
String currentline = «»;
while (client.connected()) {
 if (client.available()){
  char c= client.read();
  Serial.write(c);
}
```

```
}
client.stop();
Serial.println(«Client Disconnected.»);
}
```

В данном примере модуль будет принимать и передавать данные по 1 символу, поэтому добавим возможность принимать сразу строку данных. Для этого после строчки Serial.println(«New Client.»); добавим переменную типа String, в которую будут записываться данные:

```
String currentline = «»;
```

а после строки Serial.write(c), собственно, конструкцию для формирования этой строки:

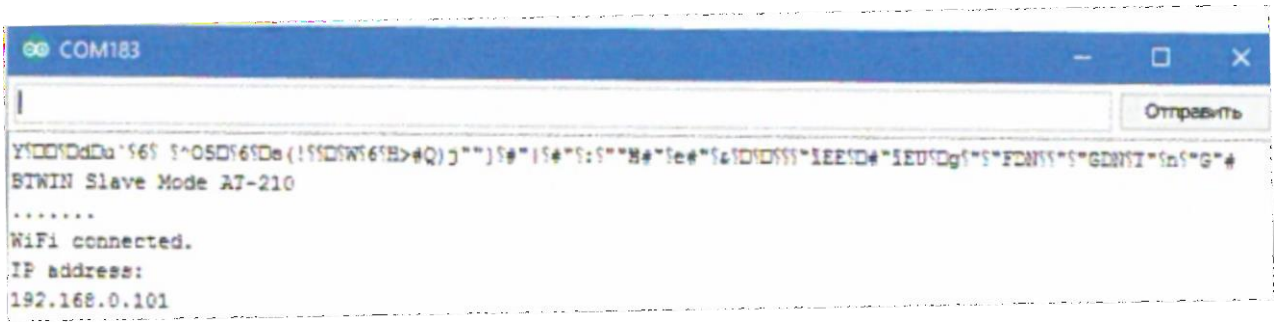
```
if (c!= '\r') {
 currentline += c;
}
```

В случае, если строка не пустая, ее можно преобразовать в числовое значение и передать с помощью цифрового пакета в контроллер:

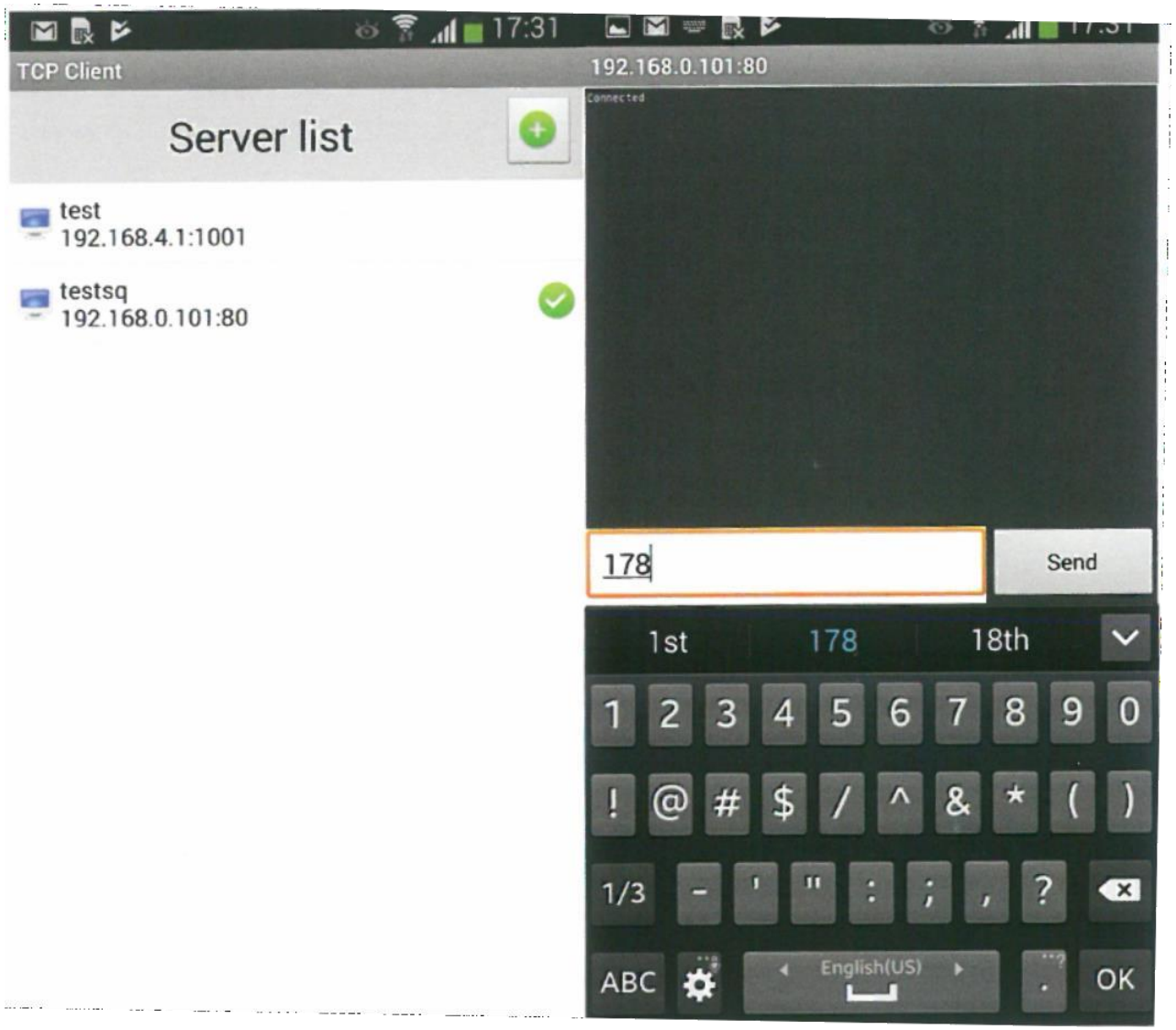
```
if (currentline.length()>0){
 uint16_tnumber = currentline.tolnt();
 Serial.println();
 Serial.println(number);
}
```

Таким образом, код для работы модуля в режиме WiFi готов, и его можно загрузить стандартным способом, рассмотренным ранее.

Перед обменом данными с модулем необходимо узнать, какой ему присваивается IP адрес в сети. Данный адрес может быть не постоянным, поэтому необходимо периодическиего проверять. Для этого необходимо, загрузив программный код в модуль, открыть терминал и дождаться подключения его к сети - в случае успешного подключения будет выведен присвоенный модулю IP адрес:



Таким образом, используя выданный IP адрес, можно выполнять обмен данными между модулем и удаленным устройством. Для этого можно воспользоваться приложением TCP Client, где произвести настройку сервера (на рисунке это testsq) и подключиться к модулю:



Пример отправки данных с TCP клиента

В случае успешного подключения, откроется окно с терминалом, в котором можно отправлять и получать данные от удаленного модуля. Для передачи данных со стороны контроллера необходимо внести простые изменения в управляющую контроллером программу, которые эти данные будет передавать. Поскольку данный процесс идентичен классическому способу передачи данных через UART, то он здесь рассматриваться далее не будет.

