УДК 004.522

д.т.н., проф. **Мясищев А.А.** (ХмНУ)
Ленков А.С. (ПФ «Арий»)

ANDROID И ARDUINO В ЗАДАЧАХ УПРАВЛЕНИЯ ГОЛОСОМ И СИНТЕЗА РЕЧИ

В работе разработаны программы для Андроид смартфонов и Ардуино, позволяющие выполнять голосовое управление устройствами, подключенными к Ардуино. Используется инструментарий Google для распознавания и синтеза речи. Причем повторный запуск активити распознавания выполняется в цикле после окончания генерации звука благодаря использованию класса UtteranceProgressListener. Связь между Андроид устройством и Ардуино выполнена через Bluetooth.

Представлена схема подключения к Ардуино устройств, которыми выполняется голосовое управление. Следует отметить высокое качество распознавания произнесенных фраз при работе с серверами Google через Интернет и более низкое при работе с библиотекой речевого поиска Google в режиме offline, если мобильное устройство поддерживает этот режим. Представленный голосовой интерфейс можно рекомендовать для создания систем простой справочной службы и голосового управления.

Ключевые слова: Андроид, Ардуино, Bluetooth HC-05, Java, Android Studio, планшет, смартфон, распознавание речи, синтез речи.

Введение. В последнее время наблюдается значительный рост интереса к технологиям, связанным с распознаванием речи. Это задачи управления устройствами с помощью голосовых команд, задачи справочной службы, которая предоставляют информацию после запроса в более естественной форме - с помощью голоса. Много задач возникает при желании взаимодействовать с помощью голоса с мобильными устройствами. Например, ввод голосовых команд для получения информации с Интернета, прокладывания маршрута движения, запуск пользовательских программ, диктовка текста. Последнее время также появилась возможность управления домашней, офисной техникой с помощью Android - устройств голосовыми командами, что широко рекламируется как технология "умный дом".

Постановка задачи. Предпосылкой развития речевых технологий является значительное увеличение вычислительных возможностей, объема памяти при значительном уменьшении габаритов компьютерных систем. Следует также отметить развитие математических методов, позволяющих выполнить требуемую обработку аудио сигнала путем выделения из него информативных признаков, однозначно характеризующих речевой сигнал. Например, это широко используемое дискретное преобразование Фурье, известное из теории цифровой обработки сигналов. В последнее время используется вейвлет - преобразования, результаты которого более информативны для выделения акустических составляющих. Дальнейшая обработка выполняется с использованием акустической модели, которая ставит в соответствие выделенным параметрам конкретные звуки(фонемы). Окончательное построение фразы, предложения, голосовой команды выполняется с помощью лингвистической модели. Следует отметить, что последние две модели используют методы статистического анализа - это метод скрытого Марковского моделирования (СММ), метод динамического программирования и метод нейронных сетей[1] .

В работе представлена программа голосового управления Ардуино с помощью Андроид - устройства через Bluetooth HC-05. К Ардуино подключены три исполнительных механизма и датчик температуры. После выполнения соответствующих команд программа синтезирует голосовое сообщения о результате работы. Этот пример является типичной подзадачей общей задачи "Умный дом". Программа также реализует простейший диалог между устройством и пользователем, отвечая на простые вопросы. Для распознавания и синтеза речи используется инструментарий Google. Если смартфон поддерживает голосовой поиск в режиме оффлайн, доступ к Интернет не обязателен. Программа для Андроид

написана в среде Android Studio(язык Java), а для Ардуино - в среде разработки Ардуино на C++(проект Wiring). Тестировалась на телефоне LG G3 Stylus и планшете Acer A500.

Результаты. В рамках этой работы на устройстве Андроид должны выполняться следующие задачи:

1. При нажатии на кнопку приложения "Нажми для начала диалога"(рис.1) с помощью механизма Intent(намерение) вызывается Активити, выполняющее прослушивание произнесенной фразы с последующей передачей ее сервису распознавания речи. Он может находиться на серверах Google или установлен на мобильном устройстве, если оно поддерживает сервис "голосовой поиск offline". Если работа производится с серверами через Интернет результаты распознавания фраз будут значительно лучше. Для решения этой части задачи используется класс RecognizerIntent.

