Introduction to Robotics

Felipe P. Vista IV

Chonbuk National University

Class Admin Matters

Grading

- Attendance 5%

Name (Original Name)	User Email	Join Time	Leave Time	Duration (Minutes)
[redacted]		4/12/2021 9:12	4/12/2021 10:14	62
		4/12/2021 9:12	4/12/2021 9:14	3
		4/12/2021 9:12	4/12/2021 9:14	3
		4/12/2021 9:12	4/12/2021 9:14	3
		4/12/2021 9:12	4/12/2021 9:14	3
		4/12/2021 9:12	4/12/2021 9:14	3
		4/12/2021 9:13	4/12/2021 9:13	1
		4/12/2021 9:13	4/12/2021 9:14	2
		4/12/2021 9:14	4/12/2021 9:14	1
		4/12/2021 9:14	4/12/2021 9:14	1
		4/12/2021 9:14	4/12/2021 10:14	60

Bad ZOOM User Name (Absent)

- Iphone → Not your name
- SiAko 202100001 → Wrong order
- SiAko → Name only
- 202100001 → ID Num only

ZOOM User Name (Present)

- University ID Num_Name
- 202100001 SiAko → GOOD (Present)

Name (Original Name)	User Email	Total Duration (Minutes)
[redacted]		62
		63
		62
		62
		63
		62
		63

Class Admin Matters

Student Responsibilities

- Download/Install ZOOM app for online lecture
  - *Zoom profile must be your **OASIS ID+name** similar to OASIS*
  - *Ex.: **202061234 YourName***
  - ***If you are asked, but no reply, then you'll be out of zoom & mark absent***
- Regularly login, check OLD IEILMS for updates, notifications
  - *https://ieilmsold.jbnu.ac.kr*
  - *Presentations & lecture videos will be uploaded after class*
- Regularly check Kakao Group Chat for class
  - *Everybody must have a Kakao talk account*
  - *Search & add account "**botjok**", introduce yourself and name of class ("Robotics"), then you will be added to the group chat*

Intro To Robotics

REACTIVE BEHAVIOUR

Reactive Behaviour

- Braitenberg Vehicles
- Reacting to Detection of an Object
- Reacting & Turning
- Line Following
- Braitenberg's Presentation of the Vehicles

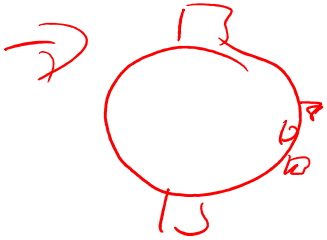
Reactive Behaviour

Intro

- Now ready for our first algorithm for robots
- Reactive behaviour
  - *Algorithm where event cause robot to react that changes its behaviour*
    - **event**: such as detecting a nearby object
    - **action**: stop the motors
- Purely reactive behaviour
  - *Action related only to event happening*
  - *Don't depend on stored data in memory (state)*
- Use Braitenberg vehicle
  - *Complex behaviour comes from simple algorithms*
- Line following
  - *Sensitive to characteristics of sensors and the lines*

Reactive Behaviour

Braitenberg Vehicles



- Valentino Braitenberg
  - *Neuroscientist, described design of virtual vehicles*
    - Suprisingly demonstrated complex behaviours
- MIT Media Lab
  - *Hardware implementation of the vehicles from programmable bricks*
    - Forerunner of LEGO® Mindstorms robotics kits
  - *Used light & touch sensors*
    - Generic robot based on horizontal proximity sensors
- Vehicle details
  - *Specification of behaviour of the robot*
  - *Formalized algorithm for specified behaviour*
  - *Activity for implementing algorithm*

Reactive Behaviour

- Braitenberg Vehicles
- Reacting to Detection of an Object
- Reacting & Turning
- Line Following
- Braitenberg's Presentation of the Vehicles

Reaction to Detection of an Object

- **Specification (Timid)**
  - *The robot moves forward if no object detected,*
    The robot stops when it detects an object.

```
1:  when object not detected in front
2:    left-motor-power ← 100
3:    right-motor-power ← 100
4:
5:  when object detected in front
6:    left-motor-power ← 0
7:    right-motor-power ← 0
```

motor power : -100 ~ 100

Algorithm 3.1: Timid

  - *Given algorithm has two event handlers*
    - Detecting an object & Not detecting an object
  - *Event handlers use of* **when** *statement as:*
    - when the event *first occurs*, perform the following actions

Reaction to Detection of an Object

- **Specification (Timid)**
  - *The robot moves forward if no object detected,*
    The robot stops when it detects an object.

```
1:  when object not detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when object detected in front
6:      left-motor-power ← 0
7:      right-motor-power ← 0
```

Algorithm 3.1: Timid

  - *Given algorithm has two event handlers*
    - Detecting an object & Not detecting an object
  - *Event handlers use of* **when** *statement as:*
    - when the event *first occurs*, perform the following actions