Работа модуля в качестве Wi-Fi точки доступа

При необходимости, модуль может сам стать источником WiFi сети. Для этого необходимо внести следующие изменения в функцию setup() написанной ранее программы, и она должна принять следующий вид:

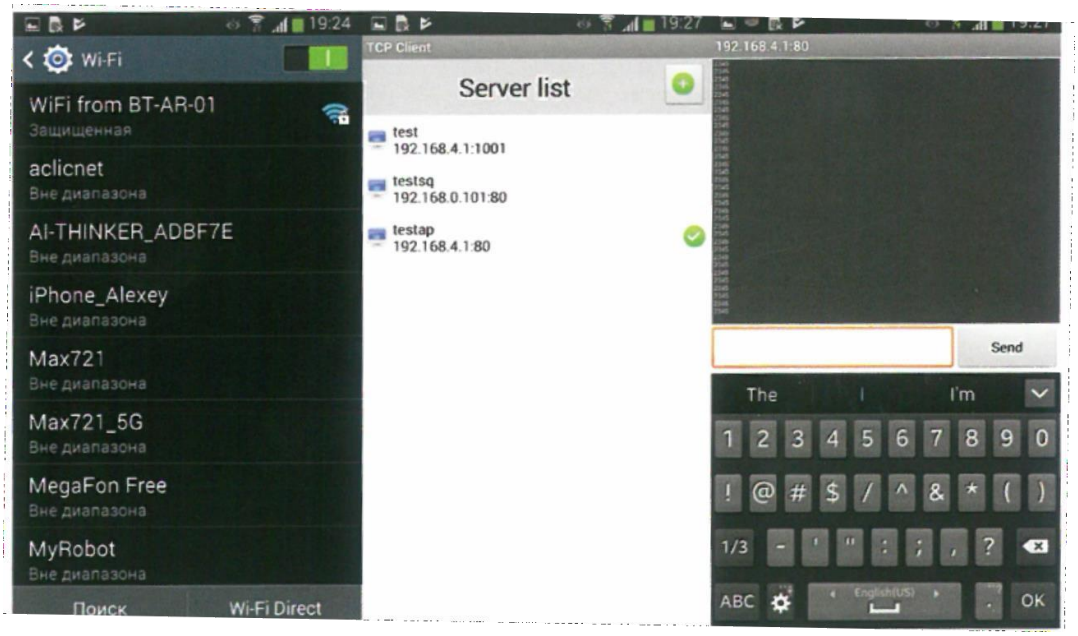
```
Serial.begin(57600);
WiFi.softAP(ssid, password);
Serial.println();
Serial.print(«IP address: «);
Serial.println(WiFi.softAPIP());
```

Загрузив измененный код в программу можно убедиться в создании сети, проверив полученный IP адрес:

Загрузка модуля в режиме точки доступа

Используя такой метод, была создана Wi-Fi сеть - WiFi from BT-AR-01 (изменено было название вместо WiFi_for_Robots), к которой можно подключиться и выполнить все те же задачи, которые были рассмотрены ранее.

Таким образом, модуль ESP-WROOM-32 может быть использован и как клиент WiFi сети, и как ее источник, что дает широкие возможности при организации обмена данными между контроллером и удаленным устройством.



Пример обмена данными с модулем в режиме точки доступа

Модуль проводной передачи данных

В качестве модуля проводной передачи данных в контроллер встроен популярный Arduino - совместимый чип от WizNet - WS100. Примеры работы с ним встроены в стандартную базу среды разработки Arduino IDE полностью совместимы с данным контроллером. В качестве демонстрационного примера можно рассмотреть пример передачи данных в браузер ПК.

В случае подключения контроллера напрямую к компьютеру через Ethernet, либо же при включении его в сеть, где все устройства имеют статический IP, необходимо указать контроллеру такой IP, чтобы он попадал в подсеть сети или компьютера. Также, для работы с чипом необходимо инициализировать следующие требующиеся библиотеки:

```
#include <SPl.h>
#include <Ethernet.h>
// Подключаем библиотеку SPI
```

```
// Подключаем библиотеку Ethernet
byte mac[] = {0xDE, 0xAD, OxBE, 0xEF, 0xFE, 0xED}; // Вводим MAC адрес
IPAddress ip(169, 254, 21, 70);// Указываем статический IP
EthernetServer server(80); //Ethernet «server» на порту 80
// Инициализация библиотеки
```

Затем задаем скорость передачи данных и ждем пока не откроется монитор порта и запускаем сервер:

```
void setup()
{
Serial.begin(115200);
while (!Serial) {;}
Ethernet.begin(mac, ip);
server.Begin();
Serial.print(«server is at «);
Serial.println(Ethernet.localIP());
}
// Задаем скорость передачи данных
// Запускаем сервер
```

а затем в функции loop() ожидаем подключение клиента (открытие вкладки браузера с указанием в адресной строки IP:port), и начинается передача данных, которые в данном примере представлены значениями с аналоговых портов:

```
void loop()
{
```

```
EthernetClient client = server.available();     // Принимаем данные,
//посылаемые клиентом
if (client) {
 Serial.println(«new    client»);
 boolean currentlinelsBlank = true;
 while (client.connected()) {
  if (client.available()) {
   char c= client.read();
   Serial.write(c);
   if (c == '\n' && currentlinelsBlank){
   client.println(«HTTP/1.1 200 OK»);
   client.println(«Content-Type: text/html»);
   client.println(«Connection: close»);
        client.println(«  Refresh:  5»);  //обновление  страницы  каждые  5  секунд
client.println();
   client.println(«<!DOCTYPE HTML>»);
   client.println(«<html>»);
   for (int analogChannel = 10; analogChannel < 16; analogChannel++) {
    int sensorReading = analogRead(analogChannel);
    client.print(«analog input «);
    client.print(analogChannel);
    client.print(« is «);
    client.print(sensorReading);
    client.println(«<br />»);
}
   client.println(«</html>»);
```

```
   break;
}
  if (c== '\n') {
   currentlinelsBlank = true;
} else if (c!= '\r') {
   currentlinelsBlank = false;
}
}
}
delay(1);
client.stop();
Serial.println(«client disconnected»);
}
}
```

Таким образом, в окне браузера появятся данные, отправляемые контроллером.

