Documentatie IoT Project

Frederik Stroobandt

1Gr – IoT Thomas More Hogeschool

Sint-Katelijne-Waver, Campus De Nayer

30 mei ’21 (Week 7)

Contents

1 Code

Screenshots van code

```
#include <WiFi.h>
#include <HCSR04.h>
#include <PubSubClient.h>

#define WHEEL_L_1 16
#define WHEEL_L_2 17 // Zal hier niet gebruikt worden maar kan gebruikt worden om achteruit te rijden
#define WHEEL_R_1 18
#define WHEEL_R_2 19 // Zal hier niet gebruikt worden maar kan gebruikt worden om achteruit te rijden

#define LED_LOST 27
#define LED_PROBLEM 26
#define LED_OK 25

#define TRIG 5
#define ECHO 4
UltraSonicDistanceSensor echo_mod(TRIG, ECHO);

#define IR_L 32
#define IR_M 35
#define IR_R 34
#define BUTTON 15

#define BLACK_LINE 1
#define WHITE_LINE 0
#define STOP_OBJ 10 // de afstand in cm van de ultrasone module tot het dichtste toegelaten object
// de auto zal in dit voorbeeld op 10cm van een object stoppen.

int color_line = BLACK_LINE;
bool interrupt = false;

const char* ssid = "SSID"; // voer hier uw netwerk's SSID in
const char* password = "PASSWORD"; // voer hier het wachtwoord in van het netwerk dat u hierboven heeft meegegeven
const char* mqttServer = "255.255.255.255"; // hier voert u het IP adres in van uw MQTT server
const int mqttPort = 1883; // vervang 1883 met de poort die uw MQTT server gebruikt
const char* mqttUser = "Jeff"; // hier voert u uw MQTT user ID in, als u deze gebruikt
const char* mqttPassword = "123"; // voer hier het wachtwoord van uw MQTT gebruiker in
const char* clientID = "Car"; // MQTT client ID - deze moet uniek zijn voor elke client op het MQTT netwerk.

void callback(char* topic, byte* payload, unsigned int length) {
  String message = String(topic); // de callback functie die controlleerd of het MQTT bericht command/continue is
  if (message == "command/continue") { // en als dit zo is, dan zal de auto doorrijden
    interrupt = true;
  }
}

void IRAM_ATTR isr() { // het declareren van een interrupt functie zodat de drukknop van het autootje zelf altijd werkt
  interrupt = true;
}

WiFiClient espClient; // het klaarzetten van de WiFi
PubSubClient client(mqttServer, mqttPort, callback, espClient); // MQTT klaarzetten

void setup() {
  Serial.begin(115200);
  pinMode(BUTTON, INPUT_PULLUP);
  attachInterrupt(BUTTON, isr, FALLING); // de drukknop een interrupt maken

  pinMode(WHEEL_L_1, OUTPUT);
  pinMode(WHEEL_L_2, OUTPUT);
  pinMode(WHEEL_R_1, OUTPUT);
  pinMode(WHEEL_R_2, OUTPUT);

  pinMode(LED_LOST, OUTPUT); // zorg ervoor dat de LEDs kunnen schijnen
  pinMode(LED_PROBLEM, OUTPUT);
  pinMode(LED_OK, OUTPUT);

  pinMode(IR_L, INPUT); // de IR sensoren leesbaar maken
  pinMode(IR_M, INPUT);
  pinMode(IR_R, INPUT);

  client.setServer(mqttServer, mqttPort);
  WiFi.begin(ssid, password); // de WiFi aanzetten
  while (WiFi.status() != WL_CONNECTED) { // verbinden met de WiFi
    delay(500);
    Serial.println("Connecting to WiFi..");
  }
}
```

```
78
79 void signal_led(int led_nr) {
80   switch (led_nr) {
81     case (LED_LOST): digitalWrite(LED_LOST, HIGH); digitalWrite(LED_PROBLEM, LOW); digitalWrite(LED_OK, LOW); break;
82     case (LED_PROBLEM): digitalWrite(LED_LOST, LOW); digitalWrite(LED_PROBLEM, HIGH); digitalWrite(LED_OK, LOW); break;
83     case (LED_OK): digitalWrite(LED_LOST, LOW); digitalWrite(LED_PROBLEM, LOW); digitalWrite(LED_OK, HIGH); break;
84     default: Serial.println("Error"); break;
85   }
86 }
87
88 void turn_left() {
89   digitalWrite(WHEEL_L_1, LOW);
90   digitalWrite(WHEEL_R_1, HIGH);
91 }
92 void turn_right() {
93   digitalWrite(WHEEL_L_1, HIGH);
94   digitalWrite(WHEEL_R_1, LOW);
95 }
96 void stop_car() {
97   digitalWrite(WHEEL_L_1, LOW);
98   digitalWrite(WHEEL_R_1, LOW);
99 }
100 void drive_forward() {
101   digitalWrite(WHEEL_L_1, HIGH);
102   digitalWrite(WHEEL_R_1, HIGH);
103 }
104
105 bool sees_line(int IR_x) { // deze fucntie controleert of de lijn die gevolgd word (zwart of wit, bepaalpaar bovenaan) zichtbaar is voor de sensor
106   bool line_seen = digitalRead(IR_x) == color_line;
107   return line_seen;
108 }
109
```

```
110 void determine_drive() {
111   double dist_obs = echo_mod.measureDistanceCm();
112   bool left_eye = sees_line(IR_L);
113   bool right_eye = sees_line(IR_R);
114   bool mid_eye = sees_line(IR_M);
115   interrupt = false; // voor de drukknop op de auto of de Raspberry Pi terug te "resetten" - deze staat hier in het geval dat de knop per ongeluk werd ingedrukt wanneer de auto niet stilstond
116   signal_led(LED_OK);
117   if (dist_obs < STOP_OBJ) {
118     stop_car();
119     signal_led(LED_PROBLEM);
120     while (echo_mod.measureDistanceCm() < STOP_OBJ) {
121     }
122     signal_led(LED_OK);
123     drive_forward();
124     return;
125   } else if (left_eye && !right_eye) { // Ik vraag hier het criteria van het middelste oog niet aan omdat dit niet relevant is als ik enkel links of rechts een lijn zie.
126     turn_left();
127   } else if (!left_eye && right_eye) { // Ik vraag hier het criteria van het middelste oog niet aan omdat dit niet relevant is als ik enkel links of rechts een lijn zie.
128     turn_right();
129   } else if (!mid_eye) { // Het middelste oog ziet geen lijn meer, de auto zal dan direct weten dat het de lijn kwijt is en zal dus stoppen.
130     signal_led(LED_LOST);
131     stop_car();
132     while (! interrupt) { // Als deze functie word opgeroepen wilt dit zeggen dat de auto gaat wachten tot als iemand op de knop drukt
133       client.loop();
134     }
135   } else if (left_eye && right_eye) { // omdat ik al heb gevraagd of het middelste oog de lijn niet ziet, ben ik zeker dat in deze gevallen mid_eye de lijn wel ziet.
136     while (sees_line(IR_L) && sees_line(IR_R)) {
137       // ik laat de auto rijden zolang deze beide lijnen ziet zodat de sensoren over de lijn staan, dan wacht de auto 20 seconden zoals gewenst
138       drive_forward();
139       if (! sees_line(IR_M)){ // in het heel uitzonderlijke geval dat de auto de lijn kwijtraakt terwijl deze vooruit rijd zal deze terugspringen uit deze tak
140         return;
141       }
142     }
143     stop_car();
144     for (int i = millis(); (i + 20000 > millis()); i) {
145       //  ik hou hier  de auto voor 20 seconden vast, tenzij de auto niet stilstaat Ik liet de auto stoppen de lijn hierboven.
146       // de enige  manier dat de auto terug begint te  rijden is als de 20 seconden gedaan  zijn of als ik via MQTT de auto weer laat rijden
147       client.loop();
148       if (interrupt) {
149         return;
150       }
151     }
152   } else if (!(left_eye || right_eye)) { // hier pas ik LeMorgan's regel toe [NOT x AND NOT Y] is equivalent met NOT[x OR Y] of kortweg NOR
153     // kan  ook vervangen worden door "else" zonder if en conditie omdat dit het geval is als geen van de bovenstaande waar is (al de andere combinaties zijn uitgesloten) - zie waarheidstabel
154     drive_forward();
155   }
156 }
```

```
157 //IR sensoren zien zwart als 1 & alles anders als 0 --> werken met zwarte lijn is IR rechtstreeks uitlezen m.a.w. sensor(IR)==1
158
159 void loop() {
160   while (!client.connected()) {
161     Serial.println("Connecting to MQTT...");
162     if (client.connect(clientID, mqttUser, mqttPassword )) {
163       Serial.println("connected");
164       client.subscribe("command/continue");
165     } else {
166       Serial.print("failed with state ");
167       Serial.print(client.state());
168       delay(2000);
169     }
170   }
171   determine_drive();
172 }
```

