Robotics Club Werenfridus

Manual MER caterpillar-pi

Ton Schuckman
Physics amanuensis/instructor

Tabor College
Werenfridus

Index

The manual is divided in three parts:

- Mechanical part
- Electronic circuit with connections
- Programming the microcomputer

Mechanical part

Rectangular baseplate

While drilling use a pair of safety glasses against flying pieces of metal. In case of a long hairstyle, tie your hair up at the back of your head to avoid entanglement with the rotating drill head.

The black arrow (figure 2) marks the upper and front side of the baseplate. The circular hole with a diameter of 15 mm is situated at the right side. Put the side plates centered in the middle under the baseplate at such a way that the motors are situated at the back. Turn the assembly upside down and mark the spots at the four places where the holes must be drilled. Next mark the holes for the microcomputer at the back of the baseplate, take a look at the demo for the exact position.

Drill the marked holes with a 3,0 mm drill. Sharp splinters can be removed by hand with a 10 mm drill. Don not assemble the parts yet, this will be done later.

Slots for the servo

Two slots are made at the front of the baseplate to facilitate the servo. This is not possible when assembled. Use a metal saw and make a slot measured 30 mm from the left side with a depth of 6 mm. Do the same measured 40 mm from the right side. When this is done mount three M2.5 x 20 bolts for the microcomputer on the baseplate.
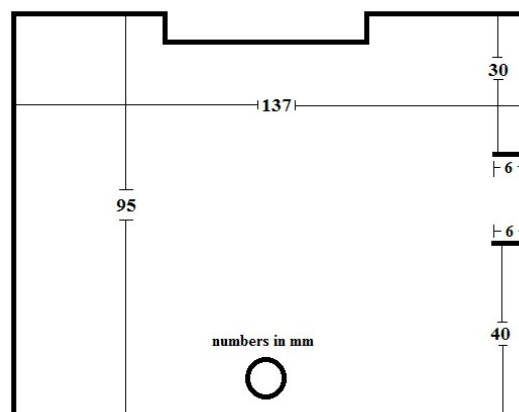

Figure 1. Slots for the servo

Middle wheels and side-plates

Put an m4 bolt trough the wheel and screw a nut against the inner site of the wheel. Then loosen it half a turn so that the wheel can rotate without friction. Now you can attach the wheels on the two side-plates with another m4 nut. Tighten them by turning the two nuts with two wrenches clockwise.

Mounting the motors

The motors are placed with the short bolts at the back of the side plates. The short bolts are essential; otherwise the gears inside the motor can be damaged. Screw gently the three bolts in the motor, this way damage of the screw-thread is avoided, then tighten them with a screw-driver. The driving wheel can be connected to the motor axis with a bolt. The bolt comes at the flat side of the motor axis. In some cases, the driving wheel most be assembled with the aluminum cylindrical piece.

Front wheels and caterpillar tracks

Sometimes the caterpillar track has to be adjusted to the right length. Put the caterpillar track around the not yet mounted front wheel and then around the rear driving wheel. Attach the front wheel the same way you attached the middle wheels.

Soldering the wires to the motors

Do not inhale the soldering smoke, it consists of heavy metals. Take care not to press the hot soldering iron against the plastic parts of the motors. In general: for the **+** wires use red and for the **–** a black wire. Cut the wires at the right length (20 cm) and solder on the left motor a red wire at the **+** eyelet and a black wire at the **–** eyelet. On the right motor it is reversed, a red wire at the **–** eyelet and a black wire at the **+** eyelet. For those who have never soldered: hold the end of the hot soldering iron for two seconds against the eyelet, which has already the wire in it, then push the soft solder at the same point and you will see the melting of the soft solder in the eyelet with wire. Then remove the soldering iron and wait a while to let it cool before you check the firmness of your connection. The other end of the wires are to be equipped with header pins.

Connecting the baseplate and side-plates

Now the two completed side-plates are placed under the baseplate with the 75 mm m3 bolts and nuts. Do this in such a way that the two motors are situated at the back of the baseplate. The 75 mm long bolts come in handy to assemble at a later point the roof beams to protect the electronics, fasten the power bank or when you want to expand your vehicle for example with a solar panel.