НАСТРОЙКА МОДУЛЯ CAMV. Использование искусственного интеллекта(ИИ).

Распознавание однотонных областей

Настройка камеры для распознавания одноцветной области сводится к заданию ее цветовых и морфологических признаков. Рассмотрим в качестве распознаваемого объекта оранжевый мяч для настольного тенниса, расположим его перед телекамерой таким образом, чтобы он целиком попадал в зону ее видимости (рис. 1.1). Это необходимо для того, чтобы выбрать все необходимые оттенки цвета, набор которых будет различен, в зависимости от внешнего освещения.

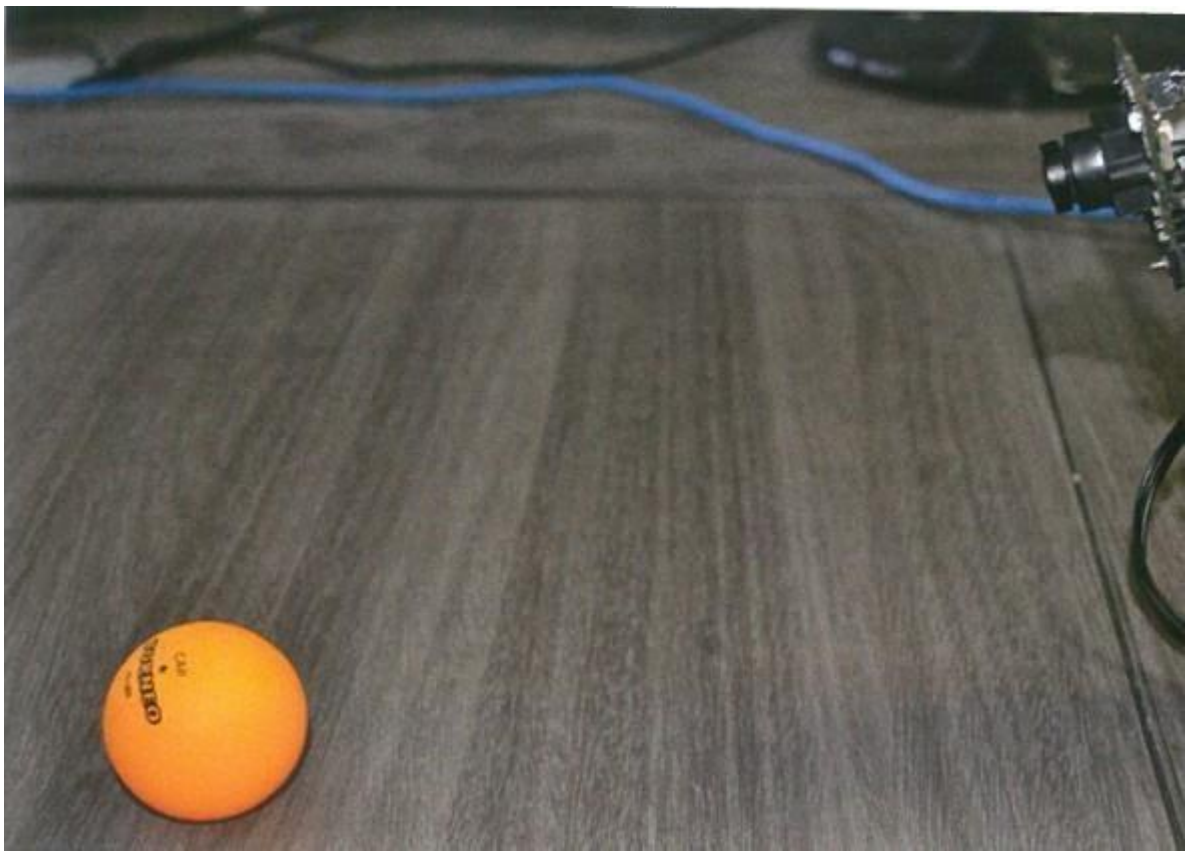


Рис. 1.1 Пример размещения одного однотонного объекта перед камерой

Для выполнения первоначальной настройки необходимо подключить модуль к компьютеру с помощью кабеля USB Mini-B HighSpeed и запустить приложение CAMVApp. В главном окне нажать кнопку Connect. В окне lmage появится изображение с телекамеры. Убедившись, что модуль видит шарик целиком, зажав кнопку Ctrl на клавиатуре нажмем левой кнопкой мыши на изображение шарика в окне lmage. Модуль отделит шарик от фона и применит к нему параметры распознавания по умолчанию. При этом программное обеспечение автоматически подберет набор цветовых оттенков, которые будут использоваться при распознавании. Выбранные оттенки можно наблюдать в окне палитры (Palette), при необходимости можно расширить область фиксируемых оттенков. Результат поиска выбранных оттенков отобразится в окне иллюстрации бинаризации (Line blobs). Окончательный результат распознавания области с учетом морфологических признаков отобразится в окне Blobs. Если же в окне Blobs не отображается