4K450
Embedded Motion Control

Presentation 2:
Coding & Composition Pattern

Group 11

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Content

- Review of the passed test
- Function block – Behaviour and Structure
- Composition pattern & Hierarchy
- Code executing and loop
- Example & Scenario
- Work to do

Review of the passed test

- **What did we get from the passed test?**

The “move ahead” function was executed in open loop, with too accurate drift control. The robot crashed into the wall.

The robot slide was not controlled, the judgement condition code needs to be rebuilt.

@TECHUNITED
TU/e Technische Universiteit Eindhoven University of Technology
BECKHOFF

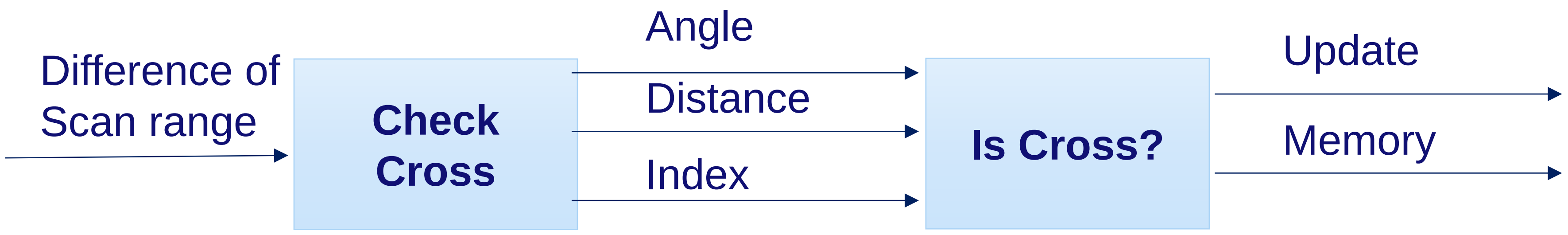
Function block – Behaviour and Structure

- **Ten functions:**

1. Find minimum value of sensor (“index”)
2. Judge the robot is in the center of the road
3. Save data (including map, robot situation, etc.)
4. Check and judge cross
5. Calculate the location and situation of robot (“update”)
6. Make the robot back to the center of the road (Feedback location)
7. Align the robot coordinate system (Feedback angle)
8. Turn around & J-turn
9. Check exit (Leave the maze)
10. Check dead end (J-turn)

Function block – Behaviour and Structure

- **Some examples of Function Block:**



Difference of Scan range

Check Cross

Angle

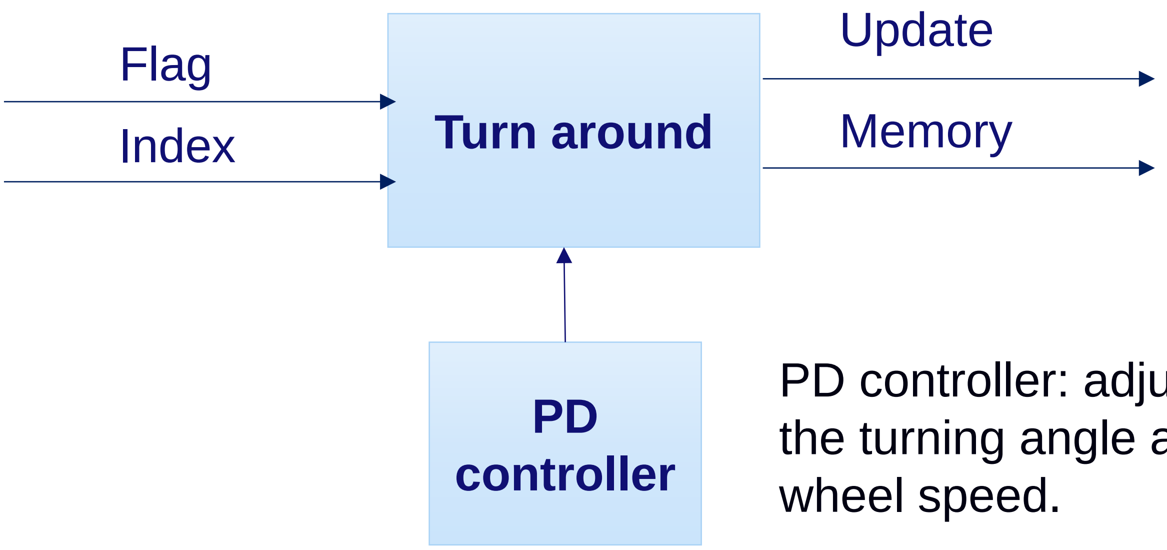
Distance

Index

Is Cross?