Assembling the roof beams

Saw two pieces of aluminum: (10 mm - 160 mm long). Drill four holes 38 mm from the end of the beams of 3,0 mm. Connect the two beams at the top of the 75 mm bolts. Fix the USB splitter with tape under the right beam. This way the electronics are protected and there is place for the power bank on top of it, and you can lift the robot from the ground for testing.

Placing the microcomputer

Attach three connectors or M2.5 bolts on the microcontroller at the positions of the holes in the baseplate. Then fasten the microcomputer with the connectors at the rear on the baseplate with M2.5 bolts or nuts. After this handle with care to avoid damage to the microcomputer! See figure 4.

The battery pack

First, put layer of duct-tape at the underside of the baseplate to avoid a short-circuit of the battery pack. Avoid to cover the holes. Cut a piece of rope at a length of 56 cm and knot it at the ends tightly together. Wrap this rope together with the USB splitter on the right roof beam, put it through the 15 mm hole and around the battery pack and up again on the left side of the baseplate. Put an elastic through the rope and wrap both ends of it at the ends of the 75 mm bolts/roof beams. The blank ends of the black and red wires from the battery plug are soldered at separate male headers. The six 1.5 volt AA make 9 volts together.

Electronic circuit

Circuit-board

The plug-in circuit-board is placed in front of the microcomputer. It has an adhesive tape at the underside. Cut with a knife a strip of 5 mm from the plastic cover of this adhesive tape. Otherwise you can't remove the circuit-board anymore without demolishing it. Now you can put it on the right spot on the baseplate. See also figure 4.
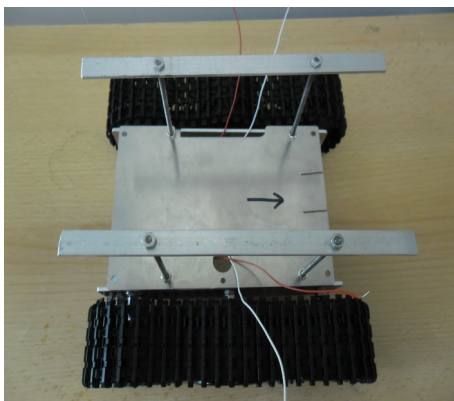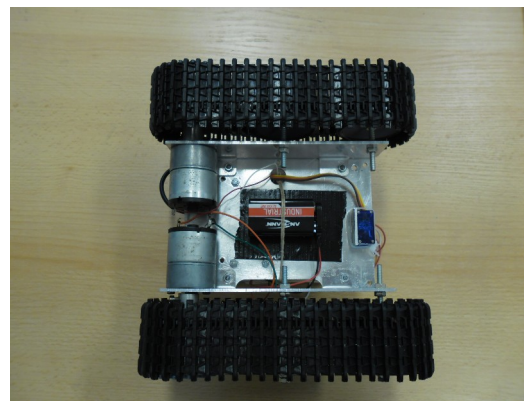

Figure 2. Upperside MER

Figure 3. Underside MER

Working of the electronic circuit - transistors

The microcomputer works, like most electronics, at a voltage of 3,3 or 5 volt, which is delivered by a USB adapter or power bank. The motors get power from the 9V battery pack. So the following scheme is going to happen: when through the program that you will write a voltage of 3,3 volt is put on pin 5, a current will flow through the resistor, decreased by it, to the basis B of the transistor. A transistor has three connections: b/basis, c/collector and e/emitter. Normally, when there is no voltage on the basis the transistor is closed. This is possible because the transistor contains a semi-conductor that will only conduct when a small current goes from the basis to the emitter. Only then the main current from the collector to the emitter can flow. The motor will run because an electrical circuit has been made. Try to follow the circuitry in figure 6. Note: crossing lines are not connected with each other.

Diodes and capacitors

The diode, a part that allows the current to flow in one direction is a safety precaution. When an electro motor is rotated by hand it will act as a dynamo and could damage the microcomputer. The diode takes care that this induction current can only flow to the battery. The capacitors function is to reduce the electromagnetic distortion from the motors.

