A-Documentation sur les voitures Radiocommandés:

Télécommande bcp plus viable que piloter la voiture avec le téléphone en bluetooth si on souhaite avoir une voiture rapide.

- Piloter une Voiture Télécommandé via bluetooth avec Arduino Source Arduino & Raspberry Pi : <https://www.tutoriel-arduino.com/voiture-telecommande-arduino/?utm_source=cpphttps://www.tutoriel-arduino.com/voiture-telecommande-arduino/>

le composant L293d qui est un double pont-H, ce qui signifie qu’il est possible de l’utiliser pour commander quatre moteurs distincts (dans un seul sens) ⇒ recherche à faire sur ce composant

Télécharger une application bluetooth pour arduino

Dans ce lien : tuto branchement + code + appli bluetooth

-Construire une voiture télécommandée Source Pobot Robotique Sophia Antipolis :

[Construire une voiture télécommandée](https://pobot.org/Moteur-a-courant-continu-avec-mosfet-et-Arduino.html)

1: Il n’est pas possible de brancher directement un moteur à une carte Arduino parce que cette dernière ne peut pas lui délivrer suffisamment de courant : mon moteur nécessite environ 1A.

**Etape 1 :** piloter le moteur dans un seul sens et à vitesse constante

Solution: utiliser un composant électronique entre la carte Arduino et le moteur : un transistor MOSFET.

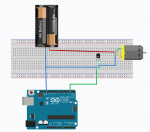
Composant: deux cartes Arduino UNO; un moteur continu en 2-12V; quelques fils de connexion

une plaque d' essai (breadboard); une source électrique (transformateur adapté au moteur);un double pont en H L298N pour les moteurs;un double pont en H L9110 pour le servo moteur;une carte SHD-JK Joystick Shield V2.0;deux émetteurs /récepteurs RF24....;un servomoteur Nikko.

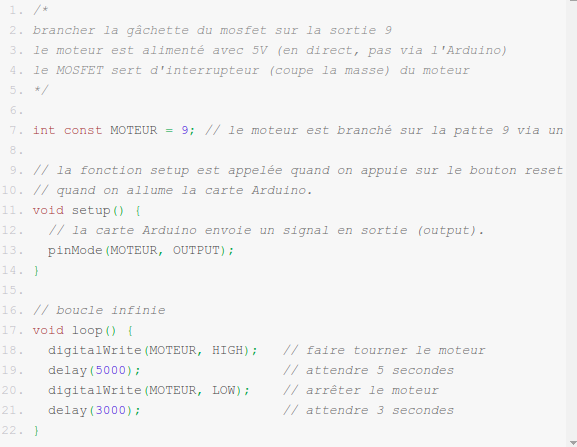
Fonctionnement du MOSFET :

Il y a 3 pattes sur le MOSFET : la source, le drain et la gâchette.

Lorsqu’on applique une tension sur la gâchette, le MOSFET laisse passer le courant entre la source et le drain, sinon le circuit reste ouvert et le courant ne passe pas.

Sur la patte 1 du MOSFET on branche l’Arduino, sur la patte 2 on branche le moteur et sur la patte 3 on branche la GND.

Faire tourner le moteur 5s puis 3s repos :



