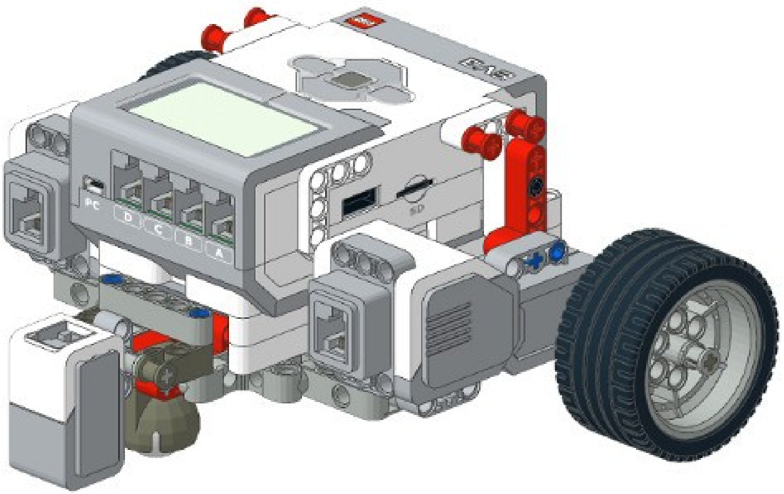
LEGO

MINDSTORMS EV3

Gyro Follower



A POSTERIORI

Play · Experience · Learn

Gyro Follower

- Very similar to line follower

Line Follower

- Look at line position
- Decide whether to...
  - Turn left
  - Turn right
  - Go straight

Gyro Follower

- Look at gyro angle
- Decide whether to...
  - Turn left
  - Turn right
  - Go straight

- Main difference; We always keep the line in the center (value = 0), but for gyro, the value depends on the direction we are following

Gyro Follower

- Uses the same algorithms as line follower (eg. 3 States, 5 States, Proportional)

```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

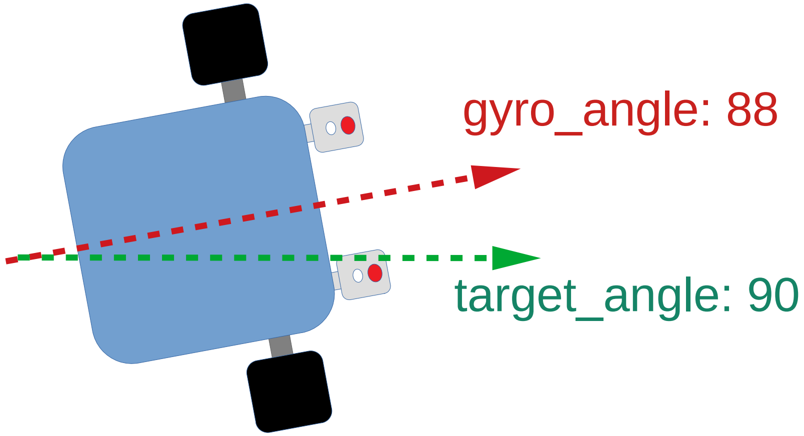
3 States

<u>Pseudo Code</u>
Don’t copy it blindly; it won’t work
Read it, understand it, write your own

Example

Target Angle is 90 degrees

Gyro angle is 88 degrees



```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

3 States

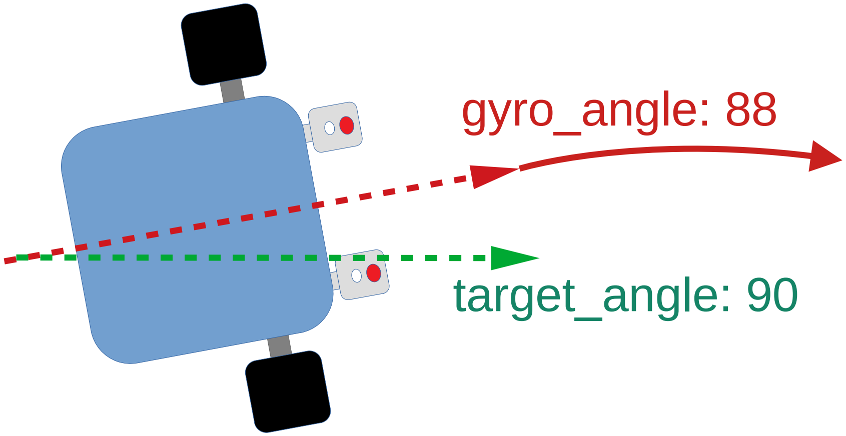
First condition is true:

- gyro_angle is less than target_angle

Example

Target Angle is 90 degrees

Gyro angle is 88 degrees



```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

3 States

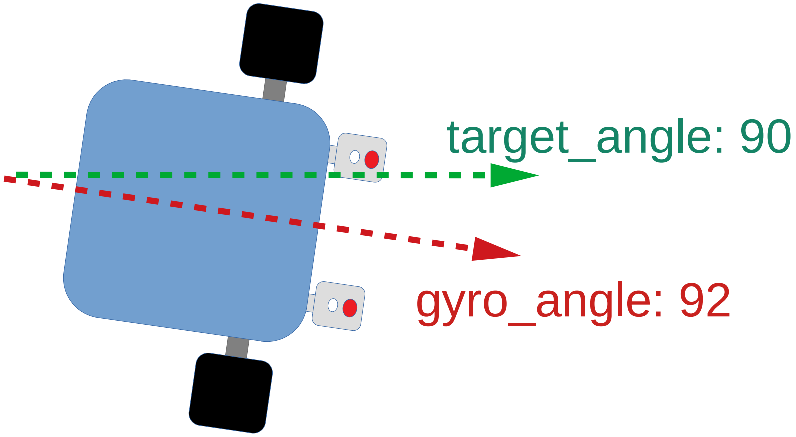
Robot turns to the right

- “move steering 10” is a slight right turn

Example

Target Angle is 90 degrees

Gyro angle is 92 degrees



```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

3 States

Second condition is true:

- gyro_angle is greater than target_angle

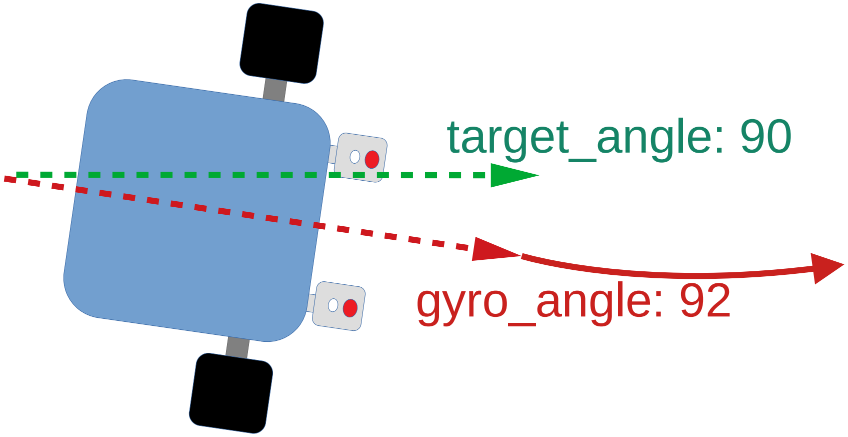
Example

Target Angle is 90 degrees

Gyro angle is 92 degrees

```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

3 States



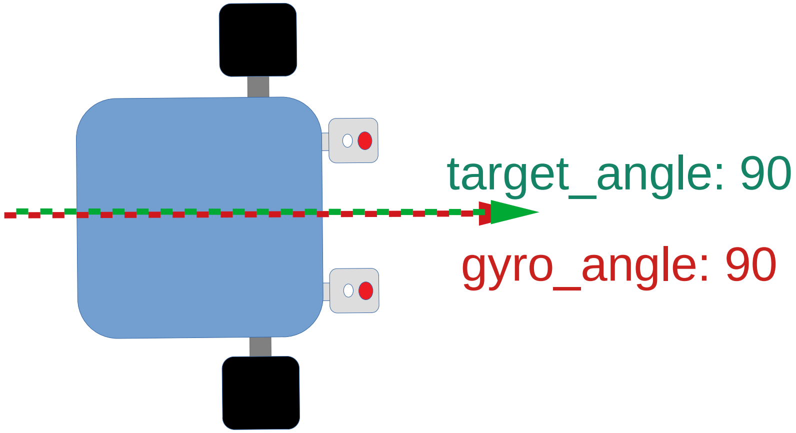
Robot turns to the left

- “move steering -10” is a slight left turn

Example

Target Angle is 90 degrees

Gyro angle is 90 degrees