Uitleg

In de eerste drie lijnen code roep ik libraries aan die ik nodig zal hebben voor deze robot te maken, de voornaamste twee zijn de WiFi.h en de PubSubClient.h libraries die ervoor zullen zorgen dat ik dankzij MQTT-berichten zal kunnen ontvangen. De HCSR04.h library is de library die mij de Ultrasone sensor laat uitmeten in centimeters zodat ik zelf geen berekening meer moet doen a.d.h.v. de snelheid van geluid.

In de lijnen 4 t.e.m. 37 declareer ik variabelen, objecten en constanten die ik later in mijn code gebruik zodat ik slechts op een plaats in mijn code een aanpassing moet maken als ik bedrading aanpas.

In dit stuk kan u aanpassen welke kleur lijn u volgt (lijn 28 int color_line=…), en u moet tussen lijnen 31 en 37 de code aanpassen zodat deze werkt met uw netwerk en uw MQTT-netwerk.

Tussen lijnen 38 en 52 declareer ik nog twee functies die Setup() nodig heeft. De callback functie voor MQTT berichten te verwerken en isr(), een functie die als interrupt werkt voor de ESP32, hierna worden de espClient en client objecten aangemaakt voor te verbinden met WiFi en MQTT.

Tussen 53 en 77 gebeurt de setup, hier declareer ik welke pinnen er als output zullen dienen en welke als input, ik vertel mijn WiFi en client objecten eveneens welke SSID en MQTT-server ze respectievelijk mee moeten verbinden. En dan wacht ik tot er een verbinding is gemaakt tussen de ESP32 en de WiFi. **Dit wil zeggen:** zonder WiFi zal de auto niet werken.

De functie signal_led krijgt meegegeven welke LED aan moet springen, en zal de twee andere uitschakelijken, dit doe ik aan de hand van een switch case, dit heb ik gedaan in een functie in het geval ik meerdere keren van LED moet veranderen en heb zo van 3x (# keer van LED veranderen) naar 7 + (#keer van LED veranderen) lijnen code gegaan.

In de turn_left, turn_right, stop_car en drive_forward functies gebruik ik heel simpele code zodat het leesbaarder is.

sees_line(int IR_x) is weeral een functie die niet per se nodig is, maar wel handig is voor de leesbaarheid van de hele code.

determine_drive() is het brein van de code van de auto, hierin wordt bepaald hoe de auto moet rijden, in de eerste vier lijnen maakt de functie 4 lokale variabelen aan en houd deze bij zodat elke vergelijking in de body met gelijke parameters verloopt. Eerst kijkt de functie of er geen object te kortbij staat, in dit geval slechts 10 centimeter, ik vertel de auto om dan onmiddellijk te stoppen en de oranje/gele LED aan te zetten zodat er visueel gesignaleerd wordt dat de auto is gestopt wegens een obstakel. Zodra het obstakel meer dan 10 cm verwijderd is zal de Auto de groene LED aanzetten en gewoon vooruitrijden (dit kan ik nog verwijderen).

In de tweede IF-tak wordt gecontroleerd of het linkeroog de streep ziet, maar het rechteroog niet, zo ja, dan slaat de auto links af. In de derde IF-tak gebeurt het omgekeerde, of het rechteroog de streep ziet maar het linkeroog niet, zo ja, dan slaat de auto rechts af.

Op lijn 129 wordt er voor de eerste (en enige) keer naar de meting van het middelste oog gekeken. Als deze geen lijn ziet dan slaat de auto stil, de rode lamp zal aanduiden dat de auto is gestopt en niet verder zal rijden tot er op een knop wordt gedrukt, en zal pas in een andere tak springen als het probleem is opgelost.

Hierna kijkt de wagen of beide sensoren de lijn zien, zo ja, dan blijft het rijden tot het over de streep staat en dan stopt het. Zodra deze of 20 seconden heeft gewacht aan de lijn, of een signaal krijgt van het commandocentrum zodat de wielen weer draaien, zal de loop voortgaan en wacht de auto niet meer.

De laatste tak is enkel vooruitrijden, als de twee buitenste ogen niets uitlezen, (en het middelste oog wel – dit is impliciet gecontroleerd door lijn 129), dan zal de auto vooruitrijden.

In de loop zal er telkens een connectie gemaakt worden met de MQTT-server, zodat de auto de MQTT-berichten van het commandocentrum kan ontvangen en deze kan verwerken a.d.h.v. de geschreven callback functie, deze zal de nuttige berichten van de nutteloze berichten scheiden. En dan op het einde van de loop roep ik determine_drive() aan zodat de robotauto de lijn kan volgen. **Dit wil zeggen:** De Auto zal niet werken zonder MQTT.

Code zonder WiFi en MQTT

Als u de code wilt zonder dat WiFi en MQTT gebruikt worden moet u een beetje code weghalen.
Te verwijderen lijnen (Ik raad aan om onderaan te beginnen, dan blijven de lijnnummers juist):
1, 3, 30 t.e.m. 44, 50 t.e.m. 52, 70 t.e.m. 76, 133, 147 en dan 160 t.e.m. 170.

