function check () {

L = robotbit.RgbUltrasonic(DigitalPin.P0)

L2 = robotbit.RgbUltrasonic(DigitalPin.P15)

if (L2 <= 5) {

avoid3()

robotbit.MotorStopAll()

basic.pause(100)

} else if (L > 0 && L < 25) {

yaoTou1()

} else {

normalMove()

}

}

// 1st ver. avoid nearby obstacles, going back and turn left

function avoid1 () {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, -255)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, -255)

basic.pause(300)

robotbit.MotorRun(robotbit.Motors.M2A, 0)

robotbit.MotorRun(robotbit.Motors.M1A, 255)

basic.pause(500)

// recount after each avoidance

count = 0

}

function lightSensingMode () {

// Mode == 0

if (robotbit.RgbUltrasonic(DigitalPin.P0) <= 20) {

robotbit.MotorRun(robotbit.Motors.M1A, -255)

robotbit.MotorRun(robotbit.Motors.M2A, -255)

basic.pause(750)

robotbit.MotorRun(robotbit.Motors.M2A, 0)

robotbit.MotorRun(robotbit.Motors.M1A, 255)

basic.pause(500)

} else if (robotbit.RgbUltrasonic(DigitalPin.P0) > 20) {

difference = pins.analogReadPin(AnalogPin.P3) - pins.analogReadPin(AnalogPin.P1)

if (30 <= Math.abs(difference)) {

if (pins.analogReadPin(AnalogPin.P3) > pins.analogReadPin(AnalogPin.P1)) {

robotbit.MotorRun(robotbit.Motors.M1A, 175 + 0.2 \* Math.abs(difference))

robotbit.MotorRun(robotbit.Motors.M2A, 185 - 0.2 \* Math.abs(difference))

} else {

robotbit.MotorRun(robotbit.Motors.M1A, 100 - 0.2 \* Math.abs(difference))

robotbit.MotorRun(robotbit.Motors.M2A, 170 + 0.2 \* Math.abs(difference))

}

} else {

robotbit.MotorRun(robotbit.Motors.M1A, pins.analogReadPin(AnalogPin.P3))

robotbit.MotorRun(robotbit.Motors.M2A, 10 + pins.analogReadPin(AnalogPin.P1))

}

}

}

function yaoTou1 () {

// 3rd ver. auto detection movements, used when there are nearby obstacles

// first, stop.

robotbit.MotorStopAll()

leastdist = 100

turning1 = 0

robotbit.Servo(robotbit.Servos.S1, 90)

basic.pause(500)

if (robotbit.RgbUltrasonic(DigitalPin.P0) < leastdist) {

leastdist = robotbit.RgbUltrasonic(DigitalPin.P0)

// turn left

turning1 = 2

}

robotbit.Servo(robotbit.Servos.S1, 180)

basic.pause(1000)

if (robotbit.RgbUltrasonic(DigitalPin.P0) < leastdist) {

leastdist = robotbit.RgbUltrasonic(DigitalPin.P0)

// turn right

turning1 = 1

}

robotbit.Servo(robotbit.Servos.S1, 135)

basic.pause(500)

if (robotbit.RgbUltrasonic(DigitalPin.P0) < leastdist) {

leastdist = robotbit.RgbUltrasonic(DigitalPin.P0)

// go straight backwards and turn

turning1 = 3

}

// execute the turning movements

if (turning1 == 1) {

robotbit.MotorRunDual(

robotbit.Motors.M1A,

160,

robotbit.Motors.M2A,

60

)

basic.pause(500)

robotbit.MotorStopAll()

count = 0

} else if (turning1 == 2) {

robotbit.MotorRunDual(

robotbit.Motors.M1A,

60,

robotbit.Motors.M2A,

160

)

basic.pause(500)

robotbit.MotorStopAll()

count = 0

} else if (turning1 == 3) {

avoid3()

}

}

// for mode == 0, the lighting controlled mode, detect lighting difference between two light sensors

// press button A to switch from lighting mode to auto mode

input.onButtonPressed(Button.A, function () {

if (mode == 0) {

mode = 1

} else {

mode = 0

}

})

function avoid3 () {

robotbit.MotorStopAll()

basic.pause(500)

robotbit.MotorRun(robotbit.Motors.M1A, -255)

robotbit.MotorRun(robotbit.Motors.M2A, 255)

basic.pause(800)

robotbit.MotorRunDual(

robotbit.Motors.M1A,

100,

robotbit.Motors.M2A,

100

)

basic.pause(200)

robotbit.MotorStopAll()

count = 0

}

function yaoTou2 () {

maxL = 0

turning2 = 0

// first look to the right

robotbit.Servo(robotbit.Servos.S1, 90)

basic.pause(500)

if (robotbit.RgbUltrasonic(DigitalPin.P0) > maxL) {

maxL = robotbit.RgbUltrasonic(DigitalPin.P0)