область соответствующая шарику, либо же видно несколько областей, вызываемых прочими внешними объектами или бликами, необходимо отрегулировать настройки детектора (Blob detector settings) в главном окне. После выполнения настройки модуля необходимо выполнить их запись в память модуля, чтобы после выключения питания они сохранялись. Для этого необходимо нажать кнопку Save. После нажатия кнопки Save начнется запись настроек в память модуля, которая займет несколько секунд, после чего кнопка Save сменится на Saved. Если в главном окне вместо кнопки «Save» отображается кнопка «Need-->», необходимо вначале выполнить синхронизацию в меню Sync. Таким образом модуль технического зрения настраивается на распознавание простых однотонных объектов (рис. 1.2).

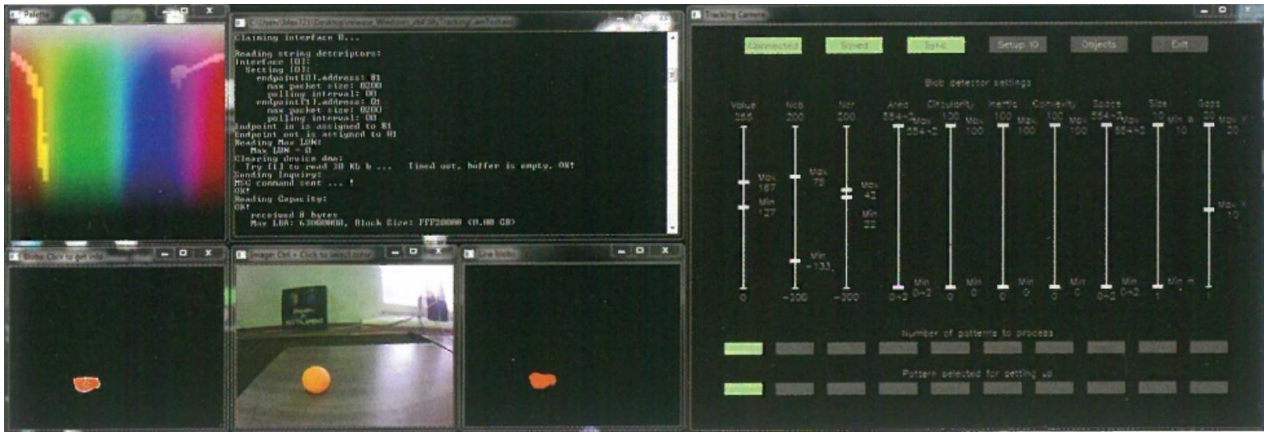


Рис. 1.2 Внешний вид интерфейса CAMVApp при настройке на однотонный объект

Модуль CAMV способен распознавать до 10 однотонных областей разных оттенков. Для настройки камеры на работу в этом режиме необходимо в главном окне выбрать требуемое число шаблонов переключателем Number of patterns to process - каждый шаблон отвечает за определенный оттенок - и затем по очереди настроить каждый шаблон по аналогии как настраивается модуль на отслеживание и распознавание одной области. Для этого необходимо поочередно выбирать текущий настраиваемый шаблон в переключателем Pattern selected for setting up и последовательно выполнять настройки для каждого требуемого цвета (оттенка). В качестве примера рассмотрим процесс настройки телекамеры на распознавание оранжевого мячика и синего кубика (рис. 1.3).

