Tomas Kvapil - tkvapi2, tkvapi2@uic.edu
Damian Ortiz - dortiz36, dortiz36@uic.edu
Javier Rojas - jrojas39, jrojas39@uic.edu

Special Operations Tactical Tank

Abstract

A fully mobile tank that has the capability to snap a picture at the push of a button. The tank is controlled by an arduino remote that sends signals to the tank via wireless transceivers. The remote also has a button to talk to a raspberry pi through serial communication. The pi then takes a picture and directly uploads the image to our very own website.

Details Project Description

We built an operational tank that moves through space, forward, and to each side. In order for both motors to run with enough power we used a DC motor control. The tank also has an arduino and a raspberry pi mounted on it that communicate with each other. A second arduino will control the tank's motors and also control a camera mounted on the tank. It will also have a 2.4GHz transceiver mounted on it to receive signals from the remote arduino which also has a transceiver. The raspberry pi will be the one that has the camera mounted on it that will take pictures at any given time. This will happen with an arduino controller that sends signals via the transceiver to the other arduino and that arduino will send signal to the raspberry pi through USB connection via serial communication. Both raspberry pi and arduino will be powered by a power bank also mounted on the tank. The second arduino will be used as a remote to communicate to the tank. It will have a button that will send a signal to the mounted arduino to take an image. It will also have 3 more buttons to go forward, go right, and go left. What we have observed that the tank does is exactly what was described. It can move forward, right, left and take pictures on command. Not only that but it also communicates with all parts without much of a problem. While we used 2.4GHz transceivers instead of bluetooth they basically work the same. These 2 transceivers are only used between the arduinos to communicate. Then the serial communication is used to communicate from tank arduino to the raspberry pi. The raspberry pi also is a wifi hotspot. So when a computer connects to it, it can go to website host where the images will be uploaded. This is basically a dump from the raspberry pi images into the raspberry pi /var/www file to correctly host the website.

Project Design

Input:

- A camera that will take pictures on command.
- Transceiver to control the movement of the tank.
- Raspberry pi serial communication

Output:

- Two motors that will move the vehicle forward, left, and right.
- Three buttons to control the motors on the vehicle movements.
- One button to take photos
- Two transceivers to send and receive signals between arduino.
- A light to notify that a picture was taken.
- USB cord
- Raspberry pi wifi hotspot

Communication Details

All micro-computers use communication one way or another. The main output of communication is the remote arduino that controls photo-taking and tank direction. The remote has a transceiver mount that will connect to the tank arduino transceiver mount located on the tank. The arduino on the tank is then connected to the raspberry pi through USB and they use serial communication. This is so that the arduino on the tank can send the signal to the pi for the raspberry pi to take a picture. Finally, the raspberry pi has a wifi hotspot. This is so that a remote computer could connect and view the website created by the pi.

Timeline of Development

Week 8 - Have the motors working. Decided not to use actuators. We swapped out one of the arduino for a raspberry pi which will have the camera.
Week 9 - Decided that instead of wheel and four motors we use only two since it works better with the tracks. Had problems with the delivery of the camera we decided to use.
Week 10 - Made the 2 buttons work with one arduino. Sending the correct signals. Had two arduinos connected but we had problems with the connections of the arduino and raspberry pi.
Week 11 - Got raspberry pi mounted to the chassis of the frame we are going to use. Decided that the raspberry pi will be independent and just take video for now.
Week 12 - Had the camera finally connected to the raspberry pi and tested.
Week 13 - Send signals to one arduino to the other to make it move. Testing mostly but they work.
Week 14 - We have everything running and mounted on the tank. It moves on command and also takes pictures.
Week 15 - Tested and Presented

