Soirée Pratique
Build your own robot

Session 4: Brains (Programming)

http://www.ieee-sb-leuven.be/soireepratiques

IEEE Student Branch Leuven

Roadmap SP 2014-2015

last semester

1. the brains: Arduino
2. the muscles: motor and power
3. the eyes: sensors

this semester

1. **more brains: programming** (16/2)
2. training session (23/2)
3. training session (9/3)
4. sumo competition Leuven (16/3)
   thermotechnical institute

5. sumo competition Gent (30/4)

Today's session

- slides and info

  http://www.ieee-sb-leuven.be/soireepratique2014

Rules of competition

- rules
  - http://people.mech.kuleuven.be/~dvanthienen/soiree_pratique/SP_sumo_rules_2015.pdf
- specifications
  - **robot** should fit at the start of the round within a box of following dimensions:
    - width: 24cm
    - length: 24cm
    - height: unlimited
    - weight: less than 1.5kg

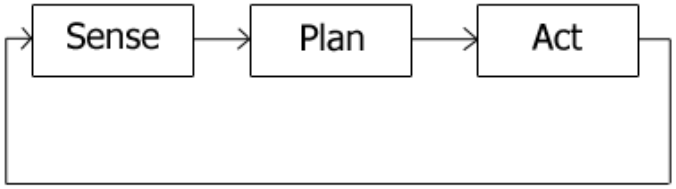
  - **field** (doyho)
    - wooden circular plate of about 1.5m diameter
    - surface is black with a white border
    - border width: about 8cm wide

Competition course

1. robots are put central on dohyo with at least 20cm distance between them
2. robots are powered
3. The **5 second countdown procedure** is activated on both robots at the same time (by releasing button). The countdown is **indicated by means of leds**.
4. everyone steps back from dohyo
5. robots start moving after 5 seconds
6. robots have **2 minutes to push contestant out of the dohyo**
7. round is over after one of the robots is pushed out or if 2 minutes are over (draw)

Programming a robot

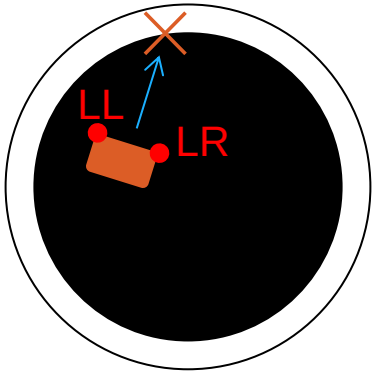
- sense-plan-act framework



  - **sense** the environment with sensors
  - **plan** the next course of action
  - **act** according to the plan
  - **iterate** to be reactive to changes

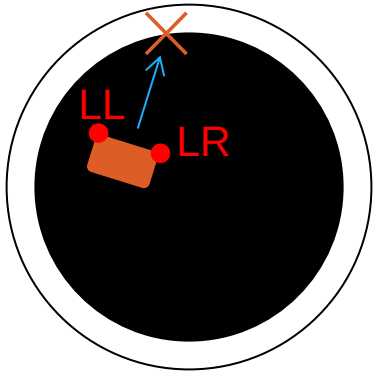
Example problem

- drive forward on the Dohyo while avoiding white border
  - sensors
    - line detection sensor 1: LL
    - line detection sensor 2: LR
  - if LL detects line
    - turn maneuver to the right for 1s
  - if LR detects line
    - turn maneuver to the left for 1s



