Arduino Based Tachometer for Gym Cycles

by Adityadp

My grandfather tries to exercise regularly on the **gym cycle** he has, however, the **display** for the cycle is **broken**. Therefore he is unable to see the distance travelled by him after exercising for a certain period of time.

The **LCD** used on the **gym cycle** to display the data was custom made and **could not be replicated**. He was unwilling to exercise without an indication of the distance travelled.

Therefore, I came up with this **solution** where my grandfather could still measure the number of **cycle revolutions** over a certain period of time and calculate the distance travelled.

This project won't apply for regular cycles, but for **exercise bikes,** it is a perfect option if the **LCD** display is **old** or **damaged**.

The design of the circuit is very similar to one of my previous projects: Distance sensor (for white cane). Please check it out if you would like to. https://www.instructables.com/Distance-Sensor-for-...

Supplies:

1) **1 x** 16x2 LCD screen (link)

2) **40 x** Male to Male and Male to female jumper wires (link). You need an assortment of male to male and male to female wire (link)

3) **1 x** HC-SR04 Ultrasonic sensor (link)

4) **1 x** Arduino Uno or Arduino Nano with its connecting cable (link)

5) **1 x** 170 point-small-breadboard (link)

6) **1 x** Potentiometer or trim pot for controlling the LCD's contrast (link)

Step 1: How the HC-SR04 Ultrasonic Sensor Works With the Arduino

The HC-SR04 ultrasonic sensor works on the principle of **sound wave reflection**. One side of the sensor **sends** an ultrasonic wave and the other side of the sensor **detects** it.

These two sides are used in **conjunction**, the **trig pin** of the HC-SR04 is activated, which causes the sensor to shoot an ultrasonic sound wave.

The **echopin** then comes into play. The sensor gives out a **high value** on the echo pin when it **detects** that the sound wave has reflected and **come back.**

The **Arduino and its code** are used to **measure** the **time difference** between the ultrasonic sensor's **trig** pin being activated and the **echo** pin turning high.

This time difference is essentially the time taken for **sound** to **traveltwo times the distance** (because it is being **reflected**) between the sensor and the object it is being pointed at.

Knowing the **speed** of sound, about **340 m/s**, and the time difference makes it very easy to calculate the distance.

2 * distance = speed of sound x time dif.

Actual distance = 0.5 x 340 x time dif.

Here is a link that might explain the circuit a little better.
https://howtomechatronics.com/tutorials/arduino/ul...

The other two pins on the sensor, **Vcc** and **GND**, are connected to the **+5V supply** and **ground** pins on the Arduino respectively. They are there just to **power** the sensor.

Step 2: How the Device Works

The entire device **sits** right **underneaththepedal** so that when the pedal with the ultrasonic **sensorpointingup** on to the underside of the pedal.

The device is **constantlymeasuring** the **distance**

is less than the set **thresholdvalue** in the code.

This causes a **counter** value on the Arduino to **increase** by 1 indicating that one **revolution** has been **completed**. This then can be translated to **actualdistance** using the **diameter** of the gym cycle **wheel**.

between the sensor and the pedal. Since the person is cycling, this value would be changing constantly similar to a **sinecurve**.

When the **pedal** is on its **lowestpoint** the distance measured by the sensor and recorded by the Arduino

The **devicestarts** immediately recording the number of revolutions once it's **USB** cable has been **plugged** into a power source. This power source can be a laptop, power bank, or an adapter from the wall.

Step 3: Wiring the LCD

Pins D2, D3, D4, D5, D11, and D12of the **Arduino** is connected to pins 14, 13, 12, 11, 6, and 4 of the **LCD**, **respectively**.

The pins will be **numbered** directly on the **Arduino**. For the LCD you can **count** the pins from **left to right** if they aren't already numbered.

These are roughly the **data pins** which transmit information from the Arduino to the LCD about what to display on the screen.