2. Результат распознавания в виде текстовой строки сопоставляется со строкой, находящейся в памяти. Если сопоставление истинно, то запускается синтезатор речи Google и произносится фраза из памяти, соответствующая сопоставленной. Например, результат распознавания "включить красный". Соответствие этой фразе в памяти "красный включила". Синтезатор сгенерирует эту фразу. Если сопоставление ложно, то синтезатор сгенерирует фразу, соответствующую распознанной строке. Для синтеза речи используется класс TextToSpeech.

3. После окончания синтеза речи основное приложение вновь запускает Активити, выполняющее прослушивание произнесенной фразы. Это повторение выполняется до тех пор, пока не будет произнесена фраза "конец связи". После этой фразы приложение должно перейти в режим ожидания пока вновь не будет нажата на кнопка " Нажми для начала диалога ".

4. После синтеза речи выполняется ее вывод через динамики компьютера. Поэтому необходимо в программе использовать специальный механизм, который бы прослушивал работу акустического вывода речи, а после окончания запускал вновь Активити по прослушиванию речи и ее распознаванию. В этом случае будет смоделирован полноценный диалог речевого общения между пользователем и компьютером. Данный механизм реализуется с помощью абстрактного класса utteranceProgressListener. Он использует методы, которым передается управление в начале, в конце высказывания синтезатором и при появлении ошибки.

5. Если произнесенная фраза соответствует команде, например "включить синий", то приложением на Андроиде выполняется передача байта на Ардуино через Bluetooth устройство и Ардуино выполняет включение синего светодиода(имитация устройства). При команде "температура" Ардуино пересылает приложению на Андроид значение температуры, считанной с температурного датчика DS18B20.

На рисунке 1 показаны скриншоты до и после активации голосового ввода.

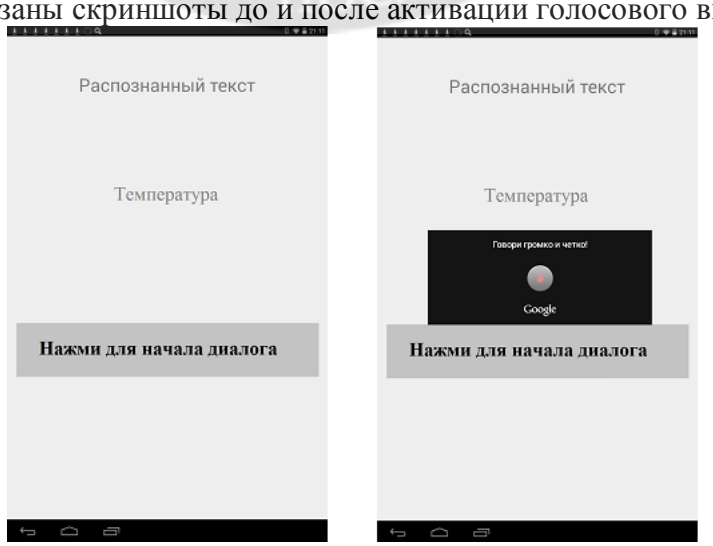

Рис. 1. Скриншоты до и после активации голосового ввода

В строке "Распознанный текст" выводится произнесенная фраза или команда, в строке температура - значение температуры, возвращенное с Ардуино после произнесения команды "темература". Строка "Нажми для начала диалога" - это кнопка, после нажатия на которую инициализируется Активити распознавателя речи с последующей обработкой произнесенных фраз.

Программа на Андроид состоит из двух больших блоков - это блок начала работы приложения (I) и блок работы приложения после нажатия на кнопку "**Нажми для начала диалога**"(II). На рисунке 2 показана блочная последовательность действий, которые выполняются каждым блоком (I, II).

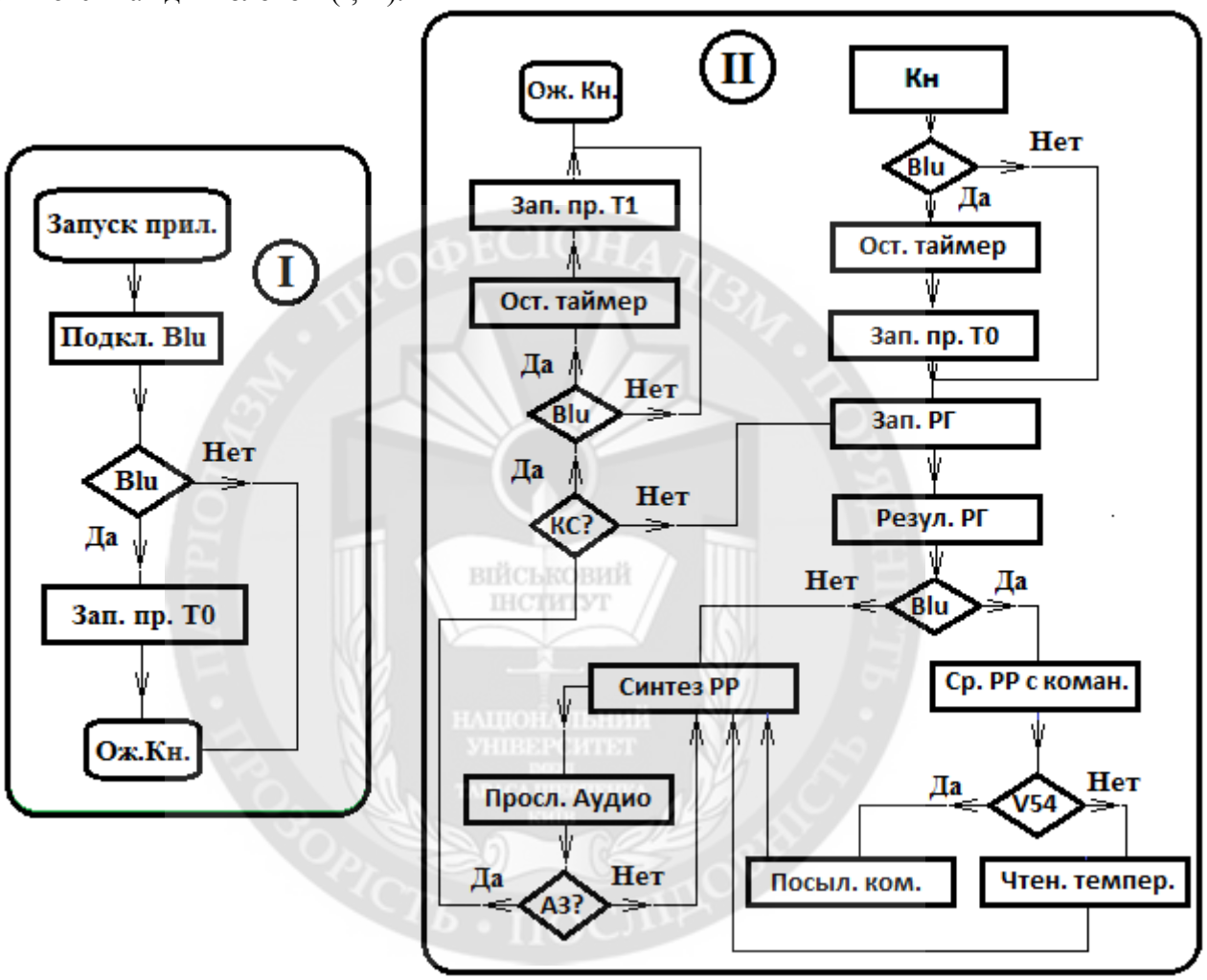

Рис. 2. Блок I и блок II, схематично поясняющие работу программы приложения

Здесь приняты следующие сокращения(см. также текст программы):

1. **Блок I**:

1.1. **Запуск прил**. - выполнить запуск приложения на Андроиде;

1.2. **Подкл. Blu** - запускается метод bluet(), который подключает Андроид к Ардуино через Bluetooth;

1.3. **Blu** - выполняется проверка, подключены ли Андроид и Ардуино по Bluetooth;

1.3. **Зап. пр. Т0** - каждую секунду запускается метод run() класса bluetoothInOut, который посылает запрос к Ардуино для определения температуры и выполняет считывание с Ардуино значения температуры. Таким образом каждую секунду Андроид получает новое значение температуры;

1.4**. Ож.Кн**. - после запуска приложение переходит в ожидание нажатия на кнопку "Нажми для начала диалога".

2. **Блок II:**

2.1. **Кн** - нажатие на кнопку "Нажми для начала диалога";

2.2. **Blu** - выполняется проверка, подключены ли Андроид и Ардуино по Bluetooth;

2.3. **Ост. таймер** - останавливается таймер, который запускает метод run() класса bluetoothInOut;

2.4**. Зап. пр. Т0** - каждую секунду запускается метод run() класса bluetoothInOut, который посылает запрос к Ардуино для определения температуры и выполняет считывание с Ардуино значения температуры.

2.5. **Зап. РГ** - запускается распознаватель голоса, метод spi();

2.6. **Резул. РГ** - метод onActivityResult получает результаты распознавания голоса и выполняет их сравнение со строкой вопросов и команд;

2.7. **Blu** - выполняется проверка, подключены ли Андроид и Ардуино по Bluetooth;

2.8. **Ср. РР с коман**. - сравнивается результат распознавания голоса с прописанной командой. При совпадении задается значение переменной value кодом символа. Например, символ "1", соответствующий включению RED(см. программу для Ардуино), представлен кодом 49 и т.д.;

2.9. **V54** - выполняется условие value != 54. Код 54 (символ "6") - это код запроса температуры от Ардуино;

2.10. **Чтен. темпер.** - распечатывается значение температуры в строке вывода текста temtext и строковой переменной spout присваивается ее значение, которое Андроид получает каждую секунду от Ардуино;

2.11. **Посыл. ком**. - через Bluetooth на Ардуино посылается байт данных (символ) для включения или выключения соответствующего устройства(RED, GREEN, BLUE);

2.12. **Синтез РР** - вызывается метод tts.speak, выполняющий аудио синтез ответа на произнесенный вопрос, результат выполнения команды, значение температуры или произнесенную фразу, если нет соответствия между вопросом и ответом(т.е. система работает как попугай);

2.13. **Просл. Аудио** - здесь работает метод onDone класса utteranceProgressListener который следит за окончанием генерации динамиком звука синтеза речи;

2.14. **АЗ** - проверка завершения аудио вывода синтезатором речи;

2.15**. КС** - проверка произнесения фразы "конец связи";

2.16. **Blu** - выполняется проверка, подключены ли Андроид и Ардуино по Bluetooth;

2.17. **Ост. таймер** - останавливается таймер, который запускает метод run() класса bluetoothInOut;

2.18. **Зап. пр. Т1** - каждую секунду запускается метод run() класса bluetoothInOut, который выполняет считывание с Ардуино значения температуры(запрос на температуру не посылается);

2.19. **Ож. кн.** - приложение переходит в ожидание нажатия на кнопку "Нажми для начала диалога".

В соответствии с перечисленными задачами рассмотрим основные отдельные блоки приложения.

1. Создается класс приложения MainActivity, наследуемый от класса Activity - базового класса для всех экранов приложения. Здесь используется интерфейс TextToSpeech.OnInitListener для инициализации синтезатора речи.

```
public class MainActivity  extends Activity implements TextToSpeech.OnInitListener  {
   // Объявление переменных
   public String sp10 = " выключить зеленый "; public String sp20 = " включить зеленый ";
...
//Socket, с помощью которого будут отправляться данные на bluetooth Arduino
   private BluetoothSocket clientSocket;
   // UUID для случая подключения к последовательному bluetooth  устройству
   private UUID uuid = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
...
```

```
    // Описание хеш карты для хранения
    // ключа (TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID - в нашем случае)
    // и любого значения - строки (например - "5678")
    private HashMap<String, String> params = new HashMap<>();
    @Override
    public void onCreate(Bundle savedInstanceState) {
...
        tts = new TextToSpeech(this, this);
        tts.setOnUtteranceProgressListener(new utteranceProgressListener()); //Установка
слушателя
//синтеза речи
        bluet(); // Подключение bluetooth
    }
// Вызов активности распознавателя голоса при нажатии на кнопку
    public void click(View view) {
if(upbluetooth==0) { // Если Bluetooth подключен
// При нажатии на кнопку останавливаю таймер работы
// программы по передаче и чтению  температуры
        time.cancel(); time.purge();
// Запускаю таймер работы программы с новыми начальными
// данными(посылка запроса температуры и прием температуры)
        bluecancel = 0; time = new Timer();
        bluetoothInOut bInOut = new bluetoothInOut(); time.schedule(bInOut,500,1000);
       }
        spi(); // Запускаю распознаватель речи
    }
  @Override
// Следующий метод выполняет инициализацию синтезатора речи.
// Он требует назначения интерфейса TextToSeech.OnInitListener для
// базового класса приложения MainActivity
 public void onInit(int status) {
 if (status == TextToSpeech.SUCCESS) {// При успешной инициализации
tts.setLanguage(Locale.getDefault()); // выполняется использование языка
// синтезатора, установленного по умолчанию
// Передача параметра params для вызова слушателя синтезируемой речи.
params.put(TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID,"HELLO"); } else
{txtText.setText("Инициализация не выполнена"); } }
... (далее классы и методы, которые кратко описаны ниже)
}
```