Wires on the microcomputer

When available you can use so called jumper-wires. Take good care not to mistakenly wrongly connect the 5V plus and GND (ground) wires on the GPIO (General Purpose Input Output) pins on the Pi and the plug-in circuit board. The microcomputer will be damaged!

Voltage differences

The Pi works at a voltage of 3,3 volt and has two pins with a voltage of 5 volt directly from the USB adapter or power-bank. With these two pins the servo and ultrasonic sensor are supplied with power. The problem is that the GPIO pins on the Pi can only handle a voltage of 3,3 volt. This means that the 5 volts from the echo pin on the ultrasonic sensor must be reduced to 3,3 volt. This is done with a so-called voltage divider that contains to resistors of 330 and 470 ohm. For the servo this is reversed: the 3,3V signal from the GPIO pin has to brought up to 5V. Fortunately for the SG90 servo this is not necessary.

Polarity

The capacitor and resistor are connected with disregard of the plus or minus side. The diode however must be connected with the white band (the blockade side) at the plus side. The transistors are mounted with the metal side to the front of the vehicle so that the e/emitter, c/collector and b/basis are connected properly. The connections of the servo are given in table 2, the yellow signal wire is connected on pin 22 of the Pi. The GPIO pins and the circuit-board have a standardized size.
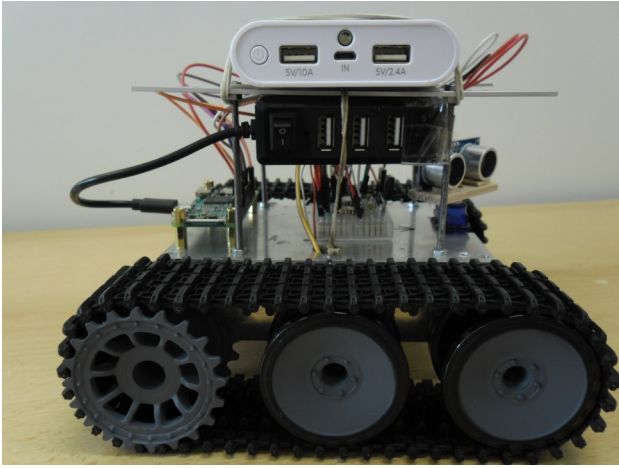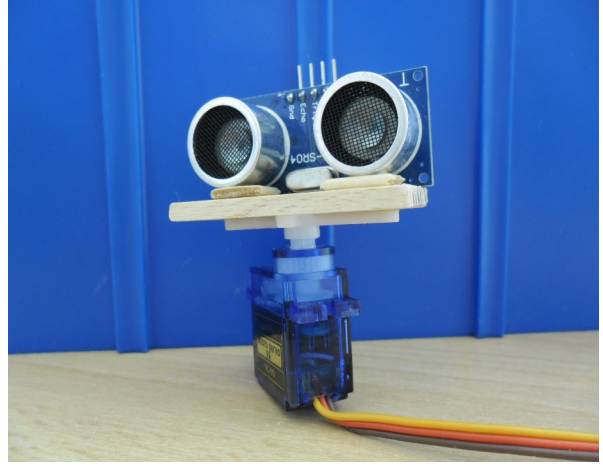
Figure 4. Components configuration

