Javier Cervilla Miguel - IES BARAJAS

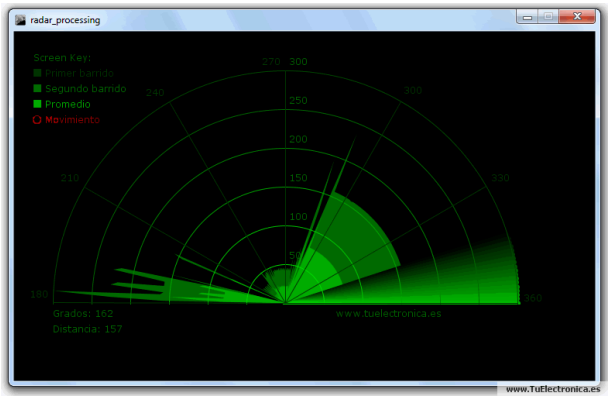
Presentación Autores ==>

ALARMA SONORA DE DETECCION DE OBSTÁCULOS EN PARADA, PEATONES, ANIMALES


© Can Stock Photo

En el momento en que el coche se detiene, ya sea por un semáforo o un paso de peatones, se activa un radar por ultrasonidos que te informa de las posibles presencias de obstáculos en la vía en zonas donde la vista no alcanza.

El dispositivo esta basado en una placa Arduino. Esta placa de código libre y hardware libre nos permitirá controlar tanto el sensor de ultrasonidos como el servo que dirige a este. El funcionamiento es el siguiente, el servo motor, comandado por la controladora Arduino, realizará un barrido de 180º grado por grado rotando el sensor ultrasonidos HC-RSC04 sobre si mismo; en cada uno de los ggrados, el sensor de ultrasonidos, emitirá una frecuencia la cual rebotará contra el obstáculo y retornará al sensor. Este medirá el tiempo que tarda la onda en retornar ; y mediante una sencilla fórmula cinematica podremos calcular la distancia a la que se encuentra el obstáculo.




Debido a que la velocidad de propagación del sonido es un dato que no varía dentro del mismo medio, en nuestro caso el aire, y que conocemos el tiempo que tarda en ir hasta el obstáculo y retornar la onda sonora (hay que aclarar, que el tiempo que recoge el sensor es el tiempo que tarda en ir y venir, por lo que para calcular la distancia lo dividiremos entre 2), podemos calcular la distancia.

Velocidad (Velocidad del sonido= 340m/s ) = espacio (distancia) * Tiempo (tiempo recopilado por el sensor/2)

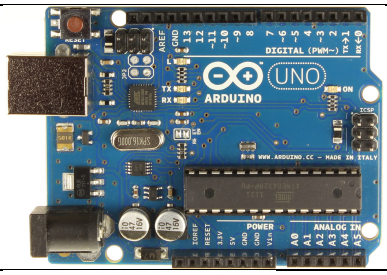
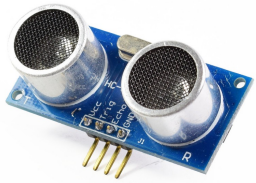
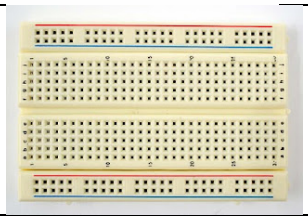
Dado que el sensor de ultrasonidos HC-RS04 tiene una apreciación de milésimas de segundo, y que la distancia la calcularemos en centímetros, hay que realizar un factor de conversión en la velocidad de propagación del sonido, quedando asi:

1 m = 100 cm

1 s = 1000 ms

Velocidad del sonido = 34 cm/ms

HARDWARE UTILIZADO EN EL PROTOTIPO:

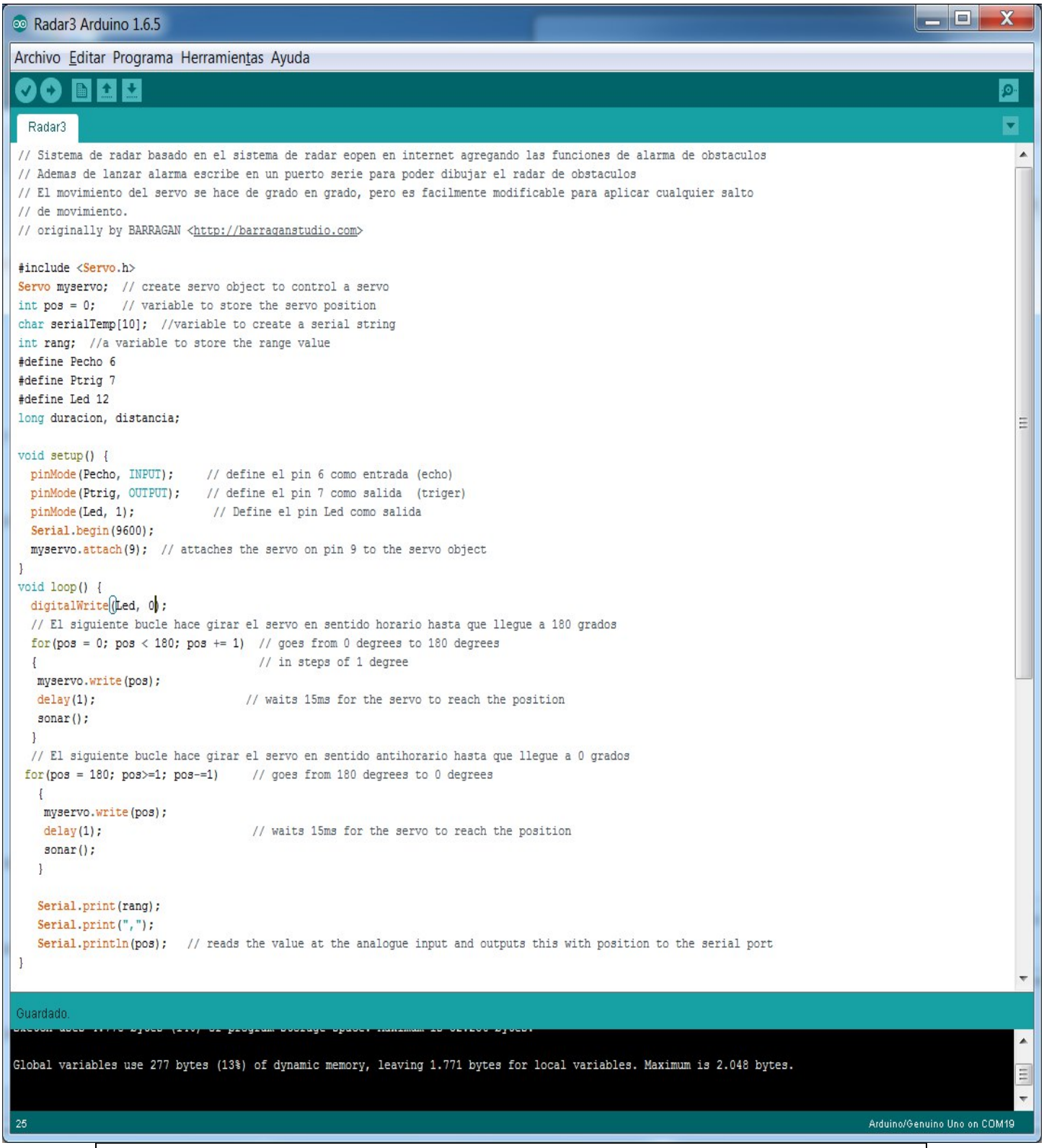
| | |
|---|---|
| 1 x Controladora Arduino UNO |  |
| 1 x Sensor de ultrasonidos HC-RSC04 |  |
| 1 x protoboard |  |
| 1 x buzzer |  |
| 1 x micro servo motor Tower Pro |   |
| 1 x diodo led |  |

| | |
|---|---|
| 10 x cables macho – macho |  |

SOFTWARE:

Para realizar el programa que controle tanto el barrido de 180º del servo, como el sensor ultrasonidos lo desarrollare en el entorno de desarrollo ARDUINO IDE 1.6.5. así como las librerías de control del servo motor ( servo.h)