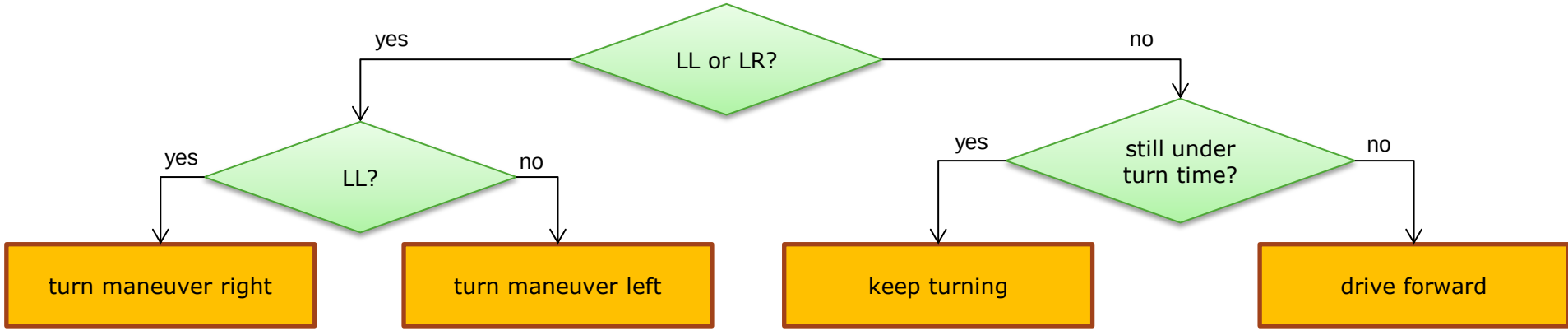
Example problem

- sense:
  - read line detection sensors LL and LR
- plan:
  - determine action
    - e.g. by using a **decision tree** (next slide)
- act:
  - actions can be seen as states in a **finite state machine**
  - transition to new state determined by decision tree outcome
  - possible states here:
    - move forward
    - turn manoever right
    - turn manoever left



Example problem

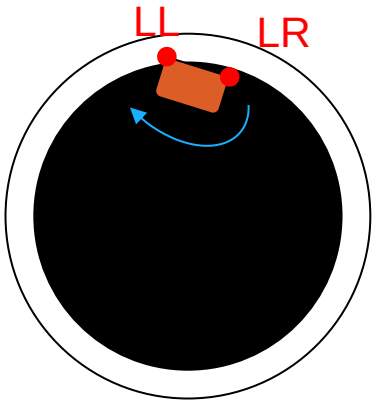
- plan using a **decision tree**:

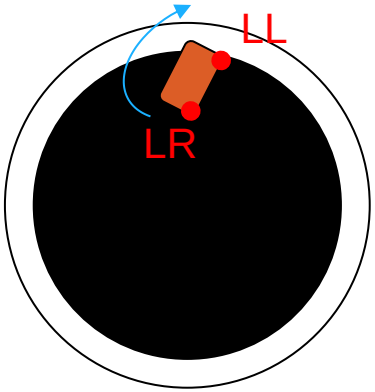


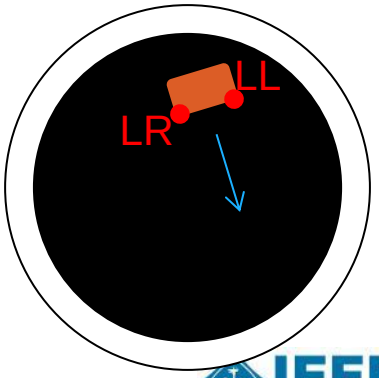
line detected

handle old line

normal







IEEE Student Branch Leuven

Example problem

- implementation in Arduino:

sense

```
void loop() {
/*<...>*/
    /*Line sensors
    *********************************/
    LL=digitalRead(LL_pin);
    LR=digitalRead(LR_pin);
/*<...>*/
```

plan

```
  if (LL || LR) { // line detected
       lineDetected = 1;
       if (LL){
         state = 21;
         turn_manoever_left = 0;
       }
       else {
         state = 22;
         turn_manoever_left = 1;
       }
  }
  else {  // no line detected
      if ((state_duration < turn_maneuver_time_limit) && lineDetected) {
        if (turn_manoever_left) {
          state = 22;
        }
        else {
          state = 21;
        }
      }
      else {
        state = 11;
        lineDetected = 0;
      }
  }
 /*<...>*/
```

act

```
switch(state) {
   /*********************************************/
                    /* NORMAL
   *********************************************/
   // DRIVE FORWARD
   case 11:
       digitalWrite(motLforward, slow_speed);
       digitalWrite(motRforward, slow_speed);
       digitalWrite(motLbackward, 0);
       digitalWrite(motRbackward, 0);
   break;

   /*********************************/
             /* TURN MANEUVERS
   *********************************/
   // Turn right
   case 21:
       digitalWrite(motLforward, slow_speed);
       digitalWrite(motRforward, 0);
       digitalWrite(motLbackward, 0);
       digitalWrite(motRbackward, slow_speed);

     break;
   // Turn left
   case 22:
       digitalWrite(motLforward, 0);
       digitalWrite(motRforward, slow_speed);
       digitalWrite(motLbackward, slow_speed);
       digitalWrite(motRbackward, 0);
     break;
 }
  delay(1); //1ms delay in loop to allow state to be executed
}
```

Programming a robot

- example code for line detection and avoidance can be found here:
  - https://code.google.com/p/sumo-robot/source/browse?repo=default

- framework is extendable with other sensors:
  - long range line sensor
  - ultrasonic sensor
  - ...

- implement different behaviours:
  - border avoidance strategy
  - search opponent strategy
  - attack strategy
  - ...

End

Next session: training (23.02.2015)