Reaction to Detection of an Object

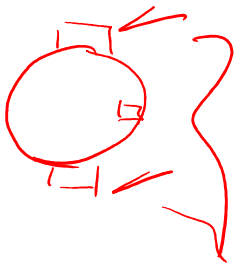
- Why not use the familiar **while**?
  - *As long as object is not detected, motors will be turned on*
    As long as object is detected, motors will be turned off
  - *Motors repeatedly turn On/Off*
    - Sensor detect object varied distances
    - Repeated comm don't damage motor
    - But waste resources
  - *Preferred command to motor*
    - "Off" at 1st detection of object
    - "On" at 1st non-detection of object
  - Therefore **when** give semantics we want

```
1:  while object not detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  while object detected in front
6:      left-motor-power ← 0
7:      right-motor-power ← 0
```

Algorithm 3.2: Timid with while

Reaction to Detection of an Object

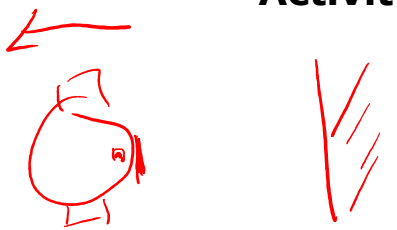
- **Specification (Indecisive)**
  - *The robot moves forward if no object detected,*
    The robot moves backward when it detects an object.



Power :
-100 ~ 0 ~ 100

```
1:  when object not detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when object detected in front
6:      left-motor-power ← -100
7:      right-motor-power ← -100
```

Activity 3.2: Indecisive



Reaction to Detection of an Object

- **Specification (Indecisive)**
  - *The robot moves forward if no object detected,*
    The robot moves backward when it detects an object.

```
1:  when object not detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when object detected in front
6:      left-motor-power ← -100
7:      right-motor-power ← -100
```

Activity 3.2: Indecisive

Reactive Behaviour

Reaction to Detection of an Object

- **Specification (Indecisive & Oscillating)**
  - *At just right distance, robot will **oscillate***
    - Move forwards and backwards in quick succession

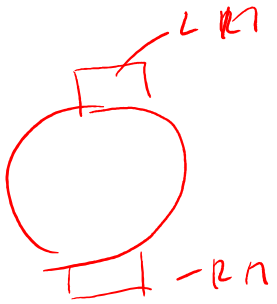
```
1:  when object not detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when object detected in front
6:      left-motor-power ← -100
7:      right-motor-power ← -100
8:
9:  when distance = rDist
10:     left-motor-power ←
11:     right-motor-power ←
10:     left-motor-power ←
11:     right-motor-power ←
```

Oscillating

Activity 3.2: Indecisive w/ oscillation

Reaction to Detection of an Object

- **Specification (Indecisive & Oscillating)**
  - *At just right distance, robot will **oscillate***
    - Move forwards and backwards in quick succession



```
1:   when object not detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:   when object detected in front
6:      left-motor-power ← -100
7:      right-motor-power ← -100
8:
9:   when distance = rDist
10:     left-motor-power ← 100
11:     right-motor-power ← 100
10:     left-motor-power ← -100
11:     right-motor-power ← -100
```

Activity 3.2: Indecisive w/ oscillation

Reaction to Detection of an Object

- **Specification (Dogged)** Dodge
  - *The robot moves forward if object detected at the back,*
    The robot moves backward if object detected in front.

```
1:  when object detected in front
2:      left-motor-power ← -100
3:      right-motor-power ← -100
4:
5:  when object detected in back
6:      left-motor-power ← 100
7:      right-motor-power ← 100
```

Activity 3.3: Dogged

~p = reverse
~0 = stop
+p = forward

Reaction to Detection of an Object

- **Specification (Dogged)**
  - *The robot moves forward if object detected at the back,*
    The robot moves backward if object detected in front.

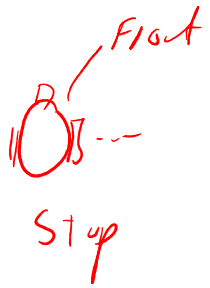
```
1:  when object detected in front
2:      left-motor-power ← -100
3:      right-motor-power ← -100
4:
5:  when object detected in back
6:      left-motor-power ← 100
7:      right-motor-power ← 100
```

Activity 3.3: Dogged

Reactive Behaviour

Reaction to Detection of an Object

- **Specification (Dogged Stop)**
  - *The robot moves forward if object detected at the back,*
    The robot moves backward if object detected in front,
    The robot stops if no object detected.




```
1:  when object detected in front
2:      left-motor-power ← -50
3:      right-motor-power ← -50
4:
5:  when object detected in back
6:      left-motor-power ← 45
7:      right-motor-power ← 45
8:
9:  when no object detected
6:      left-motor-power ← 0
7:      right-motor-power ← 0
```

Activity 3.4: Dogged (Stop)

Reaction to Detection of an Object

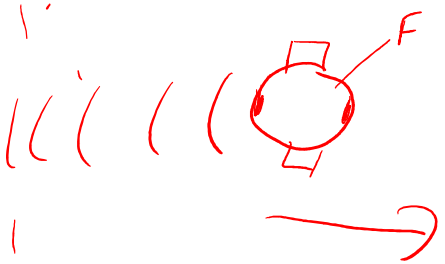
- **Specification (Dogged Stop)**
  - *The robot moves forward if object detected at the back,*
    The robot moves backward if object detected in front,
    The robot stops if no object detected.

```
1:  when object detected in front
2:      left-motor-power ← -100
3:      right-motor-power ← -100
4:
5:  when object detected in back
6:      left-motor-power ← 100
7:      right-motor-power ← 100
8:
9:  when no object detected
6:      left-motor-power ← 0
7:      right-motor-power ← 0
```