Update

Memory



Flag

Index

Turn around

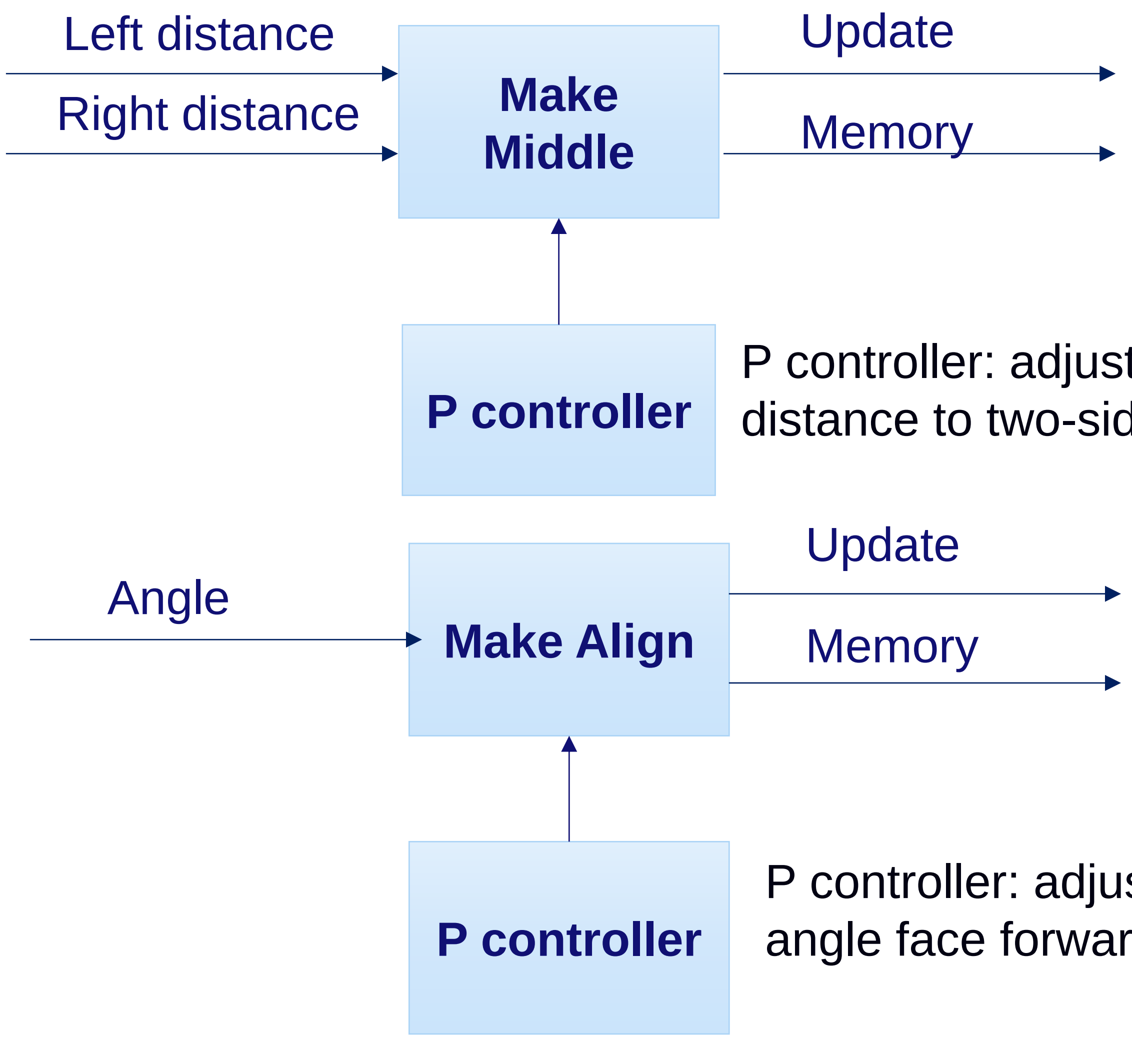
Update

Memory

PD controller

PD controller: adjust the turning angle and wheel speed.