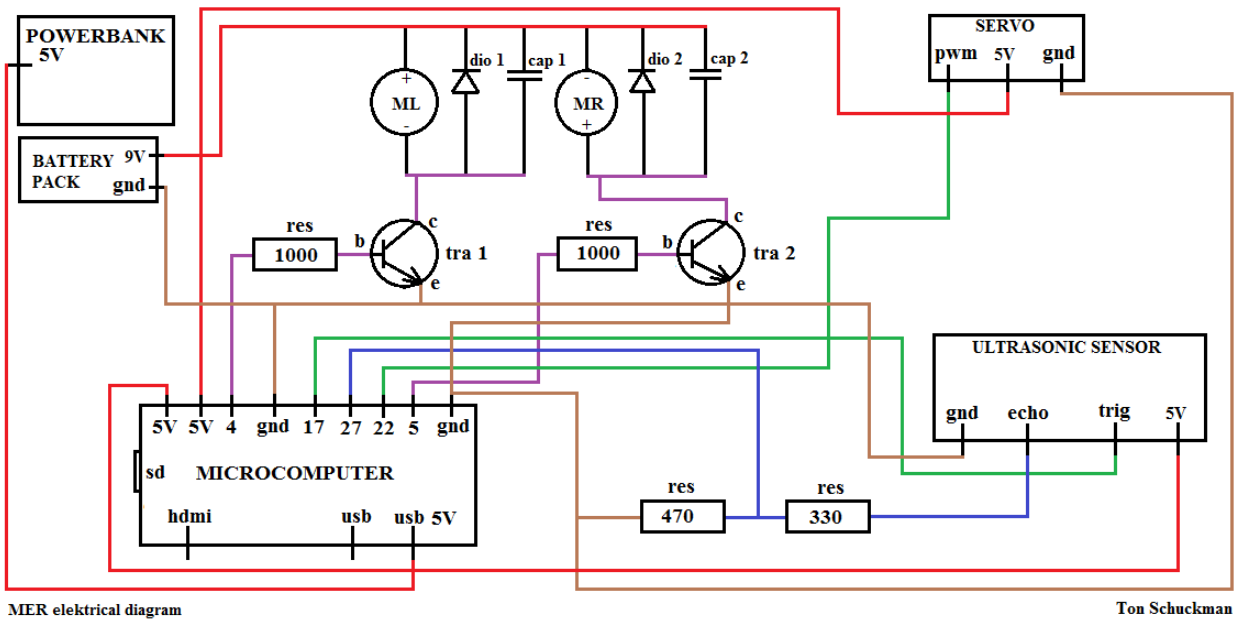Figure 5. Ultrasonic sensor and servo

Figure 6. Electrical diagram

Assembling the circuit-board

Don't make the wires to long for a good overview but you can also use so-called jumper wires. Cut the ends of the diodes, resistors and capacitors to a length of 12 mm. Now use the scheme from table 2 of the circuit-board to first connect al the electronic components and after that the wires from top to bottom on the right places. Each square on the table equals one hole at the same spot on the circuit-board. When you look closely you'll see that the holes are connected in rows of 5 holes. This way the electrical diagram in figure 6 is realized.

bas	basis (transistor)	m.r.	motor right
bat	battery	ora	orange wire (servo)
bro	brown wire (servo)	pin	pin on the pi
cap	capacitor	pwr	power (5 volt)
col	collector (transistor)	res	resistor
dio	diode	ser	servo
ech	echo (u.s. sensor)	tra	transistor
emi	emitter (transistor)	tri	trigger (u.s. sensor)
gnd	ground (-)	u.s.	ultrasonic sensor
mer	mars exploration rover	wir	wire
m.l.	motor left		

Table 1. List of abbreviations on the circuit-board

u.s. ech		res 330			tra emi	tra col	tra bas					tra emi	tra col	tra bas		
					res 470	dio 1 -			dio 1 +				dio 2 -			dio 2 +
						cap 1			cap 1				cap 2			cap 2
		pin 27														
res 330		res 470			wir c -	m.l. -	res 10^3		wir a +			wir d -	m.r. -	res 10^3		wir b +
# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #	# #
wir a +					wir c -							wir d -				
wir b +					wir f -							wir f -				
m.r. +			u.s. vcc		bat gnd		res 10^3			ser ora	ser 5v	ser bro		res 10^3		
m.l. +		u.s. tri			u.s. gnd											
bat 9v+		pin 17	pin 5v		pin gnd		pin 4			pin 22	pin 5v	pin gnd		pin 5		

Table 2. Raspberry Pi Zero connections on the plug-in circuit-board

9

	wir c -		wir a +				wir d -				u.s. ech		res 330	
	wir f -		wir b +				wir f -							
u.s. vcc	bat 9v-		m.r. +		ser ora	ser 5V	ser bru		u.s. tri					
	u.s. gnd		m.l. +										pin 27	
pin 5v	pin gnd		bat 9v+		pin 22	pin 5v	pin gnd		pin 17		res 330		res 470	