Activity 3.4: Dogged (Stop)

Reaction to Detection of an Object

- **Specification (Attractive & Repulsive)**
  - *If object approaches from behind,*
    Robot runs away until it is out of range.




```
1:  when object detected in back
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when no object detected in back
6:      left-motor-power ← 0
7:      right-motor-power ← 0
```

Activity 3.5: Attractive & Repulsive

Reaction to Detection of an Object

- **Specification (Attractive & Repulsive)**
  - *If object approaches from behind,*
    Robot runs away until it is out of range.

```
1:  when object detected in back
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when no object detected in back
6:      left-motor-power ← 0
7:      right-motor-power ← 0
```

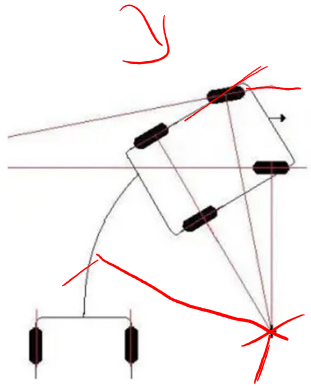
Activity 3.5: Attractive & Repulsive

Reactive Behaviour

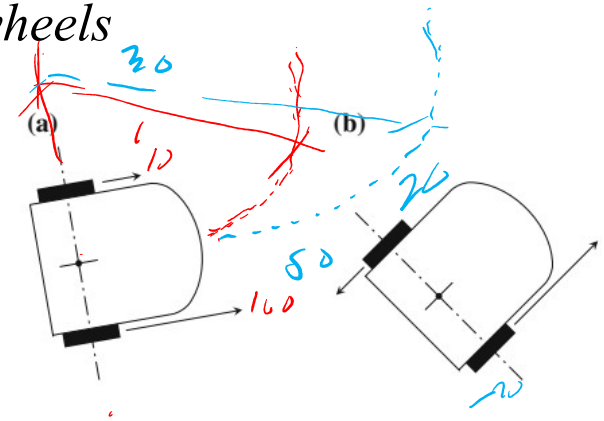
- Braitenberg Vehicles
- Reacting to Detection of an Object
- Reacting & Turning
- Line Following
- Braitenberg's Presentation of the Vehicles

Reactive Behaviour

Reacting & Turning

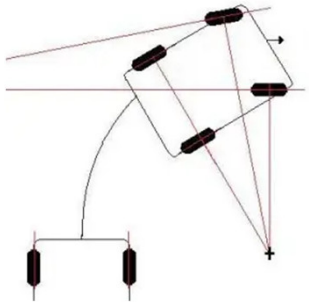


- Car turns by
  - *Angle of front wheels relative to frame (same motor power*
- Robot w/ differential drive turn by
  - *Setting different level of power to left and right wheels*
  - *(a) One wheel turning faster than the other*
    - Turn direction opposite of faster wheel
  - *(b) One wheel turns rev, other fwd*
    - Turn much faster

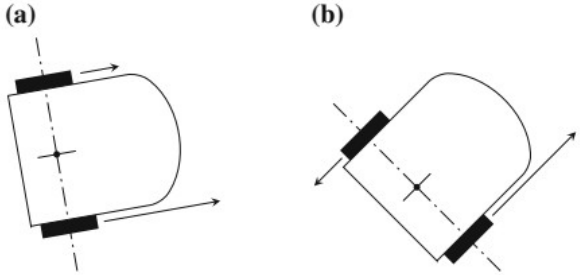


- Turning radius
  - *Radius of circle that is path of robot; "turn is tighter if radius is smaller"*
  - *m1 turn fwd, m2 turn backward, same speed?*
    - Robot turn , turning radius is

Reacting & Turning



- Car turns by
  - *Angle of front wheels relative to frame (same motor power*
- Robot w/ differential drive turn by
  - *Setting different level of power to left and right wheels*
  - *(a) One wheel turning faster than the other*
    - Turn direction opposite of faster wheel
  - *(b) One wheel turns rev, other fwd*
    - Turn much faster



- Turning radius
  - *Radius of circle that is path of robot; "turn is tighter if radius is smaller"*
  - *m1 turn fwd, m2 turn backward, same speed?*
    - Robot turn **in place**, turning radius is **zero.**

Reacting and Turning

- **Specification (Paranoid)**
  - *If robot detects an object, it moves forward (colliding with the object),*
    If robot does not detect an object, it turns left.

```
1:  when object detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when no object detected in front
6:      left-motor-power ← 20 / 10
7:      right-motor-power ← 80 / 100
```

Algorithm 3.3: Paranoid

Reacting and Turning

- **Specification (Paranoid)**
  - *If robot detects an object, it moves forward (colliding with the object),*
    If robot does not detect an object, it turns left.

```
1:  when object detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when no object detected in front
6:      left-motor-power ← -50  // left motor reverse
7:      right-motor-power ← 50 // right motor forward
```