Function block – Behaviour and Structure



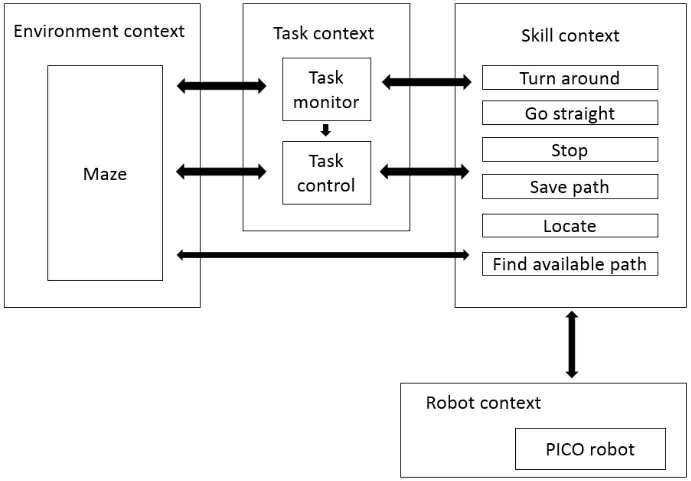
P controller: adjust the distance to two-side-walls

P controller: adjust the angle face forward.

Composition Pattern & Hierarchy

Developed and Gain

- **Smooth ↗**
- **Robustness ↗**
- **Fast ↗**
- **Optimization ↗**
- **Software structure ↗**



Environment context

Maze

Task context

Task monitor

Task control

Skill context

Turn around

Go straight

Stop

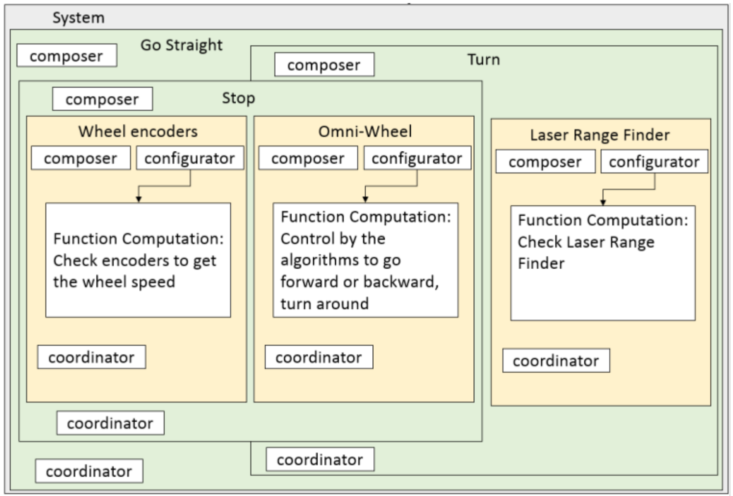
Save path

Locate

Find available path

Robot context

PICO robot



System

Go Straight

composer

Turn

composer

Stop

composer

Wheel encoders

composer

configurator

Function Computation: Check encoders to get the wheel speed

coordinator

Omni-Wheel

composer

configurator

Function Computation: Control by the algorithms to go forward or backward, turn around