```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

3 States

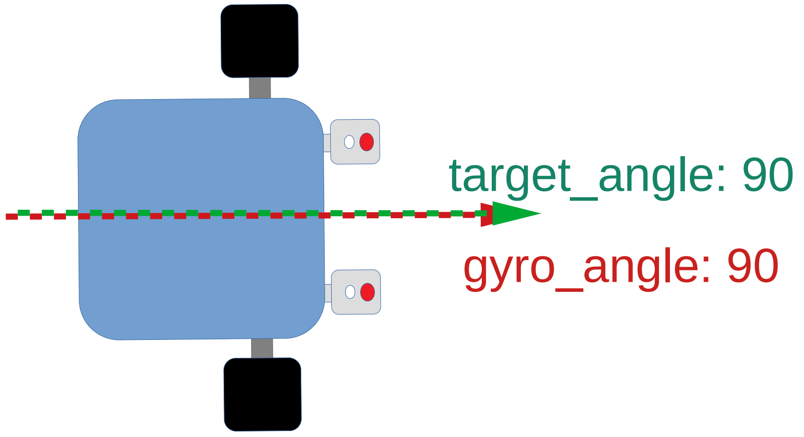
Neither the first nor second conditions are true:

- If none of the “if” matches, follow the “else” condition

Example

Target Angle is 90 degrees

Gyro angle is 90 degrees



```
def gyro_follow(target_angle, speed):
  if gyro_angle < target_angle:
    # Turn Right
    move_steering(10, speed)
  elif gyro_angle > target_angle:
    # Turn Left
    move_steering(-10, speed)
  else:
    # Go straight
    move_steering(0, speed)
```

3 States

Robot go straight

- “move steering 0” is a straight

Looping

- If you tried the program now, it won’t work
- The “gyro_follow” function only checks the gyro angle **ONE** time, then it’ll stop checking and continue moving in the same direction
- Need to use a loop to continuously check the gyro angle

```
while True:
  gyro_follow(0, 100)
```

Note

- A “while True” loop will never end, but it is useful for testing
- To make this useful, you’ll need someway to end the loop. Read the “Ending the loop” to learn how

Advanced Algorithms

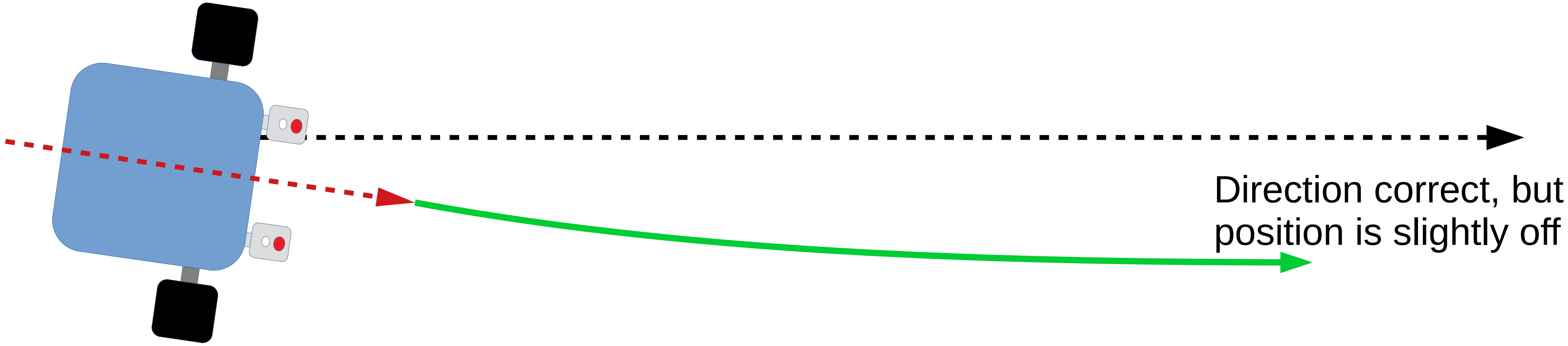
- Proportional algorithm
  - Same approach as with a line follower
  - Not as useful, as angle errors are usually only 1 degree
  - If angle errors are large after a turn, it’s better to improve your turn algorithm to be more accurate

Advanced Algorithms

- Proportional + Integral (PI) algorithm
  - Improve accuracy over longer distances
  - Not much of a difference over short to medium distances
  - Proportional / 3 States / 5 States / etc, only corrects current heading, does not correct for accumulated errors
  - PI algorithm will correct accumulated error, allowing better accuracy

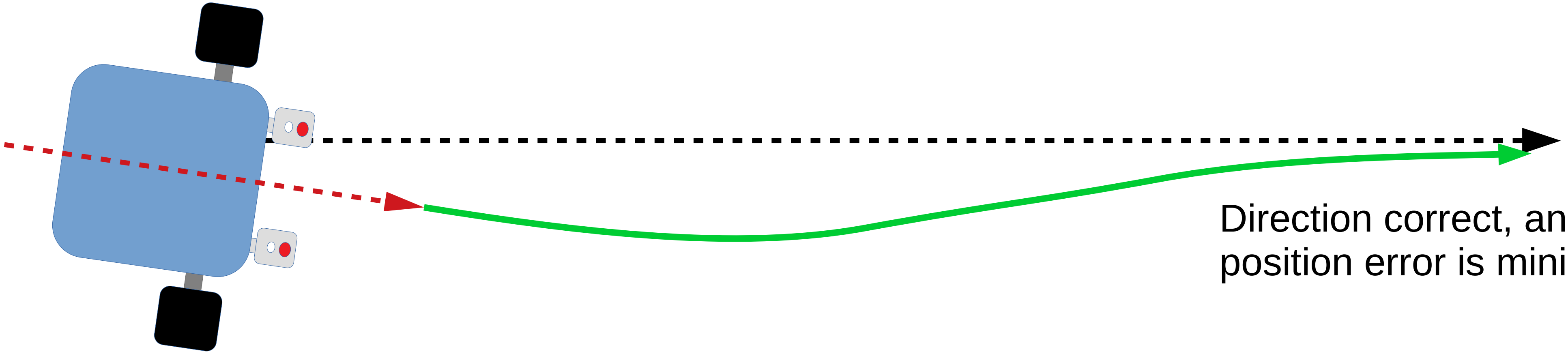
Advanced Algorithms

- Proportional / 3 States / 5 States algorithm



Direction correct, but position is slightly off

- PI (proportional + integral) algorithm



Direction correct, and position error is minimized

Angles in the EV3

- Angle when program starts is always 0 degrees
- Angles don’t rollover
  - Turning left will give -1 degree instead of 359 degrees
  - Rotating right for one round will give 360 degrees and not 0 degrees

Gyro Problems

- Not properly calibrated
  - Gyro auto-calibrates when it is plugged into the EV3
  - The reset command <u>does not</u> calibrates the gyro
  - It must be <u>perfectly still</u> during calibration
  - Don’t move it, don’t shake it, don’t even touch the table
  - If properly calibrated, the gyro reading should stay constant (...need not be zero) when the robot is not moving

Gyro Problems

- Bug in the EV3
  - Bug in the EV3 will occasionally cause the gyro to re-calibrate itself in the middle of a run
  - If it happens, any functions that relies on the gyro will go crazy
  - You can’t fix it, you can’t avoid it, but thankfully, it doesn’t happen very often

Copyright

- Created by A Posteriori LLP
- Visit http://aposteriori.com.sg/ for more tips and tutorials
-

A POSTERIORI

Play · Experience · Learn