**Etape 2 :** piloter le moteur dans les deux sens et toujours à vitesse constante

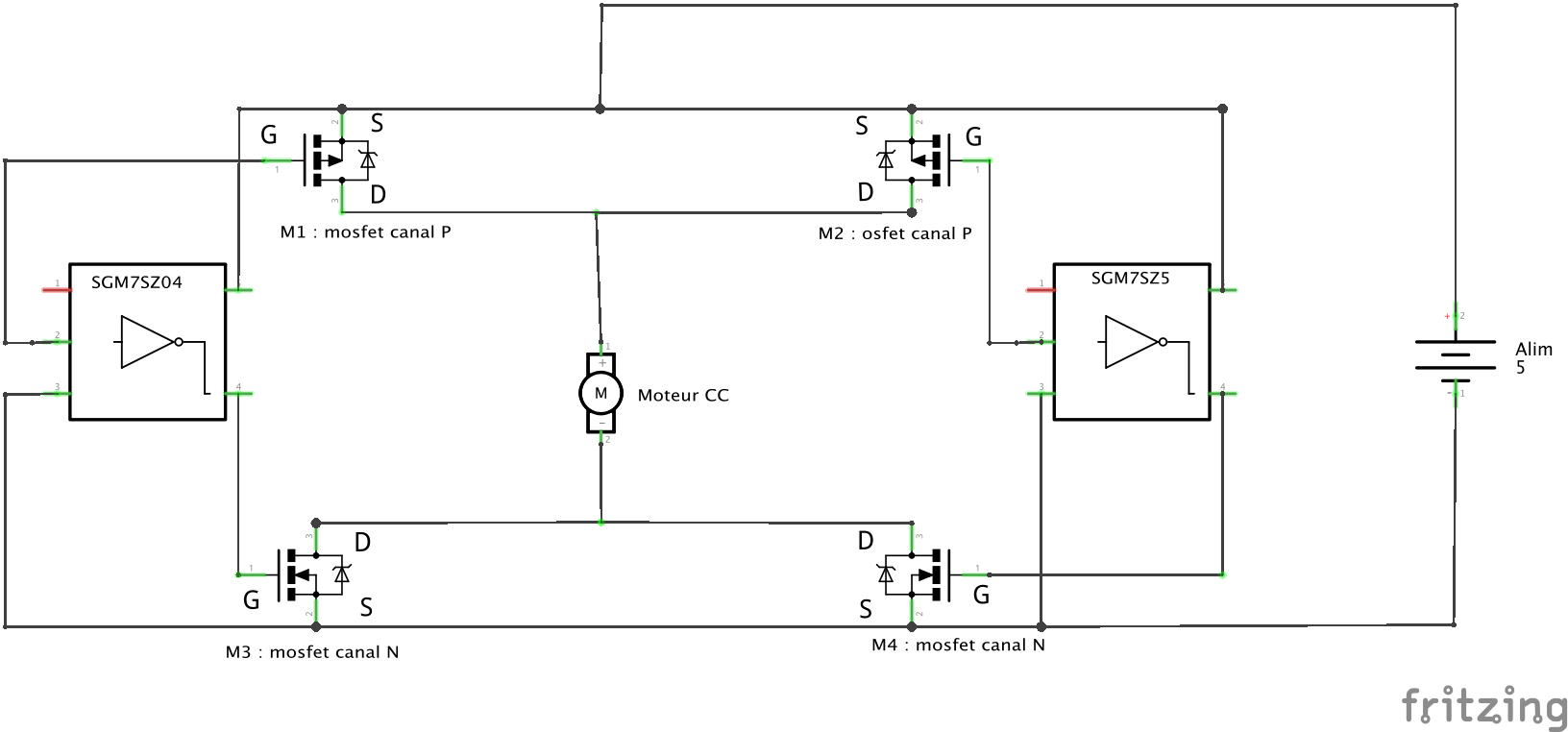
-utilisation pont en h

Rôle pont en H:

inverser la polarité aux bornes du moteur.

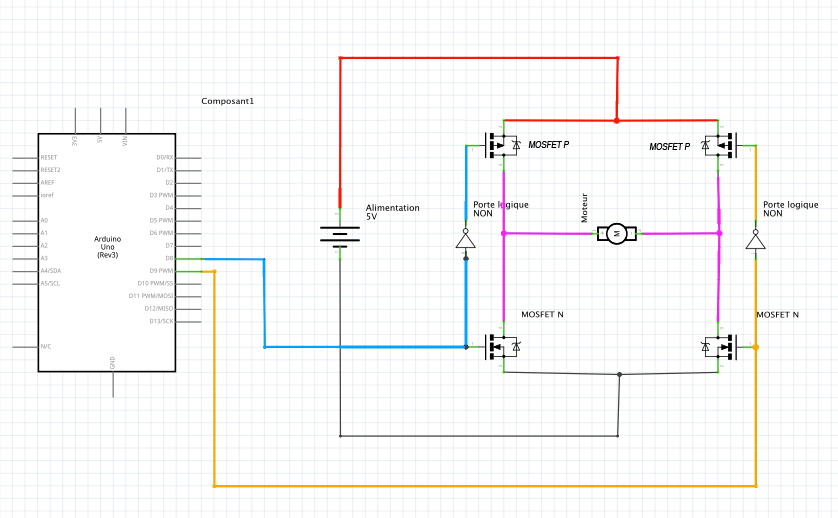
Création du pont en H :

6 fils :2 fils sur les pôles + et - de l’alimentation,2 fils sur les bornes du moteur (à la sortie des MOSFET),2 fils pour donner les ordres de tourner dans un sens ou dans l’autre ou de s’arrêter.



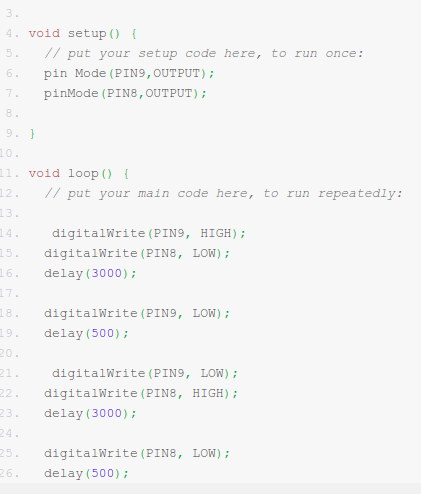
4 MOSFET et 2 portes NON

Schéma global :



Fonctionnement : Lorsque l’on veut faire tourner le moteur dans un sens, on envoie un signal "HAUT" sur la pin 8 de l’Arduino et on envoie un signal "BAS" sur la pin 9. Pour changer de sens, on inverse les signaux : "BAS" sur la pin 8 et "HAUT" sur la pin 9. Pour arrêter le moteur on envoie des signaux identiques ("HAUT "et "HAUT" ou "BAS" et "BAS".

Code :



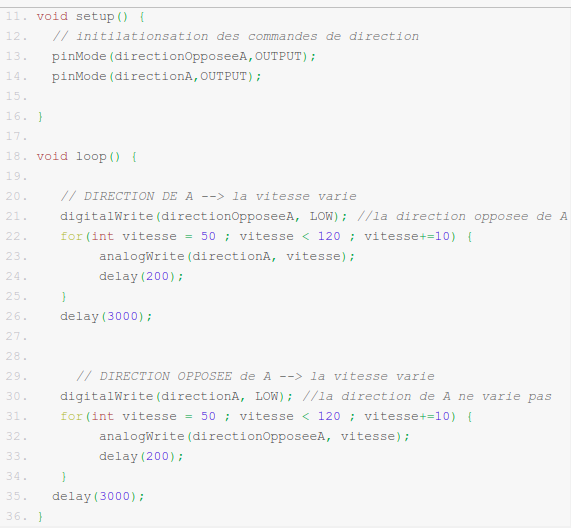
Etape 3 : piloter le moteur en variant la vitesse

-Pour faire varier la vitesse d’un moteur à courant continu, il suffit de faire varier la tension aux bornes du moteur.