coordinator

Laser Range Finder

composer

configurator

Function Computation: Check Laser Range Finder

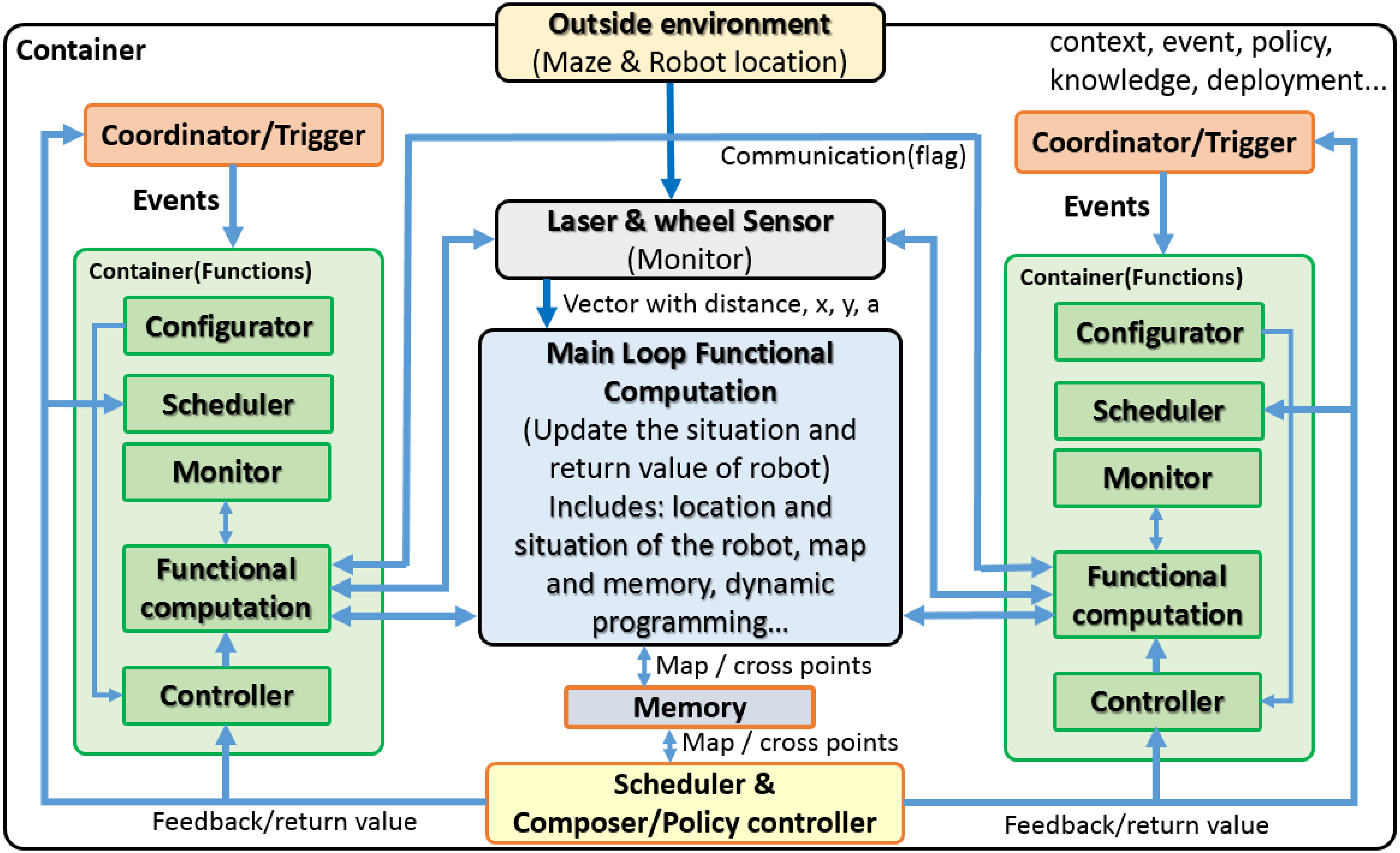
coordinator

coordinator

coordinator

coordinator

Composition Pattern & Hierarchy



Container

Outside environment
(Maze & Robot location)

context, event, policy, knowledge, deployment...

Coordinator/Trigger

Events

Communication(flag)

Coordinator/Trigger

Events

Laser & wheel Sensor
(Monitor)

Vector with distance, x, y, a

Container(Functions)

Configurator

Scheduler

Monitor

Functional computation

Controller

Main Loop Functional Computation
(Update the situation and return value of robot)
Includes: location and situation of the robot, map and memory, dynamic programming...