Raspberry Pi model 3 en 4 connections plug-in circuit-board left

tra emi	tra col	tra bas						tra emi	tra col	tra bas					
res 470	dio 1 -			dio 1 +				dio 2 -				dio 2 +			
	cap 1			cap 1		res 500			cap 2			cap 2		res 500	
wir c -	m.l. -	res 500		wir a +		pin 4		wir d -	m.r. -	res 500		wir b +		pin 5	

Raspberry Pi model 3 en 4 connections plug-in circuit-board right

3,3v pwr	5v pwr
gpio 02	5v pwr
gpio 03	gnd
gpio 04	gpio 14
gnd	gpio 15
gpio 17	gpio 18
gpio 27	gnd
gpio 22	gpio 23
3,3v pwr	gpio 24
gpio 10	gnd
gpio 09	gpio 25
gpio 11	gpio 08
gnd	gpio 07
id_sd	id_sc
gpio 05	gnd
gpio 06	gpio 12
gpio 13	gnd
gpio 19	gpio 16
gpio 26	gpio 20
gnd	gpio 21

Table 3. GPIO pins on the pi, the upper side is the side of the SD card

Attachment of the servo

When your MER is working properly, it's time to attach the servo at the front of the MER so it can look around to avoid obstacles. Two slots where already sawed for this at the start of this manual. Turn the piece of aluminum between the slots 90° downwards so the servo can be fastened with a tie-wrap to the baseplate. It must be made possible to unfasten the servo for replacement when malfunction occurs. You can find the electrical connections in table 2.
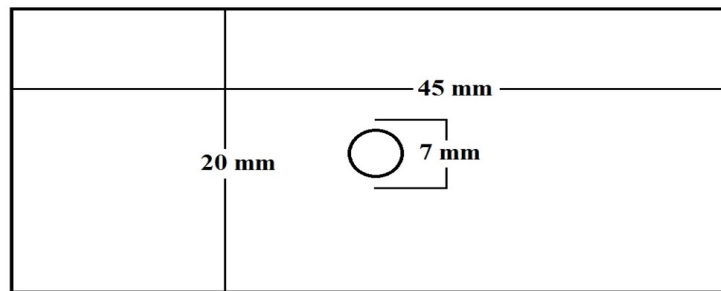
Figure 7. The sensor base plate

Mounting the ultrasonic sensor

Saw from a piece of wood or something else a rectangle of 45 mm by 20 mm and drill in the middle a hole of 7 mm (look at the demo). The sharp corners can be cut away with a cable-cutter. Use a little glue to attach the bracket of the servo at the bottom of the sensor base plate, take care to align the holes! The sensor is glued with the pins up at the upper side with the help of a small piece of wood 2 mm thick, that is placed under the aluminum tubes for horizontal alignment. Wooden ice cream sticks can be used for this.

Connecting the ultrasonic sensor

Make four wires of 20 cm (this is to enable movement of the servo) strip them 3 mm and solder them to the pins on the ultrasonic sensor. The other side of the wires can be soldered to a four-header pin, GND (black), echo, trig and vcc (red) together. But you can also use ready to use male/female jumper wires. Take good care to connect them properly on the circuit board with the help of table 2 but first you have to screw this assembly with a small screw on the servo.

The power bank and storage

Use two elastics to attach the power bank on top of the roof beams. When storing the MER, first shut it off at the terminal with the line: [sudo shutdown -h now] and make sure to disconnect the USB cable from the adapter or power bank to avoid short-circuit. Remove the six batteries to avoid leakage.