Algorithm 3.3: Paranoid

Reacting and Turning

- **Specification (Paranoid Right-Left)**
  - *If robot detects an object in front, it moves forward,*
    If object detected to it's right, robot turns right,
    If object detected to it's left, robot turns left,
    If robot does not detect an object, it stops.

```
1:   when object detected in front
2:      left-motor-power ←
3:      right-motor-power ←
4:
5:   when object detected in left side
6:      left-motor-power ←
7:      right-motor-power ←
8:
```

```
9:   when object detected in right side
10:     left-motor-power ←
11:     right-motor-power ←
12:
13:  when no object detected
14:     left-motor-power ←
15:     right-motor-power ←
```

Activity 3.7: Paranoid (right-left)

Reacting and Turning

- **Specification (Paranoid Right-Left)**
  - *If robot detects an object in front, it moves forward,*
    If object detected to it's right, robot turns right,
    If object detected to it's left, robot turns left,
    If robot does not detect an object, it stops.

```
1:   when object detected in front
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:   when object detected in left side
6:      left-motor-power ← -50  // reverse
7:      right-motor-power ← 50 // forward
8:
```

```
9:   when object detected in right side
10:     left-motor-power ← 50    // forward
11:     right-motor-power ← -50 // reverse
12:
13:  when no object detected
14:     left-motor-power ← 0
15:     right-motor-power ← 0
```

Activity 3.7: Paranoid (right-left)

Reacting and Turning

- **Specification (Insecure)**
  - *If robot does not detect an object to it's left,*
    set right motor to rotate forwards & left motor off.
    If robot detects an object to it's left,
    set right motor off & set left motor to rotate forwards.

```
1:  when object not detected in left side
2:      left-motor-power ← 0
3:      right-motor-power ← 1~100
4:
5:  when object detected in left side
6:      left-motor-power ← 1~100
7:      right-motor-power ← 0
8:
```

Activity 3.8: Insecure

Reacting and Turning

- **Specification (Insecure)**
  - *If robot does not detect an object to it's left,*
    set right motor to rotate forwards & left motor off.
    If robot detects an object to it's left,
    set right motor off & set left motor to rotate forwards.

```
1:  when object not detected in left side
2:      left-motor-power ← 0
3:      right-motor-power ← 50
4:
5:  when object detected in left side
6:      left-motor-power ← 50
7:      right-motor-power ← 0
8:
```

Activity 3.8: Insecure

Reacting and Turning

- **Specification (Driven)**
  - *If robot detect an object to it's left,*
    set right motor to rotate forwards & left motor off.
    If robot detects an object to it's right,
    set right motor off & set left motor to rotate forwards.

```
1:  when object detected in left side
2:      left-motor-power ←
3:      right-motor-power ←
4:
5:  when object detected in right side
6:      left-motor-power ←
7:      right-motor-power ←
8:
```

Activity 3.9: Driven

Reacting and Turning

- **Specification (Driven)**
  - *If robot detect an object to it's left,*
    set right motor to rotate forwards & left motor off.
    If robot detects an object to it's right,
    set right motor off & set left motor to rotate forwards.

```
1:  when object detected in left side
2:      left-motor-power ←
3:      right-motor-power ←
4:
5:  when object detected in right side
6:      left-motor-power ←
7:      right-motor-power ←
8:
```

Activity 3.9: Driven

Reacting and Turning

- **Specification (Driven)**
  - *If robot detect an object to it's left,*
    set right motor to rotate forwards & left motor off.
    If robot detects an object to it's right,
    set right motor off & set left motor to rotate forwards.

```
1:  when object detected in left side
2:      left-motor-power ← 0
3:      right-motor-power ← 100
4:
5:  when object detected in right side
6:      left-motor-power ← 100
7:      right-motor-power ← 0
8:
```

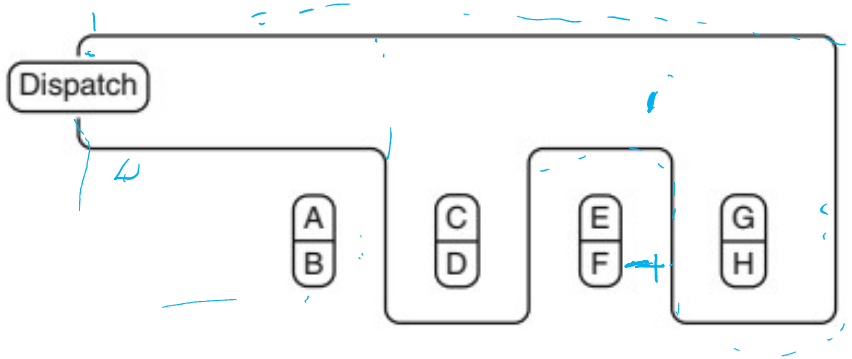
Activity 3.9: Driven

Reactive Behaviour

- Braitenberg Vehicles
- Reacting to Detection of an Object
- Reacting & Turning
- Line Following
- Braitenberg's Presentation of the Vehicles

Reactive Behaviour

Line Following



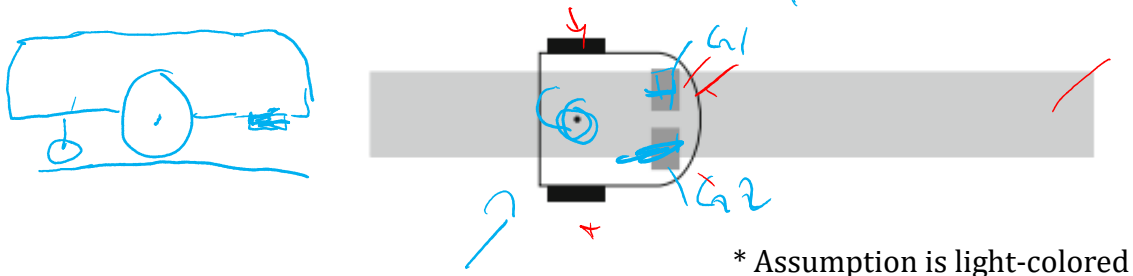
Warehouse w/ robotic carts bring items to central dispatching area

- *Lines painted on floor*
- *Robot follow lines until it reach the storage of desired item*

- Bring out all uncertainty of constructing robots in real-world
  - *Line not perfectly straight or dust may cover part of line;*
  - *Dirt cause a wheel to move slowly than the other*
  - *Robot must know if on line or not*
  - *If it starts to leave the line*
    - must turn in correct direction in order to regain line

Line Following using pair of Ground Sensors

- Ground sensor on **light-colored** floor detects lot of **reflected light**
  - *Sensor detect very little reflected light whenever over a **dark line***
  - *Darker line for increased **contrast** with white floor*
    - Line should be black but shown as gray in figure for representation purposes
  - *For two ground sensors:*
    *line **wide enough** so both sensors can sense dark line at same time*
  - ***Threshold** determine when sensor detect the line or detect the floor*
  - *Sensors **don't need** to be **totally** over line*
  - *It will be **enough** if light reflected from line below threshold for "black"*



Robot w/ two ground sensors over a line

* Assumption is light-colored floor. If dark-colored floor, then line should be white.

Reactive Behaviour

Line Following using pair of Ground Sensors

- Robot move **forward** when both sensors detect dark surface
  - *Means it is over the line*
- Robot starts to **leave** the line
  - *Either left or right ground sensor will leave line first*
- If robot moves off the line to the **left**
  - ***left** sensor will not detect line while **right** sensor still detecting it*
    → robot must turn **right**
- If robot moves off the line to the **right**
  - ***right** sensor will not detect line, while **left** sensor keep detecting it*
    → robot must turn **left**

Robot leaving the line

Line Following using pair of Ground Sensors

- **Specification (Line Following)**
  - *If sensors detect line, move forward.*
    If line not detected, robot stops.
    When robot starts to move off to the left, robot turn to the right.
    When robot starts to move off to the right, robot turn to the left.

```
1:  when both sensors detect black
2:      left-motor-power ← 50
3:      right-motor-power ← 50
4:
5:  when neither sensors detect black
6:      left-motor-power ← 0
7:      right-motor-power ← 0
8:
```

```
9:  when only left sensor detect black
10:     left-motor-power ← 20
11:     right-motor-power ← 50
12:
13: when only right sensor detect black
14:     left-motor-power ← 30
15:     right-motor-power ← 20
```

Algorithm 3.4: Line Following w/ two sensors

Reactive Behaviour

Line Following using pair of Ground Sensors

- **Specification (Line Following)**
  - *If sensors detect line, move forward.*
    If line not detected, robot stops.
    When robot starts to move off to the left, robot turn to the right.
    When robot starts to move off to the right, robot turn to the left.

```
1:  when both sensors detect black
2:      left-motor-power ← 100
3:      right-motor-power ← 100
4:
5:  when neither sensors detect black
6:      left-motor-power ← 0
7:      right-motor-power ← 0
8:
```

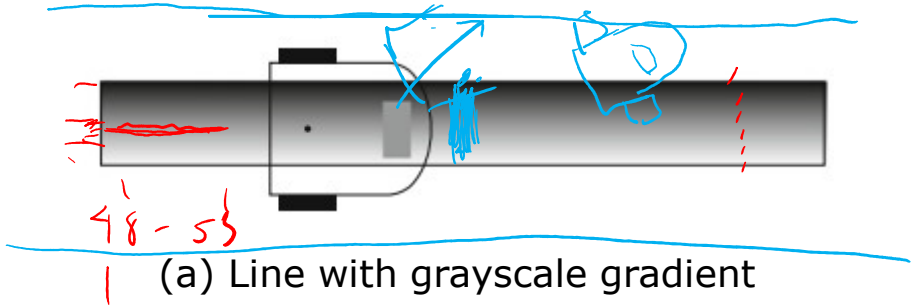
```
9:  when only left sensor detect black
10:     left-motor-power ← 0
11:     right-motor-power ← 50
12:
13: when only right sensor detect black
14:     left-motor-power ← 50
15:     right-motor-power ← 0
```

Algorithm 3.4: Line Following w/ two sensors

Reactive Behaviour

Line Following with only one Ground Sensor

- Robot can follow line with **only one** ground sensor
  - *If reflectivity of line varies across its width*
- **Ex:** Shade of line varies from fully black to fully white (a)
  - *Ground sensor returns 0 ~ 100, depending w/c part of line it is over*
  - *Sensor will return 50 if directly over center of line*
    - Don't expect to be always 50
    - Robot does not need to turn left/right unless value approaches 0 or 100
- Thresholds
  - ***Black-threshold**: below this value*
    - Robot leaving left side of the line
  - ***White-threshold**: above this value*
    - Robot leaving right side of the line



(a) Line with grayscale gradient

Line Following with only one Ground Sensor

- **Specification (Line Following w/ One Ground Sensor)**
  - *If sensor value greater than **black-threshold** and also less than **white-threshold**, move forward.*
    *If value greater than **white-threshold**, turn left.*
    *If value lesser than **black-threshold**, turn right.*

```
integer black-threshold ←
integer white-threshold ←
1:   when black-threshold ≤
       sensorValue ≤ white-threshold
2:     left-motor-power ←
3:     right-motor-power ←
4:
```

```
5:   when sensorValue > white-threshold
6:     left-motor-power ←
7:     right-motor-power ←
8:
9:   when black-threshold < sensorValue
10:    left-motor-power ←
11:    right-motor-power ←
12:
```

Algorithm 3.5: Line Following w/ one sensor

Line Following with only one Ground Sensor

- **Specification (Line Following w/ One Ground Sensor)**
  - *If sensor value greater than black-threshold and also less than white-threshold, move forward.*
    If value greater than white-threshold, turn left.
    If value lesser than black-threshold, turn right.

```
integer black-threshold ← 20
integer white-threshold ← 80
1:   when black-threshold ≤
       sensorValue ≤ white-threshold
2:     left-motor-power ← 100
3:     right-motor-power ← 100
4:
5:   when sensorValue > white-threshold
6:     left-motor-power ← -50
7:     right-motor-power ← 50
8:
9:   when black-threshold < sensorValue
10:    left-motor-power ← 50
11:    right-motor-power ← -50
12:
```

Algorithm 3.5: Line Following w/ one sensor

Line Following Without a Gradient

- ***Aperture*** of receiver of proximity sensor
  - *Opening through which light is collected*
  - *Often wide to allow more light to into the sensor*
    - So that responsive to low-light levels
- **F-steps**:
  - *The lower the f-stop, the wider the aperture*
    - Therefore pictures can be taken in relatively dark environments
- With a relatively wide aperture
  - *Gradient in the line is not needed anymore*
- Single sensor
  - *Can be utilized to follow the **edge** of a line*

Reactive Behaviour

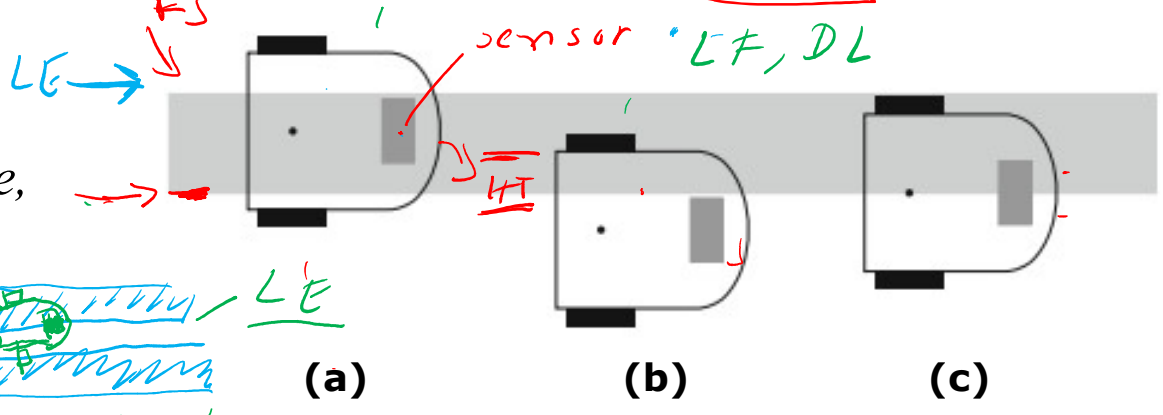
Line Following Without a Gradient

- **Ex:** Line (**right edge**) following w/ one sensor & no gradient

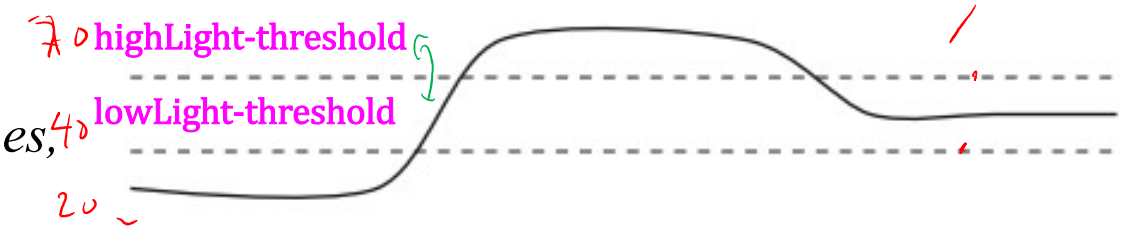
a) Sensor over the line:
   - *Little light sensed,*
   - ∴ *too far left to right edge,*
   - *Robot must turn right.*

b) Sensor off the line:
   - *Lot of light sensed,*
   - ∴ *too far right to right edge,*
   - *Robot must turn left.*

c) Sensor over right edge of line
   - *Light sensed bet two extremes,*
   - ∴ *robot where it should be,*
   - *Robot continue forward.*



Robot moving over the line (***right edge***)

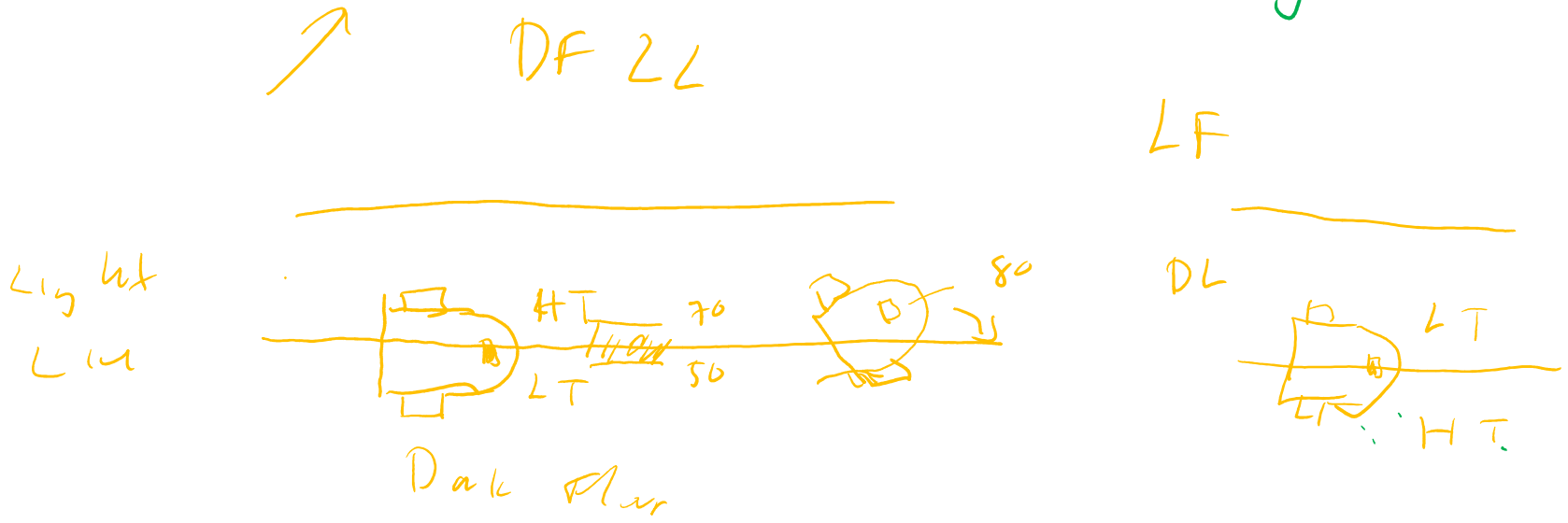


Plot of sensor values vs distance

Reactive Behaviour

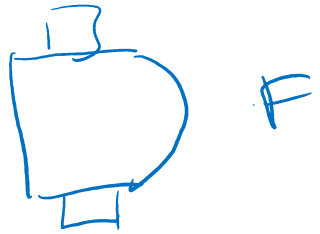
Line following without a gradient

- **Specification (Line Following w/o Gradient)**
  - *If sensor value greater than lowLight-threshold and also less than highLight-threshold, move forward.*
    If value lower than lowLight-threshold, turn right.
    If value greater than highLight-threshold, turn left.



Line following without a gradient

- **Specification (Line Following w/o Gradient)**
  - *If sensor value greater than lowLight-threshold and also less than highLight-threshold, move forward.*
  - *If value lower than lowLight-threshold, turn right.*
  - *If value greater than highLight-threshold, turn left.*




```
integer lowLight-threshold ← 20
integer highLight-threshold ← 80
1:   when lowLight-threshold ≤
       sensorValue ≤ highLight-threshold
2:      left-motor-power ←
3:      right-motor-power ←
4:
5:   when sensorValue < lowLight-threshold
6:      left-motor-power ←
7:      right-motor-power ←
8:
9:   when sensorValue > highLight-threshold
10:     left-motor-power ←
11:     right-motor-power ←
12:
```

Algorithm 3.6: Line Following w/o gradient

Line following without a gradient

- **Specification (Line Following w/o Gradient)**
  - *If sensor value greater than **lowLight-threshold** and also less than **highLight-threshold**, move forward.*
    *If value lower than **lowLight-threshold**, turn right.*
    *If value greater than **highLight-threshold**, turn left.*

```
integer lowLight-threshold ← 20
integer highLight-threshold ← 80
1:   when lowLight-threshold ≤
       sensorValue ≤ highLight-threshold
2:     left-motor-power ← 100
3:     right-motor-power ← 100
4:
5:   when sensorValue < lowLight-threshold
6:     left-motor-power ← 50
7:     right-motor-power ← -50
8:
9:   when sensorValue > highLight-threshold
10:    left-motor-power ← 50
11:    right-motor-power ← -50
12:
```