Container(Functions)

Configurator

Scheduler

Monitor

Functional computation

Controller

Map / cross points

Memory

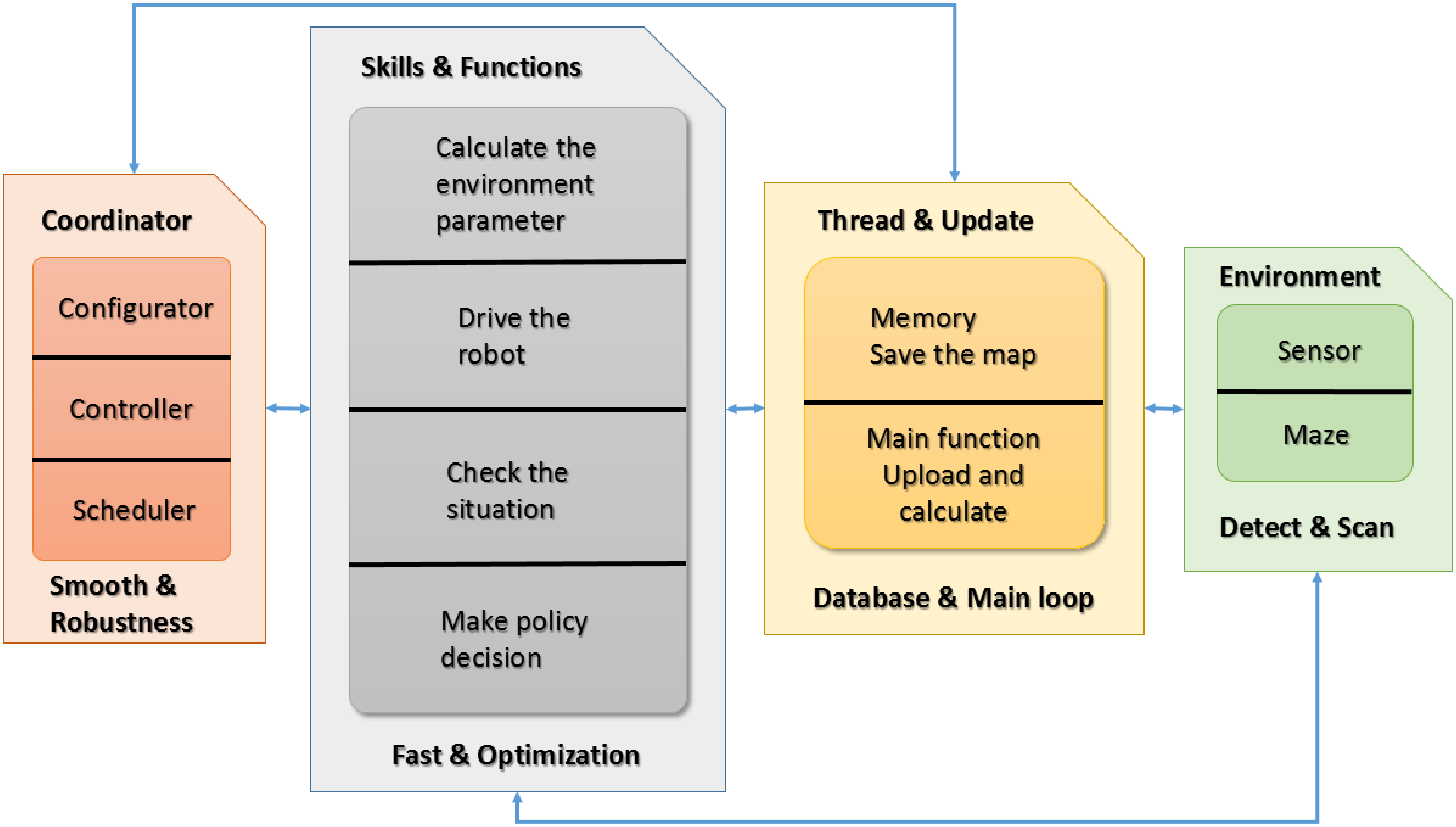
Map / cross points

Scheduler & Composer/Policy controller

Feedback/return value

Feedback/return value

Composition Pattern & Hierarchy



Coordinator

Configurator

Controller

Scheduler

Smooth & Robustness

Skills & Functions