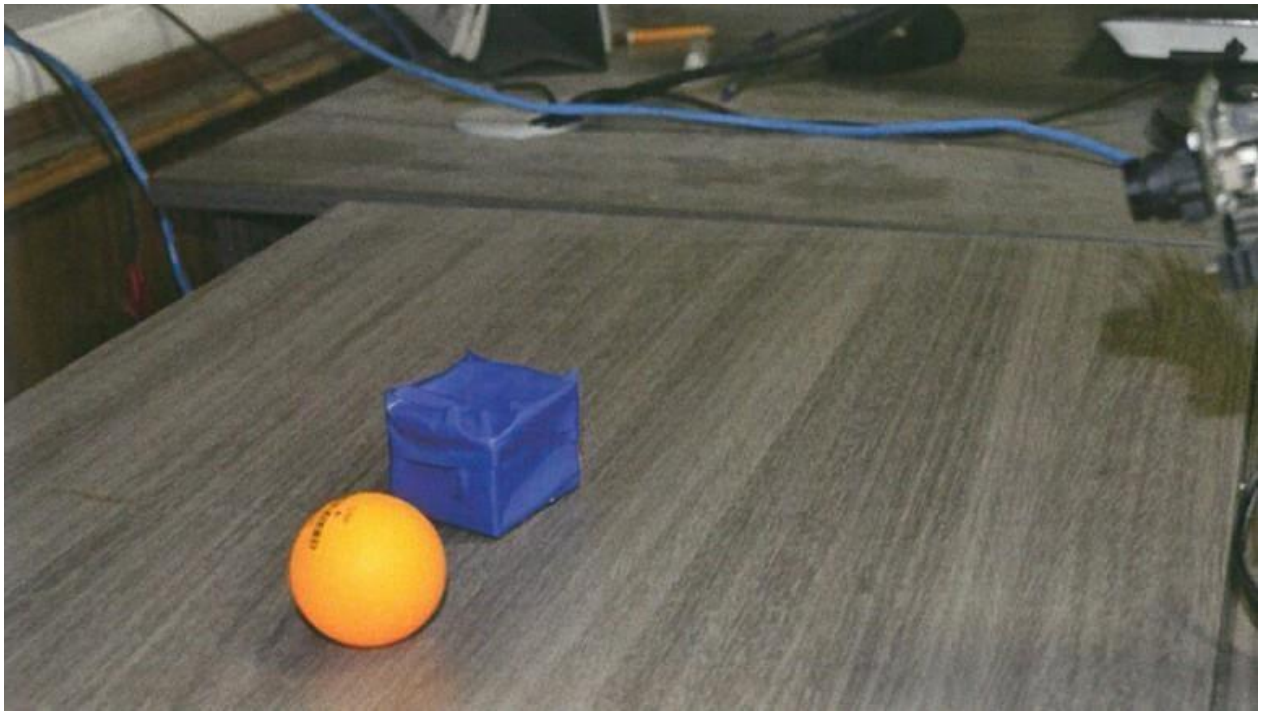


Рис. 1.3 Пример размещения двух однотонных объектов перед камерой

Первый шаблон будет отвечать за распознавание шарика, второй - кубика. После настройки, для получения информации о распознанных областях можно нажать на каждую область в окне Blobs и в консоль будет выведена вся информация о выбранной области. Эта информация будет содержать в себе данные о его типе/шаблоне (type), положении центра (cx, cy), площади в пикселях (area), площади описанного многоугольника (convex area), значения округлости (circularity), вытянутости (inertia), выпуклости (convexity), угол поворота (angle), ширину/высоту (w, h) и длину большей диагонали габаритного многоугольника (max axis). Применительно к ситуации с шариком и кубиком информация может выглядеть следующим образом (рис. 4.4):

Здесь Blob[0] - область соответствующая оранжевому шарику, а Blob[1] - синему квадрату. После выполнения настройки модуля параметры необходимо загрузить в модуль аналогично случаю с распознаванием одной области.