// turn right

turning2 = 2

}

// then look to the left

robotbit.Servo(robotbit.Servos.S1, 150)

basic.pause(500)

if (robotbit.RgbUltrasonic(DigitalPin.P0) > maxL) {

maxL = robotbit.RgbUltrasonic(DigitalPin.P0)

// turn left

turning2 = 1

}

// finally look ahead

robotbit.Servo(robotbit.Servos.S1, 120)

basic.pause(500)

if (robotbit.RgbUltrasonic(DigitalPin.P0) > maxL) {

maxL = robotbit.RgbUltrasonic(DigitalPin.P0)

// go ahead

turning2 = 3

}

// execute the movements

if (turning2 == 1) {

robotbit.MotorRunDual(

robotbit.Motors.M1A,

150,

robotbit.Motors.M2A,

-140

)

basic.pause(300)

robotbit.MotorStopAll()

} else if (turning2 == 2) {

robotbit.MotorRunDual(

robotbit.Motors.M1A,

-150,

robotbit.Motors.M2A,

160

)

basic.pause(300)

robotbit.MotorStopAll()

} else if (turning2 == 3) {

robotbit.MotorRunDual(

robotbit.Motors.M1A,

150,

robotbit.Motors.M2A,

150

)

basic.pause(300)

robotbit.MotorStopAll()

}

}

// move normally forward

function normalMove () {

robotbit.MotorRunDual(

robotbit.Motors.M1A,

150,

robotbit.Motors.M2A,

160

)

// count after each normal movement

count += 1

}

// 2nd ver. turn right at where it is, without going backwards

function avoid2 () {

robotbit.MotorStopAll()

robotbit.MotorRun(robotbit.Motors.M2A, -255)

robotbit.MotorRun(robotbit.Motors.M1A, 255)

count = 0

}

// let danger = 0

function followMode () {

L = robotbit.RgbUltrasonic(DigitalPin.P0)

if (L > 30) {

if (ll > rl) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 225)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 120)

basic.pause(200)

} else if (ll < rl) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 110)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 225)

basic.pause(200)

} else {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 200)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 220)

basic.pause(200)

}

} else if (L <= 30 && L > 20) {

if (ll > rl) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 120)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 60)

basic.pause(200)

} else if (ll < rl) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 60)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 160)

basic.pause(200)

} else if (L <= 20 && L > 10) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 90)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 110)

basic.pause(200)

} else if (L <= 10) {

robotbit.MotorStopAll()

basic.pause(400)

}

} else {

if (ll > rl) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 100)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 60)

basic.pause(200)

} else if (ll < rl) {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 60)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 100)

basic.pause(200)

} else {

// right wheel

robotbit.MotorRun(robotbit.Motors.M1A, 60)

// left wheel

robotbit.MotorRun(robotbit.Motors.M2A, 80)

basic.pause(200)

}

}

}

/\*\*

\* ---------------------

\*/

let halt = 0

let rl = 0

let ll = 0

let turning2 = 0

let maxL = 0

let mode = 0

let turning1 = 0

let leastdist = 0

let difference = 0

let count = 0

let L2 = 0

let L = 0

pins.setPull(DigitalPin.P1, PinPullMode.PullUp)

pins.setPull(DigitalPin.P2, PinPullMode.PullUp)

// 右边sensor

// 感光值

let thresh = 150

// 右边sensor

// 感光值

let baselineleft = pins.analogReadPin(AnalogPin.P2)

// 右边sensor

// 感光值

let baselineright = pins.analogReadPin(AnalogPin.P1)

basic.forever(function () {

// amount of light shone on the sensor

ll = baselineleft - pins.analogReadPin(AnalogPin.P2)

rl = baselineright - pins.analogReadPin(AnalogPin.P1)

if (ll > thresh || rl > thresh) {

mode = 2

}

if (mode == 2) {

followMode()

if (robotbit.RgbUltrasonic(DigitalPin.P0) <= 10 || (robotbit.RgbUltrasonic(DigitalPin.P15))) {

mode = 0

}

} else if (mode == 0) {

if (halt == 1) {

robotbit.MotorRun(robotbit.Motors.M2A, 0)

robotbit.MotorRun(robotbit.Motors.M1A, 0)

yaoTou2()

// recount after each halt

count = 0

halt = 0

}

// no matter halt == what, always execute check(), checking its way all the time

check()

// check for nearby obstacles to avoid + make movement choices

if (count >= 100) {

// after a while "halt" turns true and begins to detect its way in the next loop

halt = 1

}

if (ll > thresh || rl > thresh) {

mode = 2

}

} else if (mode == 1) {

lightSensingMode()

}

})

control.inBackground(function () {

while (true) {

basic.showNumber(pins.analogReadPin(AnalogPin.P3))

basic.showNumber(pins.analogReadPin(AnalogPin.P1))

}

})