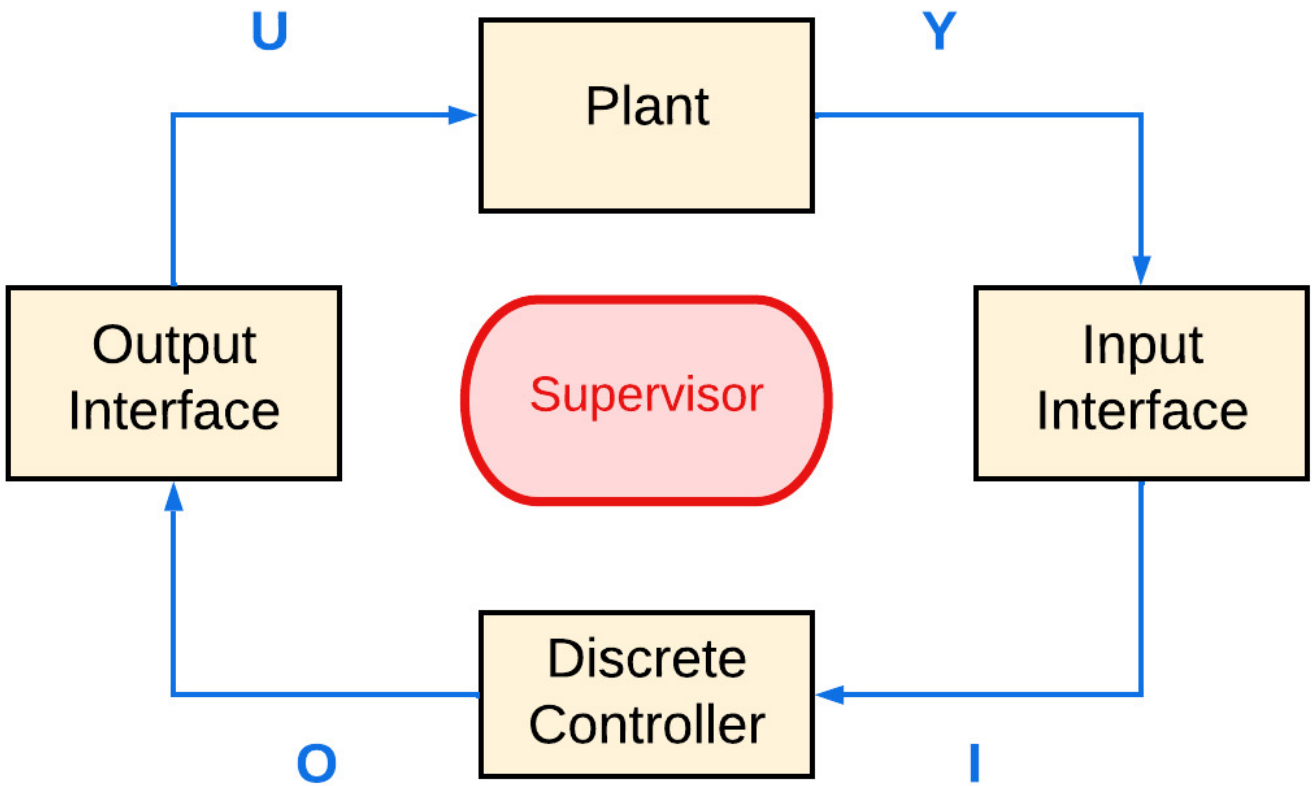