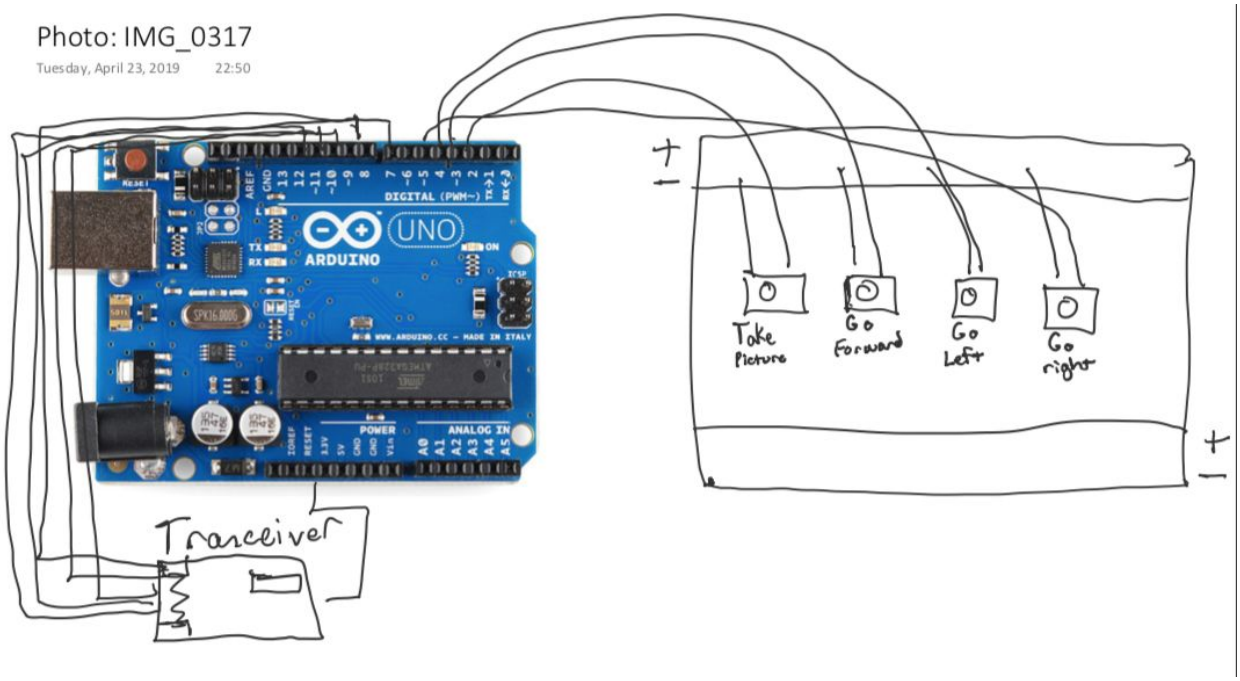
List of Materials

- 2 Arduinos and Raspberry Pi
- 2 motors
- 1 camera
- 4 buttons
- Continuous Track (tank tracks)
- 2 2.4GHz transceivers
- 1 base frame
- 1 LED light
- DC motor control
- 1 power bank

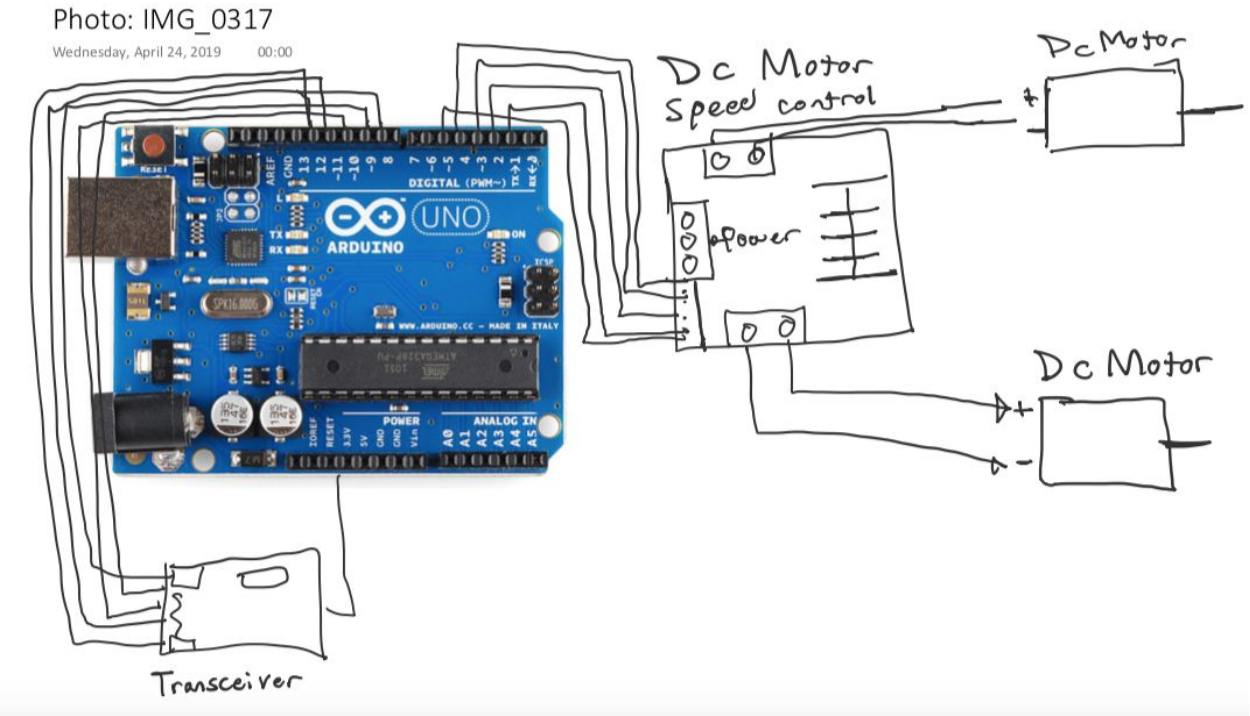
List of References

- **Arduino Bluetooth Library**
  https://www.arduino.cc/en/Main/ArduinoBoardBT?from=Main.ArduinoBoardBluetooth
- **Software Serial Library**
  https://www.arduino.cc/en/Reference/SoftwareSerial
- **Port Communication**
  https://www.arduino.cc/en/Tutorial/TwoPortReceive
- **Bluetooth Sketch**
  https://www.arduino.cc/en/Main/ArduinoBTInitializationSketch
- **Bluetooth Comm Signals**
  https://create.arduino.cc/projecthub/samanfern/bluetooth-controlled-car-d5d9ca

Remote Arduino setup



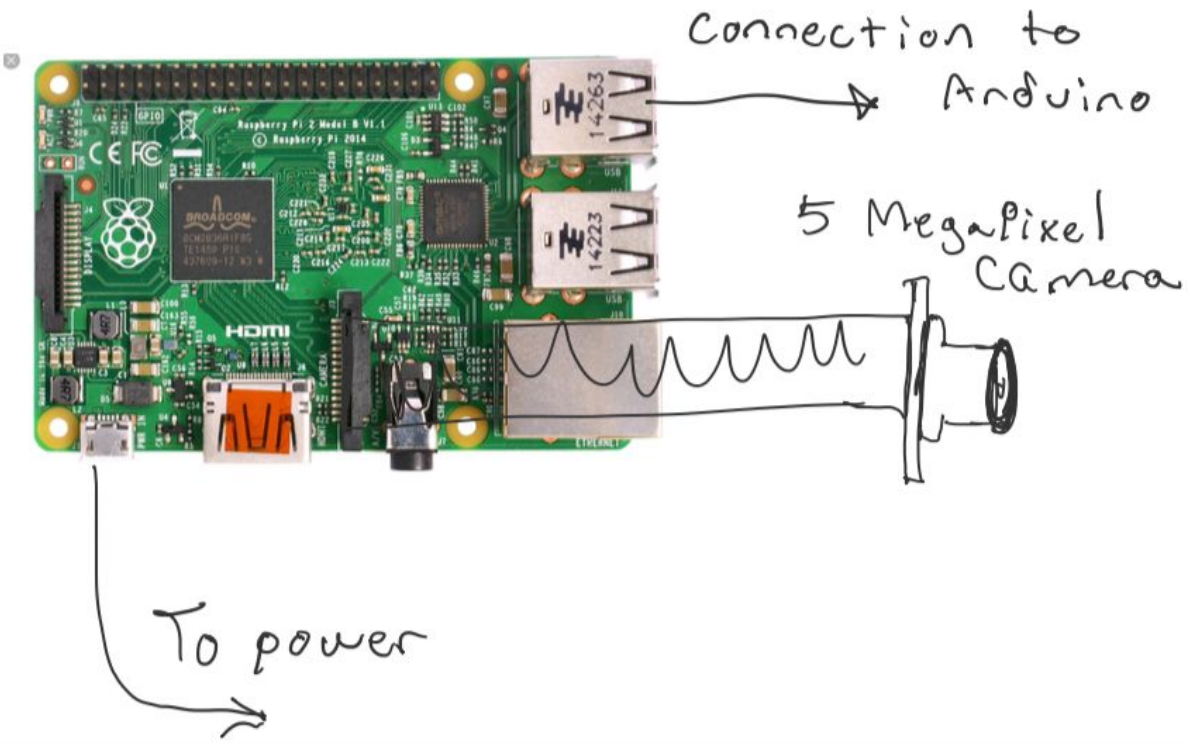
Tank Arduino Setup



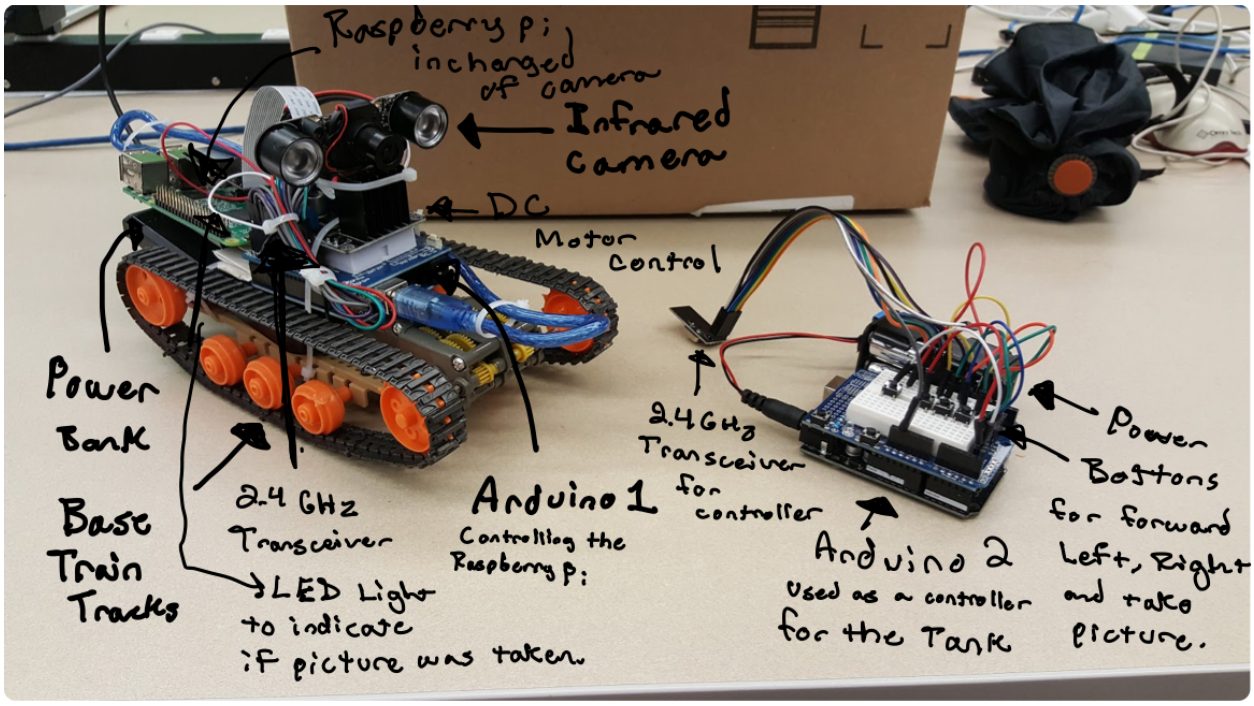
Raspberry Pi Setup

Photo: IMG_0318

Wednesday, April 24, 2019 00:13



Tank High-Level Diagram



Description of the original work

The overall objective of this project is to create a transceiver controlled vehicle that will be able to maneuver through different types of terrain, maneuver through tight spots, and will be able to take picture of the desired spots. The overall design of this project is original but it is inspired by RC Car spyware. However our architecture and use of a tank is fully original.

Remote Code

```
//libraries for transceiver
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

//Setup up transceiver and button pins
RF24 radio(7, 8);
byte switchPin[4] = { 2,3,4,5 };
const byte address[6] = "00011";
int pinVal;

void setup() {

  //Full transeiver setup
  radio.begin();
  radio.setRetries(15,15);
  radio.setPALevel(RF24_PA_MAX);
  radio.openWritingPipe(address);
  radio.stopListening();

  //Set up all 4 buttons as inputs
  for (byte i = 0; i < 4; i++){
    pinMode(switchPin[i], INPUT_PULLUP);
  }
}

void loop() {
  //Constantly checks the status of all the buttons
  for (byte i = 0; i < 4; i++) {
    pinVal = digitalRead(switchPin[i]);

    //if button pressed then send the pin number to tank
    if (pinVal == LOW) {
      sendRF(switchPin[i]);
    }
```

```
      }
    }
  }

//function that takes pin number and sends it to tank arduino
void sendRF(int data) {
    radio.write(&data, sizeof(data));
}
```

Tank Code

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

//control direction of vehivle
#define in1 2
#define in2 3
#define in3 4
#define in4 6

RF24 radio(7, 8); //declare transeiver

//declare pin numbers
const int motorPin = 5;
const int motorPin2 = 3;
const int ledPin = 10;

//declare address
const byte address[6] = "00011";

void setup() {
  Serial.begin(9600);
  //pins for output
  pinMode(motorPin, OUTPUT);
  pinMode(motorPin2, OUTPUT);
  pinMode(ledPin, OUTPUT);
  //transceiver setup
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MAX);
  radio.startListening();
  //motor ground and power set up
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,LOW);
```

```
  digitalWrite(in4,HIGH);
}

void loop() {
  digitalWrite(ledPin, LOW);
  if (radio.available()) {
    int data;
    radio.read(&data, sizeof(data));
    //if data received is 2 then go straight
    if (data == 2){
      analogWrite(motorPin, 200);
      analogWrite(motorPin2, 200);
    }
    //if data received is 3 then turn right
    if (data == 3){
      analogWrite(motorPin, 0);
      analogWrite(motorPin2, 200);
    }
    //if data received is 4 then turn left
    if(data == 4){
      analogWrite(motorPin, 200);
      analogWrite(motorPin2, 0);
    }
    //if data received is 5 then send "photo" for raspberry pi
    if (data == 5){
      Serial.println("photo");
      digitalWrite(ledPin, HIGH);
    }
    //reset components
    delay(150);
    analogWrite(motorPin, 0);
    analogWrite(motorPin2, 0);
    digitalWrite(ledPin, LOW);
  }
}
```

Raspberry Pi Code

```python
#import statements
import time
from picamera import PiCamera
import serial

#setup camera and serial communication
ser = serial.Serial('/dev/ttyACM0',9600);
camera = PiCamera()

#setup infinite loop to always listen for serial input
while 1:
    #if received information then take a photo
    if(ser.in_waiting > 0):
        line = ser.readline()
        print(line)
        image_path = '/home/pi/images/image_%s.jpg' % int(round(time.time()*1000))
        camera.capture(image_path)
        print('Took photo')
```