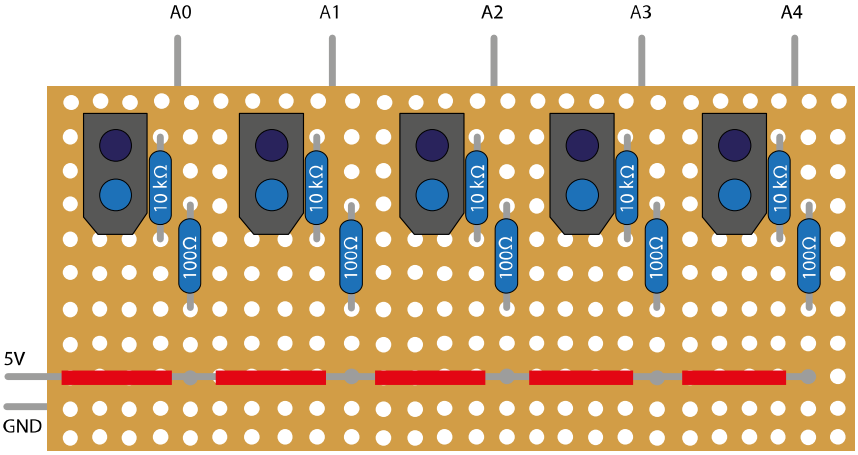
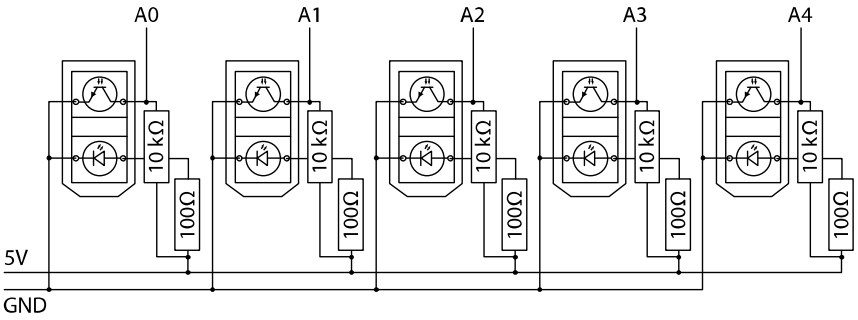
Robino-Maze-Solver

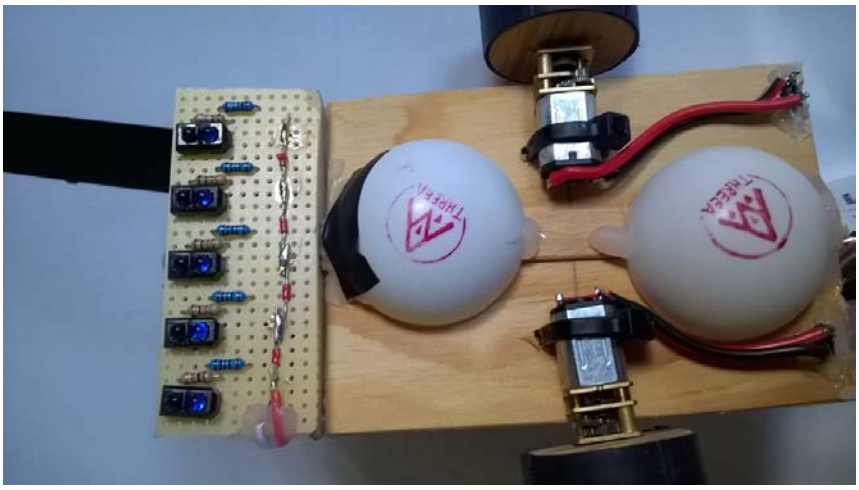
Zum Bau des „Robino-Maze-Solvers“ werden ein Robino-Fahrgestell, 5 Sensor-Module vom Typ TCRT5000, je 5 Widerstände von 100 Ohm und 10 Kiloohm sowie Kabel, Heißkleber und eine Lötstation benötigt. Die hier verwendete Lochrasterplatte kann durch ein Stück Pappe ersetzt werden.
Sensoren und Widerstände werden wie folgt angeordnet und verdrahtet:



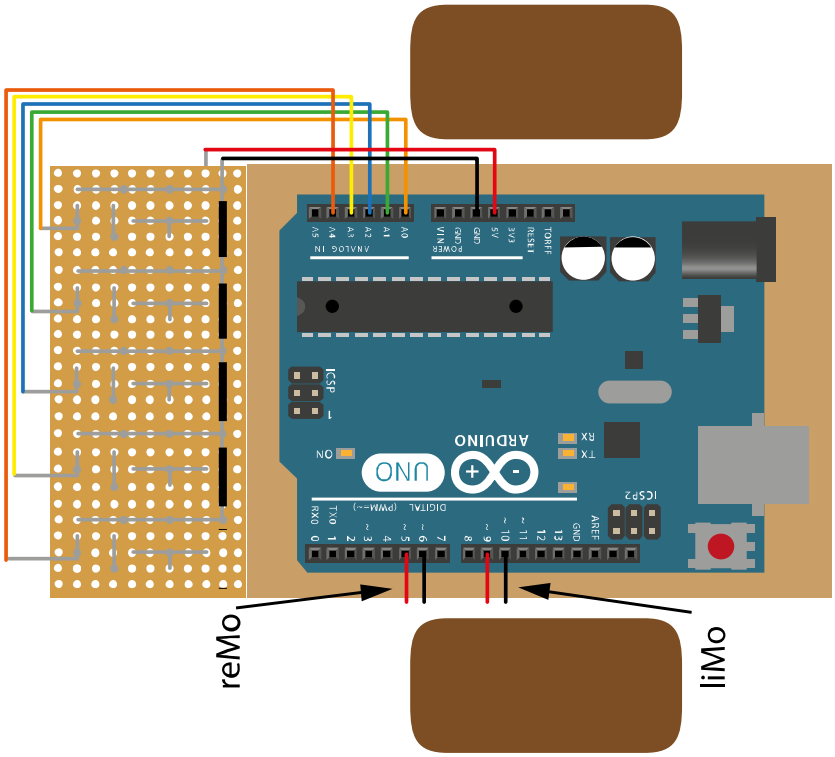
Hier der dazu passende Schaltplan:



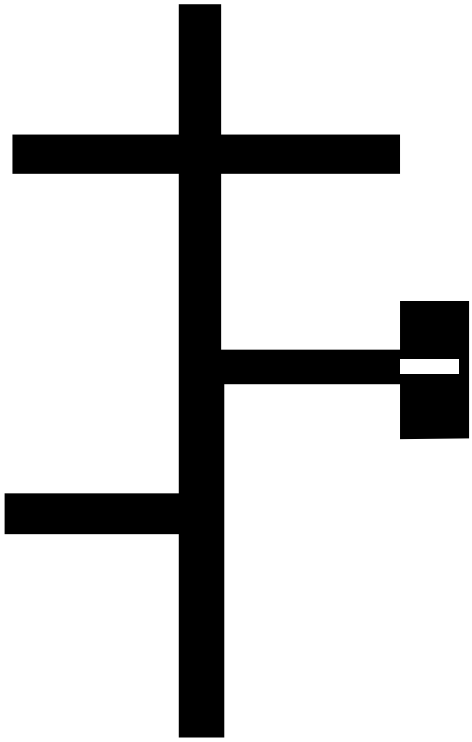
Und so sieht der „Robino-Maze-Solver“ real aus von der Unterseite betrachtet:



Die Anschlüsse für die Sensoren und Motoren können aus der folgenden Schemazeichnung und dem weiter unten befindlichen Programmcode entnommen werden:



Zum Erproben eignet sich das abgebildete Labyrinth, für das eine ca. 100 x 150 cm große Platte benötigt wird, auf die die Pfade geklebt werden können mit mattschwarzem Klebeband (z. B. Tesa 4651, schwarz, 19 mm) oder mit etwa gleich breiten Streifen aus schwarzem Tonpapier.



Die Programmierung nutzt die „Linke-Hand-Regel“. Sie besagt, dass man in einem Labyrinth entweder das Ziel oder zurück zum Ausgangspunkt findet, wenn man beim Gehen mit einer Hand (hier also die linke) Kontakt zur Wand hält.

Programmcode:

```
#define liMo_1   10   // linker Motor Anschluss 1 - PWM-Pin
#define liMo_2   11   // linker Motor Anschluss 2 - PWM-Pin
#define reMo_1   5    // rechter Motor Anschluss 1 - PWM-Pin
#define reMo_2   6    // rechter Motor Anschluss 2 - PWM-Pin

void setup() {
 Serial.begin(9600);
 pinMode(reMo_1, OUTPUT);
 pinMode(reMo_2, OUTPUT);
 pinMode(liMo_1, OUTPUT);
 pinMode(liMo_2, OUTPUT);
}

void loop() {
 sensor();
 driver();
}

void driver()      // Hier werden die Sensoren ausgelesen und die Motoren entsprechend der empfangenen Werte gesteuert
{
 int lowVal = 60;        // Erfahrungswert, der auch kleiner sein kann
 int highVal = 150;       // Erfahrungswert, der auch größer sein kann

 int s0 = analogRead(A0);          // Auslesen der 5 Sensoren an den Pins A0 bis A4
 int s1 = analogRead(A1);
 int s2 = analogRead(A2);
 int s3 = analogRead(A3);
 int s4 = analogRead(A4);

 if ((s0 <= lowVal) && (s1 <= lowVal) && (s2 >= highVal) && (s3 <= lowVal) && (s4 <= lowVal))   // 0 0 1 0 0 - Der schwarzen Linie folgen
 {
  analogWrite(reMo_1, 200);
  analogWrite(reMo_2, 0);
  analogWrite(liMo_1, 200);
  analogWrite(liMo_2, 0);
 }
 else if ((s0 <= lowVal) && (s1 <= lowVal) && (s2 >= highVal) && (s3 >= highVal) && (s4 <= lowVal)) // 0 0 1 1 0 - Richtungskorrektur bei Abweichung nach links
 {
  analogWrite(reMo_1, 0);
  analogWrite(reMo_2, 0);
  analogWrite(liMo_1, 200);
  analogWrite(liMo_2, 0);
 }
 else if ((s0 <= lowVal) && (s1 >= highVal) && (s2 >= highVal) && (s3 <= lowVal) && (s4 <= lowVal)) // 0 1 1 0 0 - Richtungskorrektur bei Abweichung nach rechts
 {
  analogWrite(reMo_1, 200);
  analogWrite(reMo_2, 0);
  analogWrite(liMo_1, 0);
  analogWrite(liMo_2, 0);
 }
 else if ((s0 >= highVal) && (s1 >= highVal) && (s2 >= highVal) && (s3 <= lowVal) && (s4 <= lowVal)) // 1 1 1 0 0 - Linksdrehung bei Abzweigung links
 {
  analogWrite(reMo_1, 200);
  analogWrite(reMo_2, 0);
  analogWrite(liMo_1, 0);
  analogWrite(liMo_2, 0);
 }
 else if ((s0 >= highVal) && (s1 >= highVal) && (s2 >= highVal) && (s3 >= highVal) && (s4 >= highVal)) // 1 1 1 1 1 - Linksdrehung an Kreuzung oder Einmündung
 {
  analogWrite(reMo_1, 200);
  analogWrite(reMo_2, 0);
  analogWrite(liMo_1, 0);
  analogWrite(liMo_2, 0);
 }
 else if ((s0 >= highVal) && (s1 >= highVal) && (s2 <= low Val) && (s3 >= highVal) && (s4 >= highVal)) // 1 1 0 1 1 - Stopp
 {
  analogWrite(reMo_1, 0);
  analogWrite(reMo_2, 0);
  analogWrite(liMo_1, 0);
  analogWrite(liMo_2, 0);
 }
 else if ((s0 <= lowVal) && (s1 <= lowVal) && (s2 <= lowVal) && (s3 <= lowVal) && (s4 <= lowVal)) // 0 0 0 0 0 - Kehrtwendung nach rechts wenn Linie endet
 {
  analogWrite(reMo_1, 0);
  analogWrite(reMo_2, 200);
  analogWrite(liMo_1, 200 );
  analogWrite(liMo_2, 0);
 }
}

void sensor() {            // Sensorkontrolle
 int s0 = analogRead(A0);
 int s1 = analogRead(A1);
 int s2 = analogRead(A2);
 int s3 = analogRead(A3);
 int s4 = analogRead(A4);
```

```
  Serial.print("s0 ");    // Anzeige der gelesenen Werte im seriellen Monitor
  Serial.print(s0);
  Serial.print(" s1 ");
  Serial.print(s1);
  Serial.print(" s2 ");
  Serial.print(s2);
  Serial.print(" s3 ");
  Serial.print(s3);
  Serial.print(" s4 ");
  Serial.print(s4);
  Serial.print(' ');
  Serial.println();
  //delay(500);
}
```