El programa queda asi:

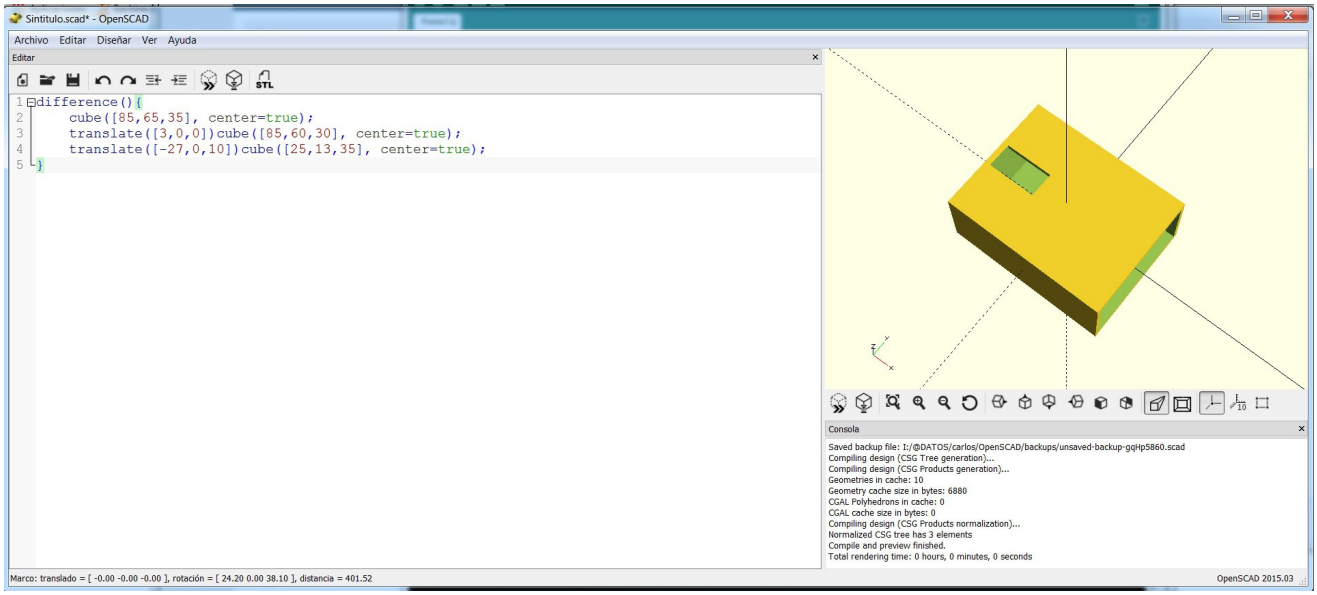


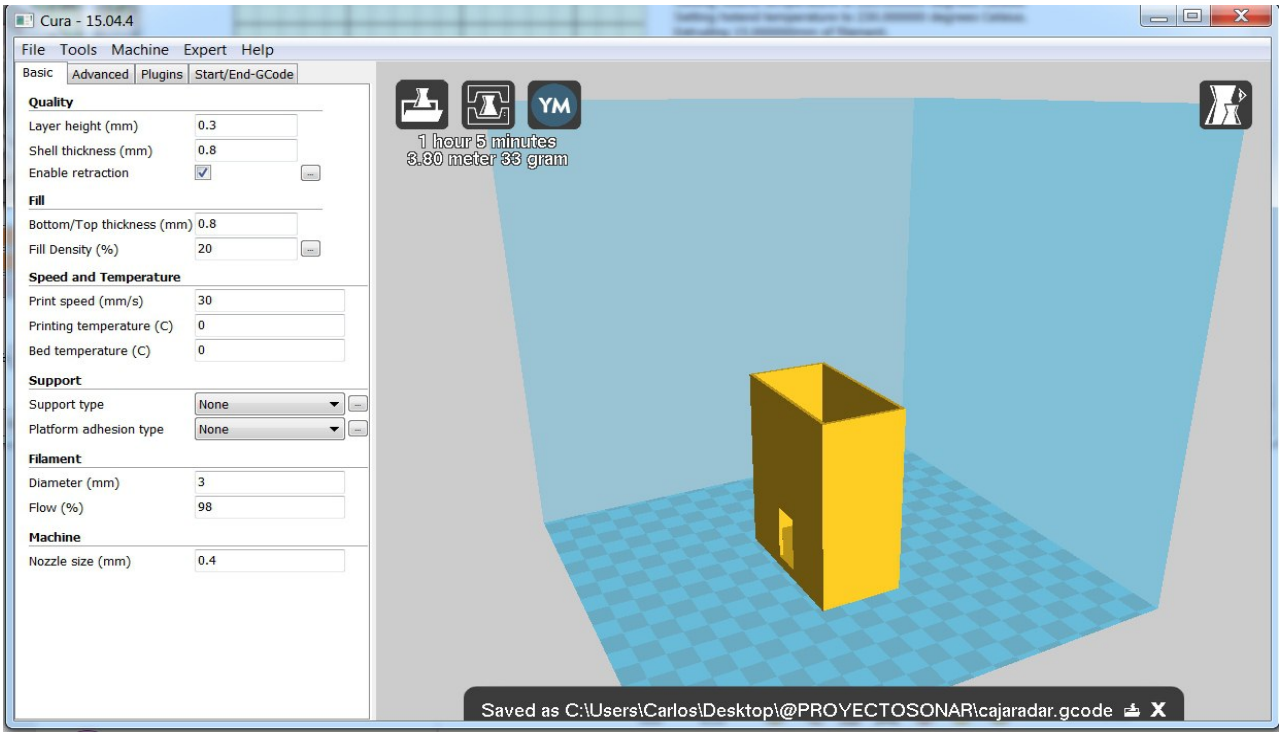

// Sistema de radar basado en el sistema de radar eopen en internet agregando

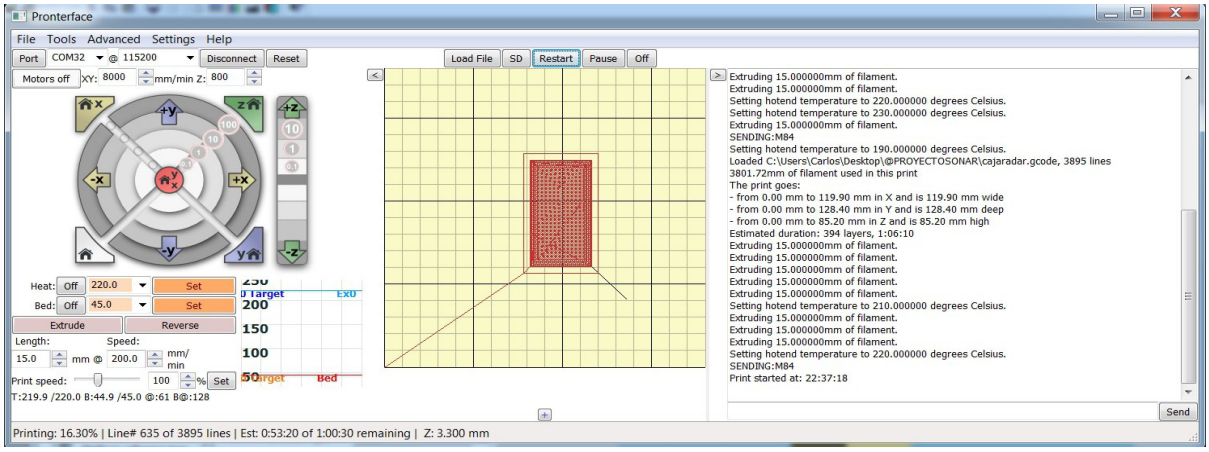
```
las funciones de alarma de obstaculos
// Ademas de lanzar alarma escribe en un puerto serie para poder dibujar el radar de obstaculos
// El movimiento del servo se hace de grado en grado, pero es facilmente modificable para aplicar cualquier salto
// de movimiento.
// originally by BARRAGAN <http://barraganstudio.com>

#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position
char serialTemp[10];  //variable to create a serial string
int rang;  //a variable to store the range value
#define Pecho 6
#define Ptrig 7
#define Led 12
long duracion, distancia;

void setup() {
  pinMode(Pecho, INPUT);     // define el pin 6 como entrada (echo)
  pinMode(Ptrig, OUTPUT);    // define el pin 7 como salida  (triger)
  pinMode(Led, 1);           // Define el pin Led como salida
  Serial.begin(9600);
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}
void loop() {
  digitalWrite(Led, 0);
  // El siguiente bucle hace girar el servo en sentido horario hasta que llegue a 180 grados
  for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 180 degrees
  {                                  // in steps of 1 degree
    myservo.write(pos);
    delay(1);                        // waits 15ms for the servo to reach the position
    sonar();
  }
  // El siguiente bucle hace girar el servo en sentido antihorario hasta que llegue a 0 grados
 for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0 degrees
   {
    myservo.write(pos);
    delay(1);                       // waits 15ms for the servo to reach the position
    sonar();
   }

   Serial.print(rang);
   Serial.print(",");
   Serial.println(pos);    // reads the value at the analogue input and outputs this with position to the serial port
}
```

```
void sonar() {
 digitalWrite(Ptrig, LOW);
 delayMicroseconds(2);
 digitalWrite(Ptrig, HIGH);   // genera el pulso de triger por 10ms
 delayMicroseconds(10);
 digitalWrite(Ptrig, LOW);

 duracion = pulseIn(Pecho, HIGH);
 distancia = (duracion/2) / 29;
 if (distancia >= 500 || distancia <= 0){  // si la distancia es mayor a 500cm o menor a 0cm
   Serial.println("---");               // no mide nada
 }
 else {
   Serial.print(distancia);          // envia el valor de la distancia por el puerto serial
   Serial.println("cm");             // le coloca a la distancia los centimetros "cm"
   digitalWrite(Led, 0);             // en bajo el pin 13
 }

  if (distancia <= 100 && distancia >= 1){
   digitalWrite(Led, 1);                     // en alto el pin 13 si la distancia es menor a 10cm
   Serial.println("Alarma.......");        // envia la palabra Alarma por el puerto serial
 }
 delay(10);                           // espera 400ms para que se logre ver la distancia en la consola
}
```

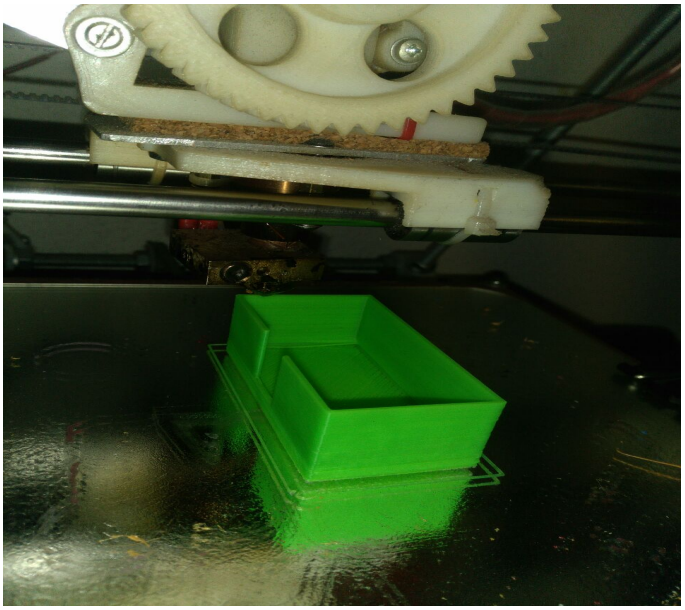
Tras compilarlo y subirlo a la placa, y comprobar su funcionamiento, diseñe con el programa de diseño 3D OPENSCAD una caja para contener la placa y sostenter el servo con el sensor, este programa funciona con código, realizando diferencias entre polígonos realice un primer prototipo de la caja. Tras convertir el archivo a STL que es un formato de modelos 3D basado en la reducción de figuras complejas a triángulos lo compile mediante CURA, un programa que convierte el STL en GCODE, fichero de lenguaje máquina, el cual lee la impresora 3D mediante el controlador PRONTERFACE y realiza la figura en 3D en plástico ABS.