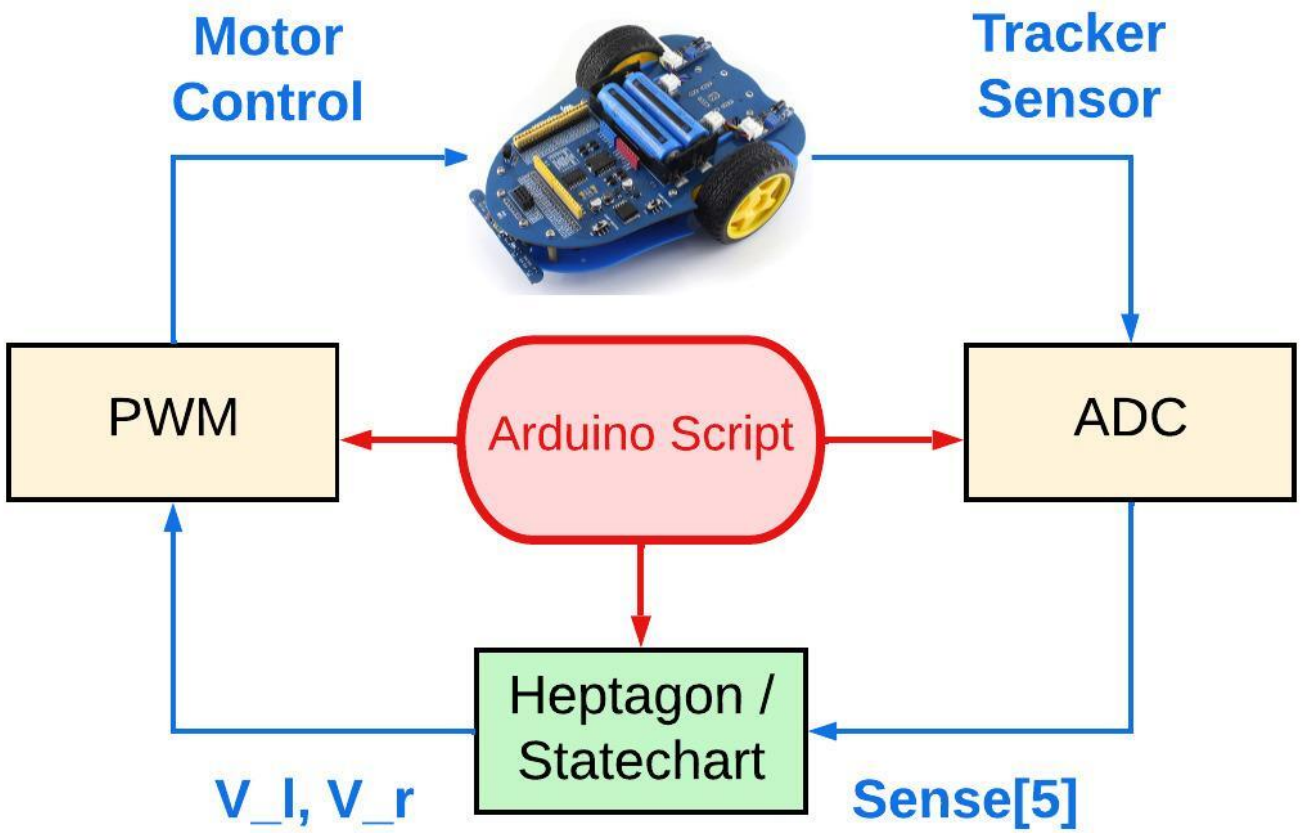
# Line Follower Robot