Solution :

le PWM

Code:



Etape 4 : télécommander le moteur

???

Etape 5 : télécommander la voiture

-utiliser un module RF24 dans la télécommande et un module RF24 dans la voiture.(=>utilise les fréquences radio 2,4 GHz)

-télécommande:

4 boutons poussoir, une LED,carte arduino (nano),pile de 9v

-voiture: pas d’info

-Série ARDUINO : Comment créer sa VOITURE RC Source Chaine Youtube iMake-Tuto :

[ARDUINO : Comment créer sa VOITURE RC - Le Servo Moteur (Part.1)](https://www.youtube.com/watch?v=uCCrLPmaa_w)

Tuto pour construire une voiture télécommandé (modèle ancienne voiture):

-servomoteur permet de diriger les roues (moteur qui fait tourner seulement à 180°) composé de 3 fils (masse, 5V, data ⇒ pour commander, envoie un signal qui donne l’angulation du moteur)

On envoie un signal sous forme de créneau de période général T=20ms où on envoie les 5V pdt un certain et dans l’autre 0V (typiquement le montage PWM). Selon la notice du cerveau moteur, on doit envoyer les 5V entre 1ms et 2ms (dans le cas de la vidéo) pour le mettre soit à 0° soit à 180°.

Suite de la vidéo :

* Montage du servomoteur avec le potentiomètre



* Partie code

Le maping permet de convertir les valeurs analogiques entre 0 et 1023 en valeurs d’angle donc entre 0 et 179.



Cette partie sert donc à faire tourner la voiture

Partie 2:

https://www.youtube.com/watch?v=O7kpAxSoF6I

Faire avancer les roues arrière.

-Contrôler un moteur à courant continu à l'aide d'Arduino

-Comprendre le fonctionnement du L298N

-Utiliser le module L298N pour contrôler le moteur

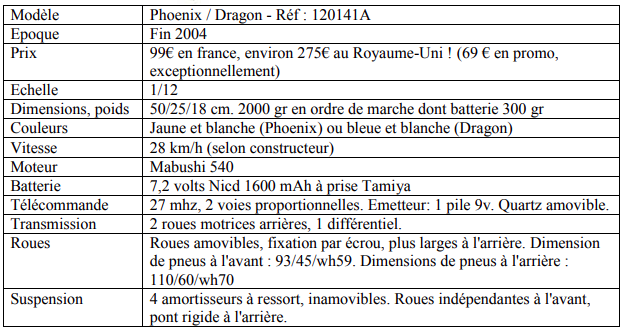
Partie 3:

Faire le module radio pour contrôler la voiture à distance

-Arduicar, une voiture commandée par Arduino Source Framboise314 :

[Arduicar, une voiture commandée par Arduino](https://www.framboise314.fr/arduicar-une-voiture-commandee-par-arduino/)

Voiture :



Téléommande:

une platine Arduino Duemillanove

Bouton de commande:

Le joystick est équipé de deux potentiomètres et d’un contact. La carte offre deux poussoirs

supplémentaires qui pourront servir pour des accessoires . De plus ce joystick est

prévu pour accueillir une carte UHF APC220 d’une portée en extérieur de 1000 mètres.

Liaison Radio :

Xbee, ZigBee,Walkthrough, APC220…

Pour des raisons de coût (il faut un shield Xbee + un module XBee de chaque côté !) et de facilité de mise en œuvre (le port APC220 est prévu sur le joystick) j’ai choisi les modules APC220 (36€).

J’ai également un afficheur LCD 2 lignes de 16 caractères, équipé de poussoirs, qui pourra servir plus tard à agrémenter la façade de la télécommande et afficher des infos en retour de la voiture (tension batterie, direction relevée par un capteur magnétique…). Un pack de piles ou une batterie fournira l’alimentation à l’ensemble.

**Voiture :**

Arduino Uno qui pilotera la voiture

-Liaison Radio :

une carte APC220

-La commande moteur:

Le moteur Mabushi 540 qui équipe la voiture (alimentée par une batterie 7,2v d’origine)

consomme environ 2A à vide, et une dizaine en charge =>circuitL298

-Servo de direction:

En fonction de l’état du servo de direction intégré et de la réutilisation possible de l’électronique,

j’utiliserai le servo d’origine ou je le remplacerai par un autre servo plus moderne

**Les étapes à réaliser:**

**-Sur la télécommande:**

Définir le codage des infos envoyées par la télécommande

Connecter le joystick sur Arduino, sans liaison radio

Tests de fonctionnement du joystick et des poussoirs, sortie sur la RS232

Configuration des cartes APC 220, tests entre un PC et un Arduino

Code de test sur le site :

Les premiers essais avec le câble USB sont concluants.

Les essais avec les APC220 le sont un peu moins. Après configuration des cartes en 9600 b/s pour la

liaison série côté PC et Arduino, et configuration de la liaison radio en 2400 b/s, les données provenant de la carte Arduino : position des 2 potentiomètres et état des 3 boutons (joystick, rouge

et bleu) arrivent correctement sur le PC. Cependant, la commande de LED depuis le PC (1=allumée,2=éteinte) ne parvient pas à l’Arduino…Après essais, il faut passer en 19200 b/s et tout fonctionne. (saturation de la liaison si elle est établie à une vitesse trop faible ?, ou charge de la liaison par les infos qui seraient échangées entre les cartes : NET ID et NODE ID… en plus des données utiles)

**Sur la voiture :**

Tester la faisabilité du pilotage du servo Nikko 6 fils

En cas d’impossibilité : montage d’un servo classique et tests de pilotage droite/gauche

Tester la commande moteur traction par le L298, accélération avant/arrière

Tester le freinage (qui sera fait par appui sur le poussoir du joystick)

Commander la voiture depuis le PC à travers les APC220

**Sur l’ensemble :**

Test de la liaison et du codage

Portée des APC 220

Fiabilité de la liaison

Blocage de la voiture en cas de perte de réception des données

Ajouter les commandes auxiliaires (feux, sirène ?)

-Voiture télécommandée en bluetooth par son smartphone Source Wikifab:

[Voiture télecommandée en bluetooth par son smartphone](https://wikifab.org/wiki/Voiture_t%C3%A9lecommand%C3%A9e_en_bluetooth_par_son_smartphone)

### Matériaux:

Contreplaqué, 1 carte Arduino, 2 moteurs 5V cc (modélisme), 1 driver pour moteur à courant continu, une alim de 9V, un bluetooth HC05 ou HC06, 2 roues, une roue libre, câbles arduino femelles, pins, une mini-breadboard

Outil:

Découpe Laser, Ordinateur avec Arduino, App Inventor 2 en ligne, Fer à souder, Tournevis, Perceuse

Étape 1 - Appli pour commander à distance:

créer notre propre appli qu'on installera directement sur notre téléphone. On va utiliser App Inventor 2 qui nécessite un compte gmail pour se connecter et un smartphone avec Android pour fonctionner. (A noter l'équivalent gratuit [Thunkable](https://thunkable.com/#/) pour Android et IOS, ne nécessitant pas gmail).

## Étape 2 - Code Arduino

bibliothèque SoftwareSerial.h pour communiquer avec le bluetooth

La vitesse sera modulée par un signal analogique (AnalogWrite) en PWM comprise entre 0 et 255.

Voir code sur le site directement

## Étape 3 - Châssis design et câblage

Dessinés sous Inkscape pour être découpés à la laser principalement, ils devaient pouvoir recevoir les différents éléments pour être facilement câblés par la suite.

Pensez à mettre les connecteurs d'alimentation orientés vers l'extérieur pour ne pas être gêné par la suite. Quelques soudures sur les fils moteurs et les pins du driver pour les maintenir.

L'alimentation a été fixée sous les châssis. Une roue libre fixée à l'avant permet un contrôle à droite ou à gauche en bloquant un moteur du côté opposé.

-Voiture commandée par arduino Source Hatlab :

[Voiture commandée par arduino — HATLAB](https://wikifab.hatlab.fr/Voiture_command%C3%A9e_par_arduino)

Etape 1 : Matériel

Rien

Etape 2:le chassis

Kit est au moins composé de :2 plaques plexi et des entretoises pour assembler un châssis;4 roues associées à 4 moteurs à courant continu;une carte Arduino uno;quelques fils permettant de connecter les entrées/sorties de l'Arduino à différents capteurs.

Donc on a déjà un châssis et des roues, on peut donc se focaliser désormais sur la partie électronique.

Etape 3 : Éléments électroniques

Afin de piloter les moteurs de la voiture, je vais utiliser :les moteurs;2 drivers moteurs;une carte Arduino Mega;un module d'alimentation;des batteries;un joystick

Comme tout projet dans lequel on se lance sans savoir, il est important d'y aller pas à pas …

Etape 5: Drivers moteurs

Donc on retient quelques éléments clés :1 moteur necessite un driver moteur qui permet d'amplifier le courant que la carte Arduino n'est pas capable de fournir.Ce driver doit recevoir des “combinaisons” de signaux pour faire avancer, reculer, stopper ou laisser aller en roue libre chaque moteur.

Driver utiliser lien

Etape 7 : Un module d'alimentation

on définit l'alimentation en dernier

Pour moins de 2€, vous recevez une alimentation à découpage qui permet de fournir une tension de sortie de mémoire jusqu'à 3A et, en prime, un afficheur permettant d'afficher soit la tension d'entrée, soit la tension de sortie, c'est assez confortable, en particulier pour la mise au point.

Etape 8 : batterie

J'ai opté pour la techno Li-ion qui a le mérite de présenter une tension par élément de 3.6V

Attention en terme de qualité et de prix, il y a de tout et de n'importe quoi.

Etape 9: Partie télécommande

un emetteur et un recepteur =>un ensemble emetteur/recepteur RF à 433MHz

Etape 10: Telecommande

La télécommande est composée d'une carte Arduino Uno, à laquelle est connecté un joystick permettant de faire tourner, avancer ou reculer la voiture.

On y trouve aussi l'emetteur RF à 433 MHz de maniere à “envoyer” à la voiture, la position du joystick et l'etat du push du Joystick (appuyé ou relaché).

-Rapport\_projet\_voiture\_base\_arduino Source Electronique mixte/Polytech Tours :

[Développement d'une voiture télécommandée à base d'Arduino](https://www.electronique-mixte.fr/wp-content/uploads/2018/06/Formation_Programmation_Arduino_cours_9.pdf)

Composants utilisés:

-Batterie : Li-Po est un élément dans lequel l'électrolyte est un polymère sous forme gélifiée. Elle se rapproche par sa forme et ses caractéristiques des batteries Li-on. Un élément Li-Po délivre 3,7 V. Livrés en 1, 2, 3, 4, 6 éléments voir plus, ils délivrent donc 3.7V, 7.4V, 11.1V, 14.8V, 22.2V. Souvent équipés d'une prise de charge spéciale, ils peuvent alors être chargés élément par élément à l'aide de chargeurs appropriés. [Source Aerowiki] Ce type d’accumulateur est très pratique par rapport aux autres accumulateurs existants, puisqu’il n’est pas sensible à l’effet mémoire (attendre la fin de la batterie pour la recharger) et son nombre de recharge est bien plus important que ses concurrents (environ 1000 charges contre 500 pour d’autres).

-Module: faire attention à la taille que l’on va donner au module (pas trop gros) pour qu’il sois facile à transporter.

-Moteur:

-Boutons poussoirs: permettant d’arrêter le véhicule s’il entre en collision avec qqch.

Reste des composants ne semble pas adapter à notre projet : shield arduino uno WIFI, carte Romeo

Pour la communication ils utilisent le WIFI mais le bluetooth semble plus adapter pour nous car:

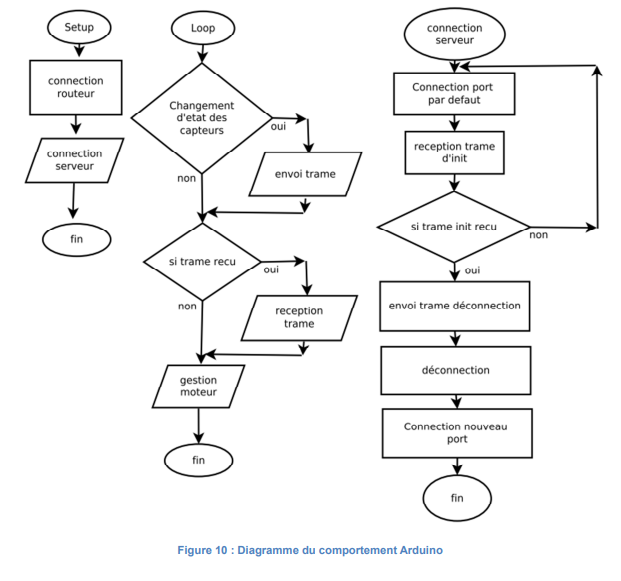
-Consomme moins que le Wifi

Peu encombrant (cet avantage n’est pas pris en compte avec un shield Arduino)

-Meilleur débit

-Module moins chère que le Wifi

Or ils favorisent le wifi car code une partie du programme en C# et veulent faire communiquer des voitures entre elles



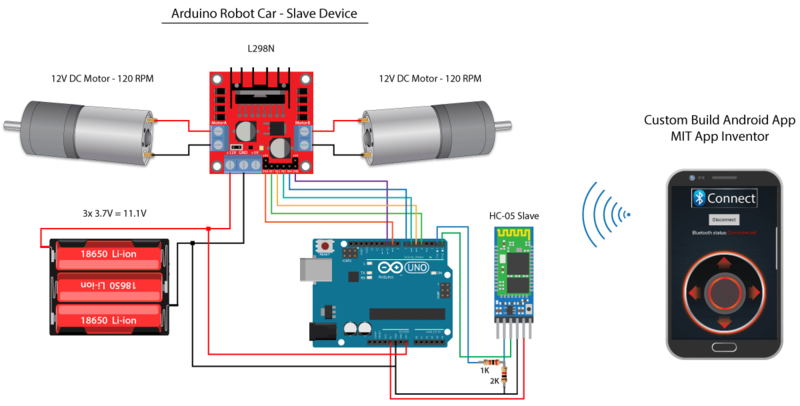
-Voiture télécommandée par bluetooth Source Les fabriques du Ponant :

[Voiture télécommandée par bluetooth](https://www.wiki.lesfabriquesduponant.net/index.php?title=Voiture_t%C3%A9l%C3%A9command%C3%A9e_par_bluetooth)

Matériel :

module Bluetooth HC-06; Variateur de vitesse L298N; 2x moteurs; Arduino Uno; 6x piles 1.5 V; Câbles

Schéma Cablage :



Code :

#define enA 9

#define in1 4

#define in2 5

#define enB 10

#define in3 6

#define in4 7

int xAxis, yAxis;

int x = 0;

int y = 0;

int motorSpeedA = 0;

int motorSpeedB = 0;

void setup() {

pinMode(enA, OUTPUT);

pinMode(enB, OUTPUT);

pinMode(in1, OUTPUT);

pinMode(in2, OUTPUT);

pinMode(in3, OUTPUT);

pinMode(in4, OUTPUT);

Serial.begin(38400); // Default communication rate of the Bluetooth module

}

void loop() {

// Default value - no movement when the Joystick stays in the center

xAxis = 510;

yAxis = 510;

// Read the incoming data from the Smartphone Android App

while (Serial.available() >= 2) {

x = Serial.read();

delay(10);

y = Serial.read();

}

delay(10);

// Makes sure we receive corrent values

if (x > 60 & x < 220) {

xAxis = map(x, 220, 60, 1023, 0); // Convert the smartphone X and Y values to 0 - 1023 range, suitable motor for the motor control code below

}

if (y > 60 & y < 220) {

yAxis = map(y, 220, 60, 0, 1023);

}

// Y-axis used for forward and backward control

if (yAxis < 470) {

// Set Motor A backward

digitalWrite(in1, HIGH);

digitalWrite(in2, LOW);

// Set Motor B backward

digitalWrite(in3, HIGH);

digitalWrite(in4, LOW);

// Convert the declining Y-axis readings for going backward from 470 to 0 into 0 to 255 value for the PWM signal for increasing the motor speed

motorSpeedA = map(yAxis, 470, 0, 0, 255);

motorSpeedB = map(yAxis, 470, 0, 0, 255);

}

else if (yAxis > 550) {

// Set Motor A forward

digitalWrite(in1, LOW);

digitalWrite(in2, HIGH);

// Set Motor B forward

digitalWrite(in3, LOW);

digitalWrite(in4, HIGH);

// Convert the increasing Y-axis readings for going forward from 550 to 1023 into 0 to 255 value for the PWM signal for increasing the motor speed

motorSpeedA = map(yAxis, 550, 1023, 0, 255);

motorSpeedB = map(yAxis, 550, 1023, 0, 255);

}

// If joystick stays in middle the motors are not moving

else {

motorSpeedA = 0;

motorSpeedB = 0;

}

// X-axis used for left and right control

if (xAxis < 470) {

// Convert the declining X-axis readings from 470 to 0 into increasing 0 to 255 value

int xMapped = map(xAxis, 470, 0, 0, 255);

// Move to left - decrease left motor speed, increase right motor speed

motorSpeedA = motorSpeedA - xMapped;

motorSpeedB = motorSpeedB + xMapped;

// Confine the range from 0 to 255

if (motorSpeedA < 0) {

motorSpeedA = 0;

}

if (motorSpeedB > 255) {

motorSpeedB = 255;

}

}

if (xAxis > 550) {

// Convert the increasing X-axis readings from 550 to 1023 into 0 to 255 value

int xMapped = map(xAxis, 550, 1023, 0, 255);

// Move right - decrease right motor speed, increase left motor speed

motorSpeedA = motorSpeedA + xMapped;

motorSpeedB = motorSpeedB - xMapped;

// Confine the range from 0 to 255

if (motorSpeedA > 255) {

motorSpeedA = 255;

}

if (motorSpeedB < 0) {

motorSpeedB = 0;

}

}

// Prevent buzzing at low speeds (Adjust according to your motors. My motors couldn't start moving if PWM value was below value of 70)

if (motorSpeedA < 70) {

motorSpeedA = 0;

}

if (motorSpeedB < 70) {

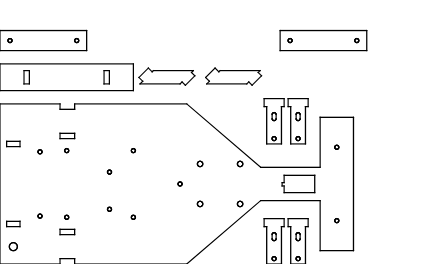
motorSpeedB = 0;

}

analogWrite(enA, motorSpeedA); // Send PWM signal to motor A

analogWrite(enB, motorSpeedB); // Send PWM signal to motor B

Modélisation du châssis :



-Recycler une voiture télécommandée (RC Car) avec un ESP8266, Shield Motor Wemos d1 mini et Blynk Source Projetsdiy :

[Recycler une voiture télécommandée (RC Car) avec un ESP8266, Shield Motor Wemos d1 mini et Blynk • Domotique et objets connectés à faire soi-même](https://projetsdiy.fr/recycler-rc-car-voiture-telecommandee-esp8266-shield-motor-wemos-d1-mini-blynk/)

Matériel :

Pour ce projet, je vous propose d’utiliser le shield Motor pour Wemos d1 mini. Elle vient s’empiler sur l’ESP8266 ce qui permet d’obtenir un montage très compact. Ls moteurs seront alimentés à l’aide de l’ancien bloc de piles. La Wemos sera alimentée à l’aide d’un batterie LiPo. Ici, j’ai utilisé une batterie de 1100mAh. Je vous conseille d’utiliser le shield Battery (présenté ici) ou utiliser une Power Bank.

Pourquoi ne pas utiliser un pont en H L293D ?

Effectivement, si vous ne voulez pas investir dans un shield Motor, il est tout à fait possible d’utiliser un pont en H, par exemple le L293D. Le prix de revient est équivalent. Par contre au niveau de la mise en œuvre, ça n’a rien à voir !

Préparation de la voiture :

Pour connecter les anciens câbles au shield Motor, vous pouvez utiliser un kit de connecteur Dupont. Voici le montage terminé.

Projet Blynk:

Facile a voir sur le lien explication

**Codage Arduino :**

Bibliothèques:

#include <BlynkSimpleEsp8266.h>

#include <ESP8266WiFi.h>

#include "WEMOS\_Motor.h"

Variables:

// Vitesse exprimée en % de la puissance - Speed depend % of the power

int \_speed = 25; // %

int STATUS\_BLYNK = D4; &nbsp;// Etat connexion serveur Blynk - Blynk connexion status

// Vos paramètres de connexion WiFi - Your WiFi credentials.

char ssid[] = "yourSSID";char pass[] = "yourPassword";

// Jeton Blynk - Blynk Token

char auth[] = "b065eb0a6e36434da42367b3fa7c3340";

Création des deux objets:

Motor M1(0x30,\_MOTOR\_A, 1000);//Motor A - Direction

Motor M2(0x30,\_MOTOR\_B, 1000);//Motor B - Propulsion

Setup :

void setup() {

Serial.begin(115200);

// Initialise la Led indiquant l'état de connexion au serveur Blynk - Init status Led inidcating Blynk connexion status

pinMode(STATUS\_BLYNK, OUTPUT);

digitalWrite(STATUS\_BLYNK, LOW);

// Authentification auprès du serveur local Blynk - Authenticate to Blynk Private Server

Serial.println("Connecting Blynk");

Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,24), 8442);

}

Loop:

void loop() {

Blynk.run();

// Actualise la Led indiquant l'état de connexion au serveur Blynk - Update Blynk led Status

if (Blynk.connected()){

digitalWrite(STATUS\_BLYNK,HIGH);

} else {

digitalWrite(STATUS\_BLYNK, LOW);

}

}

Fonction:

// Avance - Drive

BLYNK\_WRITE(V1) {

if (param.asInt()) {

Serial.println("D");

M2.setmotor(\_CW, \_speed);

} else {

M2.setmotor(\_STOP);

}

}

// Recule - Rear

BLYNK\_WRITE(V2) {

if (param.asInt()) {

Serial.println("R");

M2.setmotor(\_CCW, \_speed / 2);

} else {

M2.setmotor(\_STOP);

}

}

// Droite - Right

BLYNK\_WRITE(V3){

if (param.asInt()) {

Serial.println(">>");

M1.setmotor(\_CW,100);

} else {

M1.setmotor(\_STOP);

}

}

// Gauche - left

BLYNK\_WRITE(V4){

if (param.asInt()) {

Serial.println("<<");

M1.setmotor(\_CCW,100);

} else {

M1.setmotor(\_STOP);

}

}

-Contrôle d’une voiture RC avec Arduino Uno Source Blog Emmanuel Surleau :

[Contrôle d'une voiture RC avec Arduino Uno](http://blog.emmanuelsurleau.fr/2013/03/controle-dune-voiture-rc-avec-arduino-1x/)

Matériel :

Servomoteur S1903MG Associated Electronics;Variateur LRP A.I. Super Reverse Digital;Moteur Reedy Radon 17T;Batterie Team Orion SUPER DUTY PACK 3300 NI-MH (7.2V)

**Contrôle du servomoteur et du variateur :**

Circuit électrique:

Le + (ici Rouge) doit être relié à la borne 5V de la carte

Le – ou masse (ici Noir) doit être relié à la borne 0V (ou GND)

Le fil de l’impulsion de commande (ici Blanc) doit être relié à une broche numérique de la carte

Servomoteur de direction :

Fil Rouge : Borne 5V Arduino

Fil Noir : Borne GND Arduino

Fil Blanc : Borne Digital n°7

Variateur du moteur :

Fil Rouge : Borne Vin Arduino

Fil Noir : Borne GND Arduino

Fil Blanc : Borne Digital n°8

-La borne Vin est utilisé pour l’alimentation du variateur du moteur car celui-ci fournit du courant via la batterie de la voiture pour alimenter la carte Arduino Uno.

**Programmation:**

#include <Servo.h>

//Déclaration des servo et des variables

Servo mymotor;

Servo myservo;

int angle;

void setup()

{

//Servo branché sur la borne digitale 7

myservo.attach(7);

//Variateur branché sur la borne digitale 8

mymotor.attach(8);

//Initiation d'une connexion série 9600bauds

Serial.begin(9600);

//Initialisation de la variable angle à 90

angle = 90;

}

void loop()

{

//Si des données sont reçues sur la connexion série

if(Serial.available() > 0)

{

//Si octet reçu = 82 (donc le caractère R)

if(Serial.read() == 82)

{

//Calcul de ciffre de centaine

int r1 = 0;

switch((int)Serial.read())

{

case 48:

r1 = 0;

break;

case 49:

r1 = 1;

break;

case 50:

r1 = 2;

break;

case 51:

r1 = 3;

break;

case 52:

r1 = 4;

break;

case 53:

r1 = 5;

break;

case 54:

r1 = 6;

break;

case 55:

r1 = 7;

break;

case 56:

r1 = 8;

break;

case 57:

r1 = 9;

break;

default :

r1 = 0;

break;

}

//Calcul du chiffre de dizaine

int r2 = 0;

switch((int)Serial.read())

{

case 48:

r2 = 0;

break;

case 49:

r2 = 1;

break;

case 50:

r2 = 2;

break;

case 51:

r2 = 3;

break;

case 52:

r2 = 4;

break;

case 53:

r2 = 5;

break;

case 54:

r2 = 6;

break;

case 55:

r2 = 7;

break;

case 56:

r2 = 8;

break;

case 57:

r2 = 9;

break;

default :

r2 = 0;

break;

}

//Calcul du chiffre des unités

int r3 = 0;

switch((int)Serial.read())

{

case 48:

r3 = 0;

break;

case 49:

r3 = 1;

break;

case 50:

r3 = 2;

break;

case 51:

r3 = 3;

break;

case 52:

r3 = 4;

break;

case 53:

r3 = 5;

break;

case 54:

r3 = 6;

break;

case 55:

r3 = 7;

break;

case 56:

r3 = 8;

break;

case 57:

r3 = 9;

break;

default :

r3 = 0;

break;

}

//Calcul du nombre complet

angle = ((int)r1 \* 100) + ((int)r2 \* 10) + (int)r3;

Serial.flush();

}

}

else

{

Serial.flush();

Serial.println("Wait : Flush !");

}

//Si angle entre 70 et 104, controle du variateur de moteur

if(angle > 70 & angle < 104)

{

Serial.print("Moteur :");

Serial.println(angle);

mymotor.write(angle);

delay(15);

}

//Si angle entre 200 et 380, controle du servomoteur

else if(angle > 200 & angle < 380)

{

Serial.print("Servo :");

Serial.println(angle-200);

myservo.write(angle-200);

delay(15);

}

else

{

Serial.flush();

Serial.println("Error value !");

}

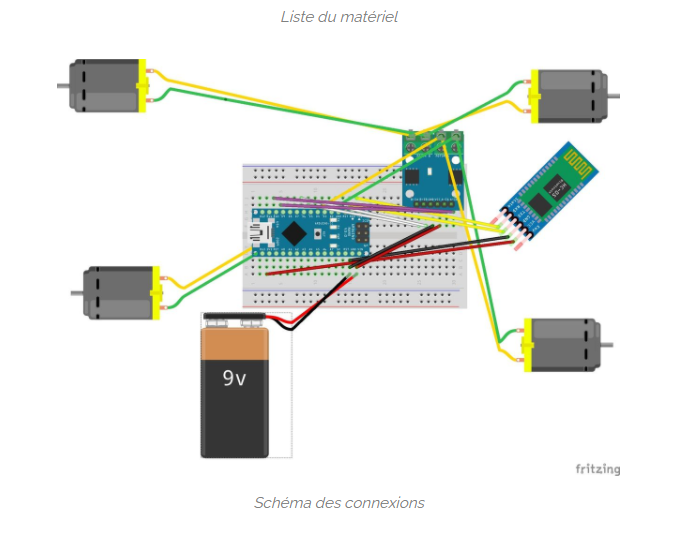
}

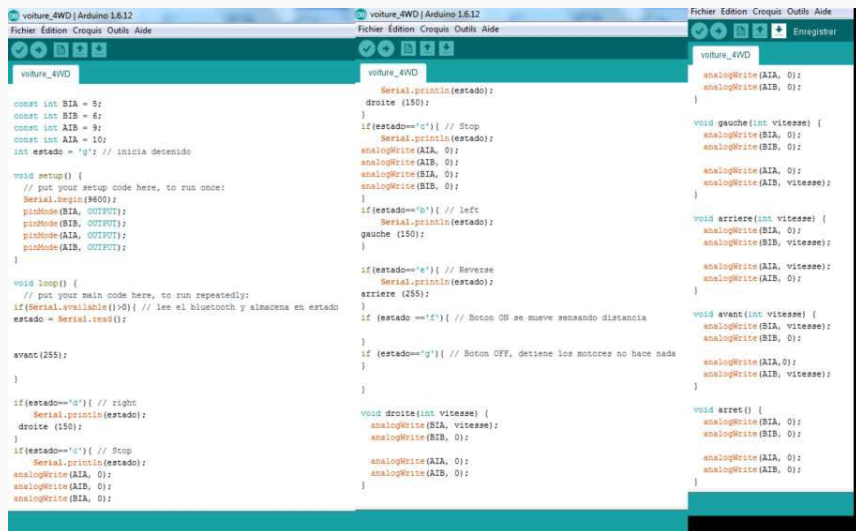
Commande via WiFi:

une carte Raspberry Pi a été utilisée pour héberger un serveur Web Node-js qui permet d’envoyer les instructions de commandes à la carte Arduino.

-Voiture télécommandée par Bluetooth Source diylisec :

[Voiture télécommandée par Bluetooth - LISEC Yourself](http://diylisec.over-blog.com/2017/03/demonstration-liste-du-materiel-schema-des-connexions-code-voici-les-connexions-precises-de-la-voiture-module-bluetooth-arduino-rxd)





-Projet Arduino : voiture télécommandée Source Futura Sciences :

[[Programmation] Projet Arduino : voiture télécommandée](https://forums.futura-sciences.com/electronique/790151-projet-arduino-voiture-telecommandee.html)

Voir pour les problèmes

-Voiture télécommandée, arduino avec connecteur bluetooth, tablette Android comme télécommande Source Chambeyron :

[Voiture télécommandée, arduino avec connecteur bluetooth, tablette Android comme télécommande](https://www.chambeyron.fr/index.php/arduino-raspberry-py/79-voiture-telecommandee-arduino-avec-connecteur-bluetooth-tablette-android-comme-telecommande)

Juste le code