Calculate the environment parameter

Drive the robot

Check the situation

Make policy decision

Fast & Optimization

Thread & Update

Memory
Save the map

Main function
Upload and calculate

Database & Main loop

Environment

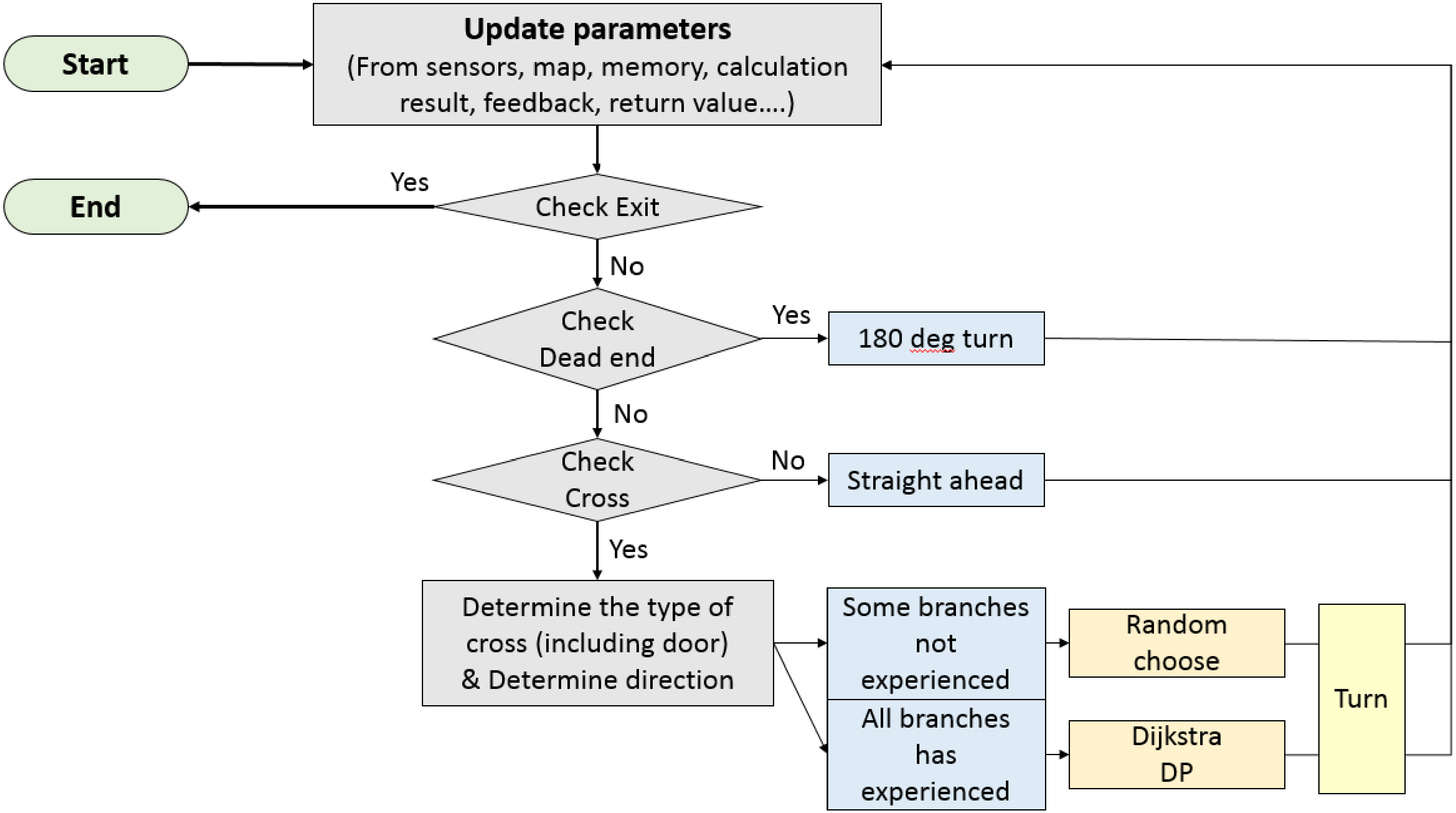
Sensor

Maze

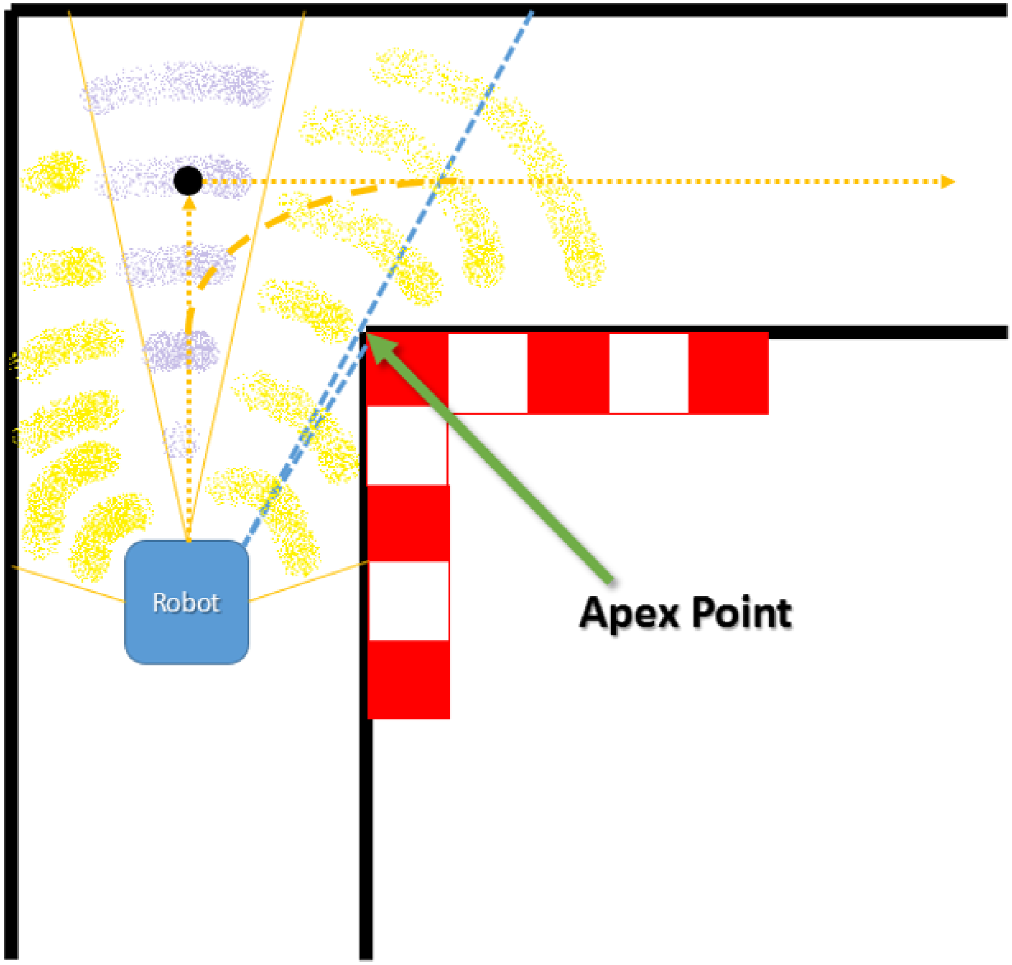
Detect & Scan

Code executing and loop

- **It is based on the scheduler and priority of tasks:**


Start
Update parameters
(From sensors, map, memory, calculation result, feedback, return value....)
End
Yes
Check Exit
No
Check Dead end
Yes
180 deg turn
No
Check Cross
No
Straight ahead
Yes
Determine the type of cross (including door) & Determine direction
Some branches not experienced
All branches has experienced
Random choose
Dijkstra DP
Turn

