OUARGA Ayoub VYTHILINGAM Kogilan GESRET Valentin

Pilotage de robot à l'aide d'un appareil Android

Projet tuteuré – Semestre 4

DUT Réseaux et Télécommunications

Encadrant : M.LODDO
Tuteur : M. FEYBESSE
Année : 2016-2017

UNIVERSITÉ PARIS 13

Remerciements

Nous tenons à remercier M. FEYBESSE pour avoir été présent lors des réunions de groupe pour notre projet, mais également pour nous avoir fourni du matériel pour le robot et pour nous avoir donné des idées pour la conception du robot ainsi que la programmation de celui-ci. Nous tenons à remercier M.LODDO pour avoir encadrer ce projet tuteuré. Nous tenons à remercier également M.SAIU pour nous avoir aidé à programmer en langage JAVA, ainsi que Mme PAULIAN pour nous avoir aidé tout au long de notre DUT à rédiger des rapports.

Table des matières

Abstract 4
Introduction 5
I. Présentation du projet 6
A. Les principaux objectifs 6
B. Les équipements et outils utilisé 6
1. L'environnement Android et le logiciel Android Studio 6
2. La carte Arduino Mega et le logiciel Arduino 7
3. Les équipements utilisés 7
C. Déroulement du projet 8
1. Répartition des tâches et mode de travail 8
2. La recherche d'information 8
II- L'évolution du projet 8
A. Les premiers pas avec la carte Arduino 8
1. Simulation en ligne 8
2. Notre premier programme « Feu Tricolore » 9
3. La carte Arduino Mega 9
B. Le robot Arduino 9
C. Les télécommandes Android 10
1. La télécommande de Google Play 10
2. La télécommande d'Android Studio 11
D. Les différents obstacles 13
1. La carte de puissance 13
2. Le mode d'alimentation 13
E. Les solutions 13
1. La marche arrière 13
2. Une alimentation original 14
III. Finalisation 15
A-La mise en forme du robot 15
B-Le parcours d'obstacles 15
C-Quelques idées pour l'avenir… 16
Conclusion 17
Table des illustrations 18
Glossaire 19
Annexe 20

Abstract

In semester 3 we, OUARGA Ayoub, GESRET Valentin and VYTHILINGAM Kogilan, we needed to chose a project for the 2nd years of our DUT. We had the choice with many subject and we had chosen the project that we have to make a robot, which we command with an Android remote.

We had chosen this project because younger, all of three loved to control a car toy. And this was the chance for us to make our own robot that we control which a remote that we make.

Our project consists to make a robot that we can command by Bluetooth with an Android remote. To do this project correctly, we caught some information on the Internet and after that we ask to Mr FEYBESSE some equipment to do ours robot.

To do this robot we have an Arduino Card, a Bluetooth module, some electronics wires.

We start the project with the fabrication of the robot. In first we programmed the Arduino Card in language C++ and use a remote that is already did on the Android platform to assure all thinks works. After that we make the remote, which control our robot with the Android Studio software.

When the semester 4 starts we had finished a major part of our work and we think about some other way to control the robot.

Introduction

Notre projet met en œuvre deux parties, un robot mobile et sa commande à distance qui se fait à l'aide d'un appareil Android. Afin de réaliser la communication ainsi que l'interface tactile, nous nous sommes servis des logiciels Android Studio pour l'application mobile et d'Arduino pour programmer la carte Arduino Mega qui sera positionné sur le robot.

Notre sujet nous a été confié par M.LODDO qui est l'encadrant du projet tuteuré, ainsi que M.FEYBESSE qui nous a suivis tout au long de notre projet.

Problématique :

Comment associer deux types de langage de programmation afin de pouvoir contrôler un robot ?

Dans un premier temps, nous présenterons le projet qui nous a été confié, dans un second temps nous verrons l'évolution de celui-ci, puis pour finir nous verrons le projet à l'état final ainsi que les améliorations qui peuvent lui être apporté.

I. Présentation du projet

Pour présenter notre projet, nous allons expliquer nos objectifs dans un premier temps, présenter les différents équipements et outils que nous avons utilisé dans un second temps, puis pour finir nous expliquerons notre méthode de travail durant le déroulement de ce projet.

A. Les principaux objectifs

Lors des choix de sujet pour notre projet tuteuré, nous avons mis le sujet « Pilotage de robot à l'aide d'un appareil Android » en premier choix. La raison était que nous aimons assez bien la programmation et nous avons voulu mettre en œuvre nos connaissances pour concevoir un robot. De plus, nous avons gardé une image d'enfant en nous qui fait que vouloir télécommander un appareil que nous allons concevoir serait un réel plaisir, ce qui fut le cas. Pour répondre à nos objectifs, nous avons donc utilisé un appareil mobile sous Android, ainsi qu'un robot sur lequel se trouve une carte Arduino.

B. Les équipements et outils utilisé

Pour notre projet, l'objectif principal est de piloter un robot par le biais d'un appareil Android à travers un module Bluetooth. Pour cela, nous avons dû utiliser le langage de programmation *Java* et le *C++*.

1. L'environnement Android et le logiciel Android Studio

Tous d'abord, Android est un système d'exploitation mobile. A l'heure actuelle, plus de deux tiers des Smartphones sont sous Android. Dans cette partie du projet, la télécommande du robot sera une application mobile adaptée aux téléphones Android. Pour cela, nous utiliserons le logiciel *Android Studio*, qui est un environnement de développement d'application mobile pour appareil Android. Pour programmer sur cette interface on utilise le langage de programmation Java.

Figure 1 Logo Android

Figure 2 Android Studio

2. La carte Arduino Mega et le logiciel Arduino

Nous avons vu comment le robot sera piloter, mais concernant le robot en question, nous utiliserons l'environnement Arduino afin d'exécuter les fonctions électronique. L'environnement Arduino est un ensemble d'éléments qui contribuent au fonctionnement d'équipements électronique. Pour cela, nous avons une carte Arduino Mega que nous allons programmer à l'aide du logiciel Arduino. Celle-ci, sera positionnée sur le robot afin de faire tourner les roues par exemple. Pour programmer sur Arduino on utilise le langage de programmation C++.

Figure 3 Logo Arduino

3. Les équipements utilisés

Comme vu précédemment, nous avons pour objectif d'assembler deux équipements différents, un Smartphone sous Android et un robot qui possède une carte Arduino Mega.

Pour lier ces deux équipements, nous n'utiliserons pas de câble directement, nous avons décidé de mettre un module Bluetooth sur la carte Arduino afin de gérer les communications entre le Smartphone et le robot.

Le module Bluetooth que nous avons est le suivant :

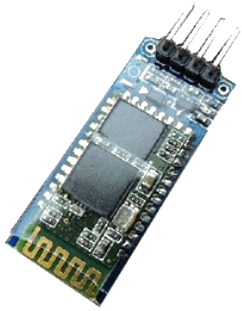

Figure 4 Module Bluetooth

Le robot est constitué d'un châssis, de deux roues, de la carte Arduino Mega, une batterie d'alimentation et le module Bluetooth.

Figure 5 Châssis et roues

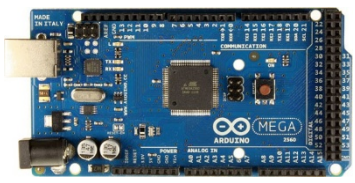

Figure 6 Carte Arduino Mega

Figure 7 Batterie

C. Déroulement du projet

1. Répartition des tâches et mode de travail

Tout au long de notre projet, nous avons toujours travaillé en groupe. Nous avons tous fais ensemble afin qu'aucun d'entre nous ne soit perdu. Nous avons débuté par la programmation de la carte Arduino, puis nous avons conçu l'application mobile, puis pour finir nous avons monté le robot.

Nous nous réunissons lors des week-ends et durant les vacances à la bibliothèque universitaire de Paris 13 Bobigny. Nous avons choisi ce lieu car nous avons la possibilité de réserver des salles afin de travailler en discutant tranquillement, mais également car elle reste ouverte jusqu'à 22h. De plus, nous n'habitons pas très loin de Bobigny, donc c'étais bénéfique pour nous.

2. La recherche d'information

La recherche d'information n'a pas été restreinte, c'est-à-dire que nous avons été ouverts à toutes aides possibles. Nous avons cherché la documentation de chacun des langages sur internet. Nous avons trouvé un cours de programmation pour appareil Android qui explique en détail les commandes à utiliser pour programmer correctement une application et l'installer sur un appareil mobile. Concernant la programmation de la carte Arduino, nous avons utilisé les différentes bibliothèques du logiciel Arduino, comme pour l'utilisation du module Bluetooth.

II- L'évolution du projet

A. Les premiers pas avec la carte Arduino

1. Simulation en ligne

Après avoir fait nos différentes recherches sur comment nous allions procéder pour la réalisation de notre projet, nous avons pris un rendez-vous avec M.FEYBESSE pour lui soumettre la liste de matériaux qu'on aurait besoin pour le prochain rendez-vous. Pendant ce temps, nous avions commencé à nous familiariser avec Arduino via des sites internet de simulation tel que Fritzing ou bien Circuits.io.

Fritzing et Circuits.io sont des sites très complets, mais Circuits.io est plus facile à utiliser. Dans un premier temps, nous avons manipulé la carte Arduino et quelques pins de la carte avec un code basique qui envoie du courant dans un pin afin d'allumer une LED. Grâce aux principales fonctions de la carte Arduino, nous avons donné quelques fonctionnements à cette LED, c'est à dire qu'elle s'allume une fois pendant 2s, puis 5s, ensuite 8s, et enfin qu'elle s'éteigne.

2. Notre premier programme « Feu Tricolore »

Petit à petit nous avons compris la base d'Arduino et ses programmations simplistes. Nous avons pensé au feu tricolore de tous les jours et nous avons essayé de le reproduire en virtuel, le temps de recevoir la vraie carte Arduino. Nous avons placé deux LEDs supplémentaire afin d'avoir les trois feux tricolores. Nous avons donné à chacune une durée pendants laquelle elles s'allumaient puis s'éteignaient. De cette façon nous avons obtenu un mini programme de feu rouge.

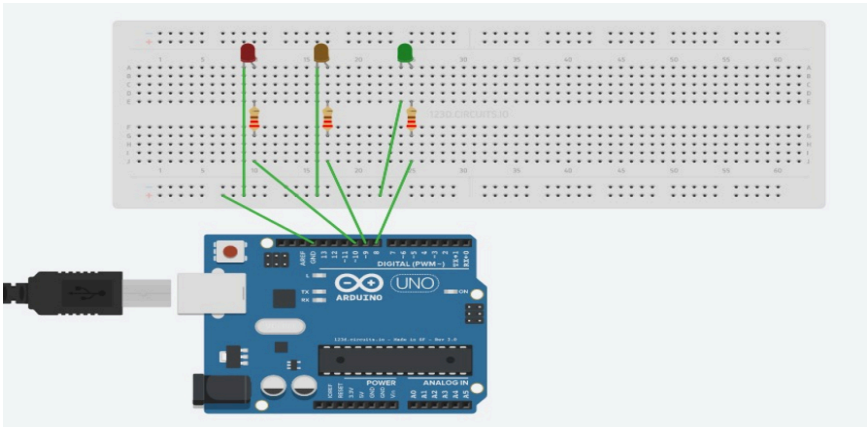

Figure 8 Virtualisation du feu tricolore

3. La carte Arduino Mega

Le moment venu, nous avions obtenu une majorité de nos équipements. Plus particulièrement la carte Arduino Mega. Nous avions installé le logiciel Arduino permettant de compiler le programme sur la carte. Nous avons télé versé notre programme feu rouge et fait les multiples branchements sur le breadboard. Nous obtenions notre feu tricolore marchant à la perfection.

Malheureusement c'est un programme automatique contrairement à notre projet qui est de télécommander un robot. Nous avons modifié ce programme et rendu chaque LED indépendante de l'autre et nous leurs avons donné un numéro de fonction. C'est ce numéro attribué qui va nous permettre d'allumer une LED à un moment voulu grâce au module Bluetooth qu'on ajoute dans le programme.

B. Le robot Arduino

Après avoir mis les LEDs indépendantes l'une de l'autre, nous avons dédié chaque LED à une roue de notre futur robot, c'est à dire que la LED verte sera associé à la roue gauche et la LED rouge sera associé à la roue droite. Ainsi pour aller tout droit on enverra du courant dans les deux pins en même temps, pour aller à droite on allumera la LED verte et inversement pour allé à gauche.

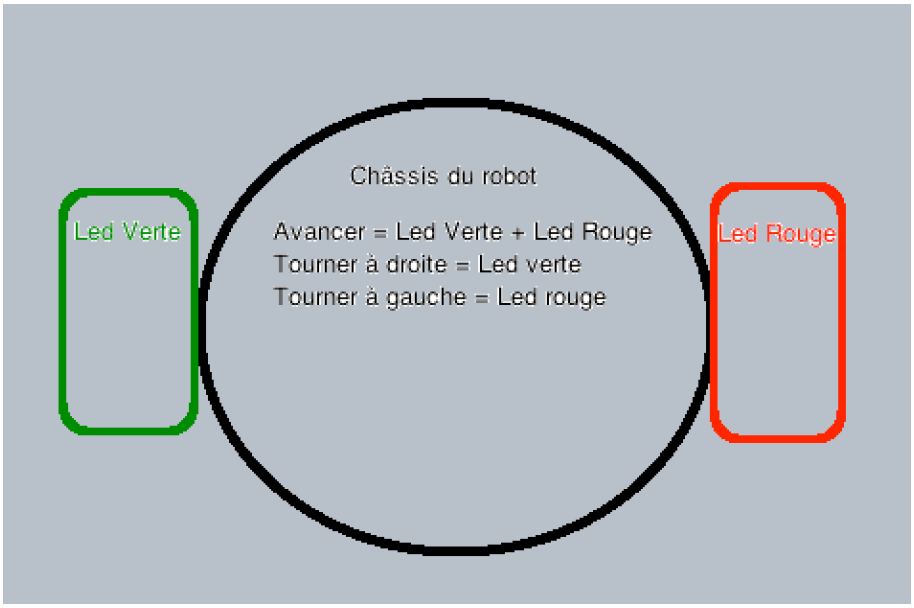

Figure 9 Correspondance des leds avec les roues

```
//Pour aller tout droit
if(ch == '3'){
  digitalWrite(5, HIGH);
  digitalWrite(3, HIGH);
 Serial1.print("led rouge et verte allumé");
 }
```

Figure 10 Code pour aller tout droit

C. Les télécommandes Android

1. La télécommande de Google Play

A ce stade, nous avons trois LEDs déterminé avec différent chiffre et associé à différentes roues qu'on peut contrôlé à l'aide du module Bluetooth. Mais pour cela, il nous faut une télécommande Android, à notre niveau on ne savait pas comment crée de toute pièce une télécommande à l'aide d'Android Studio.

Cependant nous avons pris une télécommande spécialement conçue pour notre module Bluetooth afin de tester notre expérience.

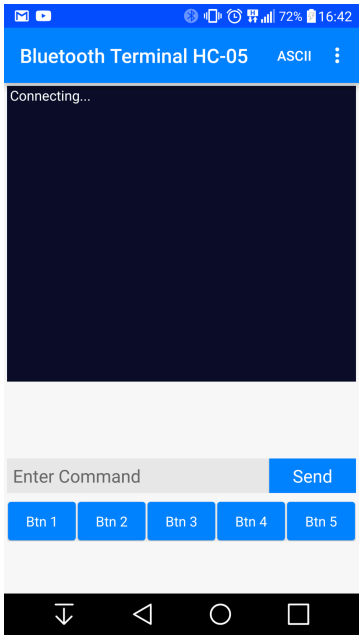

Figure 11 Télécommande Android disponible sur Google Play

2. La télécommande d'Android Studio

Grâce à la télécommande déjà programmée, nous avons pu confirmer que notre expérience de LEDs indépendante fonctionnait parfaitement. Maintenant, nous allons faire de toute pièce notre propre télécommande grâce au logiciel « Android Studio ».

Pour cette télécommande nous sommes parti sur un style très utilisé, une flèche pour chaque direction. C'est un modèle très simpliste mais compréhensible.

Par exemple, les deux boutons pour aller à droite et à gauche nous les avons définit avec un LinearLayout et nous avons divisé en trois parties égales pour placer les boutons équitablement. De plus, nous avons ajouté une option intéressante qui est la commande vocale, nous avons donné quelques fonctionnalités applicable, tel que « avancer », « tourner à droite », etc. Pour parvenir à cela, nous avons utilisé la commande vocale de Google.

```
    <!-- Bouton du milieu -->
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="100dp">

    <Button
        android:rotation="180"
        android:id="@+id/flecheGauche"
        android:background="@drawable/fleche_droite"
        android:layout_width="match_parent"
        android:layout_weight="1"
        android:layout_height="match_parent" />
    <Button
        android:id="@+id/boutonStop"
        android:background="@drawable/iut_villetaneuse"
        android:layout_width="match_parent"
        android:layout_weight="1"
        android:layout_height="match_parent" />
    <Button
        android:layout_width="match_parent"
        android:id="@+id/flecheDroite"
        android:background="@drawable/fleche_droite"
        android:layout_weight="1"
        android:layout_height="match_parent" />

</LinearLayout>
```

Figure 12 Code pour la partie milieu de la télécommande

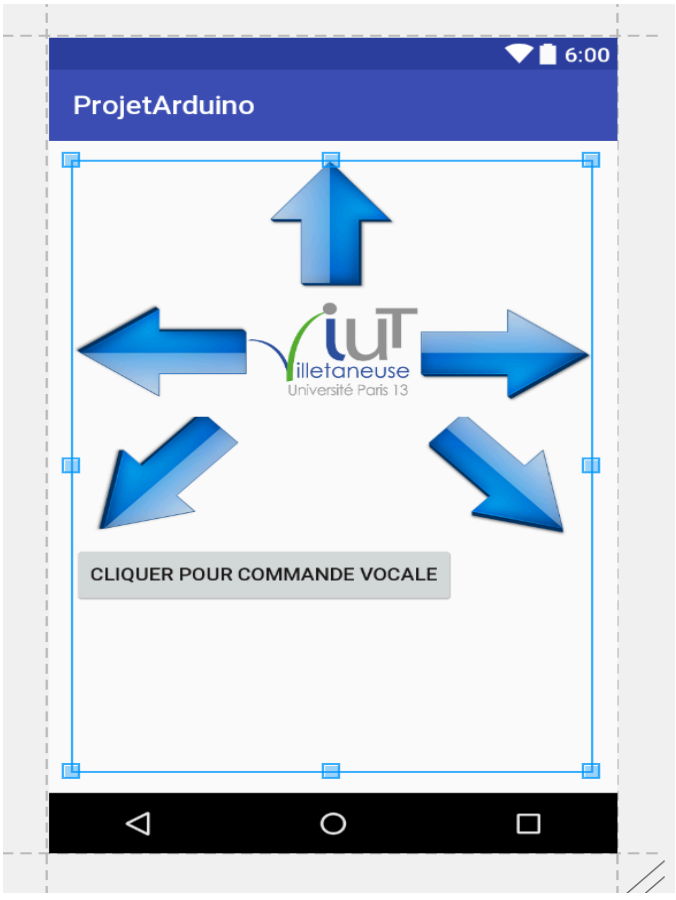

Figure 13 Télécommande crée sur Android Studio

D. Les différents obstacles

1. La carte de puissance

Une fois le robot monté sur son châssis, nous avons pensé à la marche arrière. Pour cela, nous avions besoin d'une carte de puissance. Cette carte permet d'inverser la polarité des roues et donc d'exercer une marche arrière. Nous avions eu l'autorisation de M.FEYBESSE de réutiliser l'ancien robot et donc d'utiliser la carte de puissance de celui-ci. Malheureusement cette carte de puissance n'était pas du tout compatible avec la carte Arduino Mega. De plus, nous ne pouvions pas réutiliser la carte Uno car celle-ci était limitée au niveau des pins ainsi que du processeur.

2. Le mode d'alimentation

Comme nous l'avions dis précédemment, nous pouvions réutiliser tous les équipements de l'ancien robot et donc sa batterie. Cependant la batterie utilisée avec l'ancien robot avec un câble spécifique, était inutilisable car on ne pouvait plus la recharger. Nous avons donc décidé de charger la carte Arduino directement avec un câble branché à un ordinateur, mais cela donnait une portée très restreinte.

E. Les solutions

1. La marche arrière

Pour palier au problème de l'inversion de polarisation des moteurs, nous avons ajouté deux boutons en plus sur notre télécommande pour pouvoir faire un demi-tour et repartir tout droit.

Figure 14 La marche arrière

2. Une alimentation original

Pour contourner le problème de l'alimentation et de la portée très limitée, nous avons utilisé un nouveau type de chargeur très utilisé en ce moment, la batterie externe. Cette batterie externe nous offre assez d'autonomie avec un poids léger.

III. Finalisation

A-La mise en forme du robot

Comme vous pouvez le constater précédemment, tous les composants de notre robot sont à nu et susceptibles d'être touché par les éléments extérieurs. Nous voulions mettre une sorte de boite autour de lui, mais nous trouvions que le coté esthétique était important de ce fait, une boite opaque n'était pas optimale c'est pour cela que nous avons optés pour un plastique transparent en forme de tube et ainsi laisser une vue complète sur les composants du robot et la carte Arduino.

B-Le parcours d'obstacles

Pour rendre plus intéressant l'objectif de notre projet, nous avons décidé de créer un parcours d'obstacles. Celui-ci contient un ralentisseur, trois obstacles, ainsi que un feu tricolore.

Voici le schéma de notre parcourt :

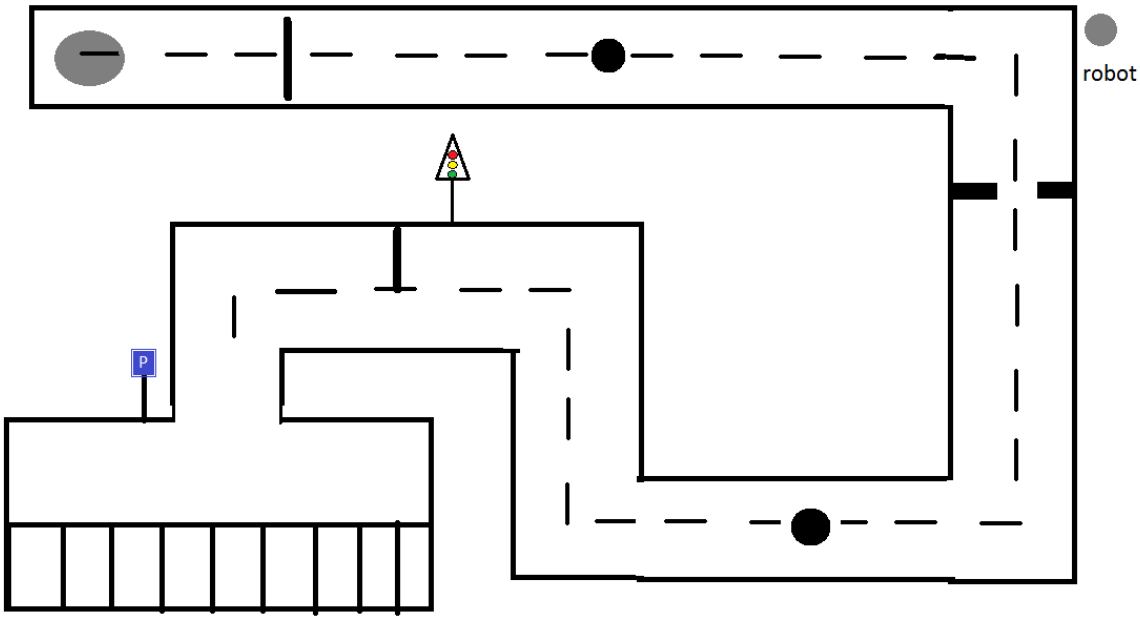

Figure 15 Plan du parcours d'obstacle

Ce parcourt sera effectué à l'aide de plusieurs télécommandes.

La première télécommande que nous allons utiliser est la première que nous avons programmée. C'est la télécommande qui nous a suivie durant tout le projet, ensuite nous allons le piloter grâce à une commande vocale, puis pour finir nous utilisions une commande avec des boutons poussoir le Controller au mieux.

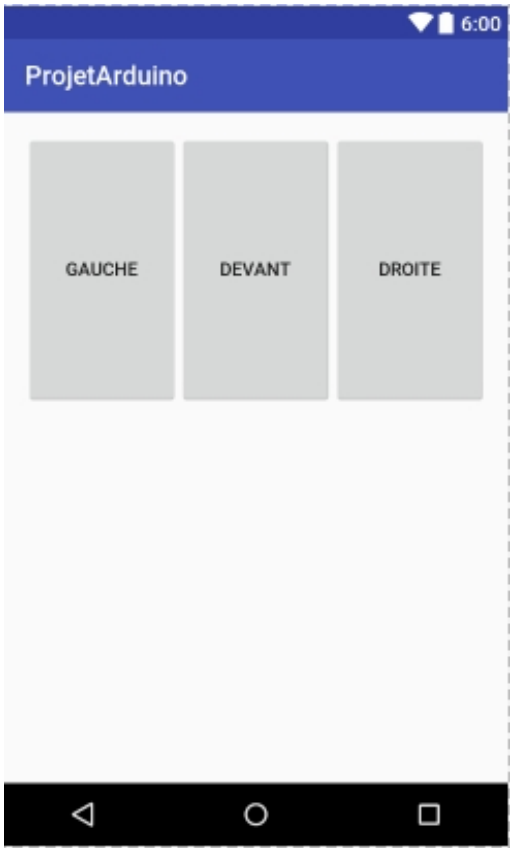

Figure 16 Télécommande amélioré

C-Quelques idées pour l'avenir…

Pour des personnes voulant continuer notre projet, nous avons pensé à plusieurs nouvelles fonctionnalités, tel que l'ajout d'une caméra pour voir où va le robot. Nous avons aussi réfléchis à changer le moyen de communication entre le robot et le téléphone et ainsi remplacer le Bluetooth par une connexion internet ce qui permettra de piloter le robot à distance, pour éviter tout accident il faudrait ajouter des radars qui arrêterais le robot lors de la rencontre avec un obstacle. Pour compléter parfaitement la télécommande, nous conseillons de développer une fonction gyroscopique à l'aide de la carte adéquate pour ainsi piloter le robot de manière originale.

Conclusion

Pour conclure, grâce à notre projet, nous avons vu que nous pouvons utiliser deux langages de programmation différents afin de pouvoir contrôler un robot. Pour cela, nous avons utilisé l'appareil Android, ainsi que la carte Arduino sur un robot. Ce projet nous a plu énormément car nous avons pu approfondir nos connaissances en programmation Java, langage que nous avons étudié tout au long de notre DUT. De plus, nous avons appris un nouveau langage qui est le C++ afin de programmer la carte du robot.

Malgré certains objectifs que nous n'avons pas pu atteindre, nous sommes satisfaits de notre projet final et nous laissons ces objectifs pour des futures personnes voulant approfondir ce projet.

Table des illustrations

Figure 1 Logo Android 6
Figure 2 Android Studio 6
Figure 3 Logo Arduino 7
Figure 4 Module Bluetooth 7
Figure 5 Châssis et roues 7
Figure 6 Carte Arduino Mega 7
Figure 7 Batterie 7
Figure 8 Virtualisation du feu tricolore 9
Figure 9 Correspondance des leds avec les roues.....10
Figure 10 Code pour aller tout droit 10
Figure 11 Télécommande Android disponible sur Google Play 11
Figure 12 Code pour la partie milieu de la télécommande.....12
Figure 13 Télécommande crée sur Android Studio.....12
Figure 14 La marche arrière 13
Figure 15 Plan du parcours d'obstacle 15
Figure 16 Télécommande amélioré 16

Glossaire

Java : Langage de programmation informatique orienté objet.

C++ : Langage de programmation informatique compilé

Android : Système d'exploitation mobile

Android Studio : Logiciel de développement d'application Android

Arduino : Logiciel de programmation pour carte Arduino

Programmation : activité qui permet de créer un programme informatique

Carte de puissance : Carte qui permet d'inverser la polarité des moteurs et permet de gérer la vitesse des moteurs

Bluetooth : Système de communication sans fil

Batterie externe : Source d'énergie permettant un appareil quelconque par le biais d'un câble USB sans être brancher au secteur

Annexe

Code Première Télécommande :

Première Classe ConnectBT :

```
package com.example.kogi.projetarduino;

import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.location.Address;
import android.os.AsyncTask;
import android.widget.Button;
import android.widget.SeekBar;

import java.io.IOException;
import java.util.UUID;

/**
 * Created by Kogi on 14/01/2017.
 */

class ConnectBT extends AsyncTask<Void, Void, Void>  // UI thread
{
   private boolean ConnectSuccess = true; //La connexion est bonne


   Button btnOn, btnOff, btnDis;
   SeekBar brightness;
   String address = null;
   private ProgressDialog progress;
   BluetoothAdapter myBluetooth = null;
   BluetoothSocket btSocket = null;
   private boolean isBtConnected = false;
   static final UUID myUUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");


   public ConnectBT(String address){
      this.address=address;



      try
      {

         if (btSocket == null || !isBtConnected)
         {
            myBluetooth = BluetoothAdapter.getDefaultAdapter();//récupere le module blutooth
            BluetoothDevice dispositivo = myBluetooth.getRemoteDevice(address);//Sec connecte au blutooth et
regarde si c'est ok
            btSocket = dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);
            BluetoothAdapter.getDefaultAdapter().cancelDiscovery();
            btSocket.connect();//la connexion commence
         }
      }
      catch (IOException e)
      {
         ConnectSuccess = false;//Soulève une exeption si la connexion n'est pas faite
      }
```

```
    }


    @Override
    protected Void doInBackground(Void... devices) //
    {

        return null;
    }
    @Override
    protected void onPostExecute(Void result) //montre si tout est ok
    {
        super.onPostExecute(result);

        if (!ConnectSuccess)
        {
            //Affiche la connexion établie

        }
        else
        {

            isBtConnected = true;
        }
        progress.dismiss();
    }


    public void mouvement(String a){
        if (btSocket!=null)
        {
            try
            {
                btSocket.getOutputStream().write(a.toString().getBytes());
            }
            catch (IOException e)
            {

            }
        }
    }


    public void tournerDroite(){
       mouvement("1");
    }

    public void tournerGauche(){
        mouvement("2");
    }

    public void avancer(){
        mouvement("3");
    }
    public void stop(){
        mouvement("0");
    }

    public void reculerGauche() {mouvement("4");}

    public void reculerDroite() {mouvement("5");}
}
```

Deuxième Classe MainActivity.java :

```
package com.example.kogi.projetarduino;

import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.speech.RecognizerIntent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.Set;

public class MainActivity extends AppCompatActivity {

   boolean isBlueToothActive;
   private final static int REQUEST_CODE_ENABLE_BLUETOOTH = 0;
   BluetoothAdapter bluetoothAdapter;

   ConnectBT connectBT;

   Set<BluetoothDevice> pairedDevices ;


   private final BroadcastReceiver bluetoothReceiver = new BroadcastReceiver() {
      public void onReceive(Context context, Intent intent) {
         String action = intent.getAction();
         Log.i("ici","action = "+action);
         if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            Toast.makeText(MainActivity.this, "New Device = " + device.getName(),
Toast.LENGTH_SHORT).show();



         }
      }
   };



   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);


      //j'ai rajouté tout ce qu'il y a en dessous
      isBlueToothActive=true;
```

```
        activerBluetooth();


        try {
            Thread.sleep(3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        Toast.makeText(this,"Connexion établie",Toast.LENGTH_SHORT).show();
        ajouterListenerFleches();


        ajouterListenerBoutonVocal();



    }

    private void ajouterListenerBoutonVocal() {
        Button boutonSpeech = (Button) findViewById(R.id.speechButton);

        boutonSpeech.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
                intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                        RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
                intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "Que puis-je faire pour vous?");
                startActivityForResult(intent, 2);
            }
        });


    }

    private void ajouterListenerFleches() {
        //Bouton Avancer
        Button boutonHaut = (Button) findViewById(R.id.flecheHaut);
        boutonHaut.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if(connectBT!=null) {
                    connectBT.avancer();
                }else{
                    activerBluetooth();
                }
            }
        });


        Button boutonGauche = (Button) findViewById(R.id.flecheGauche);
        boutonGauche.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if(connectBT!=null) {
                    connectBT.tournerGauche();
                }
                    else{
                        activerBluetooth();
```

```
            }
        }
    });


    //Bouton Droite
    Button boutonDroite = (Button) findViewById(R.id.flecheDroite);
    boutonDroite.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(connectBT!=null) {
                connectBT.tournerDroite();
            }else{
                activerBluetooth();
            }
        }
    });

    Button boutonStop = (Button) findViewById(R.id.boutonStop);
    boutonStop.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(connectBT!=null) {
                connectBT.stop();
            }else{
                activerBluetooth();
            }
        }
    });


    Button boutonDemiTourGauche = (Button) findViewById(R.id.flecheDemiTourGauche);
    boutonDemiTourGauche.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(connectBT!=null) {
                connectBT.reculerGauche();
            }else{
                activerBluetooth();
            }
        }
    });

    Button boutonDemiTourDroite = (Button) findViewById(R.id.flecheDemiTourDroite);
    boutonDemiTourDroite.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(connectBT!=null) {
                connectBT.reculerDroite();
            }else{
                activerBluetooth();
            }
        }
    });



}

private void activateBluetooth(){
    //Activer Bluetooth
    if (!bluetoothAdapter.isEnabled()) {
```

```
        Intent enableBlueTooth = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableBlueTooth, REQUEST_CODE_ENABLE_BLUETOOTH);
    }
}

private void desactivateBluetooth(){
    bluetoothAdapter.disable();

}


private void activerBluetooth() {

    //Verifier la présence du bluetooth sur le téléphone
    bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (bluetoothAdapter == null) {
        Toast.makeText(MainActivity.this, "Pas de Bluetooth",
                Toast.LENGTH_SHORT).show();
    }else {
        Toast.makeText(MainActivity.this, "Avec Bluetooth",
                Toast.LENGTH_SHORT).show();
    }
    activateBluetooth();


    IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
    registerReceiver(bluetoothReceiver, filter);

    //bluetoothAdapter.startDiscovery();


    pairedDevices = bluetoothAdapter.getBondedDevices();



    for (Iterator<BluetoothDevice> it = pairedDevices.iterator(); it.hasNext(); ) {
        BluetoothDevice f = it.next();
        if(f.getName().equals("HC-05")){

            connectBT = new ConnectBT(f.getAddress());


        }
    }

}




@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (resultCode == RESULT_OK) {
        // L'utilisation a activé le bluetooth
    } else {
        // L'utilisation n'a pas activé le bluetooth
    }

        if (requestCode == 2 && resultCode == RESULT_OK) {
```

```
            ArrayList<String> matches = data.getStringArrayListExtra(
                  RecognizerIntent.EXTRA_RESULTS);
            TextView speechText = (TextView)findViewById(R.id.speechText);

            String texteResultat = matches.get(0).toString();
            texteResultat = texteResultat.trim();
            speechText.setText(texteResultat);

            if(texteResultat.equals("avancer") && connectBT!=null){
               connectBT.avancer();
            }else if(texteResultat.equals("tourner à droite")){
               connectBT.tournerDroite();
            }else if(texteResultat.equals("tourner à gauche")){
               connectBT.tournerGauche();
            }else if(texteResultat.equals("stop")){
               connectBT.stop();
            }
            else if(texteResultat.equals("reculer gauche")){
            connectBT.reculerGauche();
            }
            else if(texteResultat.equals("reculer droite")){
               connectBT.reculerDroite();
            }



         }
         super.onActivityResult(requestCode, resultCode, data);

   }



   @Override
   protected void onDestroy() {
      super.onDestroy();
      bluetoothAdapter.cancelDiscovery();
      unregisterReceiver(bluetoothReceiver);
   }
}
```

Code deuxième télécommande

Réutilisation de connectBT

Nouvelle classe MainActivity.java

```
package com.example.kogi.projetarduino;


import android.bluetooth.BluetoothAdapter;

import android.bluetooth.BluetoothDevice;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;
```

```
import android.content.IntentFilter;
import android.speech.RecognizerIntent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.Set;

public class MainActivity extends AppCompatActivity {

   boolean isBlueToothActive;
   private final static int REQUEST_CODE_ENABLE_BLUETOOTH = 0;
   BluetoothAdapter bluetoothAdapter;

   ConnectBT connectBT;

   Set<BluetoothDevice> pairedDevices;



   private final BroadcastReceiver bluetoothReceiver = new BroadcastReceiver() {
      public void onReceive(Context context, Intent intent) {
         String action = intent.getAction();
         Log.i("ici", "action = " + action);
         if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
```

```
            Toast.makeText(MainActivity.this, "New Device = " + device.getName(),
Toast.LENGTH_SHORT).show();
        }
    }
};


@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Button boutonGauche = (Button) findViewById(R.id.btnLeft);
    Button boutonDroite = (Button) findViewById(R.id.btnRight);
    Button boutonDevant = (Button) findViewById(R.id.btnDev);

    boutonGauche.setOnTouchListener(handleTouch);
    boutonDroite.setOnTouchListener(handleTouch);
    boutonDevant.setOnTouchListener(handleTouch);

    //j'ai rajouté tout ce qu'il y a en dessous
    isBlueToothActive = true;
    activerBluetooth();


    try {
        Thread.sleep(3000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }

    Toast.makeText(this, "Connexion établie", Toast.LENGTH_SHORT).show();
}
```

```
private View.OnTouchListener handleTouch = new View.OnTouchListener() {
    @Override
    public boolean onTouch(View v, MotionEvent event) {

        if(connectBT!=null) {
            switch (event.getAction()) {
                case MotionEvent.ACTION_DOWN:
                    switch (v.getId()){
                        case R.id.btnLeft:
                            connectBT.tournerGauche();
                            break;
                        case R.id.btnRight:
                            connectBT.tournerDroite();
                            break;
                        case R.id.btnDev:
                            connectBT.avancer();
                            break;
                    }
                    break;
                case MotionEvent.ACTION_UP:
                        connectBT.stop();
                    break;
            }
        }
        else{
            activateBluetooth();
        }
        return true;
    }
};
```

```
  private void activateBluetooth() {
    //Activer Bluetooth
    if (!bluetoothAdapter.isEnabled()) {
      Intent enableBlueTooth = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
      startActivityForResult(enableBlueTooth, REQUEST_CODE_ENABLE_BLUETOOTH);
    }
  }

  private void desactivateBluetooth() {
    bluetoothAdapter.disable();

  }


  private void activerBluetooth() {

    //Verifier la présence du bluetooth sur le téléphone
    bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (bluetoothAdapter == null) {
      Toast.makeText(MainActivity.this, "Pas de Bluetooth",
          Toast.LENGTH_SHORT).show();
    } else {
      Toast.makeText(MainActivity.this, "Avec Bluetooth",
          Toast.LENGTH_SHORT).show();
    }
    activateBluetooth();


    IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
    registerReceiver(bluetoothReceiver, filter);

    //bluetoothAdapter.startDiscovery();
```

```
        pairedDevices = bluetoothAdapter.getBondedDevices();


        for (Iterator<BluetoothDevice> it = pairedDevices.iterator(); it.hasNext(); ) {
            BluetoothDevice f = it.next();
            if (f.getName().equals("HC-05")) {

                connectBT = new ConnectBT(f.getAddress());



            }
        }

    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        bluetoothAdapter.cancelDiscovery();
        unregisterReceiver(bluetoothReceiver);
    }
}
```

Code Carte Arduino :

```
void setup() {
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(3, OUTPUT);
```

```
 pinMode(9, OUTPUT);
 Serial1.begin(9600);
 }

void loop() {
 if (Serial1.available()){
  char ch = Serial1.read();

  //Pour aller à droite
  if(ch =='1') {
   digitalWrite(3, HIGH);
   digitalWrite(5, LOW);
   Serial1.print("led verte allumer ");
 }

  //Pour aller à gauche
  if(ch =='2') {
   digitalWrite(5, HIGH);
   digitalWrite(3, LOW);
   Serial1.print("led rouge allumer ");
 }

  //Pour aller tout droit
  if(ch == '3'){
    digitalWrite(5, HIGH);
    digitalWrite(3, HIGH);
   Serial1.print("led rouge et verte allumé");
   }


  if(ch == '0'){
    digitalWrite(3, LOW);
```

```
    digitalWrite(5, LOW);

   Serial1.print("toute les leds eteinte, machine arret ");
   }


    if(ch == '4'){
    digitalWrite(5, HIGH);
    digitalWrite(3, LOW);
    delay(4000);
    digitalWrite(5, LOW);

   Serial1.print("Demi tour par la gauche");
    }

    if(ch == '5'){
    digitalWrite(5, LOW);
    digitalWrite(3, HIGH);
    delay(4000);
    digitalWrite(3, LOW);

   Serial1.print("Demi tour par la droite");
    }


  delay(100);
 }
```

```
}
```