Embedded Real-Time Systems (ERTS) Lab
Indian Institute of Technology, Bombay

# Model of Cyber-Physical System



# Model of Line Follower



# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances
  - Initialise Devices required - Tracker sensor, Motors, PWM, Serial (for debugging).

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances
  - Initialise Devices required - Tracker sensor, Motors, PWM, Serial (for debugging).
- While Loop

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances
  - Initialise Devices required - Tracker sensor, Motors, PWM, Serial (for debugging).
- While Loop
  - Read data from sensors

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances
  - Initialise Devices required - Tracker sensor, Motors, PWM, Serial (for debugging).
- While Loop
  - Read data from sensors
  - Call discrete Controller With sensors data as input

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances
  - Initialise Devices required - Tracker sensor, Motors, PWM, Serial (for debugging).
- While Loop
  - Read data from sensors
  - Call discrete Controller With sensors data as input
  - Set speed of the robot based on values obtained from discrete controller

# Supervisor

Arduino Script acts as a supervisor which reads data from sensor and give as input to reactive kernel. Also takes velocity values from kernal and drives the motor through PWM.

Steps involved are:

- Setup
  - Initialise Heptagon or Statechart related instances
  - Initialise Devices required - Tracker sensor, Motors, PWM, Serial (for debugging).
- While Loop
  - Read data from sensors
  - Call discrete Controller With sensors data as input
  - Set speed of the robot based on values obtained from discrete controller

# Arduino Script

```
int sensorValues[NUM_SENSORS];
Linefollower__main_mem mem;
Linefollower__main_out _res;

void setup()
{
  Linefollower__main_reset(&mem);
  motion_init();
  forward();
  sensor_init();
  Serial.begin(115200);
}

void loop()
{
  AnalogRead(sensorValues);
  Linefollower__main_step(sensorValues[0], sensorValues[1], sensorValues[2],
                          sensorValues[3], sensorValues[4], &_res, &mem);
  SetSpeed(_/res.v_l, _res.v_r);
}
```

# Line Tracker sensor

# Line Tracker sensor

- Five line sensor - five analog outputs

# Line Tracker sensor

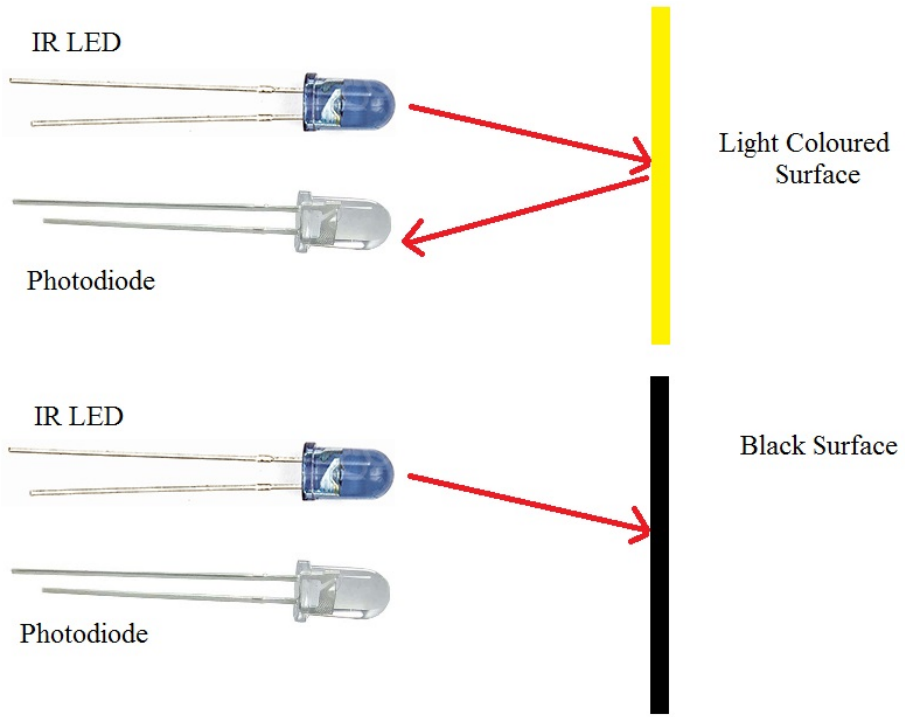
- Five line sensor - five analog outputs
- Higher infrared reflectance (in white) – larger output value

# Line Tracker sensor

- Five line sensor - five analog outputs
- Higher infrared reflectance (in white) – larger output value
- Lower infrared reflectance (in black) – smaller output value

# Line Tracker sensor

- Five line sensor - five analog outputs
- Higher infrared reflectance (in white) – larger output value
- Lower infrared reflectance (in black) – smaller output value



# Calibration

1. Different sensors – different results – same color and distance

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results
3. We may get:

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results
3. We may get:
    1. Actual Min – higher than 0

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results
3. We may get:
   1. Actual Min – higher than 0
   2. Actual Max – lower than 1023

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results
3. We may get:
   1. Actual Min – higher than 0
   2. Actual Max – lower than 1023
4. Normalization process – linear transformation from [Min Max] to the range of [0 1000]

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results
3. We may get:
   1. Actual Min – higher than 0
   2. Actual Max – lower than 1023
4. Normalization process – linear transformation from [Min Max] to the range of [0 1000]
5. Calibrated value = (Value - Min) * 1000 / (Max - Min)

# Calibration

1. Different sensors – different results – same color and distance
2. Environmental Factors – Lighting Conditions – different results
3. We may get:
   1. Actual Min – higher than 0
   2. Actual Max – lower than 1023
4. Normalization process – linear transformation from [Min Max] to the range of [0 1000]
5. Calibrated value = (Value - Min) * 1000 / (Max - Min)
6. For Line switching: Calibrated value = 1000 - Calibrated value

# Thank You!