Programming the microcomputer

When all the connections are made the Pi can be brought to life. The o.s. (operating system) must be downloaded (Raspberry Pi Imager) on a pc from the raspberrypi.com website by following the instructions on this website. When downloaded start the Raspberry Pi Imager, put an SD card in the pc and follow the instructions. When this is all done put the SD card in the Pi, connect the tv/monitor with a HDMI mini to HDMI normal cable, connect the mouse and keyboard and finally the USB adapter. The Pi shall start automatically installing the o.s. Just follow the instructions on the screen. Name it pi. The Pi is not equipped with a battery so you have to fill in the date and time manually. Example: [sudo date -s 'yyyy-mm-dd hh:mm:ss'] however, this is only necessary to avoid errors when surfing the internet. Open: [Python3] [file] [new file] and type your program in the opened IDE (integrated development environment). Name your program mer_snake.py and save it in the map Documents.

Programming a snake in Python3

With the lines underneath the MER will drive a snake. After a [#] explanations can be add that will not be read by the microcomputer. When you type these lines in your own Pi IDE you have written your first program.

Software snake

```python
import RPi.GPIO as GPIO; import time
time.sleep(5) # time to put the MER on the ground
motor_left = 4; motor_right = 5; run_time = 0
GPIO.setmode(GPIO.BCM)
GPIO.setup(motor_left, GPIO.OUT); GPIO.setup(motor_right, GPIO.OUT)
while run_time < 2:
    #straight
    GPIO.output(motor_left, True); GPIO.output(motor_right, True); time.sleep(1.0)
    GPIO.output(motor_left, False); GPIO.output(motor_right, False)
    #right
    GPIO.output(motor_left, True); time.sleep(1.5); GPIO.output(motor_left, False)
    # straight
    GPIO.output(motor_left, True); GPIO.output(motor_right, True); time.sleep(1.0)
    GPIO.output(motor_left, False); GPIO.output(motor_right, False)
    #left
    GPIO.output(motor_right, True); time.sleep(1.0); GPIO.output(motor_right, False)
```

```
#straight
GPIO.output(motor_left, True); GPIO.output(motor_right, True); time.sleep(1.0)
GPIO.output(motor_left, False); GPIO.output(motor_right, False)
#right
GPIO.output(motor_left, True); time.sleep(1.5); GPIO.output(motor_left, False)
#straight
GPIO.output(motor_left, True); GPIO.output(motor_right, True); time.sleep(1.0)
GPIO.output(motor_left, False); GPIO.output(motor_right, False)
#left
GPIO.output(motor_right, True); time.sleep(1.0); GPIO.output(motor_right, False)
run_time = run_time +1
```

To start the program, open the terminal and type [cd Documents] then [sudo python3 mer_snake.py]. Make sure the MER is lifted from the table otherwise it will drive from the table…

When the MER operates properly connect the keyboard cable, and type [ctrl c] and the program will stop. Next step is to shutdown the Pi with [sudo shutdown -h now] and disconnect the USB adapter. Start the Pi again with the power bank connected to it. Lift the MER from the ground, start your program again and pull out the HDMI, mouse and keyboard cables. Your robot can now drive independently. To shut it down again you only will have to connect the keyboard cable again and type [ctrl c] [enter]. This can be done by lifting it from the floor or ask someone to hold it for you.

Software with ultrasonic sensor and servo
```
# !/usr/bin/env python3
# to start the program; open the terminal and type: sudo python3 [file name]
# to stop the program type [ctrl] and [c] simultaneously at the terminal
# the indentations are 4, 8 or 12 spaces, do not use tabs

import RPi.GPIO as GPIO; import time; GPIO.setwarnings(False)
time.sleep(7) # time to put the MER on the ground
# constants used for driving logic
line_of_sight = 40; turn_delay = 1

# duty cycle values for the pulse width modulation of the servo. Default min.2 max.12
# check these values to make sure the servo properly rotates 90 degrees each side
min_duty = 2; max_duty = 12
```

```python
class Robot:
    # define all pins as used on the raspberry pi board
    servo_pin = 22; motor_left = 4; motor_right = 5; echo_pin = 27; trig_pin = 17

    # initial setup of the robot
    def __init__(self):
        # set up the pin layout
        GPIO.setmode(GPIO.BCM)

        # set up the pins for output and input
        GPIO.setup(self.motor_left, GPIO.OUT); GPIO.setup(self.motor_right, GPIO.OUT)
        GPIO.setup(self.trig_pin, GPIO.OUT); GPIO.setup(self.echo_pin, GPIO.IN)
        GPIO.setup(self.servo_pin, GPIO.OUT)

        # start pulse width modulation for servo and motors
        self.pwm_servo = GPIO.PWM(self.servo_pin, 50)
        self.pwm_left = GPIO.PWM(self.motor_left, 1000)
        self.pwm_right = GPIO.PWM(self.motor_right, 1000)

        # default values: servo straight ahead, motors not moving, ultrasonic sensor inactive
        self.pwm_servo.start(7); self.pwm_left.start(0); self.pwm_right.start(0)
        GPIO.output(self.trig_pin, GPIO.LOW)

    # cleanup of the robot
    def __del__(self):
        # stop the servo
        self.pwm_servo.stop()

        # stop the motors
        self.motor_speed(0,0); self.pwm_left.stop(); self.pwm_right.stop()

    # set the motor speed of the left and right wheels to a value between 0% and 100%
    def motor_speed(self, left, right):
        # set the left engine
        self.pwm_left.ChangeDutyCycle(left)

        # set the right engine
        self.pwm_right.ChangeDutyCycle(right)
```

```python
    # turn the servo to an angle between -90 and +90 degrees
    def turn_servo(self, angle):

        # calculate the duty cycle by linearly distributing the angles
        duty_cycle = (angle + 90) / 180 * (max_duty - min_duty) + min_duty

        # set the duty cycle
        self.pwm_servo.ChangeDutyCycle(duty_cycle)

        # wait for the servo to turn
        time.sleep(0.30)

    # use the ultrasonic sensor to look how far the nearest object is
    def look(self):
        # trigger the sensor by setting the pin to high, and then low again
        GPIO.output(self.trig_pin, GPIO.HIGH); time.sleep(10e-6)
        GPIO.output(self.trig_pin, GPIO.LOW)

        # define start and end time
        pulse_start = time.time(); pulse_end = time.time()

        # start counting when the echo pin is low
        while GPIO.input(self.echo_pin) == GPIO.LOW:
            pulse_start = time.time()

        # stop counting when the echo pin is high again (signal has come back)
        while GPIO.input(self.echo_pin) == GPIO.HIGH:
            pulse_end = time.time()

        # Calculate the distance (speed of sound 34300 cm/s)
        return (pulse_end – pulse_start)*34300/2

# driving code starts here
my_robot = Robot()

# initial speed of the robot
my_robot.motor_speed(50,50)
```

```python
# the main loop
while True:
    # look in each direction and check the distance to the nearest object
    my_robot.turn_servo(45); distance_left = my_robot.look()
    my_robot.turn_servo(0); distance_straight = my_robot.look()
    my_robot.turn_servo(-45); distance_right = my_robot.look()

    # print the distances and a blank line
    print("Distance left     %4.1f cm" % distance_left)
    print("Distance straight %4.1f cm" % distance_straight)
    print("Distance right    %4.1f cm" % distance_right)
    print(" ")

    # driving logic
    if(distance_left<line_of_sight and distance_straight < line_of_sight and
    distance_right<line_of_sight): my_robot.motor_speed(0,0)

    elif distance_left < line_of_sight:
        my_robot.motor_speed(100,0)
        time.sleep(turn_delay)
        my_robot.motor_speed(50,50)

    elif distance_straight < line_of_sight:
        if distance_right > distance_left:
            my_robot.motor_speed(100,0)
            time.sleep(turn_delay)
            my_robot.motor_speed(50,50)

        else:
            my_robot.motor_speed(0,100)
            time.sleep(turn_delay)
            my_robot.motor_speed(50,50)

    elif distance_right < line_of_sight:
        my_robot.motor_speed(0,100)
        time.sleep(turn_delay)
        my_robot.motor_speed(50,50)
```

Control the MER through your mobile phone with Bluedot

Open the terminal and install bluedot: sudo pip3 install bluedot. Typ the program underneath in python3 and save it in documents as: b_dot. Install the app bluedot on your phone. Connect your phone with the MER through bluetooth and open in the terminal cd Documents and start the program: sudo python3 b_dot.py

```python
import RPi.GPIO as gpio; import time; import sys; import tkinter as tk
from bluedot import BlueDot

bd = BlueDot()
def init():
    gpio.setmode(gpio.BCM); gpio.setup(4, gpio.OUT)
    gpio.setup(5, gpio.OUT); gpio.setwarnings(False)

init()
def forward(tf):
    gpio.output(4, True); gpio.output(5, True); time.sleep(tf)
def left(tf):
    gpio.output(4, False); gpio.output(5, True); time.sleep(tf)
def right(tf):
    gpio.output(4, True); gpio.output(5, False); time.sleep(tf)
def stop(tf):
    gpio.output(4, False); gpio.output(5, False)
    sys.exit(); time.sleep(tf)
def move(pos):
    init(); sleep_time = 0.030

    if   pos.top:     forward(sleep_time)
    elif pos.left:    left(sleep_time)
    elif pos.right:   right(sleep_time)
    elif pos.bottom: stop(sleep_time)

def init():
    gpio.setmode(gpio.BCM); gpio.setup(4, gpio.OUT)
    gpio.setup(5, gpio.OUT); gpio.setwarnings(False)

def stop1():
    init()
    gpio.output(4, False); gpio.output(5, False)
bd.when_pressed = move; bd.when_moved = move; bd.when_released = stop1
command = tk.Tk(); command.mainloop()
```

part	type	amount
tank chassis	he0102-460	1
raspberry pi micro pc	zero w	1
micro sd card	8 or 16 gb	1
circuit-board	he0109-002	1
nuts and bolts	m2.5 x 20 3x	< >
nuts and bolts	m3 x 75 mm 4x	< >
aluminum strip	10 - 400 mm	1
diodes	50v 1a	2
transistors	npn bd139	2
wires	he0167-001	1
male headers	he0236-600	1
resistors	500 ohm	2
resistor	470 ohm	1
resistor	330 ohm	1
capacitors	10 nanofarad	2
wires male-male	he0108-002	< >
ultrasonic sensor	hc-sr04	1
servo	sg90	1
usb hub micro to a	kw2407	1
hdmi cable	mini to normal	1
battery holder	six aa	1
battery 1.5 aa	rechargeable	6 *
powerbank > 10000 mah	5volt > 2.4 a output	1 *
usb cable	micro to a	1 *
usb adaptor	output >2.4 amps	1 *
mouse	with usb connector	1 *
keyboard	with usb connector	1 *
tv or monitor	with hdmi socket	1 *

Table 4. Parts list   * available at home or to be purchased

At home start sequence
> Connect de Pi Zero with a HDMI cable to your monitor or TV set.
> Connect the keybord and mouse into the USB hub on the MER.
> Connect the Pi Zero to the powerbank.
> Open the terminal and type [cd Documents], [enter].
> Then type: [sudo python3 mer_us_sv.py], [enter].
> Disconnect the HDMI, mouse and keybord cables, and of you go.

To stop de MER connect the keybord and type: [ctrl c]. After that you can disconnect the HDMI and mouse cables again.

When finished
This is the end of the support from the physics department but you're always welcome to attend the robotica afternoons to expand your robot on your own or together with classmates, or even use the Pi platform for something totally different. You can also take a look on the internet for ideas.

Warranty and Ownership
During all your years on Werenfridus, you have a full warranty on the MER, excluding the microcomputer, it has its own factory warranty. So, if your MER doesn't work anymore and you can't solve the problem by yourself, bring it to the physics cabinet to have it repaired. When finished and taken home, the MER becomes your property, therefore the school is not accountable for damage caused by the MER during its use and storage.

Disposal and Recycling
In the (unhappily) event that you want to dispose the MER, please don't throw it away but bring it back to the physics cabinet so that it can be taken apart for spare parts. This is also valid for the situation when the Mer is not finished. Their will be no refund.

Tabor College
Werenfridus

Ton Schuckman
Physics amanuensis/instructor
aj.schuckman@tabor.nl