TEMA

“Volante para simulación con Arduino”

Materia: gestión de proyectos y software de calidad

Trabajo final

Carrera: Desarrollo de software.

Instituto: Alfredo Coviello.

Año: 2022

Profesor: Velárdez Rubén Emiliano

Alumnos: Santiago David Viva - David Olivera

Índice

Introducción

Este articulo detalla los elementos necesarios para la construcción y el funcionamiento de un volante y pedal de simulación en PC y consolas

Objetivo del proyecto

- Introducirnos en el mundo de la tecnología como lo son la robótica y la programación con Arduino
- Construir un volante de bajo costo, pero de gran calidad respecto a sus similares en el mercado.
- Despertar el interés en las personas por la robótica

Problemática

- Debido a los altos costos de dispositivos de simulación de calidad tanto para pc como consolas. Donde estos dispositivos más económicos que se encuentran actualmente en el mercado no llegan a cumplir con componentes que brinden una buena experiencia al usuario final. Decidimos abordar esta problemática realizando de manera casera un volante utilizando diferentes tipos de software y hardware.

Marco teórico

Descripción de los materiales utilizados para el proyecto

- Arduino IDE 1.8.19
- El entorno de desarrollo integrado (IDE) de Arduino es una aplicación multiplataforma (para Windows, macOS, Linux) que está escrita en el lenguaje de programación Java. Se utiliza para escribir y cargar programas en placas compatibles con Arduino, pero también, con la ayuda de núcleos de terceros, se puede usar con placas de desarrollo de otros proveedores.
- UnoJoy: UnoJoy es un pequeño software que te permite transformar una placa Arduino en un dispositivo de juego USB como un joystick, por ejemplo.

Componentes

Arduino UNO R3

Es una placa de desarrollo basada en el microcontrolador ATmega328P. Es una de las más conocidas y usadas de la extensa familia de placas Arduino, siendo una mejora de un diseño anterior (Arduino Duemilanove) que mantiene el 100% de compatibilidad y muestra algunos cambios significativos en su diseño

Otra diferencia importante entre Arduino y otras placas de microcontroladores es el costo. En 2006, otro microcontrolador popular, el Basic Stamp, costaba casi cuatro veces más ($ 119) que un Arduino ($ 32). Hoy, un Arduino Uno cuesta solo $ 22.

LCD 16x2 i2c

La pantalla LCD de 16×2 es un periférico muy común, que se utiliza ampliamente en proyectos con Arduino y microcontroladores en general, sin embargo, es bien sabido por todo aquel entusiasta que ha incluido una en sus proyectos, que este tipo de pantalla requiere muchos pines del microcontrolador debido a que utiliza un bus paralelo para comunicarse.

Afortunadamente existe una solución muy fácil y económica para este problema: un adaptador basado en el PCF8574 que permite conectar la pantalla al Arduino usando solamente dos líneas digitales a través del bus I2C. Dichos pines, pueden además ser compartidos por otros periféricos como el RTC o memorias EEPROM.

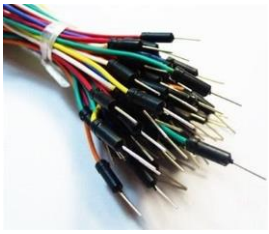
Encoder Rotativo Incremental 600 Pulsos Fotoeléctrico

El Encoder rotativo incremental fotoeléctrico se puede utilizar para medir la velocidad de rotación, el ángulo y la aceleración del objeto. También para la medición de la longitud.

Potenciómetro 10k

Este producto es un **potenciómetro** de un amplio uso debido a que permite fijarse en una tarjeta de circuito impreso así como en un panel o tablero de control de muchos equipos y proyectos electrónicos. El potenciómetro tiene una resistencia de variación lineal que va desde 0 Ohmios hasta **10K Ohmios**, la cual es controlada por el giro de un eje estriado. Su operación es muy sencilla y viene diseñado para que a pesar de su suavidad, no se gire solo o con alguna vibración.

Cableado



Un cable puente para prototipos (o simplemente puente para prototipos), es un cable con un conector en cada punta (o a veces sin ellos), que se usa normalmente para interconectar entre sí los componentes en una placa de pruebas. P.E.: se utilizan de forma general para transferir señales eléctricas de cualquier parte de la placa de prototipos a los pines de entrada/salida de un micro controlador. Los cables puente se fijan mediante la inserción de sus extremos en los agujeros previstos a tal efecto en las ranuras de la placa de pruebas, la cual debajo de su superficie tiene unas planchas interiores paralelas que conectan las ranuras en grupos de filas o columnas según la zona. Los conectores se insertan en la placa de prototipos, sin necesidad de soldar, en los agujeros que convengan para el conexionado del

4x18 5/16 varilla roscada

4 varillas que nos van a servir para armar la estructura del volante

Tuercas Hexagonal Hierro Zincado 5/16 Uss (8mm)

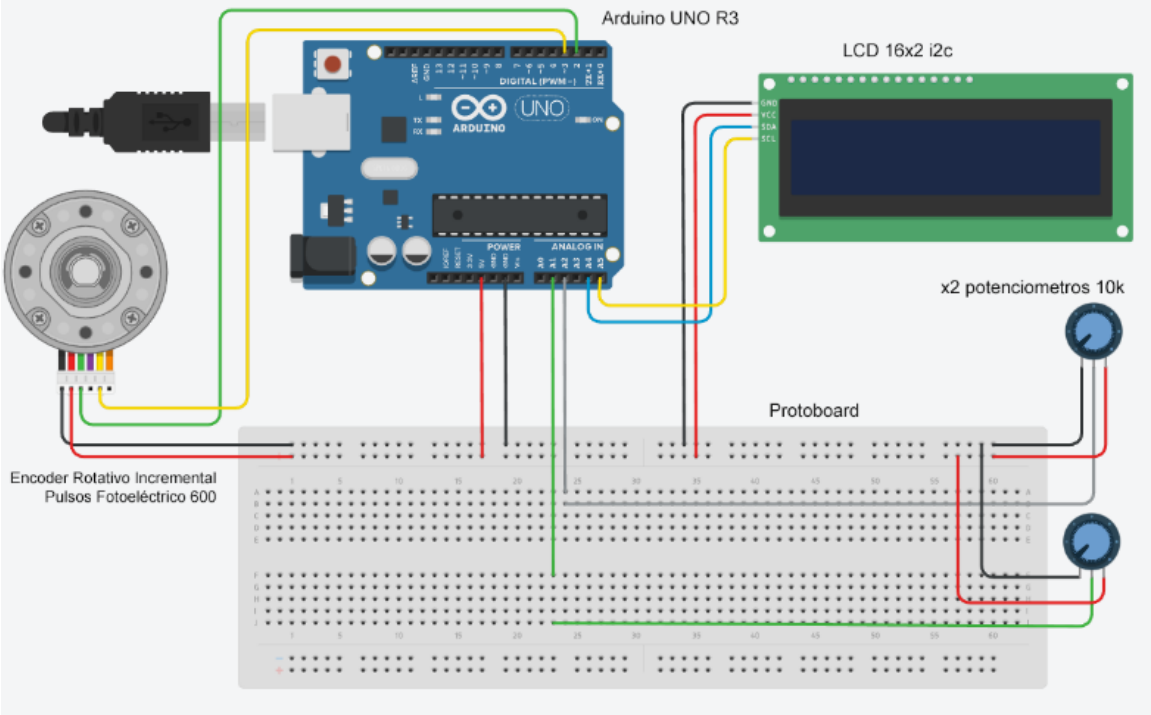
Tuercas que para unión de varillas y tablas

Planchas madera 15x30 2u

Planchas madera para estructura

Desarrollo

Se realiza el diagrama correspondiente para luego comenzar a trabajar en el código fuente y el armado:



Código fuente:

```
//Primero incluimos las librerias necesarias para el funcionamiento del proyecto.

#include "UnoJoy.h"
#include <math.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

//creación de un objeto con los parámetros indicados.
LiquidCrystal_I2C lcd(0x27,16,2);


//definicion 8 nuevos caracteres que nos serviran para representar la fuerza con la que aceleremos y frenemos en el vehiculo.

byte A[8] = {
B00000,
B00000,
B00000,
B00000,
B00000,
B00000,
B00000,
B11111,
};
byte B[8] = {
B00000,
B00000,
B00000,
B00000,
B00000,
B00000,
```

```
B11111,
B11111,
};
byte C[8] = {
B00000,
B00000,
B00000,
B00000,
B00000,
B11111,
B11111,
B11111,
};
byte D[8] = {
B00000,
B00000,
B00000,
B00000,
B11111,
B11111,
B11111,
B11111,
};
byte E[8] = {
B00000,
B00000,
B00000,
B11111,
B11111,
B11111,
B11111,
B11111,
};
byte F[8] = {
B00000,
B00000,
B11111,
B11111,
B11111,
B11111,
B11111,
B11111,
};

byte G[8] = {
B00000,
B11111,
B11111,
B11111,
B11111,
B11111,
B11111,
B11111,
};

byte H[8] = {
B11111,
B11111,
B11111,
B11111,
B11111,
```

```
B11111,
B11111,
B11111,
};

//Creamos dos variables que tomaran los datos de los pines A1 y A2, que son donde están
conectados nuestros acelerador y freno.
int pinA1 = A1;
int pinA2 = A2;

//Inicialización las variales para el acelerador, el  frenodo y el giro del volante. Es
muy importante asignarle un valor correcto ya que en el caso del acelerador y freno,
mostrarian un valor erroneo en el display, y en el caso del valante, este comenzaria
con un angulo de giro incorrecto.
int dataAcelerador = 0;
int dataFreno = 0;

//Nuestro Encoder tomara datos desde el punto 0 al 1000, por lo que el volante se
mantendrá centrado en el valor 500
volatile float counter = 500;


//funcion para crear caracteres con base a los diseños previamente realizados.
void crearCaracteres(){
 lcd.createChar (8,A);
 lcd.createChar (9,B);
 lcd.createChar (10,C);
 lcd.createChar (11,D);
 lcd.createChar (12,E);
 lcd.createChar (13,F);
 lcd.createChar (14,G);
 lcd.createChar (15,H);
}

//Función que generar los datos a mostrar para mostrar tanto del acelerador como el
frenado.
La variable data sera quien trae los datos provenientes tanto del acelerador como el
freno.
Fila le indicará al display donde se verán reflejados estos dotos. Uno para la fila
1(acelerador) y 2(freno) en el display.
void acelAndBrakes(int data, int fila){

  if (data <= 3){
    lcd.setCursor(8,fila);
    lcd.print(" ");
  }
  else if (data <=  125  && data >=4 ){
    lcd.setCursor(8,fila);
    lcd.write(byte (8));
    lcd.setCursor(9,fila);
    lcd.print(" ");

  }
  else if(data >=126 && data <=251){
    lcd.setCursor(9,fila);
    lcd.write(byte (9));
    lcd.setCursor(10,fila);
    lcd.print(" ");
  }
  else if(data >=252 && data <=377){
    lcd.setCursor(10,fila);
```

```
    lcd.write(byte (10));
    lcd.setCursor(11,fila);
    lcd.print(" ");
  }
  else if(data >=378 && data <=503){
    lcd.setCursor(11,fila);
    lcd.write(byte (11));
    lcd.setCursor(12,fila);
    lcd.print(" ");
  }
  else if(data >=504 && data <=629){
    lcd.setCursor(12,fila);
    lcd.write(byte (12));
    lcd.setCursor(13,fila);
    lcd.print(" ");
  }
  else if(data >=630 && data <=755){
    lcd.setCursor(13,fila);
    lcd.write(byte (13));
    lcd.setCursor(14,fila);
    lcd.print(" ");
  }
  else if(data >=756 && data <=881){
    lcd.setCursor(14,fila);
    lcd.write(byte (14));
    lcd.setCursor(15,fila);
    lcd.print(" ");
  }
  else if(data >=882){
    lcd.setCursor(15,fila);
    lcd.write(byte (15));
  }
}



void setup(){

  //creamos los nuevos caracteres
  crearCaracteres();

  //Iniciamos el LCD
  lcd.init();

 //Encender la luz de fondo.
  lcd.backlight();

//El ciclo 'for' nos mostrara en pantalla un texto indicando la carga del dispositivo,
este se mostrará 2 segundos parpadeando. Su función no es mas que decorativa para el
volante.
  for (int i=0;i<=1;i++){
    lcd.display();
    lcd.setCursor(4,0);
    lcd.print("Cargando");
    delay(700);
    lcd.noDisplay();
    delay(700);
  }


  //SetUp pines unoJoy
```

```
  setupPins();
  setupUnoJoy();

  pinMode(2, INPUT_PULLUP);

  pinMode(3, INPUT_PULLUP);

  attachInterrupt(0, ai0, RISING);

  attachInterrupt(1, ai1, RISING);


//Se inicia nuevamente el display y se limpia la pantalla para luego colocar los textos
"SpeedUp"(aceleración) y brakes(frenos)
  lcd.display();
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("SpeedUp");
  lcd.setCursor(0,1);
  lcd.print("Brakes");

}

// Loop que iniciara las funciones para el funcionamiento del volante
void loop(){

  dataForController_t controllerData = getControllerData();
  setControllerData(controllerData);


  dataAcelerador = analogRead(pinA1);
  acelAndBrakes(dataAcelerador, 0);

  dataFreno = analogRead(pinA2);
  acelAndBrakes(dataFreno, 1);
}


//Cada vez que movamos el volante, una función sumara o restara el valor
correspondiente para la variable counter.
void ai0() {

  if(digitalRead(3)==LOW) {
  counter = counter + 1.32;
  }else{
  counter= counter - 1.32;
  }
  }

void ai1() {

  if(digitalRead(2)==LOW) {
  counter= counter - 1.32;
  }else{
  counter= counter + 1.32;
  }
  }

void setupPins(void){

  for (int i = 4; i <= 12; i++){
```

```
    pinMode(i, INPUT);
    digitalWrite(i, HIGH);
  }
  pinMode(A4, INPUT);
  digitalWrite(A4, HIGH);
  pinMode(A5, INPUT);
  digitalWrite(A5, HIGH);
}

  dataForController_t getControllerData(void){
  dataForController_t controllerData = getBlankDataForController();


  controllerData.squareOn = !digitalRead(4);
  controllerData.crossOn = !digitalRead(5);
  controllerData.dpadUpOn = !digitalRead(6);
  controllerData.dpadDownOn = !digitalRead(7);
  controllerData.dpadLeftOn = !digitalRead(8);
  controllerData.dpadRightOn = !digitalRead(9);
  controllerData.l1On = !digitalRead(10);
  controllerData.r1On = !digitalRead(11);
  controllerData.selectOn = !digitalRead(12);


//Tope de giro para el volante
  if(counter >= 1000){
  counter=1000;
  }
  if(counter <= 0){
  counter= 0;
  }

  controllerData.leftStickX = round(counter) >> 2;
  controllerData.leftStickY = analogRead(A1) >> 2;
  controllerData.rightStickX = analogRead(A2) >> 2;
  controllerData.rightStickY = analogRead(A3) >> 2
  return controllerData;
}
```

Armado

El volante se realizó con una pieza de madera fenólica de 2cm de espesor.

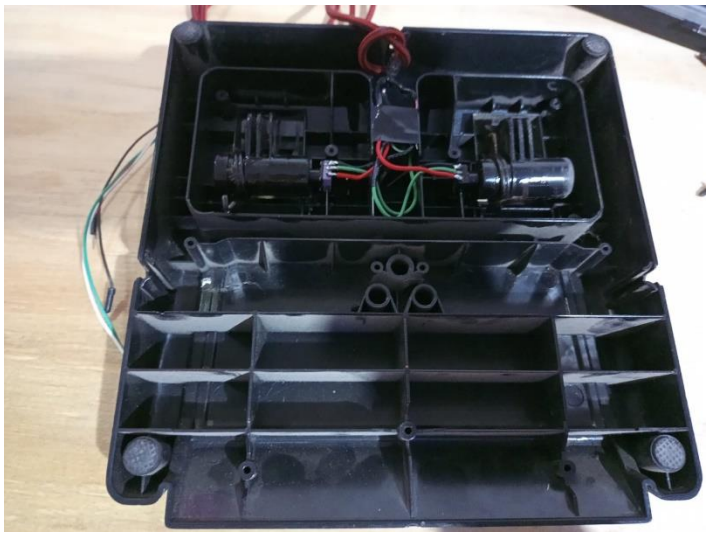
El molde se copió de la foto de un volante de internet.

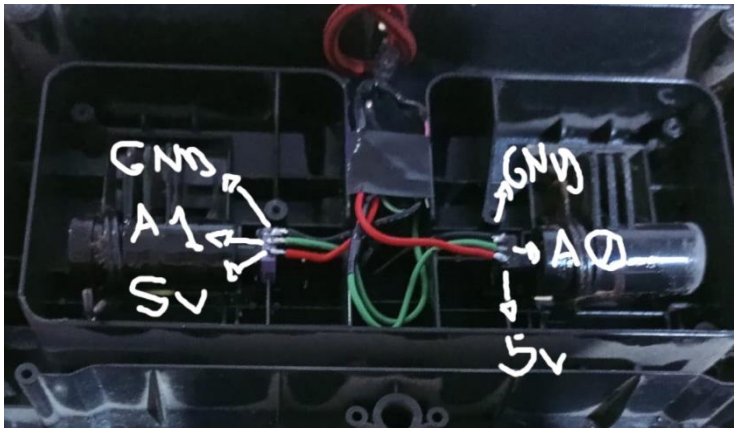
El diámetro es de 30 cm.

Cortamos la madera con una caladora y se la lijó y barnizó.

Luego se comenzó el armado de los pedales. La estructura del pedal utilizada es de un volante económico sin funcionar “Net Runner NR-V21”:

El funcionamiento del pedal era simple, con solo una conexión a GND y otro en IN(5v), sin medir aceleración o frenado de acuerdo con la fuerza con la que se pise, por lo que se optó por realizar la modificación correspondiente mostrada en la siguiente imagen.





De esta manera se logra un mejor funcionamiento a la hora de utilizar el pedal

Se realizó el armado de la caja en madera donde iría conectados Arduino junto al volante y la pantalla lcd y un cable hacia los pedales, quedando el producto físico terminado.

Recursos y costos

1. Los recursos virtuales utilizados fueron una serie de guias tanto en videos como pdf, donde se indica el funcionamiento de cada elemento.

RECURSOS	COSTO
Tarjeta Arduino UNO	$3.990
Protoboard	
Cable USB de la tarjeta Arduino	
Cables de conexión arduino a protoboard	
Potenciometro 10kΩ	
LCD 16x2	
i2c	
Encoder rotativo	$4.679
Maderas MDF 15*30*3 ½ X 2	
Varilla Roscada 5/16 X 1 Metro De Largo	$660
Tuerca Hº 5/16 8 Unid 2x	$185
Computadora personal	
Juego American Truck simulator	$275
TOTAL	**$9789**

Conclusiones

El proyecto logró alcanzar los objetivos planteados dentro de lo que el presupuesto nos permitía. La combinación de elementos de hardware y software tiene la eficiencia deseada, que fue probada con éxito en más de un juego.

Los puntos fuertes son:

- Con el equipo de volante y pedales se logró alcanzar una calidad superior a los volantes económicos que se ofrecen en el mercado.
- El volante tiene un menor tiempo de respuesta gracias a su encoder
- Los pedales tienen una respuesta acorde a la presión con la que se pisen, logrando una experiencia más realista.

Los puntos a mejorar:

- Usar un motor paso a paso para lograr el force feedback, con el que cuentas la gran mayoría de los volantes profesionales, que no pudimos conseguir para este proyecto por estar fuera del presupuesto.
- Un sistema de poleas conectado a un motor paso a paso y al encoder, que permite proteger ambos mecanismos y ayuda a su funcionamiento simultáneo.
- Una mejor terminación estética de la estructura.

Bibliografía.

UnoJoy

https://github.com/AlanChatham/UnoJoy

Guia de uso LCD con I2C

https://www.bolanosdj.com.ar/MOVIL/ARDUINO2/LCDconI2C.pdf

Código base con el cual se comenzo a utlizar el encoder rotativo

https://forum.arduino.cc/t/incremental-encoder/553033