Lego Spike Prime

Af: Michael Hansen, Coding Pirates Furesø, 2022, version 0.50

Dokumentet ligger her: https://github.com/mhfalken/lego

Her er lidt ideer til hvordan man kan bruge Lego Spike Prime sættet til Coding Pirates, hvor hoved fokus er på programmering og specielt **Python**. I de skoleopgaver som 'følger med' sættet får man generelt udlevet al koden (scratch) og primært skal man bare se hvad der sker. Det egner sig i mine øjne ikke særligt godt til Coding Pirates, hvor fokus er mere på programmering.
Note: Jeg har tegnet et 'kort/bane' på et A0-ark ala FLL, og det er det jeg refererer til i nogle af opgaverne. Nogle af disse opgaver kan ikke løses uden dette kort eller noget tilsvarende af køre efter. Der er et billede af mit kort i bunden af dette dokument. Jeg har også lavet 2 paprør, som afstandssensoren har lettere ved at se end nogle Legoklodser.

1 Bil kontrol

1.1 Klods til cirkel

Bilen skal starte i startområdet og skubbe en klods ind i cirklen i midten og køre baglæns tilbage til start området.
Opret et nyt program (Fil->Nyt Projekt), vælg Python og kald det *bil-klods* og tryk **Opret**.
Byg den bil som hedder *Grundmodel 1* under **Modeller**.

1. Få bilen til at køre frem

```
motor_pair = MotorPair('C', 'D')
motor_pair.move(20, "cm", steering=0, speed=50)
```

Brug hjælpen ude til høje i Lego programmet til at få hjælp til hvad de forskellige parametre gør.

2. Få bilen til at skubbe klodsen ind i cirklen og køre tilbage igen

1.2 Firkant

Vi skal nu have bilen til at køre i en firkant.
Lav et nyt projekt og kald det *bil-firkant*.
Når bilen skal dreje om sig selv, kan man bruge følgende kommando:

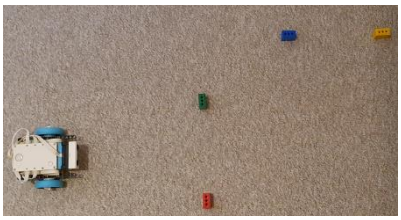
```
motor_pair.move(180, 'degrees', steering= -100, speed=50)
```

1) Få bilen til at køre 20 cm frem og derefter dreje 90 grader til venstre.
2) Lav en loop rundt om disse linjer, så den laver det 4 gange og der med en firkant.

Bilen skal gerne ende samme sted som den startede.

1.3 Bane

Byg en lille bane af Legoklodser og få bilen til at følge banen vha. simple motor kommandoer.
Lav et nyt projekt og kald det *bil-bane*.



1) Byg en lille bane af Legoklodser
2) Få bilen til at følge banen

1.4 Hent en klods

Sæt en gribearm på bilen og få den til at hente en klods.
Lav et nyt program og kald det *bil-arm*.

1) Byg en gribearm som er styret af motoren foran. Søg for at gribearmen er monteret på bilen og stikket sat i **port E**.
2) Sæt bilen i start området og få den til at køre frem til klodsen og tilbage igen
3) Få gribearmen til at åbne sig inden bilen kører frem

```
motorArm = Motor('E')
...
motorArm.run_for_rotations(0.15)
```

4) Få gribearmen til at lukke sig når bilen når klodsen
5) Test at bilen nu kan hente klodsen

1.5 Hent flere klodser

Få bilen til at hente flere klodser, ved at køre flere gange
Lav et nyt program og kald det *bil-klodser*.

1) Læg to klodser med ca. 15 cm imellem sig, så bilen kan hente en af gangen
2) Kopier koden fra før til at hente den første klods

Når bilen kommer tilbage fra første tur, så skal den drejes lidt, så den peger mod den næste klods.

3) Tilføj en linje som drejer bilen lidt så den peger mod den næste klods
4) Lav den kode som henter klods to. (Det er det samme kode som henter klods et)
5) Test at bilen kan hente begge klodser

1.6 Accelerometer kontrol (avanceret)

Vi skal nu bruge accelerometeret til at styre vinklen som bilen drejer.
Lav et nyt program og kald det *bil-vinkel*.
Brug nu bevægelsessensoren i HUB'en til at få bilen til at dreje 90 grader.

1) Først skal du lave et lille program som printer vinklen og så dreje bilen i hånden for at se hvordan sensoren virker.

```
for i in range(50):
  v = hub.motion_sensor.get_yaw_angle()
  print(v)
```

Læg mærke til at værdien kan 'hoppe' fra 180 til -180! Det gør det hele lidt sværere. For at undgå det når man skal styre bilen, kan man nulstille vinklen inden start.

```
hub.motion_sensor.reset_yaw_angle()
```

Man kan også se vinklen direkte i Lego programmet ved at trykke på ikonet i øverste venstre hjørne – se billedet til højre:

2) Lav nu et program som for bilen til at dreje og som venter på at den har drejet 90 grader.

Her skal du bruge start og stop motor funktionerne:

```
motor_pair.start(steering=100, speed=50)
...
motor_pair.stop()
```

3) Når programmet er slut, så få det til at skrive den reelle vinkel ud. Overvej hvorfor bilen ikke drejer præcist 90 grader og prøv at gøre noget ved det.

Hint: Det har noget med motorhastigheden at gøre.

4) `print()` tager lang tid for CPU at udføre, så programmet kører meget langsomt. Fjern `print(…)` inde i løkken og se hvad der sker! Fiks det nye problem som opstår.

Hint. Det har noget at gøre med at programmet kører meget hurtigere nu så loop'en bliver for hurtig færdig.

2 Bil farvesensor

2.1 Kør til stregen

Få bilen til at køre frem og stoppe når den kører over en sort streg.
Lav et nyt program og kald det *bil-streg*.

1) Sæt farvesensoren fast på bilen så den peger nedad ca. 15 mm over gulvet. Sæt stikket i port A.
2) Placer bilen i startområdet så den peger til venstre (mod de 4 sorte linjer).

Farvesensoren aflæses på følgende måde:

```
colorsensor = ColorSensor('A')
...
color = colorsensor.get_color()
```

Første linje sætter sensoren op og den anden læser den aktuelle farve som den ser.
Vi skal også have bilen til at køre 'af sig selv' dvs. at den kører indtil at den skal stoppe. Det gøres med følgende kommandoer:

```
motor_pair.start(steering=0, speed=80)
...
motor_pair.stop()
```

3) Lav følgende program og se at bilen kører frem

```
motor_pair = MotorPair('C', 'D')
afstandSensor = DistanceSensor('F')

motor_pair.start(steering=0, speed=50)
for i in range(50):
    wait_for_seconds(0.05)
motor_pair.stop()
```

4) Få nu bilen til løbende at læse farvesensoren og stoppe bilen når den ser en sort streg.

2.2 Tæl antal streger

Få bilen til at køre et stykke og tæl hvor mange linjer den kører over.
Lav et nyt program og kald det *bil-count*.

1) Ændre koden fra før til at tælle hver gang den kører over en sort streg.
2) Skrev tallet ud på displayet `hub.light_matrix.write(count)`

Husk at stoppe efter lidt tid. Hvis man læser farverne for hurtigt, så kan man komme til at tælle den samme streg flere gange. Brug `wait_for_seconds(0.05)` til at få koden til at køre lidt langsommere.

2.3 Lav en liste over farverne

Få bilen til at køre et lille stykke og lav en liste over de farver den ser.
Lav et nyt program og kald det *bil-farve-liste*.

1) Placer bilen i start området udfor den liste af farver som har 5 farver.
2) Kopier koden fra før
3) Opret en tom liste `colorl = []`
4) Hver gang man kører over en ny farve, så gem farven i listen. `Colorl.append(color)`
5) Udskriv listen til sidst `print(colorl)`.

Farvestriberne ligger meget tæt, så koden må ikke have nogen pause (`wait_for_seconds()`). For at undgå at man læser den samme farve flere gange, så må man gemme den forrige farve for at se om den aktuelle farve er ny.
Hint: Brug følgende linjer i din kode:

```
colorPrev = 'white'
...
if color != 'white' and color != None and color != colorPrev:
  colorPrev = color
  ...
```

2.4 Reager på farverækkefølgen (avanceret)

Få bilen til at gøre noget forskelligt afhængig af farverækkefølgen (lidt lige som en Ozobot).
Lav et nyt program og kald det *bil-farve-sekvens*.

1) Placer bilen foran en af farvelisterne
2) Kopier koden fra før
3) Tilføj de tre farvelister på denne måde i starten af koden (kun liste1 vist):
   `list1 = ['blue', 'yellow', 'black', 'green', 'red']`

Man sammenligner to lister på følgende måde:

```
if colorl == list1:
  ...
```

4) Tilføj nu 3 `if` sætninger i bunden af koden, en for hver liste og få bilen til at gøre noget forskelligt for hver af dem.
5) Test det virker for alle tre farvelister

2.5 Farvereaktion

Lav et program, som få bilen til at køre i forskellige mønstre afhængig af hvilken farve farvesensoren ser.
Fx *rød* får bilen til at køre 10 cm frem, *grøn* får bilen til at dreje 90 grader etc.
Lav et nyt program og kald det *bil-farve*.

1) Sæt farvesensoren fast på bilen, så man kan holde en klods op foran den.
2) Tilføj følgende linjer, og test det virker med en rød klods.

```
motor_pair = MotorPair('C', 'D')
colorsensor = ColorSensor('A')

while True:
  color = colorsensor.get_color()
  if color == 'red':
    motor_pair.move(10, "cm", steering=0, speed=50)
```

3) Tilføj support for flere farver, så bilen gør noget forskelligt afhængig af farven.

3 Bil afstandssensor

3.1 Kør tæt på en klods

Få bilen til at køre så tæt på en 'væg' som muligt.
Lav et nyt program og kald det *bil-afstand*.

Afstandssensoren sidder højt oppe på bilen og er desværre ikke særlig følsom og har derfor svært ved at se klodser. Brug derfor et 'paprør' som 'væg'.

5) Sæt afstandssensoren på bilen og stikket i **port F**.
6) Sæt et rør ca. 30 cm fra bilen.

Afstandssensoren aflæses på følgende måde:

```
afstandSensor = DistanceSensor('F')
...
afstand = afstandSensor.get_distance_cm()
```

Vi skal også have bilen til at køre 'af sig selv' dvs. at den kører indtil at den skal stoppe. Det gøres med følgende kommandoer:

```
motor_pair.start(steering=0, speed=80)
...
motor_pair.stop()
```

7) Lav følgende program og se at bilen kører frem

```
motor_pair = MotorPair('C', 'D')
afstandSensor = DistanceSensor('F')

motor_pair.start(steering=0, speed=50)
for i in range(50):
    wait_for_seconds(0.05)
motor_pair.stop()
```

Vi skal nu have afstandssensoren til at stoppe bilen, når den er tæt på røret

8) Tilføj en `if` sætning som får bilen til at stoppe når afstanden er mindre end 7 cm. Selve `if` sætningen skal desværre se lidt 'mærkelig'.

```
if afstand and afstand < 7:
```

9) Test nu at bilen stopper tæt på røret

Afstandssensoren har 'øjne' som kan lyse.

10) Sæt følgende linje ind i starten af koden

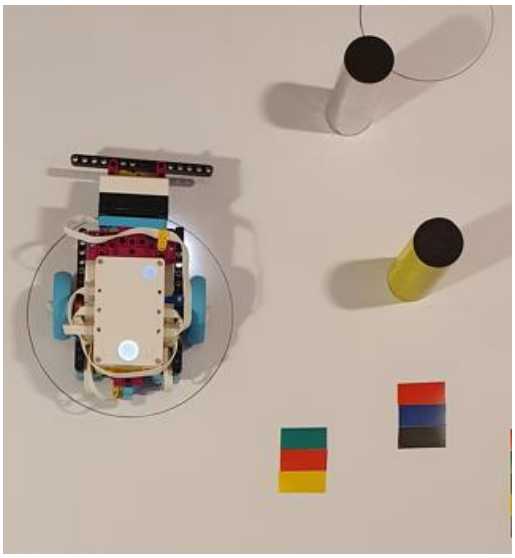
```
afstandSensor.light_up_all(100)
```

11) Tilføj yderlige en linje som slukker for 'øjnene' når bilen stoppet ved røret.

3.2 Ryd området

Placer bilen i cirklen i midten og de to paprør uden om. Bilen skal nu drejerundt og skubbe de to paprør ud til siderne.
Lav et nyt program og kald det *bil-ryd*.

1) Placer bilen i cirklen i midten. Man kan med fordel montere en pind på tværs på gribearmen for at gøre den lidt bredere – se billedet.
2) Placer de to paprør ca. 15 cm uden for cirklen
3) Kopier koden fra før (Kør tæt på en klods)
4) Få bilen til at dreje langsomt rundt om sig selv
5) Aflæs afstandssensoren og stop bilen hvis den ser noget tættere på end 30 cm
6) Test at bilen stopper når den peger på røret
7) Kør bilen frem og tilbage for at skubbe røret væk (koden skal stå inde i `if` sætningen)
8) Forsæt med at dreje bilen rundt og kik efter næste rør



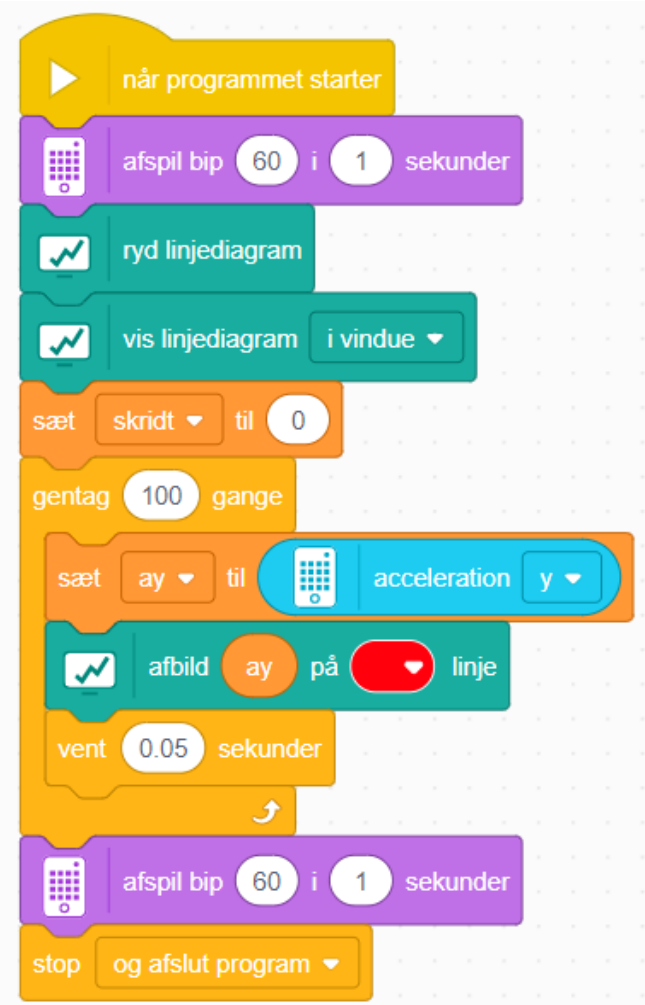
4 Bil følg en streg

TBD

5 Skridt tæller

Vi skal nu lave et program der kan tælle hvor mange skridt man har taget.
Lav et nyt projekt og vælg "Ordblokke" og kald det *skridt*.
HUB'en skal enten puttes i lommen, eller man skal bygge en holder, så den kan sidde i bæltet. **Det er vigtigt at den ikke bliver tabt på gulvet**! Man kan eventuelt bruge byggevejledningen: **Træningstrackere->Skridt for skridt**.

1) Tilføj følgende blokke (man kan tilføje de blokgrupper som mangler med  nede i venstre hjørne.



2) Sæt HUB'en fast på kroppen (eller put den i lommen), som en skridttæller og prøv at gå mens programmet kører. Kik på grafen og finde ud af hvad den viser og hvordan det kan bruges til at tælle antallet af skridt.
3) Tilføj kode så den kan tælle antal skridt man tager. Antallet af skridt kan vises som en ekstra graf – det gør det lettere at fejlsøge. (Koden må kun have ét flow, da den senere skal skrives om til Python)
4) Få antallet af skridt vist på displayet på HUB'en.

5.1 Lav nu den samme kode i Python

Her er der desværre lidt udfordringer, da Legos Python implementering er mangelfuld!

1) Man kan ikke lave en graf, så det er lidt sværere at fejlsøge.
2) Man kan ikke tilgå accelerometeret direkte uden at lave et hack i koden, så det gør vi.

Lav et nyt Python program og kald det: *skridt*.
Tilføj denne linje lige under de andre import (linje 4).

```
import hub as hub2 # Special hack to access more features
```

For at kunne aflæse accelerometeret skal man bruge følgende linje:

```
ay = hub2.motion.accelerometer()[1]
```

Skriv nu Python koden så den følger det samme flow som i blokkoden og bruger de samme værdier.

1) Start med at lave et program, som printer `ay` ud, og se at det virker. (Husk delayet)
2) Tilføj et start og stop beep

3) Indsæt nu de 2 if sætninger og print `skridt` ud i stedet for `ay`
4) Skriv antallet af `skridt` ud på displayet.

6 Plotter (CNC-maskine)

Lego lektion: **Opfinderholdet->Defekt**
Lav koden i Python

7 Løsninger

7.1 Bil

7.1.1 Bane

```
motor_pair = MotorPair('C', 'D')

motor_pair.move(50, "cm", steering=0, speed=50)
motor_pair.move(30, "cm", steering=-40, speed=50)
motor_pair.move(20, "cm", steering=0, speed=50)
```

7.1.2 Hent flere klodser

```
motor_pair = MotorPair('C', 'D')
motorArm = Motor('E')

motorArm.run_for_rotations(0.15)
motor_pair.move(40, "cm", steering=0, speed=50)
motorArm.run_for_rotations(-0.15)
motor_pair.move(-40, "cm", steering=0, speed=50)

motor_pair.move(30, 'degrees', steering= 100, speed=50)
motorArm.run_for_rotations(0.15)
motor_pair.move(40, "cm", steering=0, speed=50)
motorArm.run_for_rotations(-0.15)
motor_pair.move(-40, "cm", steering=0, speed=50)
```

7.1.3 Accelerometer kontrol

```
motor_pair = MotorPair('C', 'D')
hub.motion_sensor.reset_yaw_angle()
print("--- Start ---")
motor_pair.start(steering=100, speed=20)

for i in range(10000):
    v = hub.motion_sensor.get_yaw_angle()
    #print(v)
    if v >= 90:
        break

motor_pair.stop()
wait_for_seconds(0.1)
vf = hub.motion_sensor.get_yaw_angle()
print("Vinkel [%i] %i" %(i, vf))
```

7.2 Bil farvesensor

7.2.1 Bil-streg

```
motor_pair = MotorPair('C', 'D')
colorSensor = ColorSensor('A')

motor_pair.start(steering=0, speed=50)
for i in range(1000):
    color = colorSensor.get_color()
    if color == 'black':
        break
motor_pair.stop()
```

7.2.2 Bil-count

```
motor_pair = MotorPair('C', 'D')
colorSensor = ColorSensor('A')

count = 0
motor_pair.start(steering=0, speed=50)
for i in range(50):
    color = colorSensor.get_color()
    if color == 'black':
        count += 1
    wait_for_seconds(0.05)
motor_pair.stop()
hub.light_matrix.write(count)
```

7.2.3 Bil-farve-liste

```
motor_pair = MotorPair('C', 'D')
colorsensor = ColorSensor('A')

colorl = []
colorPrev = 'white'
print("----------")
motor_pair.start(steering=0, speed=80)
for i in range(400):
    color = colorsensor.get_color()
    if color != 'white' and color != None and color != colorPrev:
        colorPrev = color
        colorl.append(color)
```

```
        hub.light_matrix.write(len(colorl))
motor_pair.stop()
print(colorl)
```

7.2.4 Bil-farve-sekvens

```
motor_pair = MotorPair('C', 'D')
colorsensor = ColorSensor('A')

list1 = ['blue', 'yellow', 'black', 'green', 'red']
list2 = ['black', 'blue', 'red']
list3 = ['yellow', 'red', 'green']
colorl = []
colorPrev = 'white'
print("----------")
motor_pair.start(steering=0, speed=80)
for i in range(400):
    color = colorsensor.get_color()
    if color != 'white' and color != None and color != colorPrev:
        colorPrev = color
        colorl.append(color)
        hub.light_matrix.write(len(colorl))
motor_pair.stop()
print(colorl)
if colorl == list1:
    print('List1')
    motor_pair.move(720, 'degrees', steering=100, speed=50)
if colorl == list2:
    print('List2')
    motor_pair.move(-40, 'cm', steering=0, speed=50)
if colorl == list3:
    print('List3')
    motor_pair.move(360, 'degrees', steering=100, speed=30)
    motor_pair.move(40, 'cm', steering=0, speed=80)
```

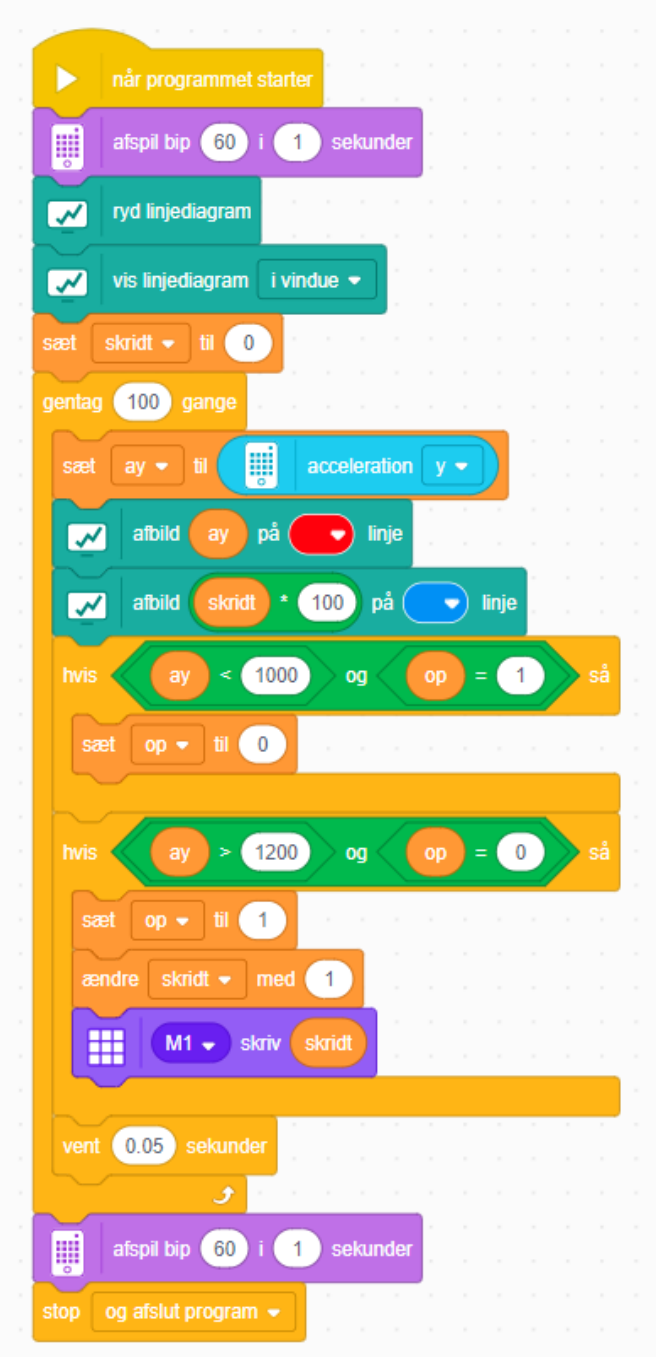
7.3 Bil afstandssensor

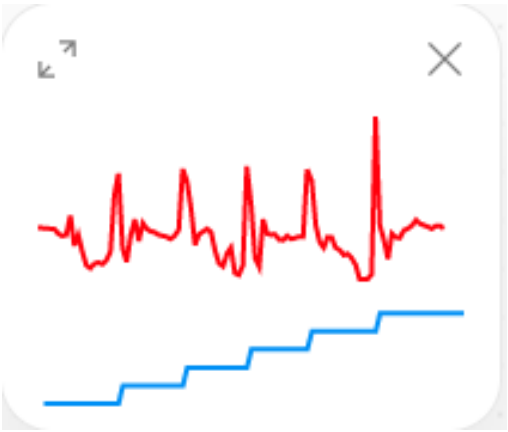
7.3.1 Ryd området

```
motor_pair = MotorPair('C', 'D')
afstandSensor = DistanceSensor('F')
afstandSensor.light_up_all(0)
motor_pair.start(steering=100, speed=5)

for i in range(150):
    afstand = afstandSensor.get_distance_cm()
    if afstand and afstand < 30:
        afstandSensor.light_up_all(100)
        motor_pair.stop()
        motor_pair.move(30, "cm", steering=0, speed=50)
        motor_pair.move(-30, "cm", steering=0, speed=50)
        afstandSensor.light_up_all(0)
        motor_pair.start(steering=100, speed=5)
    wait_for_seconds(0.05)
motor_pair.stop()
```

7.4 Skridttæller

7.4.1 Blokkoden





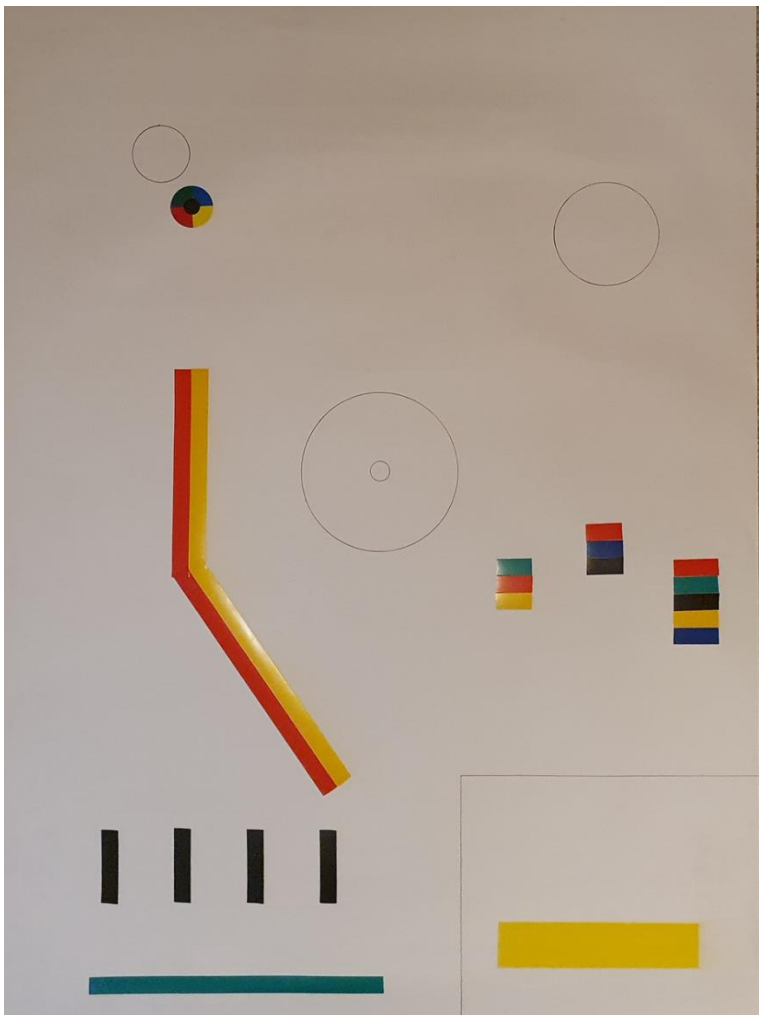
7.4.2 Pythonkoden

```
import hub as hub2 # Special hack to access more features

hub = PrimeHub()

hub.light_matrix.show_image('HAPPY')

skridt = 0
op = False
hub.speaker.beep(60, 1)
for i in range(100):
    ay = hub2.motion.accelerometer()[1]
    if ay < 1000 and op == True:
        op = False
    if ay > 1200 and op == False:
        op = True
        skridt = skridt + 1
        hub.light_matrix.write(skridt)
        print(skridt)
    wait_for_seconds(0.05)
hub.speaker.beep(60, 1)
```

8 Appendiks

8.1 Kort/bane

Kortet er tegnet på et A0 papir og striberne er lavet med tape.



8.2 Links