Algorithm 3.6: Line Following w/o gradient

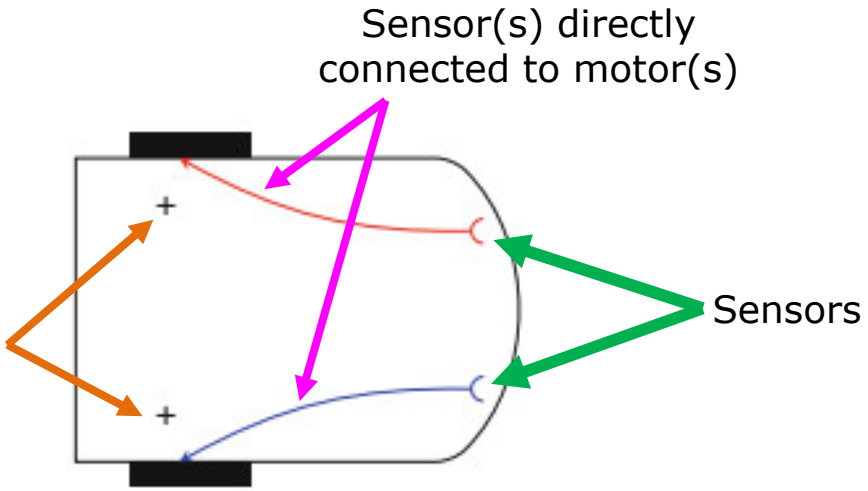
Reactive Behaviour

- Braitenberg Vehicles
- Reacting to Detection of an Object
- Reacting & Turning
- Line Following
- Braitenberg's Presentation of the Vehicles

Braitenberg's Presentation of Vehicles

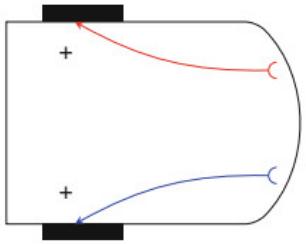
- Conducted as thought experiments
  - *Not for implementation w/ electronic components*
  - *Not for implementation in software*
- Sensors directly connected to motors
  - *As in nervous system of living creatures*
- Some also with memory
  - *Similar to brain*

(+) – if higher sensor data value, faster wheel turning.
(-) – if lower sensor data value, slower wheel turning.



Braitenberg's Presentation

- Demonstration of Braitenberg's presentation
  - *Light sensors directly connected to motor(s) of the wheel(s)*
  - *The more light detected, the **faster** the wheel will turn*
  - *Strong light source straight ahead*
    - Both sensors return same value, move forward **fast**

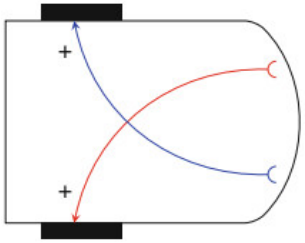


(a)

- If light source is off to the left

  ***a) Coward** Vehicle*
    - Left wheel turns rapidly, right wheel turns slowly
    - Robot turn sharply away from light source

  ***b) Aggressive** Vehicle*
    - Left wheel turns slowly, right wheel turns rapidly
    - Robot turns toward the light source



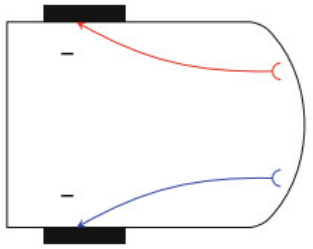
(b)

Braitenberg's Presentation

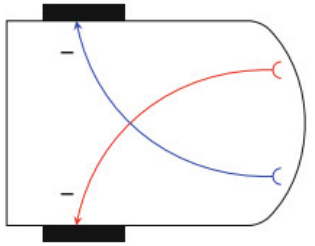
- Demonstration of Braitenberg's presentation
  - *Light sensors directly connected to motor(s) of the wheel(s)*
  - *The more light detected, the **slower** the wheel will turn*
  - *Strong light source straight ahead*
    - Both sensors return same value, move forward **slow**
- If light source is off to the left

  ***a) Loves** Vehicle*
    - Left wheel turns slowly, right wheel turns rapidly
    - Robot turns towards the light source

  ***b) Explorer** Vehicle*
    - Left wheel turns rapidly, right wheel turns slowly
    - Robot turns sharply away from the light source



(a)



(b)

Reactive Behaviour

Summary

- Reactive behaviour
  - *Exhibited if actions depend only on returned sensor values*
- Braitenberg Vehicles
  - *Change setting of motors related to sensor value*
  - *Demonstrate that complex behaviour can result from simple reactive algo's*
- Line following
  - *Fundamental task in robotics*
  - *Use landmarks (i.e. lines) in ensuring movement to desired location*
  - *Reactive behaviour (reacts due to value from ground sensors)*
  - *Sample algorithms given*
    - Depends on sensor threshold & motor speeds
    - Detemined by experimentation

Thank you.