Verslag IPOMEDT

GREEKGODX (Groep 14)

Delano Teske	S1107904
Stefan de Keijzer	S1106980
Bart van der Lans	S1107097
Roy van Dijk	S1106628

Bouw

We zijn als team gesplitst begonnen. Het bouwen van de structuur en het aanleggen van de draadjes hebben we apart gedaan. Zo hebben wij voorkomen dat het hele team zit te kijken naar één persoon die de robot bouwt. Met de bouw zijn we begonnen met het schuren van de scherpe randjes van het frame, vervolgens het bevestigen van de sensor en het aanleggen van de spacers.

Na de structuur werden de printplaatjes, motors en Pi in de behuizing gemonteerd. Deze hebben we apart bedraad (vóór het inbouwen) volgens het schema op de website.

Programmeren

Voor de code zijn we (uiteraard) begonnen met Google. Hier hebben we voorbeeldcode gevonden van een stappenmotor. Dit is de basis van onze code geworden maar met alleen deze code werkte het nog niet. De juiste pin-nummers moesten nog worden aangegeven in de code en de code deed letterlijk alleen het aandrijven van een enkele motor in één richting. Ook draaide de motor erg langzaam waardoor we hebben moeten zoeken hoe dit sneller kon.

Na het weten aan te drijven van een enkele motor dachten we de tweede motor op precies dezelfde manier aan te drijven. Dit werkte natuurlijk niet omdat de motors in spiegelbeeld ten opzichte van elkaar zijn gemonteerd. De tweede motor moet dus eigenlijk andersom draaien. Dit hebben we simpel opgelost door handmatig de sequence om te draaien.

Daarna zijn er een aantal beslissingen die moeten worden genomen tijdens het uitvoeren van de loop. Rechtsaf (linker sensor ziet wit, rechter ziet zwart), Linksaf (linker sensor ziet zwart, rechter wit) of een kruispunt, waar alle sensors zwart zullen zien. In alle andere gevallen gaat de robot gewoon rechtdoor.

In de functies voor rechts- en linksaf zijn we begonnen met alleen het tegenovergestelde wiel te draaien. Bijvoorbeeld: bij een bocht naar rechts gaat alleen het linker wiel draaien. Dit werkte echter niet op scherpe bochten omdat de robot van de lijn af gaat. Om dit op te lossen hebben we gezorgd dat in aanvulling tot het linker wiel, ook het rechter wiel andersom zou draaien en vice versa. Op deze manier krijgen we een scherpere bocht.

Als er een kruispunt wordt gedetecteerd (alle sensors op 0), hebben we ervoor gezorgd dat de robot als het ware de sensors even “negeert”. We laten de robot dan handmatig afslaan met een voorgeprogrammeerde functie. De richting waar de robot op draait is uiteraard door de gebruiker aan het begin aangegeven. Een stopteken is voor de robot precies hetzelfde als een kruispunt, maar er is maar één kruispunt per baan. Om deze reden hebben we ervoor gekozen om gewoon te stoppen bij de tweede detectie van een kruispunt.

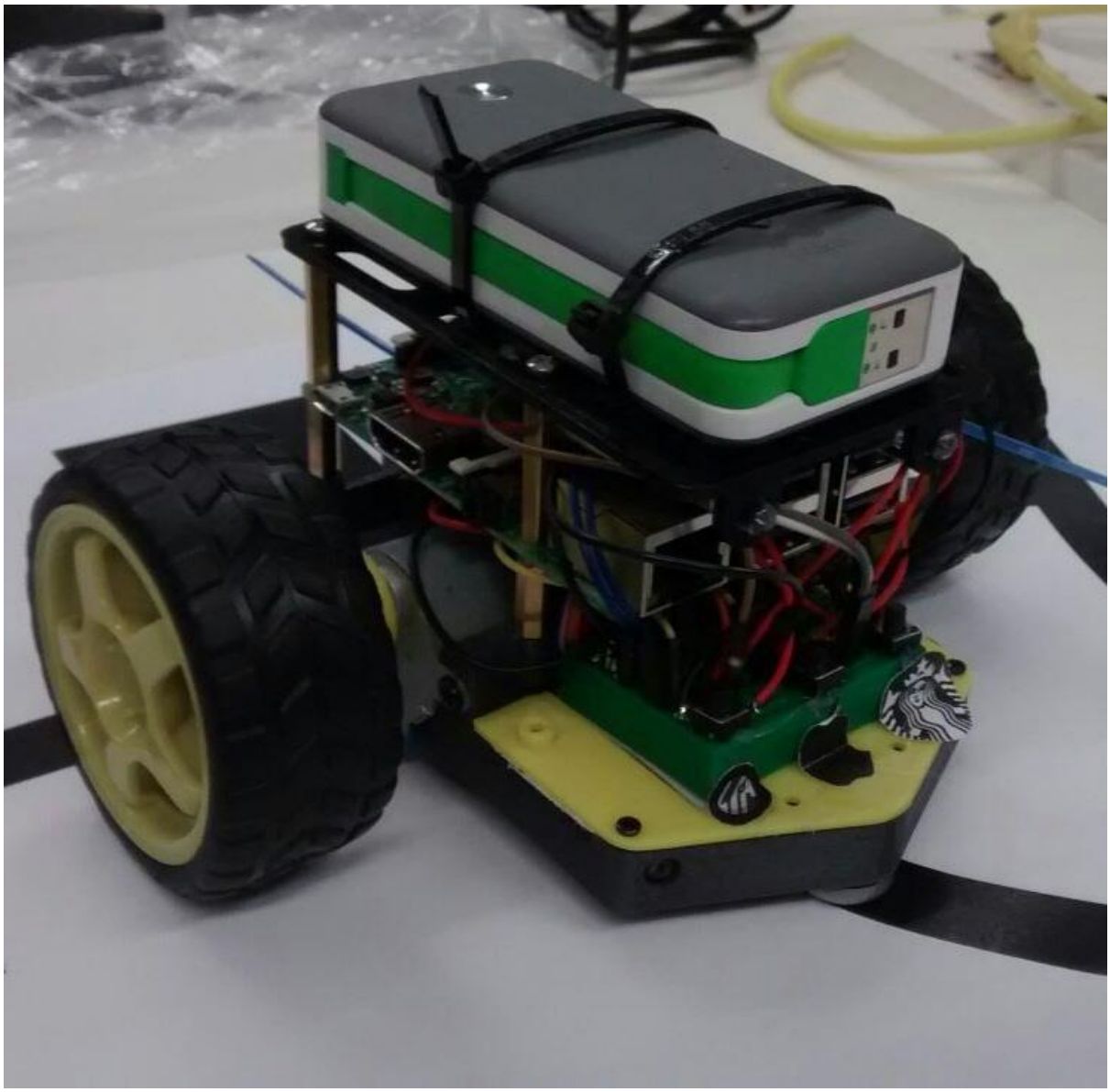
Afwerking

Na het programmeren van alle logica hebben we ook nog LED-lampjes toegevoegd. Voor gebruikersfeedback. De lampjes werken als het ware als een knipperlicht op een auto. Als beide lampjes knipperen gaat de robot rechtdoor. Als de robot stopt knipperen alle lampjes.

Deze feedback bleek echter niet genoeg te zijn en hierom hebben we een web interface aan de robot gekoppeld om de gebruiker feedback te geven.

Resultaat

Onze definitieve Pi-zzabot!



Code

```
import time
import RPi.GPIO as GPIO
import threading
import urllib.request

#Geen Waarschuwingen en juiste schema voor pins
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#GPIO pins motoren
motor1 = [2,3,4,17]
motor2 = [27,22,10,9]

#GPIO PINS sensoren en knoppen
sensorl = 14
sensorm = 15
sensorr = 18
knop    = 26
knopl   = 6
knopm   = 19
knopr   = 13
lichtRA = 5
lichtRV = 12
lichtLA = 25
lichtLV = 24

#Lichtrichting en richting begin waardes
lichtrichting = "vooruit"
richting      = "vooruit"

#Standaardwaarde lichtknipperen
lichtknipper = True
lichtenaan   = True

#Sequences
SeqVoorAchter1 = [[1,0,0,0],[1,1,0,0],[0,1,0,0],[0,1,1,0],[0,0,1,0],[0,0,1,1],[0,0,0,1],[1,0,0,1]]
SeqVoorAchter2 = [[0,0,0,1],[0,0,1,1],[0,0,1,0],[0,1,1,0],[0,1,0,0],[1,1,0,0],[1,0,0,0],[1,0,0,1]]

#Lengte van sequence en controle of beide even groot zijn
stepcountVoorAchter1 = len(SeqVoorAchter1)
stepcountVoorAchter2 = len(SeqVoorAchter2)
stepcountvoorAchter  = 0

if (stepcountVoorAchter1 == stepcountVoorAchter2):
    stepcountVoorAchter = stepcountVoorAchter1

#Vertragingsvariabele voor motors
delay = 2/float(1000)

#Pinnen van motoren op outputs zetten en op 0
for pin in motor1:
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, 0)

for pin in motor2:
    GPIO.setup(pin, GPIO.OUT)
    GPIO.output(pin, 0)
```

```
#GPIO pinnen op inputs zetten
GPIO.setup(sensorl, GPIO.IN)
GPIO.setup(sensorm, GPIO.IN)
GPIO.setup(sensorr, GPIO.IN)
GPIO.setup(knop, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(knopl, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(knopr, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(knopm, GPIO.IN, GPIO.PUD_UP)

#GPIO pinnen op outputs zetten
GPIO.setup(lichtRA,GPIO.OUT)
GPIO.setup(lichtRV,GPIO.OUT)
GPIO.setup(lichtLA,GPIO.OUT)
GPIO.setup(lichtLV,GPIO.OUT)

#Juiste lichtjes aan
def lichtjesuit() :
    GPIO.output(lichtRA, GPIO.LOW)
    GPIO.output(lichtRV, GPIO.LOW)
    GPIO.output(lichtLA, GPIO.LOW)
    GPIO.output(lichtLV, GPIO.LOW)

def lichtjesLinks() :
    GPIO.output(lichtRA, GPIO.LOW)
    GPIO.output(lichtRV, GPIO.LOW)
    GPIO.output(lichtLA, GPIO.HIGH)
    GPIO.output(lichtLV, GPIO.HIGH)

def lichtjesAan() :
    GPIO.output(lichtRA, GPIO.HIGH)
    GPIO.output(lichtRV, GPIO.HIGH)
    GPIO.output(lichtLA, GPIO.HIGH)
    GPIO.output(lichtLV, GPIO.HIGH)

def lichtjesRechts() :
    GPIO.output(lichtRA, GPIO.HIGH)
    GPIO.output(lichtRV, GPIO.HIGH)
    GPIO.output(lichtLA, GPIO.LOW)
    GPIO.output(lichtLV, GPIO.LOW)

#Lichtjes uitzetten
lichtjesuit()

def rij(motor, seq) :
    GPIO.output(motor[0], seq[0])
    GPIO.output(motor[1], seq[1])
    GPIO.output(motor[2], seq[2])
    GPIO.output(motor[3], seq[3])

def vooruit(step) :
    global motor1
    global motor2
    global lichtrichting
    #Beide motortjes een andere richting draaien
    rij(motor1, SeqVoorAchter1[step])
    rij(motor2, SeqVoorAchter2[step])
    lichtrichting = "vooruit"

def rechts(step):
    global motor1
    global motor2
    global lichtrichting
    #Beide motortjes dezelfde richting draaien
    rij(motor1, SeqVoorAchter1[step])
    rij(motor2, SeqVoorAchter1[step])
    lichtrichting = "rechts"
```

```
def links(step):
    global motor1
    global motor2
    global lichtrichting
    #Beide motortjes dezelfde richting draaien
    rij(motor2, SeqVoorAchter2[step])
    rij(motor1, SeqVoorAchter2[step])
    lichtrichting = "links"

def draaiwiel(richting, hoeveelheid):
    global delay
    global stepcountVoorAchter
    aantal = hoeveelheid * 256
    for step in range (0, aantal) :
        time.sleep(delay)
        if (richting == "links") :
            links(step % stepcountVoorAchter)
        if (richting == "rechts") :
            rechts(step % stepcountVoorAchter)
        if (richting == "vooruit") :
            vooruit(step % stepcountVoorAchter)

#Rijden van auto
def rijden():
    step           = 0
    kruispuntGehad = False
    while True:
        #Kruispunt gedetecteerd
        if (not GPIO.input(sensorl) and not GPIO.input(sensorm) and not GPIO.input(sensorr)) :
            #Controleren op tweede keer kruispunt
            if (kruispuntGehad) :
                #Informeer web interface over huidige status
                urllib.request.urlopen("http://127.0.0.1:3000/bestemmingbereikt/").read()
                #Tweede keer kruispunt dus stoppen
                break
            else :
                #Informeer web interface over huidige status
                urllib.request.urlopen("http://127.0.0.1:3000/kruispuntgevonden/").read()
                #3 if-statements voor handmatige bochten op kruising
                if (richting == "links") :
                    draaiwiel("vooruit", 3)
                    draaiwiel("links", 8)
                    draaiwiel("vooruit", 2)
                if (richting == "vooruit") :
                    draaiwiel("vooruit", 2)
                if (richting == "rechts") :
                    draaiwiel("vooruit", 3)
                    draaiwiel("rechts", 8)
                    draaiwiel("vooruit", 2)
                    #Informeer web interface over huidige status
                    urllib.request.urlopen("http://127.0.0.1:3000/onderwegnaarbestemming/").read()
                kruispuntGehad = True
        #Naar links gedetecteerd
        elif (not GPIO.input(sensorl) and GPIO.input(sensorr)) :
            links(step)
        #Naar recht gedetecteerd
        elif (GPIO.input(sensorl) and not GPIO.input(sensorr)) :
            rechts(step)
        #Niets gedetecteerd
        else  :
            vooruit(step)
        if(step == stepcountVoorAchter - 1):
                step = 0
        else :
                step = step + 1
        time.sleep(delay)
```

```
#Licht laten knipperen
def knipper() :
    global lichtrichting
    global lichtknipper
    global lichtenaan
    while lichtenaan:
        if (lichtknipper):
            lichtknipper = False
            if (lichtrichting == "links"):
                lichtjesLinks()
            if (lichtrichting == "rechts"):
                lichtjesRechts()
            if (lichtrichting == "vooruit"):
                lichtjesAan()
        else:
            lichtknipper = True
            lichtjesuit()
        time.sleep(1)

#Lichtknipper code apart laten draaien
lichtenknipperen = threading.Thread(target=knipper)
lichtenknipperen.start()

#Start code
#Informeer web interface over huidige status
urllib.request.urlopen("http://127.0.0.1:3000/wachtenopbestemming/").read()
while True:
    #Knop starten ingedrukt
    if (not GPIO.input(knop)) :
        kruispunt = False
        #Informeer web interface over huidige status
        urllib.request.urlopen("http://127.0.0.1:3000/onderwegnaarkruispunt/").read()
        rijden()
        lichtenaan = False
        break
    #Knop rechts ingedrukt
    elif (not GPIO.input(knopr)) :
        richting = "rechts"
        lichtenknipperen = True
        lichtjesRechts()
        #Informeer web interface over huidige status
        urllib.request.urlopen("http://127.0.0.1:3000/klaaromtestarten/").read()
    #Knop vooruit ingedrukt
    elif (not GPIO.input(knopm)) :
        richting = "vooruit"
        lichtenknipperen = True
        lichtjesAan()
        #Informeer web interface over huidige status
        urllib.request.urlopen("http://127.0.0.1:3000/klaaromtestarten/").read()
    #Knop links ingedrukt
    elif (not GPIO.input(knopl)) :
        richting = "links"
        lichtenknipperen = True
        lichtjesLinks()
        #Informeer web interface over huidige status
        urllib.request.urlopen("http://127.0.0.1:3000/klaaromtestarten/").read()
    lichtrichting = richting
    time.sleep(0.1)
```