Example & Scenario



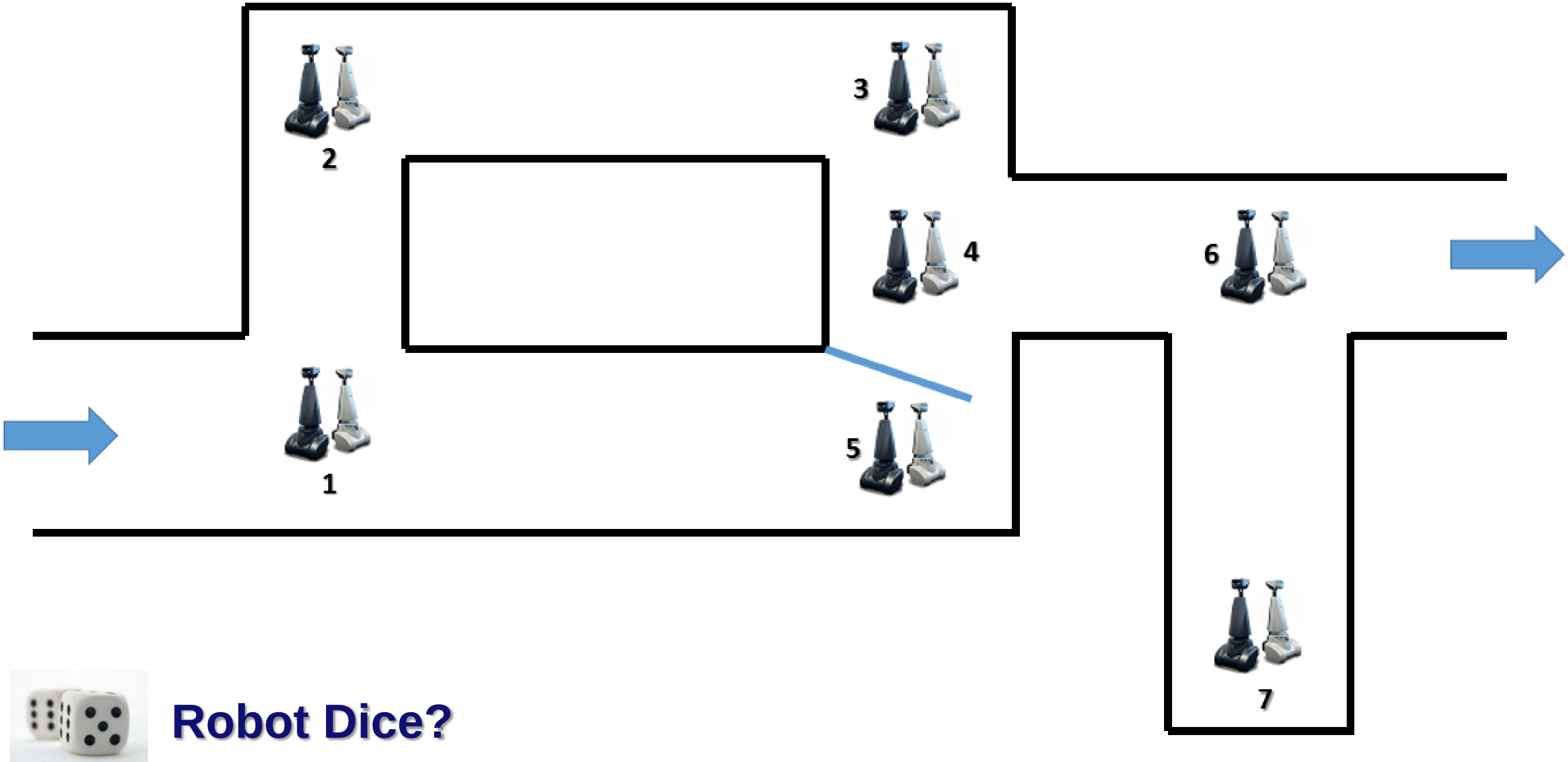
Detect range:
-20°-80°,20°-80°

Check the differences between the sensor return values

Oval trace

Memory the corner for DP

Example & Scenario



Robot Dice?

Work to do…

- Save Map: save the paths & get whole maze map in memory
- DP: Dynamic Programming, based on map
- Controller: change PD controller parameters, or using more advanced controller
- Door judgement and make corresponding door function
- Maybe more functions…

Thank you!

Any questions?

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts