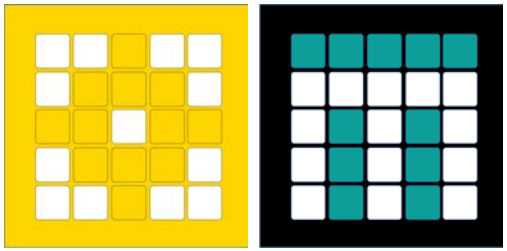
PRIME LESSONS

By the Makers of EV3Lessons



ÎNTO ARCERILE CU GYRO

BY SANJAY AND ARVIND SESHAN

This lesson uses SPIKE 3 software

OBIECTIVELE LECȚIEI

- Învățăm cum să îmbunătățim acuratețea întoarcerilor
- Învățăm moduri alternative de a realiza întoarceri pivot și spin.

CÂT DE PRECISĂ ESTE ÎNTOARCEREA?

Rulează acest cod și utilizează Tabloul de bord pentru a vedea dacă la întoarcerea de 90 de grade robotul întoarce de fapt 90 de grade.

```
from hub import port, motion_sensor
import runloop, motor_pair, sys

# Funcția care poate returna valoarea de adevăr când valoarea absolută a ,,yaw angle'' este 90 de grade
def turn_done():
    # convertește tuple decidegree în același format ca în app și block-uri
    return abs(motion_sensor.tilt_angles()[0] * -0.1) > 90

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
    motion_sensor.reset_yaw(0)
    await runloop.until(motion_sensor.stable)
    # Utilizează o întoarcere de 100 în loc de 50 pentru a efectua întoarcerea spin
    motor_pair.move(motor_pair.PAIR_1, 50, velocity=500)
    await runloop.until(turn_done)
    motor_pair.stop(motor_pair.PAIR_1)
    sys.exit("Done")

runloop.run(main())
```

CÂT DE PRECISĂ ESTE ÎNTOARCEREA?

- Observă că noi am setat viteza motorului la 500 în loc de 200 în lecția anterioară.
- Pentru Drive Base 1, ei întorc 98 de grade
- Aceasta este din două motive
  1. Ia un pic de timp să citești giroscopul. În acest timp, robot se mișcă. Această întârziere de pe SPIKE Prime este relativ mică dar va produce câteva grade de eroare.
  2. Este nevoie de ceva timp pentru a opri robotul din cauza inerției. Aceasta produce câteva grade de eroare adițională.

IMBUNĂTĂȚIREA ACURATEȚEI ÎNTOARCERILOR

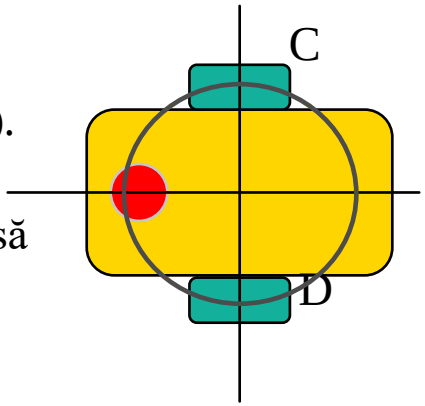
- Așa cum am menționat în slide-ul anterior, utilizarea Drive base 1 la viteza 500, robotul se întoarce 98 de grade în loc de cerința de 90 de grade. Cum rezolvăm această problemă?
- O soluție ar fi să cerem robotului să întoarcă 8 grade mai puțin pentru Drive Base 1.
  - Numărul de grade cu care e nevoie să reduci întoarcerea va depinde de viteza întoarcerii și de design-ul robotului. Vei avea nevoie să încerci câteva valori pentru a fi totul perfect.
- O soluție mai bună este să utilizezi o viteză mai mică. Cu o viteză de 200, eroarea la Drive Base 1 am redus de la 8 grade la doar 2 grade.
- Nu am observat o diferență semnificativă utilizând move vs move_tank. Ajustarea vitezei a făcut o mare diferență.
- Întoarcerea de tip Pivot și spin tpar să aibă tipare de eroare similară.

ÎNTOARCEREA FĂRĂ A UTILIZA GYRO

- Este posibil să calculezi numărul de grade de întoarcerea a roții pentru o valoare specifică, utilizând geometria robotului.
- Amintiți-vă, întoarcerile motorului nu sunt la fel pe măsură ce robotul întoarce!
- Utilizăm Drive Base 1 (DB1) ca exemplu:
- Mai întâi, trebuie să știm circumferința roții. Pentru DB1, aceasta este 17.5cm
- Apoi, trebuie să cunoaștem distanță dintre roți. Aceasta este cunoscut ca și TRACK. Poți să măsurați. Pentru DB1, este aproximativ 11.2 cm.
- Pentru DB1, motoarele acționează direct roțile de tracțiune, fără roți dințate de transfer. Așa că, noi știm nu avem nevoie de rația de multiplicare (i.e., it is 1)
- Acum, noi calculăm matematic pentru întoarcerile de tip spin si pivot.

GEOMETRIC SPIN TURN

- O întoarcere de tip spin întoarce robotul în jurul centrului dintre roțile de tracțiune.
- **Diametru** cercului de întoarcere de tip spin este egală cu cursa (TRACK). Circumferința dată este: Cs = Track * Pi
- Pentru a face o întoarcere întreagă de 360 de grade, fiecare roată trebuie să se miște o distanță egală cu circumferința cercului de întoarcere.
- Rotația cu un singur motor= rotația unei singure roți
- Dacă circumferința roții este dată de Cw, apoi:



Numărul de rotații a motorului pentru întoarcerea robotului 360 de grade = (Cs /Cw)

Numărul de grade a motorului pentru întoarcerea robotului 360 de grade = (Cs /Cw)*360

- În termeni generali, pentru întoarcerea un anumit număr de grade “robot_turn” :

motor_degrees = (Cs/Cw) * robot_turn

- Aceasta funcționează pentru **orice** număr de grade. Nu este un caz special pentru trecerea la 180, 360 sau orice altă valoare, așa cum am văzut când am utilizat „yaw angle”.

PROVOCARE- SPIN

- Scrie o funcție care să facă robotul efectueze o întoarcere de tip spin, în sensul acelor de ceasornic, sau în sens opus acelor de ceasornic, utilizând funcțiile MotorPair.
- Utilizează parametrii pentru a face viteza variabilă.
- Scrie teste pentru funcția ta.

PROVOCAREA – SPIN: SOLUȚIA PAGINA 1 DIN 2

```
from hub import port
import motor_pair, runloop, sys, math

# Constante pentru Drive Base 1
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
WHEEL_CIRCUMFERENCE = 17.5 # cm – vă rugăm să ajustați în conformitate cu roata robotului
TRACK = 11.2 # cm – vă rugăm să măsări pe robotul propriu.
SPIN_CIRCUMFERENCE = TRACK * math.pi

async def spin_turn(robot_degrees, motor_speed):
    # Adaugă rații ale roții de transfer dacă utilizezi asta.
    motor_degrees = int((SPIN_CIRCUMFERENCE/WHEEL_CIRCUMFERENCE) * abs(robot_degrees))
    if robot_degrees > 0:
        # întoarcere spin în sensul acelor de ceasornic
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, motor_degrees, 100, velocity=motor_speed)
    else:
        # întoarcere spin în sens opus acelor de ceasornic
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, motor_degrees, -100, velocity=motor_speed)
```

Define the imports, the globals that are fixed for your robot, and the spin_turn function that computes the motor move degrees.

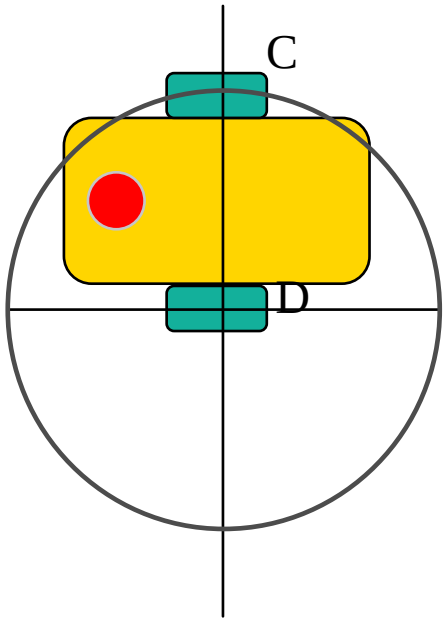
PROVOCARE – SPIN: SOLUȚIA PAGINA 2 DIN 2

```
async def main():
    # Spin 720 în sensul acelor de ceasornic cu viteza 200
    await spin_turn(720, 200)
    await runloop.sleep_ms(1000)
    # Spin 180 în sensul opus acelor de ceasornic cu viteza 200
    await spin_turn(-180, 200)
    await runloop.sleep_ms(1000)
    # Spin 90 în sensul acelor de ceasornic cu viteza 300
    await spin_turn(90, 300)
    # stop și ieși din program
    sys.exit("Stopping")

runloop.run(main())
```

Scrie testele în funcția principală. Când ești mulțumit cum merge codul, funcția poate fi adăugată la librărie dacă alegi o rotație geometrică.

GEOMETRIC PIVOT TURN



- O întoarcere pivot întoarce robotul în jurul **centrului** dintre cele 2 roți de tracțiune.
- **Raza** cercului de întoarcereeste egală cu cursa. Circumferința este dată de: Cp = 2 * Track * Pi
- Pentru a efectua o întoarcere completă (360 degrees), fiecare roată trebuie să se miște o distanță egală cu circumferința acestui cerc de întoarcere.
- Rotația cu un singur motor= rotația unei singure roți
- Dacă circumferința roții este dată de Cw, apoi:

Numărul de rotații a motorului pentru întoarcerea robotului 360 de grade = (Cs /Cw)

Numărul de grade a motorului pentru întoarcerea robotului 360 de grade = (Cs /Cw)*360

- În termeni generali, pentru întoarcerea un anumit număr de grade “robot_turn” :

motor_degrees = (Cp/Cw) * robot_turn

- Aceasta funcționează pentru **orice** număr de grade. Nu este un caz special pentru trecerea la 180, 360 sau orice altă valoare, așa cum am văzut când am utilizat „yaw angle”.

PROVOCARE - PIVOT

- Scrie o funcție pentru a face întoarceri de tip pivot, în sensul acelor de ceasornic sau în sens opus acelor de ceasornic, utilizând doar funcțiile MotorPair.
- Utilizează un parametru pentru a face viteza variabilă
- Scrieți teste pentru funcția voastră.

PROVOCARE – PIVOT: SOLUȚIA PAGINA 1 DIN 2

```
from hub import port
import motor_pair, runloop, sys, math

# Constante pentru  Drive Base 1
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
WHEEL_CIRCUMFERENCE = 17.5 # cm
TRACK = 11.2 #cm – vă rugăm să măsurați pe robotul vostru.
PIVOT_CIRCUMFERENCE = 2 * TRACK * math.pi

async def pivot_turn(robot_degrees, motor_speed):
    # Adaugă rate de multiplicare dacă folosiți roți de transfer
    motor_degrees = int((PIVOT_CIRCUMFERENCE/WHEEL_CIRCUMFERENCE) * abs(robot_degrees))
    if robot_degrees > 0:
        # pivot în sensul acelor de ceasornic
        await motor_pair.move_for_degrees(motor_pair.PAIR_1, motor_degrees, 50, velocity=motor_speed)
    else:
        #pivot în sensul opus acelor de ceasornic
await motor_pair.move_for_degrees(motor_pair.PAIR_1, motor_degrees, -50, velocity=motor_speed)
```

Definește importurile, variabilele globale sunt fixate pentru robotul vostru, și funcția pivot_turn va calcula numărul de grade pe care motorul trebuie să meargă.

PROVOCARE – PIVOT: SOLUȚIA PAGINA 2 DIN 2

```
async def main():
    # Pivot 360 în sensul acelor de ceasornic cu viteza 200
    await pivot_turn(360, 200)
    await runloop.sleep_ms(1000)
    # Pivot 360 în sensul opus acelor de ceasornic cu viteza 200
    await pivot_turn(-360, 200)
    await runloop.sleep_ms(1000)
    sys.exit("Stopping")

runloop.run(main())
```

Scrie testele în funcția principală. Când ești mulțumit cum merge codul, funcția poate fi adăugată la librărie dacă alegi o rotație geometrică.

ÎNTOARCEREA FĂRĂ GYRO – PRO ȘI CONTRA

- Pro
  - Codul este simplu, și nu e nevoie de cazuri speciale atunci când întoarcem 180, 360 etc. Funcționează orice număr de grade în orice direcție
- Contra
  - Depinde de mărimea roții robotului, distanța dintre roți și rația roții dințate. Schimbarea robotului presupune schimbarea constantelor codului. Abordarea Yaw angle nu depinde de geometria robotului, așa de mult. Totuși va trebui să faci un set-up special al brick-ului dacă acesta este orientat altfel pe robot pe DB1

- Această abordare funcționează mai bine la viteze mai mici și inadvertențele cresc mai mult, pe măsură ce viteza crește.
- Soluția pe care o alegi depinde de situație. Rulează teste și află ceea ce funcționează mai bine pentru tine.

CREDITS

- Această lecție a fost creată de Sanjay Seshan și Arvind Seshan for SPIKE Prime Lessons
- La această lecție au contribuit membrii comunității FLL Share & Learn.
- Mai multe lecții sunt disponibile pe www.primelessons.org
- Această lecție a fost tradusă în limba romană de echipa de robotică FTC – ROSOPHIA #21455 RO20