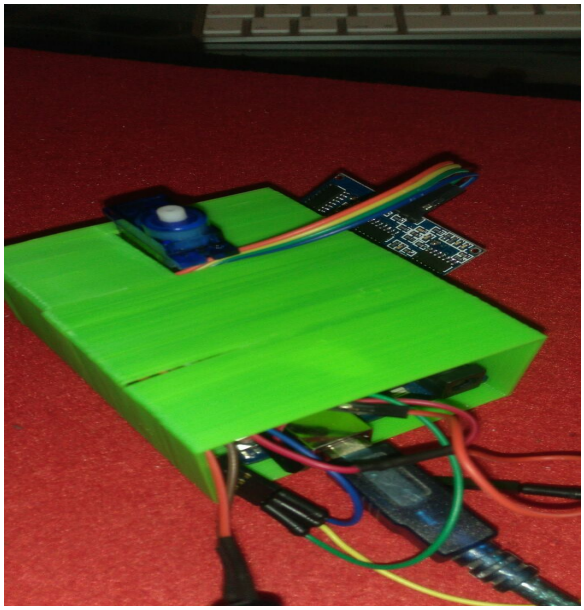

Sintitulo.scad* - OpenSCAD
Archivo Editar Diseñar Ver Ayuda
Editar
1 difference(){
2 cube([85,65,35], center=true);
3 translate([3,0,0])cube([85,60,30], center=true);
4 translate([-27,0,10])cube([25,13,35], center=true);
5 }
Consola
Saved backup file: I:/@DATOS/carlos/OpenSCAD/backups/unsaved-backup-gqHp5860.scad
Compiling design (CSG Tree generation)...
Compiling design (CSG Products generation)...
Geometries in cache: 10
Geometry cache size in bytes: 6880
CGAL Polyhedrons in cache: 0
CGAL cache size in bytes: 0
Compiling design (CSG Products normalization)...
Normalized CSG tree has 3 elements
Compile and preview finished.
Total rendering time: 0 hours, 0 minutes, 0 seconds
Marco: translado = [ -0.00 -0.00 -0.00 ], rotación = [ 24.20 0.00 38.10 ], distancia = 401.52
OpenSCAD 2015.03

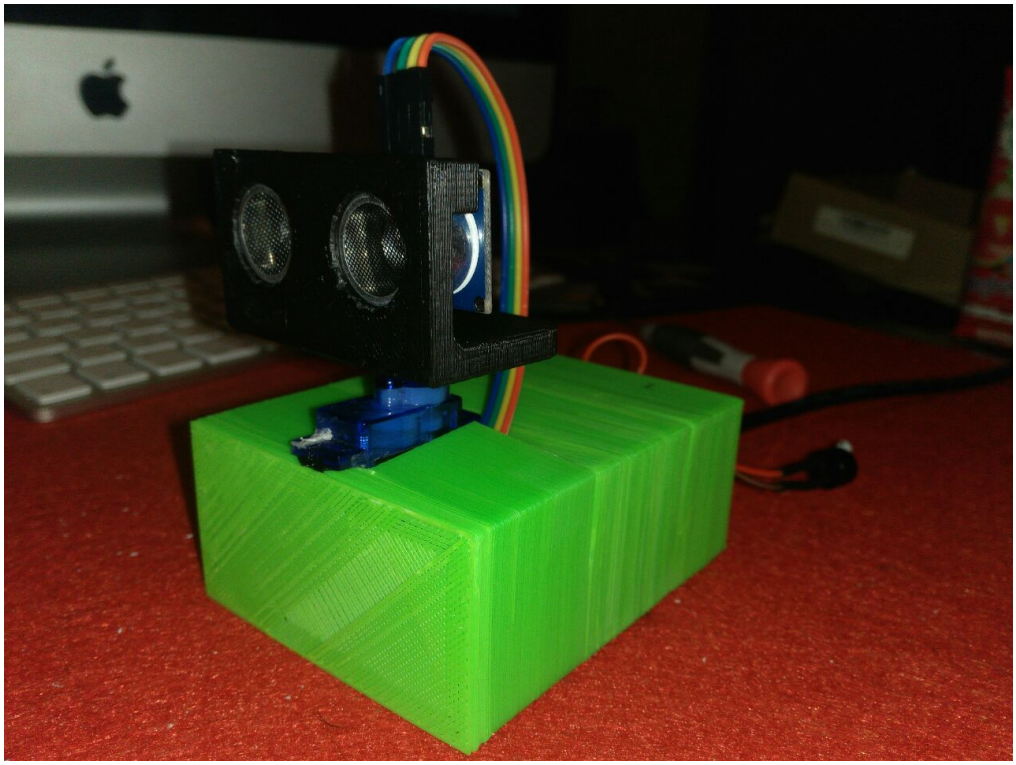

Cura - 15.04.4
File Tools Machine Expert Help
Basic Advanced Plugins Start/End-GCode
Quality
Layer height (mm) 0.3
Shell thickness (mm) 0.8
Enable retraction
Fill
Bottom/Top thickness (mm) 0.8
Fill Density (%) 20
Speed and Temperature
Print speed (mm/s) 30
Printing temperature (C) 0
Bed temperature (C) 0
Support
Support type None
Platform adhesion type None
Filament
Diameter (mm) 3
Flow (%) 98
Machine
Nozzle size (mm) 0.4
YM
1 hour 5 minutes
3.80 meter 33 gram
Saved as C:\Users\Carlos\Desktop\@PROYECTOSONAR\cajaradar.gcode


Pronterface
File Tools Advanced Settings Help
Port COM32 @ 115200 Disconnect Reset
Load File SD Restart Pause Off
Motors off XY: 8000 mm/min Z: 800
Heat: Off 220.0 Set
Bed: Off 45.0 Set
Extrude Reverse
Length: Speed:
15.0 mm @ 200.0 mm/min
Print speed: 100 % Set
T:219.9 /220.0 B:44.9 /45.0 @:61 B@:128
Extruding 15.000000mm of filament.
Extruding 15.000000mm of filament.
Setting hotend temperature to 220.000000 degrees Celsius.
Setting hotend temperature to 230.000000 degrees Celsius.
Extruding 15.000000mm of filament.
SENDING:M84
Setting hotend temperature to 190.000000 degrees Celsius.
Loaded C:\Users\Carlos\Desktop\@PROYECTOSONAR\cajaradar.gcode, 3895 lines
3801.72mm of filament used in this print
The print goes:
- from 0.00 mm to 119.90 mm in X and is 119.90 mm wide
- from 0.00 mm to 128.40 mm in Y and is 128.40 mm deep
- from 0.00 mm to 85.20 mm in Z and is 85.20 mm high
Estimated duration: 394 layers, 1:06:10
Extruding 15.000000mm of filament.
Extruding 15.000000mm of filament.
Extruding 15.000000mm of filament.
Extruding 15.000000mm of filament.
Setting hotend temperature to 210.000000 degrees Celsius.
Extruding 15.000000mm of filament.
Extruding 15.000000mm of filament.
Extruding 15.000000mm of filament.
Setting hotend temperature to 220.000000 degrees Celsius.
SENDING:M84
Print started at: 22:37:18
Send
Printing: 16.30% | Line# 635 of 3895 lines | Est: 0:53:20 of 1:00:30 remaining | Z: 3.300 mm

Para poder compactarlo todo en la caja impresa, sustituyo la proto board por cables macho – hembra







Este es el resultado del primer prototipo funcional, el siguiente paso es reducir el tamaño del conjunto utilizando un ARDUINO MICRO, el cual reduciría el conjunto hasta un tamaño de  4x3x3 cm, el cual pudiera entrar en la parrilla frontal del automóvil.