2. Метод spi() используется для запуска активности распознавания голоса Google с помощью механизма Intent. Активность формирует запрос к серверам Google и посылает его через Интернет.

```
    public void spi() {
// Intent подготавливает запуск активности распознавания голоса
 Intent intent =  new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
// Передача в активность параметров. Задает подсказку пользователю
intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "Говори громко и четко!");
// Запускается активити, выполняющее распознавание произнесенных
// фраз и возвращающая результаты распознавания
        startActivityForResult(intent, CODE);
}
```

3. Метод onActivityResult используется в программе для получения результатов работы Активити распознавания речи. Фактически он принимает результат распознавания с серверов Google через Интернет и передает их основному приложению.

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
// Возврат результатов распознавания речи
    super.onActivityResult(requestCode, resultCode, data);
// Если это результаты распознавания речи (CODE)
// и процесс распознавания прошел успешно
    switch (requestCode){ case CODE: { if (resultCode == RESULT_OK && null != data)
{
 // то получаем список текстовых строк - результат распознавания.
// Строк может быть несколько, а правильные результаты идут в начале
            ArrayList<String> spee =
data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
sp =spee.get(0); // Из массива строк - результатов выбирается самая первая
txtText.setText(sp); // и выводится на экран
                spout = sp;
 // Сопоставления результата распознавания со строками вопросов
                int r10 = sp.compareTo(sp10);   ...
// В зависимости от результата сопоставления выбирается тот или иной ответ
                if (r10 == 0) spout = sp20;  ...
if (upbluetooth == 0) {  // Если подключение к bluetooth существует то
                // результат сравнения представляем символом
                // Например, если результат распознавания голоса соответствует строке
                // "включить зеленый" то на bluetooth посылаем символ 3 (код 51)
                ... if (r3 == 0) value = 51; ...
                // Если запрашиваем температуру, то посылаем данные
                if (value != 54) outData(value);
                if (value == 54) { temtext.setText(readMessage);    // Распечатываем
температуру
                spout = readMessage;  } }
// Синтез речи для выбранного ответа
                tts.speak(spout, TextToSpeech.QUEUE_ADD, params);
        }       } break; } }
```

4. Объявляем класс utteranceProgressListener, наследуемый от класса UtteranceProgressListener. Он необходим для выполнения действий после прослушивания результата синтеза речи. Выше было отмечено, что Активити распознавания речи должно запускаться автоматически после произнесенной синтезатором речи фразы до тех пор, пока не встретятся голосовая команда "конец связи". Этот класс содержит необходимый нам метод, который выполняется после завершения звука динамиком, сгенерированным синтезатором.

```
public class utteranceProgressListener extends UtteranceProgressListener {
  @Override
  public void onDone(String utteranceId) { // Действия после окончания речи
синтезатором
    r19 = sp.compareTo(sp19);
    if(r19 != 0) spi(); // Если не "конец связи", то активити распознавания голоса
запускается вновь
    else  { if(upbluetooth==0) {  // Если Bluetooth включен
      // и если произнесено "конец связи" - останавливаю таймер
      time.cancel(); time.purge();
```

```
// и запускаю таймер работы программы с новыми начальными данными
// (без посылки запроса температуры но с приемом температуры)
            bluecancel = 1;
            time = new Timer();
            bluetoothInOut bInOut = new bluetoothInOut();
            time.schedule(bInOut,500,1000);  } } }
        @Override
        public void onStart(String utteranceId) { }
        @Override
        public void onError(String utteranceId) { txtText.setText("Error"); } }
```

5. Класс bluetoothInOut, который наследуется от класса TimerTask необходим для запроса температуры с Ардуино один раз в секунду.

```
public class bluetoothInOut extends TimerTask {
    public void run() {
       try { if( bluecancel == 0 ) // Только в этом случае посылаем запрос температуры
          {  OutputStream outStream = clientSocket.getOutputStream();
outStream.write(54);}
            // Получаем входной поток для приема данных
            InputStream inb = clientSocket.getInputStream();
            // Преобразование входного потока от bluetooth в строку
            DataInputStream in = new DataInputStream(inb);  bytes = in.read(buffer);
            if (bytes > 10 )  // Если через bluetooth получено (например) больше 10 байт, то
            { // преобразуем байты в строку с нулевого индекса до индекса bytes
              readMessage = new String(buffer, 0, bytes); }
        } catch (IOException e) {  }  }  }
```

Необходимо подчеркнуть следующие особенности работы программы.

1. Перед запуском синтезатора речи необходима его инициализация. Для этого основной класс MainActivity создается с использованием интерфейса TextToSpeech.OnInitListener. При инициализации используется метод onInit(int status).

2. Для работы метода onDone класса utteranceProgressListener, который следит за окончанием генерации динамиком звука синтеза речи, необходимо описание хеш карты для хранения ключа в params. При инициализации синтезатора необходимо выполнить передачу ключа с помощью метода params.put(TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID," HELLO"). Необходимо также установить слушатель синтеза речи tts.setOnUtteranceProgressListener(new utteranceProgressListener());

3. Чтение температуры с Ардуино выполняется раз в одну секунду с помощью классов Timer и TimerTask, которые выполняют запуск задачи в определенное время в будущем через определенные интервалы(500 и 1000 миллисекунд соответственно по программе). Попытка организации запроса температуры с последующим ее чтением в основном потоке Thread приводит к тому, что для получения текущего значения температуры необходим ее повторный запрос. Т.е. для получения правильного текущего значения температуры необходимо два раза сказать: "температура". Использование отдельного потока Thread с классом Handler работает правильно но приводит к ошибочной передаче кириллицы. Латиница передается верно, как это представлено в работе[6]. Поэтому здесь использовался способ чтения температуры с одновременным ее запросом с Андроид каждую секунду в переменную readMessage, которая доступна в любой момент синтезатору голоса. При команде "температура" ее значение берется из строки readMessage и синтезируется синтезатором.

Текст программы на языке Java с файлом разметки и манифестом для Android Studio 1.0.1. представлен в источнике [4]. Видео - ролик работы разработанной здесь системы представлен в источнике [3].

На рисунке 3. показана схема подключения к Arduino UNO устройства связи Bluetooth HC-05, датчика температуры DS18B20 и трех светодиодов красного, зеленого и синего цвета (альтернатива трем исполнительным механизмам, например свет, телевизор и кондиционер).

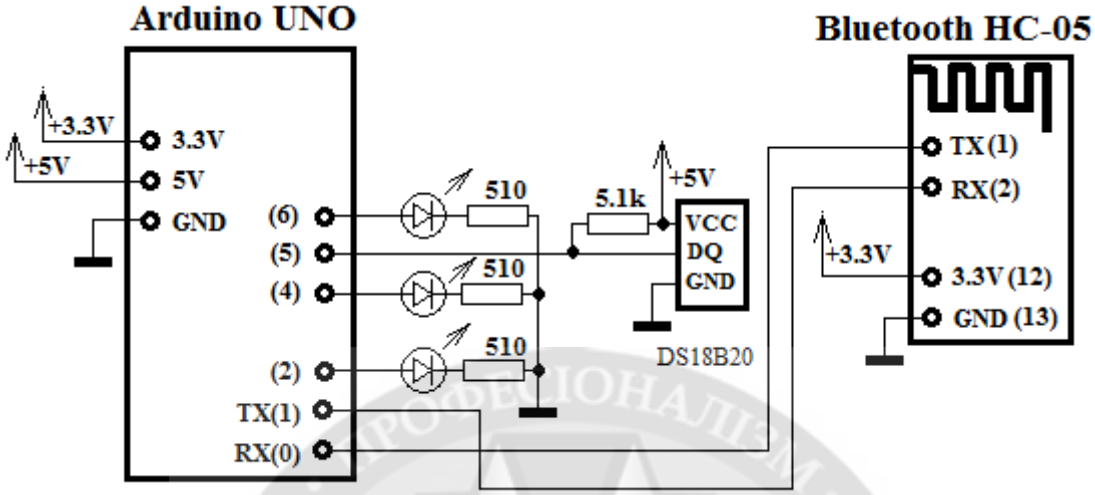

Рис. 3. Схема подключения устройств к Ардуино

Программа для Ардуино написана в стандартной среде разработки Ардуино[5] и ее тест приведен в источнике[4].

Выводы.

1. Разработаны программы для смартфонов и планшетов на ОС Андроид и Ардуино , позволяющие выполнять голосовое управление устройствами, подключенными к Ардуино. Здесь используется инструментарий Google для распознавания и синтеза речи. Причем повторный запуск Активити распознавания выполняется в цикле после окончания генерации звука благодаря использования класса UtteranceProgressListener.

2. Представлена схема подключения к Ардуино устройств, которыми выполняется голосовое управление.