Er zal ook een download link zijn op GitHub voor de code zonder WiFi en MQTT.

Waarheidstabel voor de 3 IR-ogen

↓ Scenario	left_eye	mid_eye	right_eye
Lost	0	0	0
Right	0	0	1
Drive	0	1	0
Right	0	1	1
Left	1	0	0
Lost	1	0	1
Left	1	1	0
Stop	1	1	1

In deze waarheidstabel kunnen we zien dat er slechts 5 scenario's zijn die we nodig hebben. Dus om van 8 scenario's (alle mogelijke combinaties) naar 5 te gaan en meerdere gevallen tegelijk te tackelen zullen we kijken naar de variabelen die het meeste doen om een scenario te bepalen.

Voor Left zien we dat left_eye gelijk moet zijn aan 1 en right_eye moet gelijk zijn aan 0, dit wil zeggen dat mid_eye er niet toe doet, hetzelfde geldt voor Right, maar de waardes van right_eye en left_eye zijn 1 en 0 respectievelijk voor dit scenario. Dit wil zeggen dat we met deze twee geen variabelen kunnen uitsluiten en er zijn nog 4 andere "scenario's" mogelijk.

Met Lost zien we dat mid_eye uitsluitend 0 mag zijn, maar de andere twee doen er niet meer toe. Dit neemt 2 van de 4 overblijvende basis-scenario's weg. En als we kijken zien we dat deze twee beide mid_eye gelijk aan 1 over houden, en sterker nog left_eye en right_eye zijn gelijk aan elkaar.

Dit wil zeggen dat ik zelfs door enkel een van de twee te testen kan zien in welke tak ik zal belanden. Maar ik had besloten om deze twee nog altijd expliciet te testen en geen "else" tak te gebruiken.

Moest ik het nog compacter willen schrijven dan kon ik het volgende doen.

```
If (left_eye && !right_eye){ // dit is de left-tak
…} else if (right_eye && !left_eye){ // dit is de right-tak
…} else if (!mid_eye){ // dit is de lost-tak
…} else if (right_eye){ // test of ik in stop of drive zit – dit is de stop tak
…} else {…} // dit is de drive-tak
```

2 Hardware & aankopen

Kosten

Ik laat hier enkele prijzen buiten beschouwing, namelijk de prijzen van de soldeerbout, het soldeersel, de bouten en moeren of vijzen die u gebruikt, de female-to-male en male-to-male kabels, de 3 LEDs en de drukknop, deze maakte deel uit van een kit waarvan ik de prijs niet ken.

Naam	Aankoop prijs	Hoeveelhe	Prijs per s	Hoeve	Prijs per auto	link
IR-sensoren	6.49	5	1.30	3	3.90	https://www.amazon.d
Echolocatie module	11.99	5	2.40	1	2.40	https://www.amazon.d
Breadboard 170 puntjes	10.99	12	0.92	1	0.92	https://www.amazon.d
chassis - motor - wielen kit	18.90	1	18.90	0.5	9.45	https://www.amazon.d
ESP32 38pin	29.99	5	6.00	1	6.00	https://www.amazon.d
Blanke PCB 5x7 cm en pins	10.99	5	2.20	1	2.20	https://www.amazon.d
Zwenkwieltje	13.99	4	3.50	1	3.50	https://www.amazon.d
L298N	12.99	5	2.60	1	2.60	https://www.amazon.d
Powerbank	9.79	1	9.79	1	9.79	https://www.amazon.d
Totaal:	126.12	/	/	/	40.76	

U zult hier zien dat ik in een kit 0.5 als een hoeveelheid nodig heb gelijst, dit is omdat de kit die ik gebruikt heb gemaakt is om één auto met 4 wielen en motoren te maken, hierdoor moet men het zwenkwiel apart kopen. Maar hierdoor krijgt u wel voldoende chassis, motoren en wielen om 2 wagens te maken. Dit komt goedkoper uit dan de 2 wielen kit te kopen per auto.

Hier gebruik ik een powerbank in plaats van een veel goedkopere 9V batterij en batterij aansluiting omdat de ESP32 die hier gebruikt wordt te veel stroom vraagt, dit kan niet geleverd worden als de batterij ook de wielen moet aandrijven. De powerbank levert 2.1A maximaal wat meer als voldoende is.

Doordat ik gebruik maak van Amazon hier en al de gekozen producten genieten van Amazon's levering kost verlaging moet u geen kosten rekenen voor verzendingen.

Benodigdheden

Hardware:

- 1 chassis die 15 bij 20 meet
- 1 blanco 5x7cm PCB of 2 Breadboards die minstens 20 x 5 gaten zijn elk (om de ESP32 op te pinnen)
- 1 klein breadboard
- 1 L298N motor drive controller
- 38-pin ESP32
- 1 drukknop
- 3 LEDs – 1 groen, 1 geel, 1 rood
- 3 IR sensoren type FC-03
- 1 HC-SR04 Ultrasone sensor
- 2 wielen en een zwenkwiel (of knikker en een knikker houder)
- 2 dc-motoren
- 1 powerbank – zo compact mogelijk met 5V/2.1A output
- 1 USB-to-micro-USB kabel
- Een bundel male-to-male kabels
- Een bundel male-to-female kabels

Tools:

- Schroevendraaier
- Mes
- Bouten & moeren
- Dubbelzijdige plakband
- Soldeerbout
- Soldeersel
- Een wel geventileerde ruimte

Assemblage

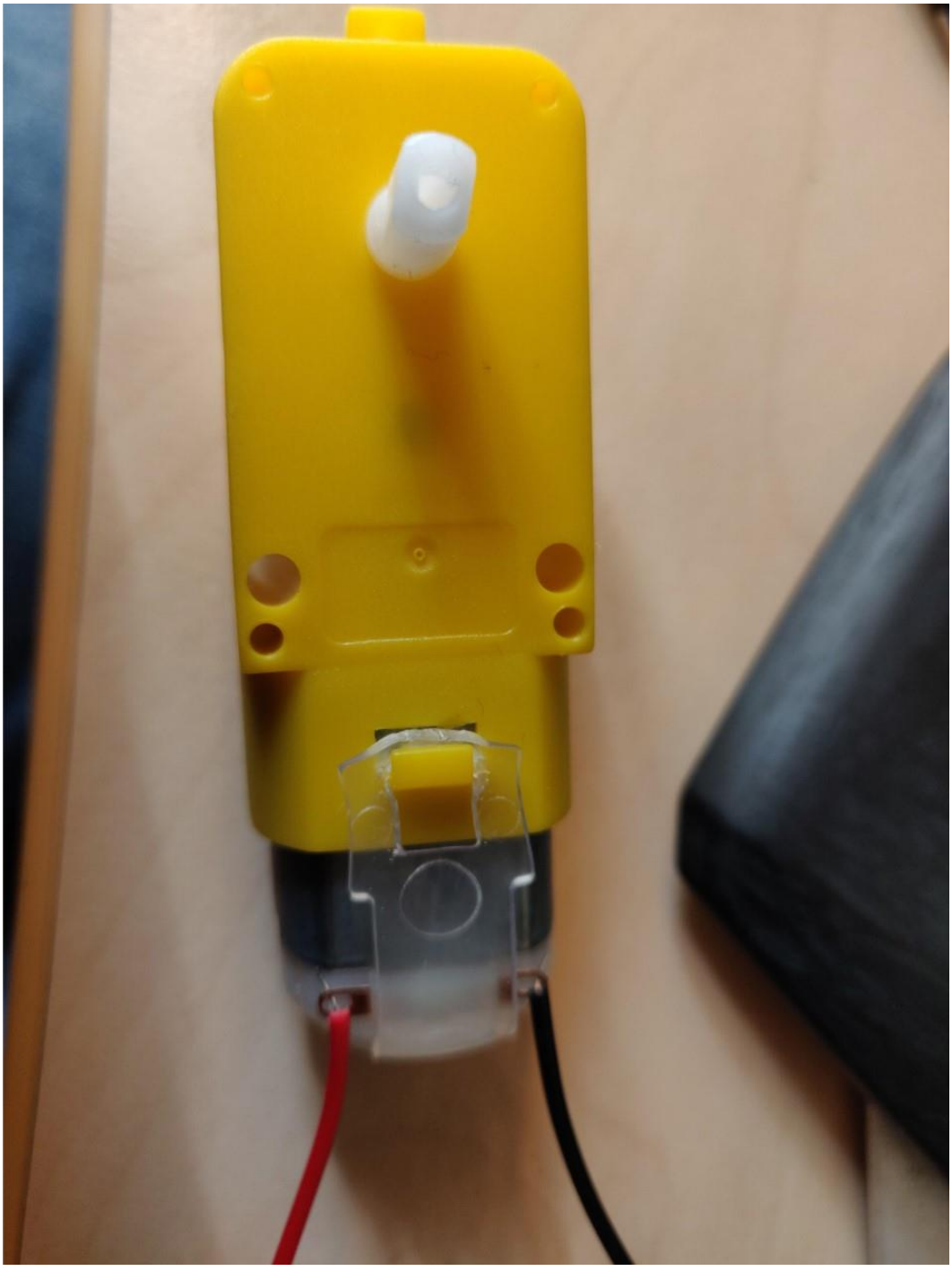
Disclaimer: ik heb geen handleiding voor het ineensteken van het autootje met de custom PCB. Ik heb namelijk zelf de custom PCB niet laten maken en kan dus niet demonsteren hoe dit moet.

Stap 0)

Zorg dat je al de benodigdheden klaar hebt zodat u in een keer alles kan maken.

Stap 1)

Neem uw dc-motoren, u zult zien dat er twee kleine koperen loops zijn, aan deze soldeert u een rode en een zwarte draad, zodat u een plus en een min kan bepalen. Dit is belangrijk voor de aansluitingen later dat u dit juist doet.

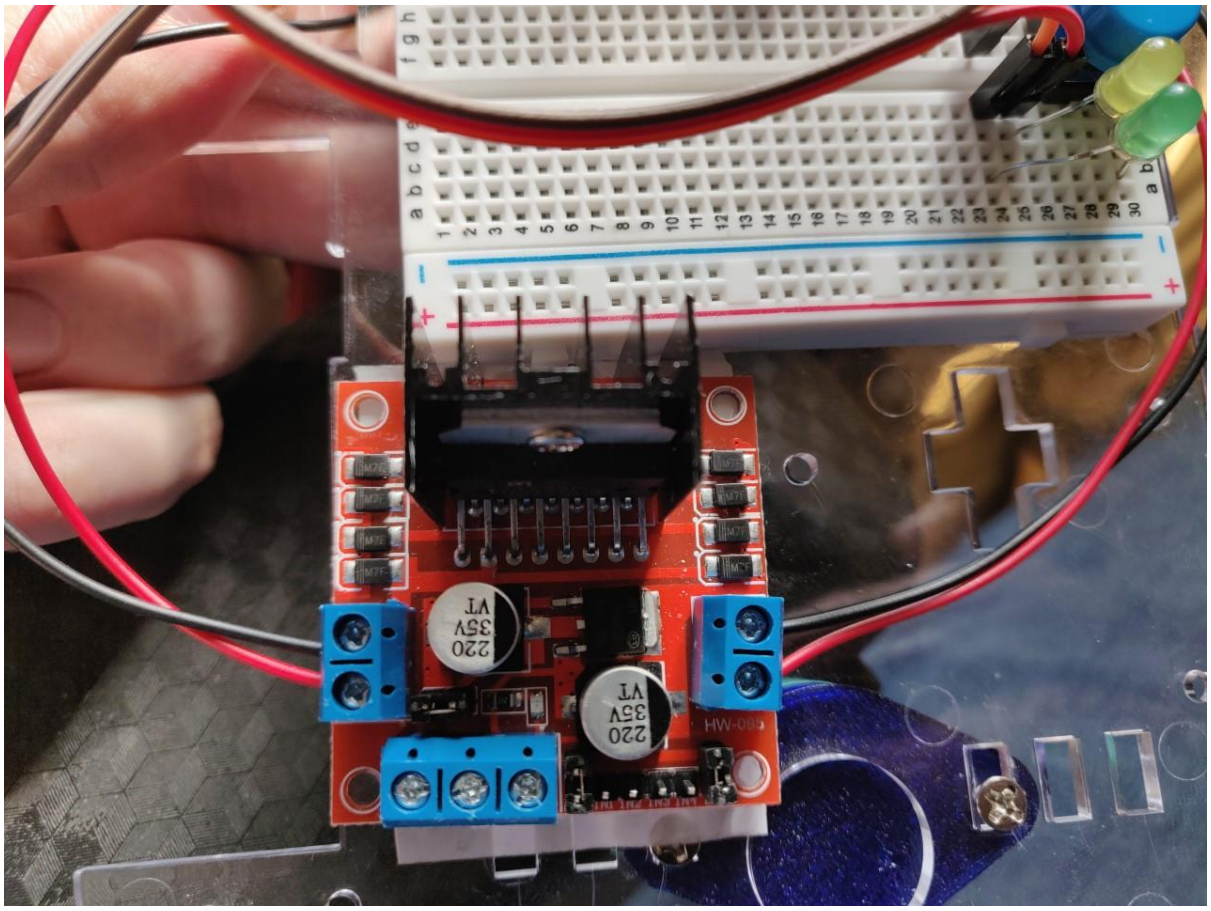


Stap 2)

Monteer uw motoren op uw chassis op uw methode, ik heb ze onderaan mijn plastiek chassis gemonteerd zodat de auto niet gekanteld stond door het zwenkwiel, u mag nu ook al het zwenkwieltje monteren.