Рис. 1.4 Пример вывода информации о распознанных областях

Распознавание разноцветных объектов

Помимо задач распознавания однотонных областей, зачастую возникает задача отслеживания разноцветных объектов, например, многоцветных меток - маркеров. Для этого модуль CAMV настраивается на распознавание составных объектов, которые описываются взаимными размерами, положением, ориентацией и прочими параметрами, входящих в них однотонных областей. При объединении однотонных областей в составные объекты разработчики модуля условно называют их узлами (Node) объекта, а векторы связывающих их параметров связями (Link). Модуль CAMV способен отслеживать одновременно до 5 объектов, состоящих максимум из 3-х узлов.

Разберем на примере процесс настройки модуля на распознавание объекта, состоящего из оранжевой и синей областей. Для необходимо аналогично тому, как проводилась настройка на распознавание оранжевой

области, настроить его на распознавание еще одной синей области, выбрав 2-й шаблон переключателем Number of patterns to process и 2-й шаблон переключателем Pattern selected for setting up в главном окне. После этих манупуляций в окне Blobs будут отображаться сразу 2-е разных области. Затем необходимо зайти в окно Object Detector (рис. 4.5), нажав кнопку Object и выбрать из какого количества узлов он будет состоять переключателем «Type (Number of nodes)».

В данном случае объект состоит из 2-х цветных областей (Two blobs). Также необходимо выбрать один шаблон переключателем количества распознаваемых объектов Number of objects to detect. После этого на схеме объекта окажутся подсвеченными 2 узла и образуется связь между ними. При необходимости можно подкрутить их параметры в соответствующих окнах. После настройки параметров искомого объекта необходимо нажать кнопку Close в окне Object Detector и затем в окне Blobs можно будет увидеть оба ранее определенных области, объединенных в один объект. После выполнения настройки модуля параметры необходимо сохранить в модуль аналогичным ранее рассмотренному способом. Текущую информацию о параметрах как отдельных областей, так и целиком всего объекта можно посмотреть в терминале. Информация о параметрах областей, входящих в состав объекта, получается аналогично случаю распознавания отдельных областей - путем нажатия на интересующую область в окне Blobs. Для вывода сведений обо всех найденных объектах необходимо в окне Object Detector нажать кнопку Print, после чего в консоль будет выведена информация о структуре каждого объекта (из какого количества узлов он состоит) и его узлах: номер шаблона, к которому он относится, вошедшая в него цветная область (Type), порядковые номера цветных областей, степень соответствия распознанного объекта

заданному шаблону (Similarity), размер объекта (Size) в единицах заданной меры и угол ориентации (рис. 1.6).