3. Следует отметить высокое качество распознавания произнесенных фраз при работе с серверами Google через Интернет и более низкое при работе с библиотекой речевого поиска Google в режиме offline, если мобильное устройство поддерживает этот режим.

4. При создании библиотеки голосовых команд необходимо подбирать такие команды, которые распознаются с меньшими ошибками. Особенно это относится к режиму offline.

5. Опыт показал, что при создании систем голосового управления для более надежного распознавания и автономности целесообразно использовать технологию распознавания с коротким словарем команд, которая работает всегда в режиме offline.

6. Представленный здесь голосовой интерфейс можно рекомендовать для создания систем простой справочной службы и голосового управления.

ЛИТЕРАТУРА:

1. А.В. Фролов Г.В. Фролов. Синтез и распознавание речи. Современные решения. / Интернет-ресурс. - Режим доступа: http://www.frolov-lib.ru/books/hi/index.html. - 2003.

2.Android / Интернет-ресурс. - Режим доступа: http://developer.android.com/reference/android/package-summary.htm.

3. Андроид и ардуино в задачах голосового управления через Bluetooth. / Интернет-ресурс. - Режим доступа: https://youtu.be/zis9Q8AuxuU

4. Мясищев А.А. Андроид и Ардуино в задачах управления голосом и синтеза речи с использованием Bluetooth. / Интернет-ресурс. - Режим доступа:

https://sites.google.com/site/webstm32/8-upravlenie-golosom-s-pomosu-android-i-arduino. - 2015.
5. Arduino. Официальный сайт. [Electronic resource]. - Mode of access: http://arduino.cc , 2014.
6. Передача данных по Bluetooth между Android и Arduino. / Интернет-ресурс. - Режим доступа: http://cxem.net/arduino/arduino64.php. - 2012.

Без рецензії.

д.т.н., проф. Мясищев О.А., Лєнков О.С.

ANDROID І ARDUINO В ЗАВДАННЯХ УПРАВЛІННЯ ГОЛОСОМ І СИНТЕЗУ МОВИ

У роботі розроблені програми для Андроїд смартфонів і Ардуїно, що дозволяють виконувати голосове управління пристроями, підключеними до Ардуїно. Використовується інструментарій Google для розпізнавання і синтезу мови. Причому повторний запуск активіті розпізнавання виконується в циклі після закінчення генерації звуку завдяки використання класу UtteranceProgressListener. Зв'язок між Андроїд пристроєм і Ардуїно виконаний через Bluetooth.

Представлено схему підключення до Ардуино пристроїв, якими виконується голосове управління. Слід зазначити високу якість розпізнавання вимовлених фраз при роботі із серверами Google через Інтернет і більше низьку при роботі з бібліотекою мовного пошуку Google у режимі offline, якщо мобільний пристрій підтримує цей режим. Представлений голосовий інтерфейс можна рекомендувати для створення систем простої довідкової служби й голосового управління.

Ключові слова: Андроїд, Ардуїно, Bluetooth HC-05, Java, Android Studio, планшет, смартфон, розпізнавання мови, синтез мови.

Prof. Myasischev A.A., Lenkov O.S.

ANDROID AND ARDUINO CONTROL PROBLEMS IN VOICE AND SPEECH SYNTHESIS

The work program developed for Android smartphones and Arduino, which perform voice control devices connected to the Arduino. Google tools used for speech recognition and synthesis. And restart Activiti recognition is performed in the cycle after the sound generation due to the use of class UtteranceProgressListener. Communication between Android and Arduino device is made via Bluetooth.

The scheme of connection to the Arduino device which performs voice control. It should be noted the high quality of recognition of spoken phrases when dealing with Google's servers via the Internet and lower when working with library Google voice search mode offline, if the mobile device supports this mode. Presented voice interface can be recommended to create a simple help desk systems and voice control.

Keywords: Android, Arduino, Bluetooth HC-05, Java, Android Studio, tablet, smartphone, speech recognition, speech synthesis.