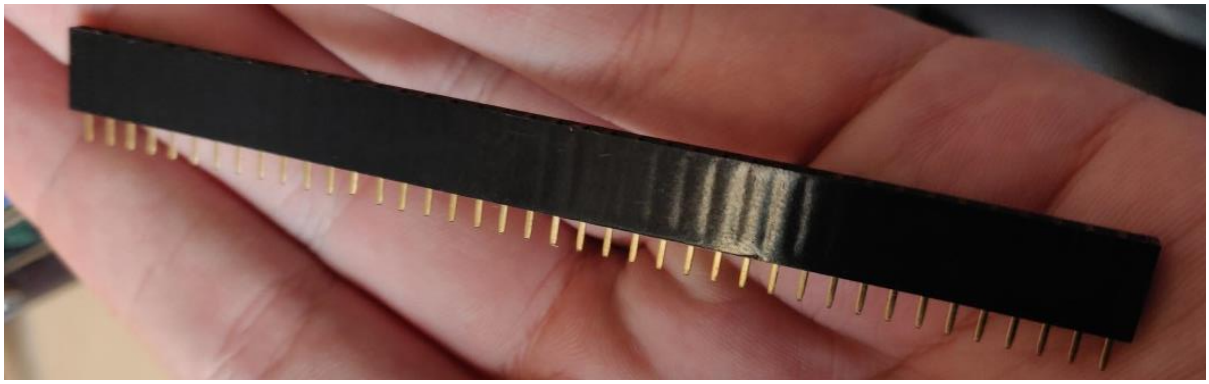
Stap 3)

Verbind de vers gesoldeerde dc-motoren met de L298N op volgende wijze:

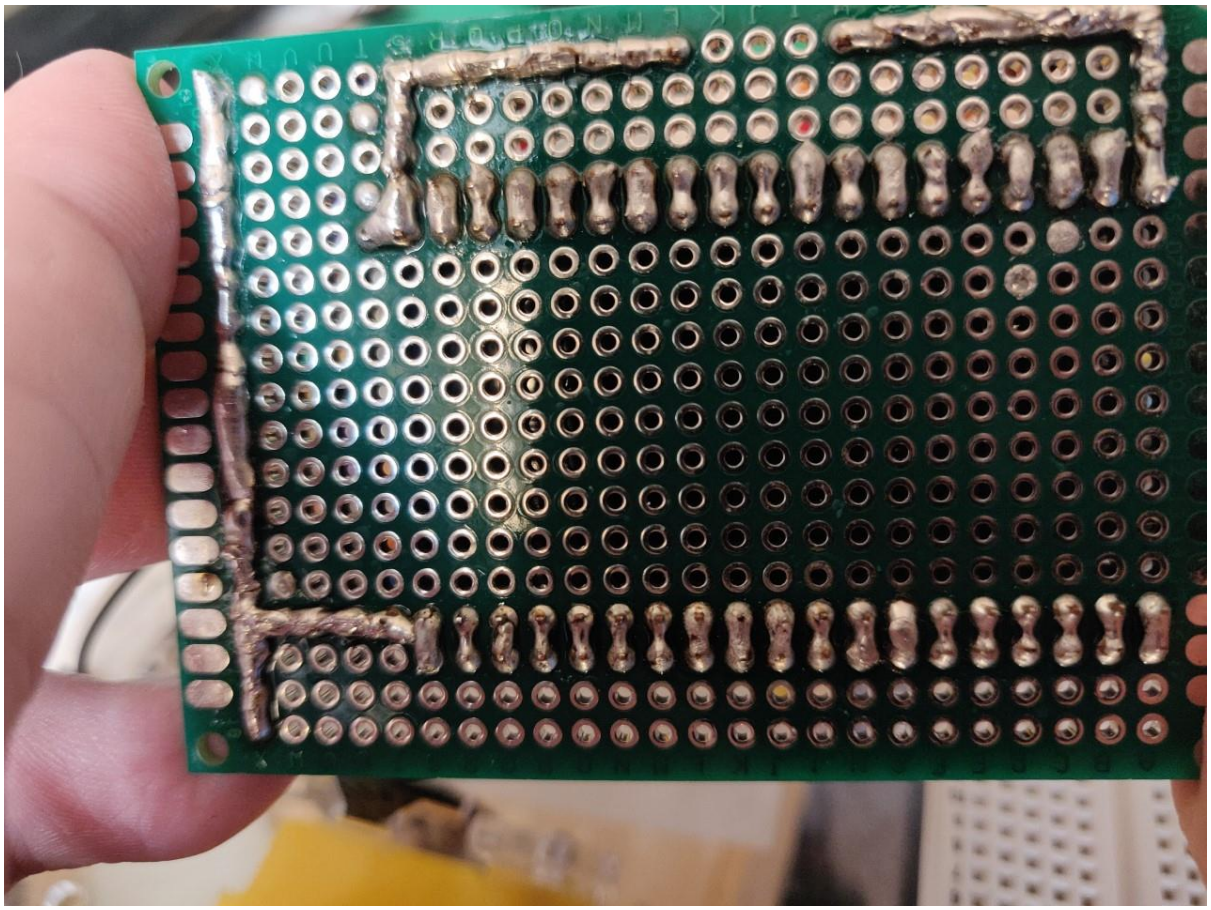


Stap 4)

Neem de blanco 5x7cm PCB en neem enkele van de volgende pinnen. Zie dat u 4 strookjes heeft van 19 pinnen lang elk, en dan nog een strookje van 18 lang en dan nog eens twee van 5 tot 7 lang.



Soldeer deze vast zodat u dit patroon krijgt:



Stap 5)

Monteer uw IR sensoren en HC-SR04 vooraan op uw wagen en plaats uw PCB hier achter (zet deze vast met dubbelzijdige plakband als u geen passende bouten heeft), met genoeg ruimte tussen de sensoren en het PCB voor kabeltjes te buigen waar nodig.