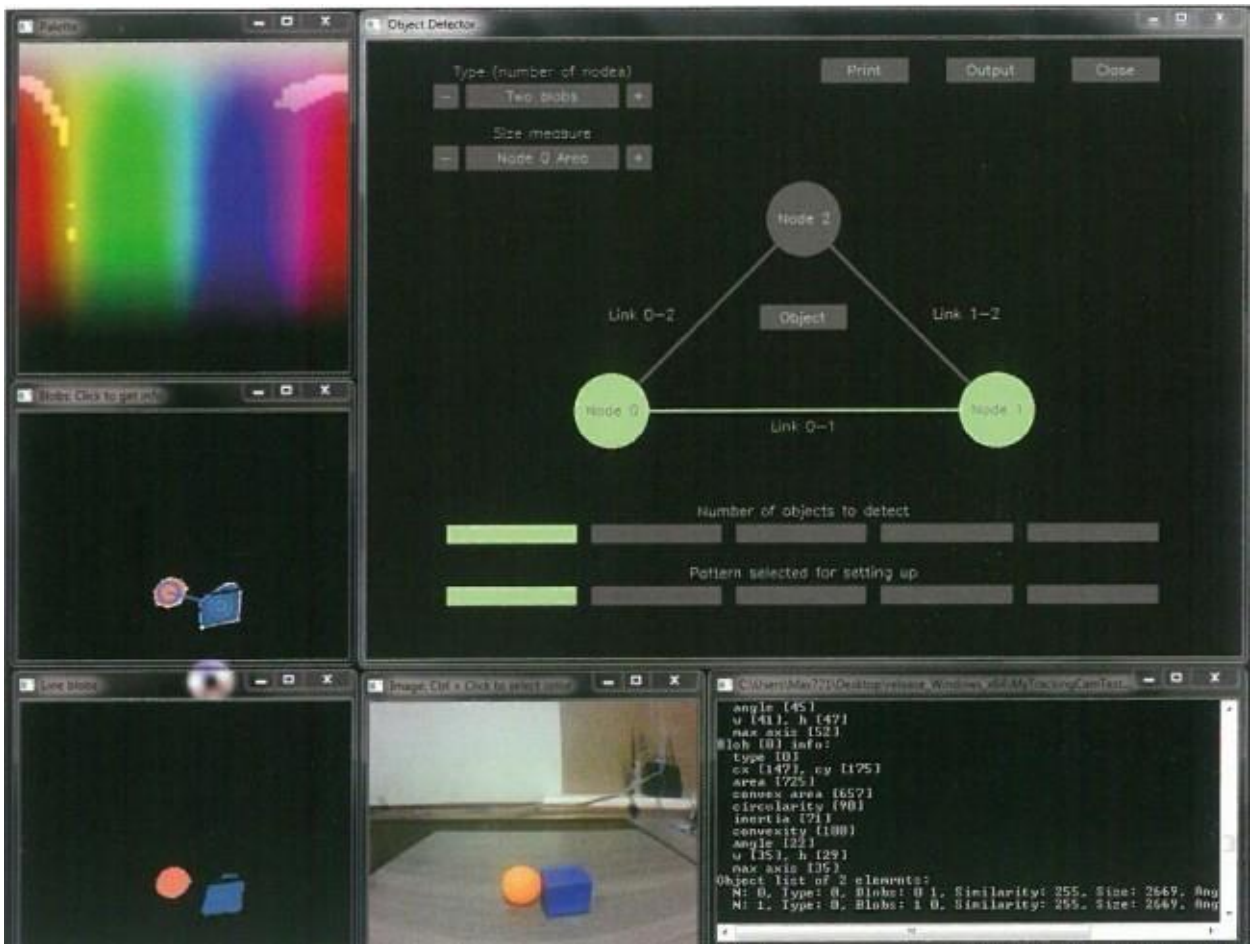


Рис. 1.5 Пример размещения двух однотонных объектов перед камерой

Работа модуля в режиме распознавания объектов может быть полезна при решении задач соревновательного направления - распознавания различных цветовых меток, следования по линиям, состоящим из разноцветных областей, классификация окружающих объектов.

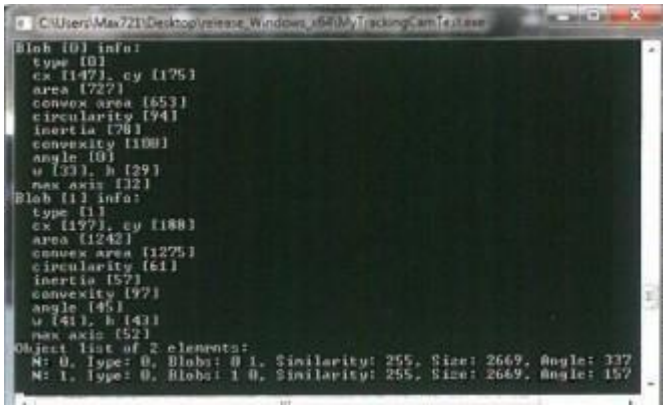


Рис. 1.6 Пример вывода информации об узлах распознанного объекта.