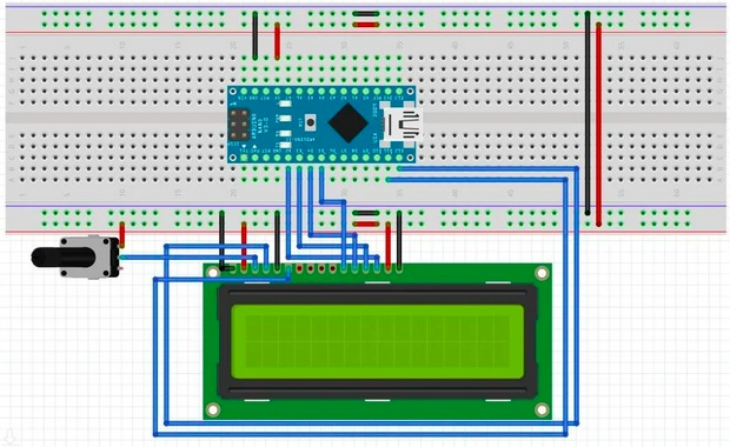
Pins1, **5**, and **16** of the **LCD** are connected to **ground**.

Pins2 and **15** of the **LCD** are connected to **+5V**.

Pin 3 of the LCD is **connected** to the **middle** terminal of the **Potentiometer** or trim pot. One of the other two **terminals** of the **Potentiometer** or trim pot should be connected to **ground** and the other to **+5V**.

Pins 7, 8, 9, and 10 of the LCD are not connected to anything.

Follow the diagram for more clarity.

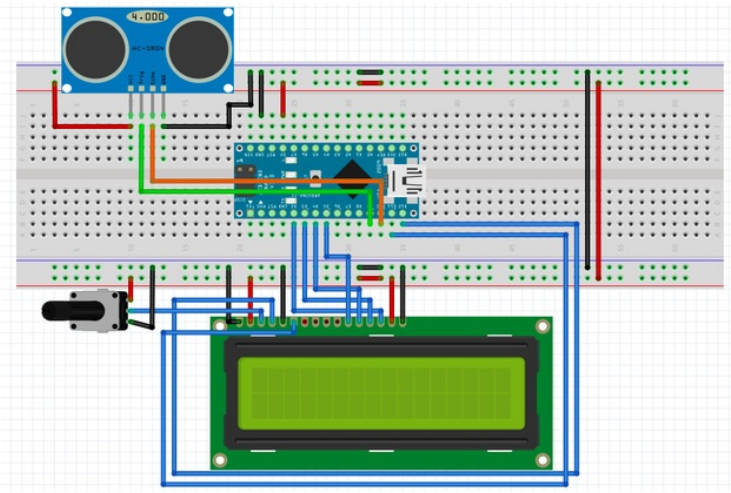


Step 4: Wiring the Ultrasonic Sensor

The ultrasonic sensor has **4 pins**. The outermost pins, named **Vcc** and **GND**, are connected to the **+5V rail** and **ground rail,** respectively.

The pin labelled **trig** on the ultrasonic sensor is connected to **pin 9** of the **Arduino**. This connection is shown as the **green** wire.

The pin labelled **echo** on the ultrasonic sensor is connected to **pin 10** of the **Arduino**. This connection is shown as the **orange** wire.



Step 5: Code

```
#include // includes the LiquidCrystal Library

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // sets the pins for the LCD

const int TP = 9; // stands for Trigger pin

const int EP = 10; // stands for Echo pin

const int Buzzer = 6; // sets the pin for the buzzer

float dur; // Time taken for the sound waves to travel is now recorded to two decimal places

float distcm; // Distance is now recorded to two decimal places

float val1 = 1000000;

float val2 = 1000000;

bool controlswitch = false;
```

```
int pedalcounter = 0; \\ Counts the number of pedals

float threshold1 = 0; \\ This is the minimum distance between the sensor and the pedal

float radius = 0; \\ Radius of the cycle's wheel

float threshold2 = threshold1 + radius; \\ This is the minimum distance + the radius of the wheel

float distancetravelled = 0; \\ This is a changing variable. It shows the total distance travelled by the cycle.

void setup()

{

lcd.begin(16,2); // configures the dimensions of the LCD

pinMode(TP, OUTPUT); // sets the trigger pin as an output

pinMode(EP, INPUT); // sets the echo pin as an input

Serial.begin(9600); // Sets the serial baud. Allows the arduino to communicate with the serial monitor

pinMode(Buzzer, OUTPUT); //sets the buzzer pin as an output

}

void loop()

{

digitalWrite(TP, LOW);

delayMicroseconds(5);

digitalWrite(TP, HIGH); // Previous three commands tell the ultrasonic sensor to send out a pulse. Going from low to high back to low.

delayMicroseconds(10); // The ultrasonic pulse lasts for 10 Microseconds

digitalWrite (TP, LOW); // Bringing the trigger pin back to low.

dur = pulseIn(EP, HIGH); // PulseIn records the time it takes for the pin to change its state to HIGH. dur is the time taken for the sounds waves to travel from the trigger, hit the object, and bounce back to be detected. //Serial.println(dur);

if (dur < 10000) // This statement prevents any malfunctioning of the sensor. If an object is brought too close for the sounds waves to reflect back properly than the buzzer will sound.

{

distcm = (dur*0.017); // Converts the time taken for the sound to travel to the distance of the object from the sensor.

Serial.println(distcm); // Prints the distance in cm on the serial monitor to help identify if there is an issue with the LCD.

val2 = val1;

val1 = distcm;



if (val1 < threshold1 && controlswitch == false)

{

controlswitch = true;

pedalcounter++;

lcd.clear();

lcd.print(pedalcounter);

distancetravelled = pedalcounter*2*3.141592*radius;

}
```

```
if(val1>threshold2)
{
controlswitch = false;
}
}
else
{
lcd.clear();
lcd.print("Too far!");
delay(500);
}
}
```

Step 6: Housing Design

Obviously, the circuit cannot be left like this for its final use. A proper housing has to be made which holds the ultrasonic sensor in place.

I used my 3D printer to build the housing, however, there are many other options if you don't have access to a 3D printer. Laser cutting the sides of the box is a possibility. You can build housing from spare balsa wood as it is easy to cut and drill.

Designing

1. I used **Onshape** to design the housing for the circuit.

2. The model's **.step file** has been attached below. If you want to alter the housing download the .step file and use a 3D modelling software to edit it. Onshape is online and free.

3. Rough dimensions of the model are **110mm x 90mm x 80mm.**

4. I used Ultimaker **Cura** as the slicer and **Ender 3** as the 3D printer.

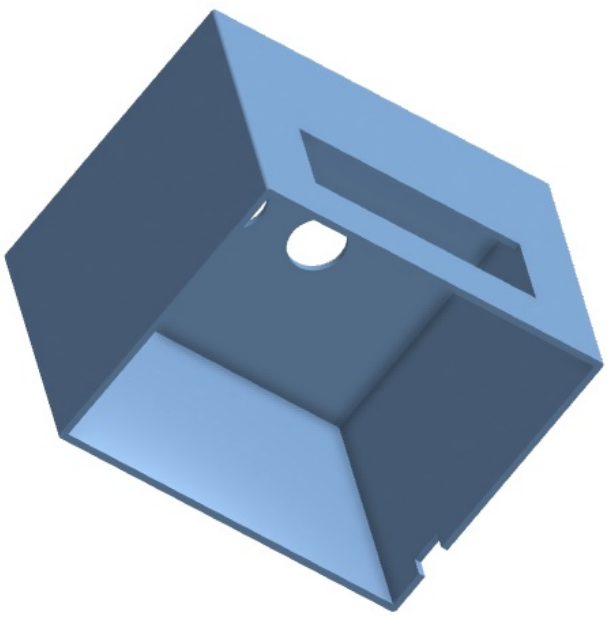
For this project

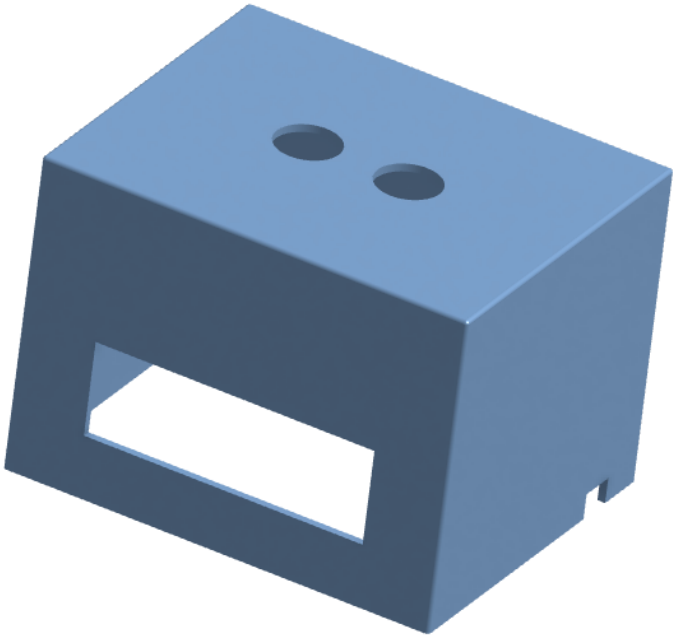
1. The Ultrasonic **sensor's** two shells fit through the two **holes** on the top of the design

2. There is a **hole** on the side of the design with the same dimensions as a **16x2 LCD screen**, 24mm x 72mm. In my model, I only had a 20x4 LCD screen but you don't have to. If you have access to the 16x2 screen use that. It is cheaper, lighter, and more effective.

3. I do not have a base for the housing. It was done so that it is very **easy** to **access** the **circuit** for repairs. A piece of paper can be placed at the bottom to keep it all together when moving.

4. There is also a small **notch** near the **back-side** of the design. This is for the USB **cable** that would be powering the Arduino to **pass through**. Now if you place the device down it will sit flat on the floor.





View in 3D | Download

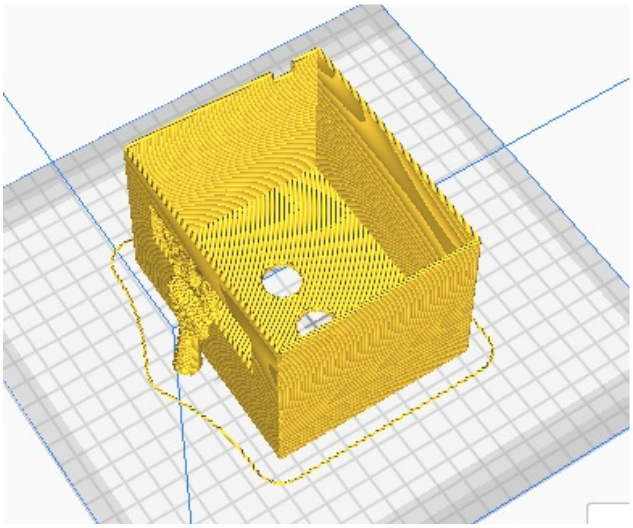
https://www.instructables.com/ORIG/FCJ/G40J/KGXP4G84/FCJG40JKGXP4G84.stl

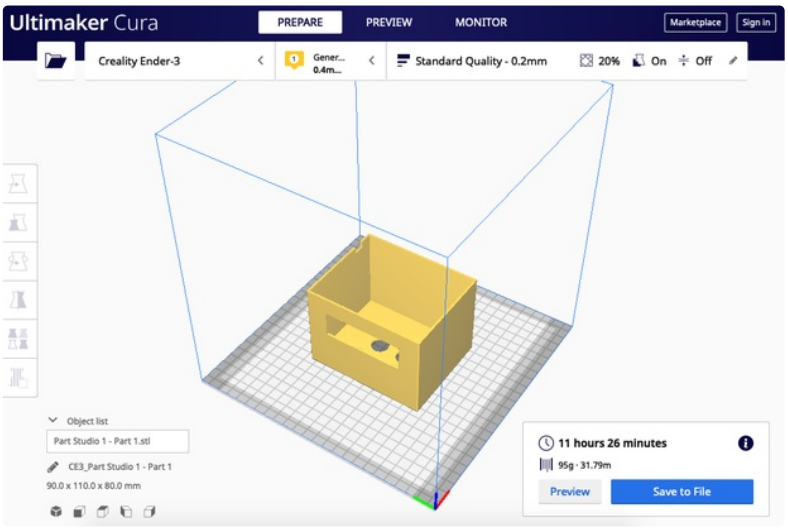
Step 7: 3D Printing

1. I used Ultimaker Cura as the slicer and Ender 3 as the 3D printer.

2. File was uploaded to the 3D printer. The temperature presets were **200** degrees C for the **nozzle** and **50** degrees C for the **bed**.

3. The print took about **11.5 hours** and **30 meters** of filament. Using pliers I removed the model off the platform and picked off the supports.





Step 8: Assembly

1. I pressed the ultrasonic sensor into the holes. Same for the LCD screen.

2. The rest of the circuit was carefully pushed into the housing.

3. The USB cable was connected to the Arduino and now everything was good to go.