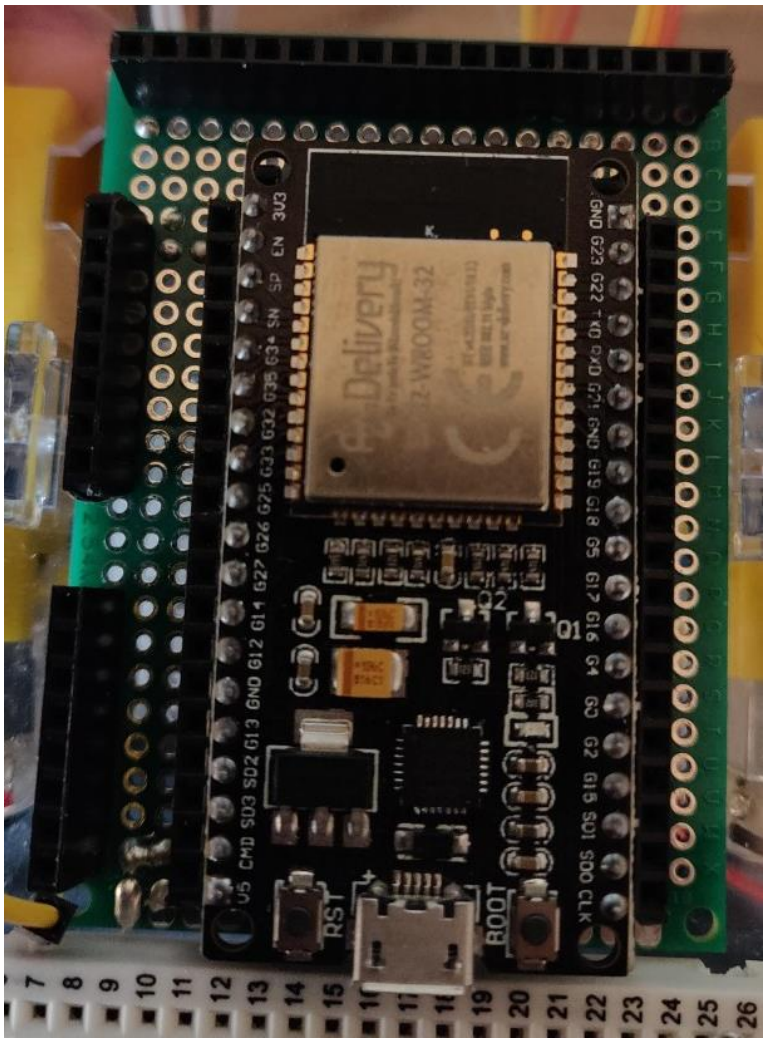
Om deze sensoren vast te steken heeft u bouten en moeren, zipties of vijzen nodig.

Stap 6)

Achter uw PCB kan u uw breadboardje plakken voor uw knop en LEDs op te monteren. U kunt hier ook ineens uw powerbank plaatsen voor gewichtsdistributie zolang dat dit genoeg plaats laat voor uw L298N te plaatsen.

Stap 7)

Plug uw 38-pin ESP32 in uw PCB georiënteerd als volgende:



Ga naar Deel 3 – Schema's en sluit de bekabeling aan zoals aangegeven op dat schema.

Stap 8)

Open uw webbrowser en installeer Arduino.exe als u dit nog niet geïnstalleerd heeft, en installeer de ESP32 libraries.

Navigeer naar https://github.com/fstroobandt/IR_line_follower hier downloadt u "Week_7_IoT.ino" als u een wilt met WiFi en MQTT, zo niet dan downloadt u de andere .ini file

Stap 9) – enkel voor WiFi & MQTT

Open Week_7_IoT.ino en pas hier de code aan om te passen bij uw WiFi netwerk en uw MQTT-netwerk, hier past u ook aan welke kleur lijn u gaat volgen, u kan kiezen tussen zwart en wit.

Stap 10) – enkel voor WiFi & MQTT

Zet uw MQTT-server aan en zorg dat het controlecentrum dat u gebruikt, en dat de knop die u wilt gebruiken het volgende bericht zenden om de auto voort te laten rijden "command/continue".

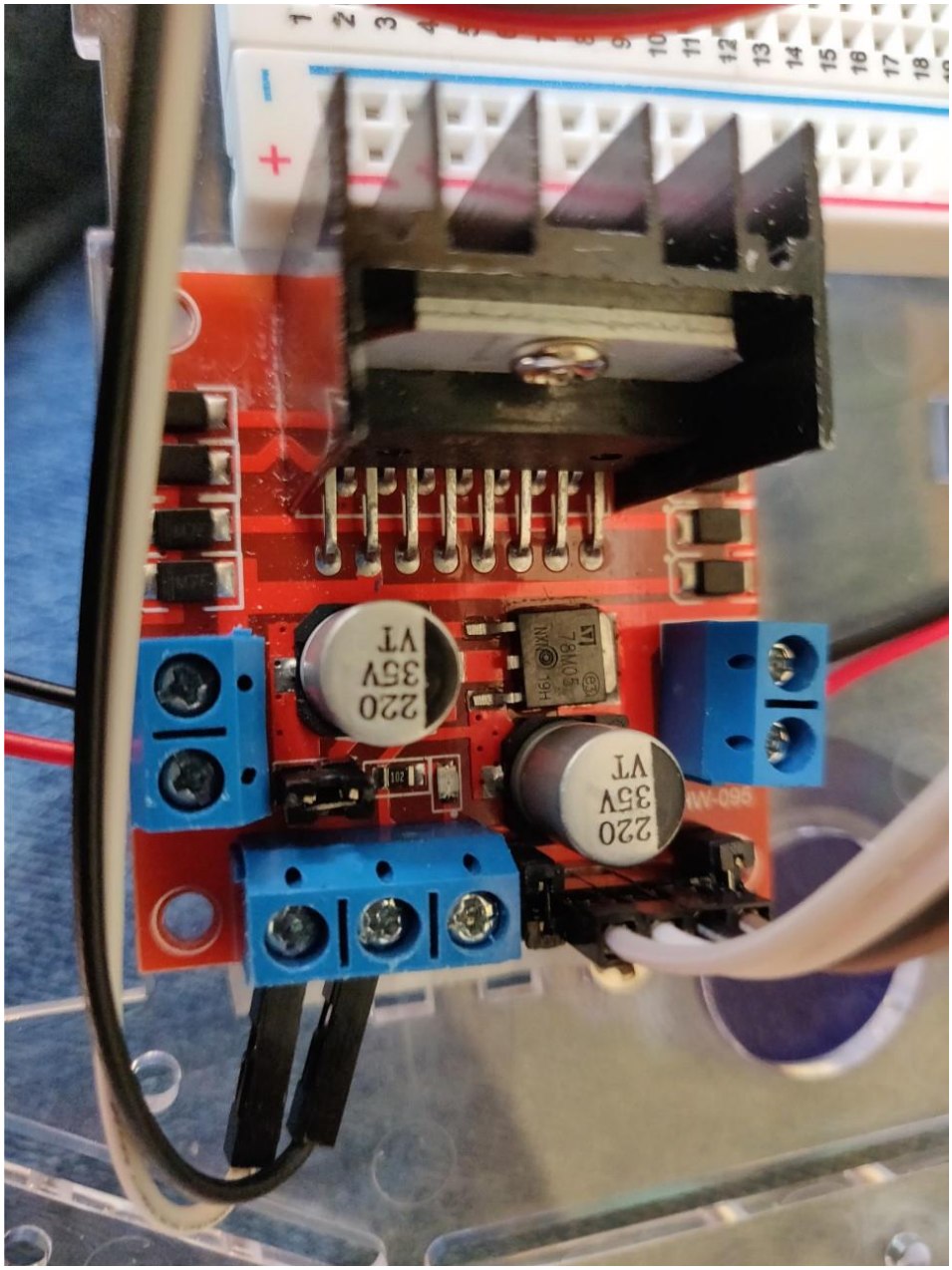
Stap 11)

Verbind uw ESP32 met uw computer via een microUSB naar USB kabel die data kan schrijven. Dan selecteerd u in het .ino bestand onder tools “NodeMCU-32S” als microcontroller. Dan uploadt u de code.

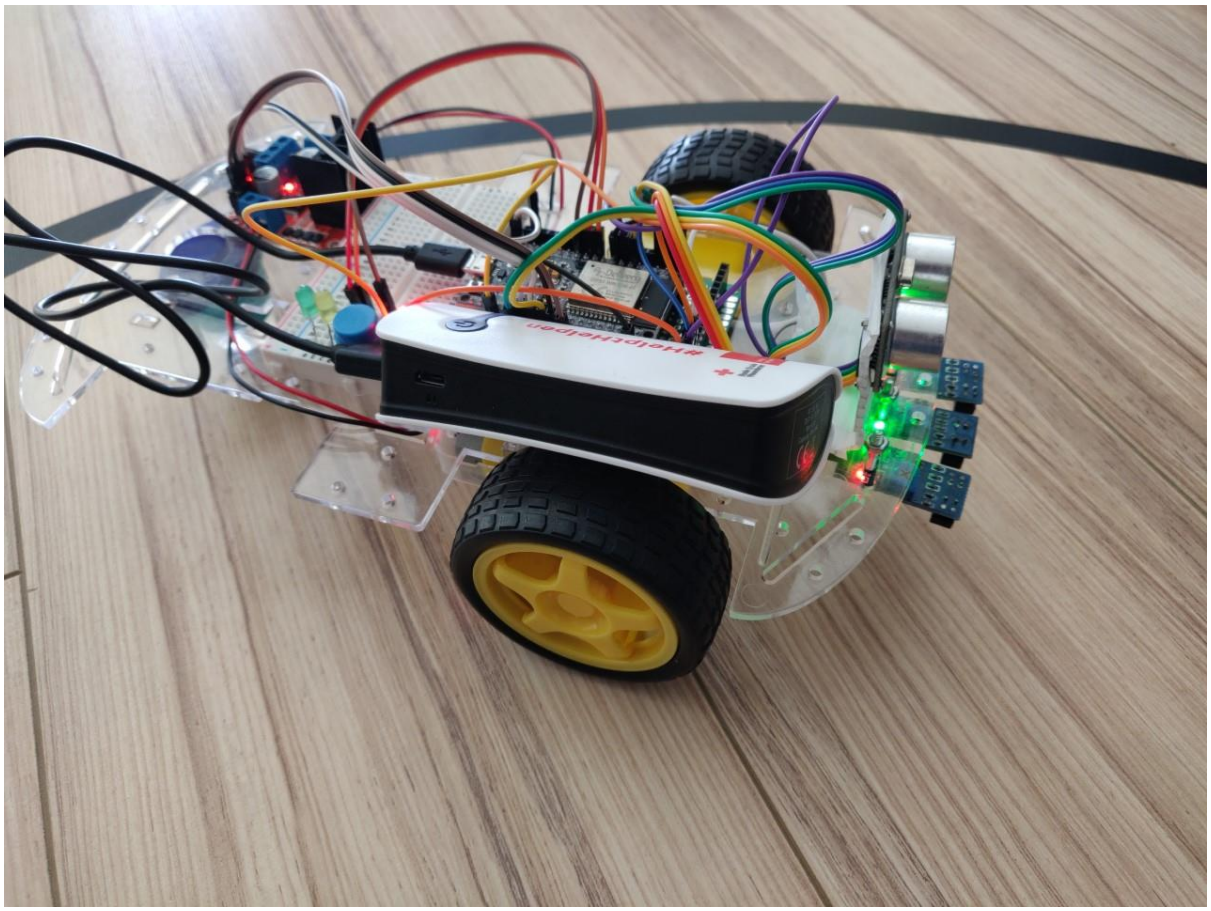
Stap 12)

Eenmaal de code is geüpload kan u de ESP32 loskoppelen van de computer. U moet nog één kabeltje trekken normaalgezien. Daarna is het uw powerbank verbinden met uw ESP32 en deze zal de kleur lijn volgen die u heeft gekozen*. De zwarte kabel leid naar GND van de ESP32 en de witte/grijze naar de 5V lijn. – deze voeden de L298N (en de motoren).

* Het kan zijn dat u nog uw IR-sensoren moet afstellen voor de donkerheid/lichtheid van uw vloer.



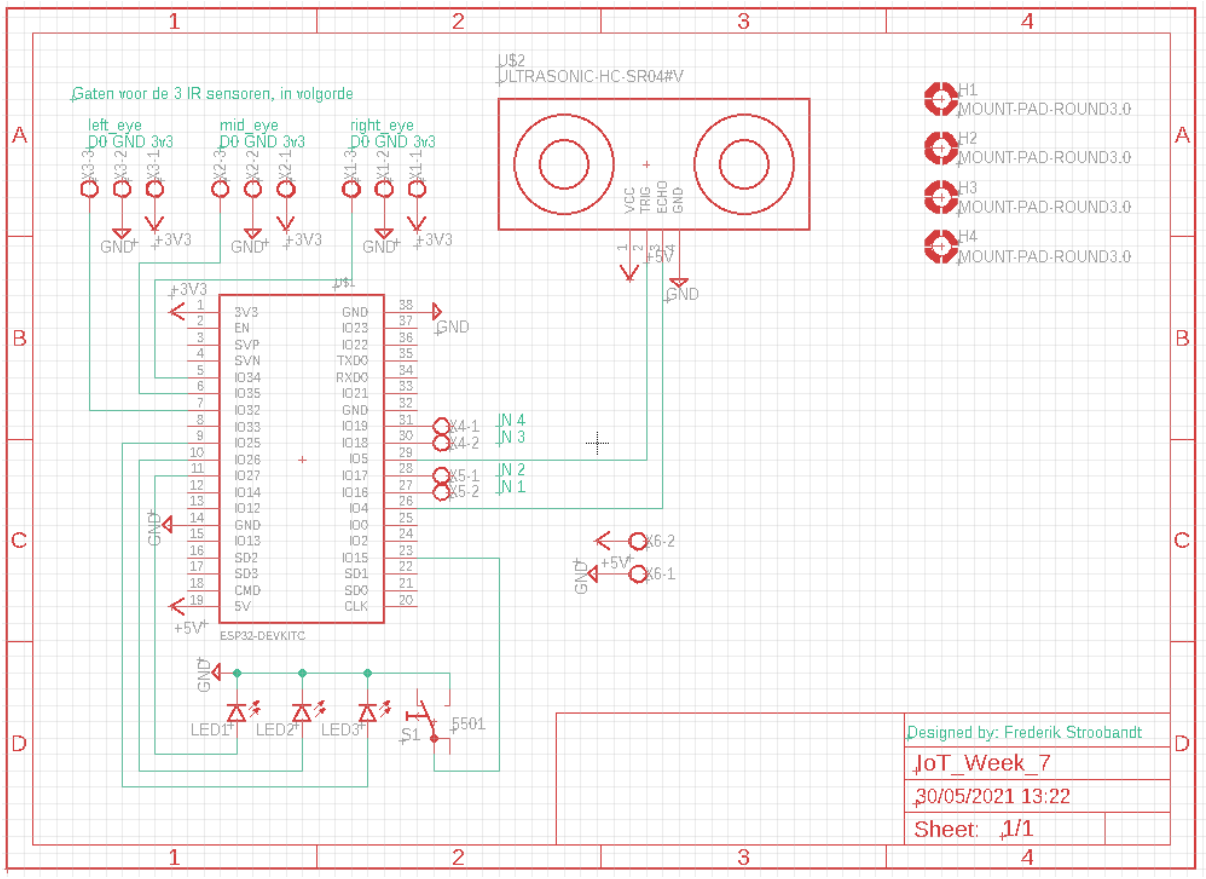
Dit zou gelijkaardig moeten zijn aan uw eindresultaat.



3 Schema’s

Electronisch schema

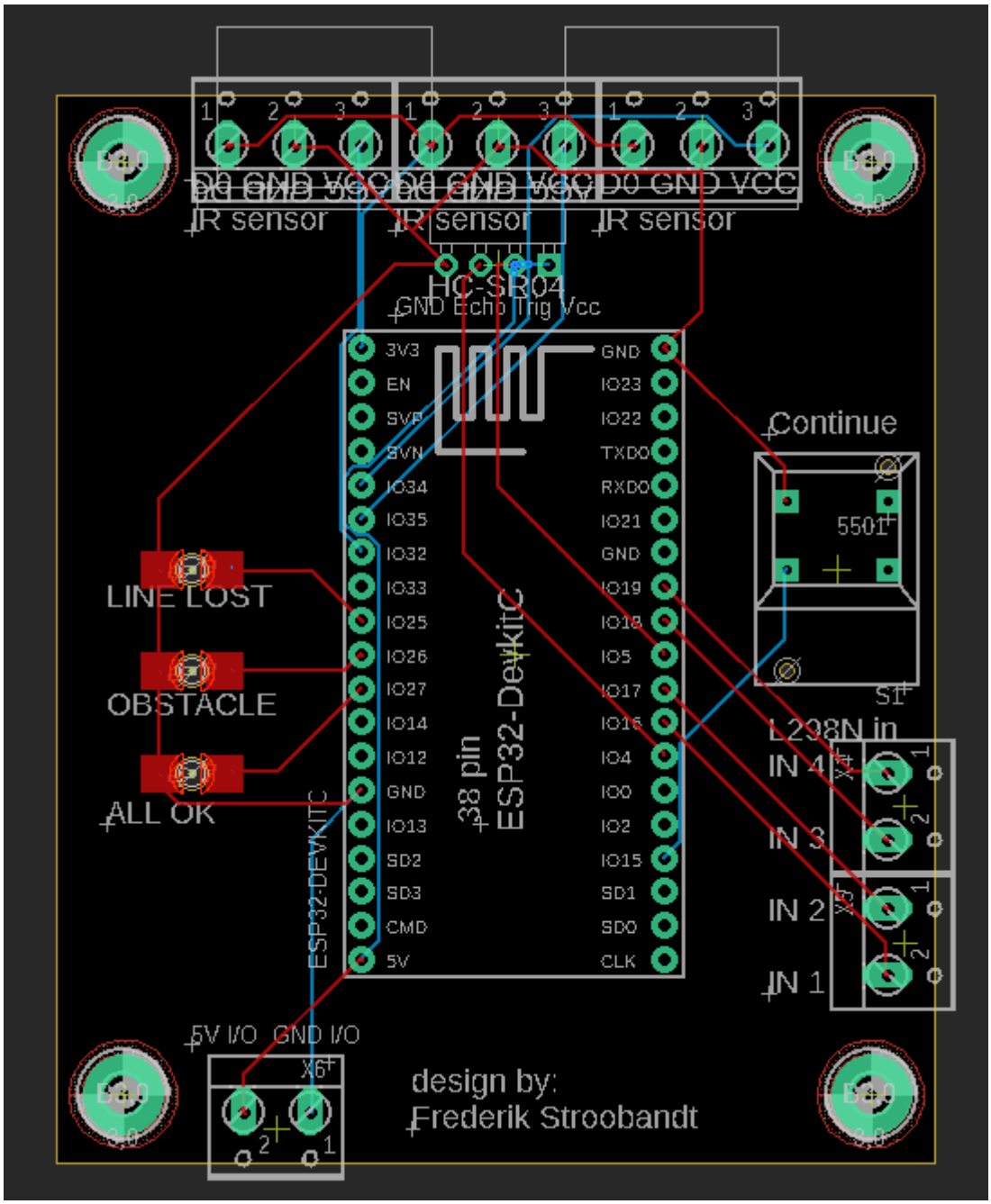
Dit elektronisch schema laat enkel zien wat er op de PCB wordt aangesloten.



Ik kan hierop de dc-motoren niet laten zien gezien deze op de L298N worden aangesloten en deze wordt dan op keer aangesloten op dit schema via Screwterminals (zie IN 4, IN 3, IN 2 en IN 1) De IR-sensor aansluitingen zijn hier vervangen door drie 3-poortige screwterminals. Dit heb ik gedaan zodat u male-to-female jumpers kan gebruiken om aan te sluiten en omdat er geen goede library te vinden was voor de IR sensoren die ik gebruik.

PCB

De PCB meet 67.31 mm bij 86.34 mm, een zeer compacte PCB.



Het PCB is mogelijks niet volledig leesbaar. Dit komt omdat ik zowel op de onderkant als bovenkant tekst laat printen.

U ziet ook dat de ESP32 nog veel pinnen over heeft, dit wil zeggen dat ik deze eigenlijk nog compacter zou kunnen maken als ik een kleinere microcontroller had gebruikt. Ik heb ook expres de I2C pinnen niet gebruikt zodat hier op nog uitbreiding kunnen komen zo gewenst.

Natuurlijk ziet u hierop geen aansluitingen voor de IR-ogen op het eerste gezicht, dit is omdat hiervoor geen libraries bestonden, want de ogen zijn al PCBs, net zoals de L298N motorcontroller die wij gebruiken. De plaatsen om deze aan te sluiten staan vermeld op de PCB zelf. Voor de L298N heb je de screwterminals rechtsonder en de screwterminal linksonder voor stroom. Voor de 3 IR-sensoren zijn er drie keer drie punten voorzien onder de HC-SR04 Ultrasone module. Ik adviseer hier de draden rechtstreeks op te solderen in plaats van de screwterminals te installeren, deze screwterminals zouden te hoog zijn als u de HC-SR04 rechtstreeks op de PCB wilt solderen.

4 Link naar de GitHub pagina

https://github.com/fstroobandt/IR_line_follower