AsebaCode  
  
<!DOCTYPE aesl-source>

<network>

<!--list of global events-->

<!--list of constants-->

<constant value="200" name="THRESHOLD"/>

<constant value="20" name="SEARCH\_DISTANCE"/>

<constant value="500" name="SEARCH\_ANGLE"/>

<constant value="1" name="WANDER"/>

<constant value="2" name="TURNING"/>

<constant value="0" name="STOP"/>

<constant value="4300" name="PER\_SPACE"/>

<constant value="900" name="TURN\_ANGLE"/>

<constant value="1800" name="TURN\_BACK"/>

<constant value="3500" name="NEAR\_VIEW"/>

<constant value="250" name="SPEED"/>

<constant value="500" name="MAX\_VIEW"/>

<!--show keywords state-->

<keywords flag="true"/>

<!--node thymio-II-->

<node nodeId="1" name="thymio-II">#variables

var state = WANDER

var dir = 1 #Direction 1 as right

var dis = 0 #Distance before turn

timer.period[0]=10000 #Timer for 10 sec

#Move forward

motor.left.target = SPEED

motor.right.target = SPEED

call leds.top(0, 32, 0) #Green LED

onevent prox

if state == WANDER then

if prox.ground.delta[0] &lt; THRESHOLD or

prox.ground.delta[1] &lt; THRESHOLD then

if prox.ground.delta[0] &lt; THRESHOLD then #left Sensor

#right Turn

motor.left.target = SPEED

motor.right.target = -SPEED

timer.period[0] = TURN\_ANGLE

elseif prox.ground.delta[1] &lt; THRESHOLD then #right Sensor

#left Turn

motor.left.target = -SPEED

motor.right.target = SPEED

timer.period[0] = TURN\_ANGLE

end

state = TURNING

dis = 0 #Resetting Distance Counter

call leds.top(0, 0, 32) #Blue LED

elseif dis >= SEARCH\_DISTANCE then

if dir == 1 then

#Turn Right

motor.left.target = SPEED

motor.right.target = 0

timer.period[0] = TURN\_ANGLE

else

#Turn Left

motor.left.target = 0

motor.right.target = SPEED

timer.period[0] = TURN\_ANGLE

end

state = TURNING

call leds.top(0, 0, 32)

dir = -dir #Changing Direction to left

dis = 0 #Resetting Distance Counter

else

#Moving Farward

motor.left.target = SPEED

motor.right.target = SPEED

dis++

end

if prox.horizontal[0] > PER\_SPACE or

prox.horizontal[1] > PER\_SPACE or

prox.horizontal[2] > PER\_SPACE or

prox.horizontal[3] > PER\_SPACE or

prox.horizontal[4] > PER\_SPACE then

# Turning Back

motor.left.target = SPEED

motor.right.target = -SPEED

timer.period[0] = TURN\_BACK

state = TURNING

dis = 0 #Resetting Distance Counter

call leds.top(32, 0, 0) #Red LED

elseif prox.horizontal[0] > NEAR\_VIEW or

prox.horizontal[1] > NEAR\_VIEW then

# Turning Right

motor.left.target = SPEED

motor.right.target = 0

timer.period[0] = SEARCH\_ANGLE

state = TURNING

call leds.top(0, 0, 32)

dis = 0 #Resetting Distance Counter

elseif prox.horizontal[3] > NEAR\_VIEW or

prox.horizontal[4] > NEAR\_VIEW then

# Turning Left

motor.left.target = 0

motor.right.target = SPEED

timer.period[0] = SEARCH\_ANGLE

state = TURNING

call leds.top(0, 0, 32)

dis = 0

elseif prox.horizontal[1] > MAX\_VIEW then

# Turning Left

motor.left.target = 0

motor.right.target = SPEED

timer.period[0] = SEARCH\_ANGLE

state = TURNING

call leds.top(0, 32, 32)

dis = 0

elseif prox.horizontal[3] > MAX\_VIEW then

# Turning Right

motor.left.target = SPEED

motor.right.target = 0

timer.period[0] = SEARCH\_ANGLE

state = TURNING

call leds.top(0, 32, 32)

dis = 0

end

end

return

onevent buttons

if button.center == 1 and state != STOP then #Stopping on center

state = STOP

motor.left.target = 0

motor.right.target = 0

end

if button.forward == 1 and state == STOP then #Starting on Forward

state = WANDER

motor.left.target = SPEED

motor.right.target = SPEED

end

return

onevent timer0

if state == TURNING then #Changeing State

motor.left.target = SPEED

motor.right.target = SPEED

timer.period[0]=0

state = WANDER

call leds.top(0, 32, 0)

end

return

</node>

</network>