IMPLEMENTATION and EVALUATION of an e-puck2 MINIATURE MOBILE ROBOT. (ACS6501)

Pranav Patil, Yifan Zhu and Lingyue Pan

***Abstract*— The first of the two tasks assigned was to explore the surroundings using the e-puck2 robot whilst avoiding obstacles and hitting the wall boundaries. The second task aimed at chasing an object in an open environment without bumping into it.**
For tackling both tasks, a program was fed to the robot which instructed it to make use of the 8 infrared sensors on its periphery to change its state while maintaining a certain speed. Additionally, LEDs were switched on which indicated the current state of the robot.

I. STRATEGIES

A. Exploration of the robot within a bounded environment

This task aimed at establishing and examining the motion of the robot in an environment surrounded by hurdles without stumbling upon them and at the same time avoiding collision with the environment boundaries. To execute this task, the use of 8 infrared sensors arranged on the robot's circumference was made. Firstly, a threshold distance is generated using the 8 infra-red sensors. The robot was programmed to halt if it detects anything within the proximity threshold range. Moreover, the robot should stop immediately whenever the object is placed within the threshold range to avoid a collision.

After this, the front 6 infrared sensors were used to scan the range of vision for the robot. Making use of these sensors, the robot was programmed to move forward unless the obstacle is out of the threshold range. The change in direction of the robot (left and right) was controlled by the sensor it sensed. To elaborate, the robot's turn was dependent upon which sensor sensed the larger distance from the threshold.

While moving forward, the robot turned when the front sensors detect any obstacle within the proximity range, and for the robot to move forward after turning it is essential that the front sensors do not detect any object within the proximity threshold range. During all these operations, a certain speed for the bot was set to keep it moving smoothly.

B. Trailing an object

The aim of this task was to follow a target object whenever it is in the sensing range or the range defined by the robot. In addition to this, if the robot gets too close to the target it is following, it must move backward to a secure position. To achieve this task, a threshold value for the proximity sensors was set to keep the robot from bumping into the obstacle.

Patil, Zhu and Pan are with the Department of Automatic Control and Systems Engineering, The University of Sheffield, UK {email1, email2}@papercept.net

To start with, the robot was programmed to scan for the object using the 8 infrared sensors. At any point in time, if any of the sensors sensed the object the robot changed its state. After sensing the object, the bot moved forward and chased the object until it stopped which in turn altered the state of the robot. When the object changed its direction, the robot was made to follow it by using proximity sensors. While the object was moving towards the robot and it crossed the threshold value, the robot was programmed to start moving backward in order to stop it from colliding with the object.

II. IMPLEMENTATION

A. Task 1: Exploration of the bot within a bounded environment

In order to maintain a proper sequence of operations of this task, different functions were created and initialized as seen in lines 1 to 42 of the Appendix.

The first step carried out was to set the threshold limit for all the infrared sensors in use to 300. Furthermore, the speed of the robot while moving forward and while taking a turn was set to 800 steps per second.

The robot was programmed using the conditions mentioned below for executing the following tasks.

a) Turning left: A program was written for the robot to turn left switching on LED 7 when sensor 0, 1 or 2 detected the obstacle when it was within the threshold range.

b) Turning right: The robot turned right activating LED 3 when either one of the sensors 7, 6 or 5 identified the obstacle in the threshold range.

c) Moving forward: When all of the 8 infrared senors did not detect the object, the robot was programmed to go straight switching on LED 1.

d) Changing direction when approaching a dead end: When the robot was nearing a dead end, and sensors 0, 1, 2, 5, 6 and 7 got activated the same time, it turned right turning on LED 5.

In this way, the task of exploring a bounded environment whilst avoiding the obstacle was successfully implemented.

B. Task 2: Trailing an object

The first thing done was to initialize the motors and calibrate the proximity sensors. The next step done was to set different threshold values. The target proximity threshold was set at 200 while the inner threshold (target too close) and outer threshold (target out of vicinity) of the object position were set at 100 and 50 respectively.

the robots speed was set at 600 steps per seconds while moving forward and backward while, a speed of 800 steps per seconds was set when the robot changed its state to turn to either left or right.

A set of conditions were implemented for achieving the tasks mentioned below.

a) Moving straight: Whenever sensor 0 value was within 50, the robot was programmed to progress forward turning LED 1 on. On the other hand when the value of sensor 0 was between 10 and 50, the robot stopped immediately.

b) Moving back: A program was fed to the robot to move it in the reverse direction whilst turning LED 5 on when the sensor 0 value exceeded 50, whereas it was made to stop when the sensor 0 value lied in the range of 10 and 50.

c) Turning right: The robot turned right when proximity sensors 1 or 2 or 3 exceeded the threshold value of 60 turning on LED 3.

d) Turning left: When the reading of sensors 4 or 5 or 6 was greater than the threshold value of 60, the robot was instructed to turn left activating LED 5.

III. RESULTS AND DISCUSSIONS

APPENDIX

```
************   Task 1: Exploration of the bot within a bounded environment   **************

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#include "ch.h"
#include "hal.h"
#include "memory_protection.h"
#include <main.h>


#include "leds.h"
#include "spi_comm.h"
#include "sensors/proximity.h"
#include "motors.h"
#include "sensors/VL53L0X/VL53L0X.h"


messagebus_t bus;
MUTEX_DECL(bus_lock);
CONDVAR_DECL(bus_condvar);

//declare function
//int max(int num1, int num2, int num3, int num4, int num5, int num6, int num7, int num8);

int main(void)
{

    halInit();
    chSysInit();
    mpu_init();
   /** Inits the Inter Process Communication bus. */
  messagebus_init(&bus, &bus_lock, &bus_condvar);
  clear_leds();
  set_body_led(0);
  set_front_led(0);
    motors_init(); //motor initialization
    proximity_start(); //Start the proximity measurement module
    calibrate_ir(); //Calibrate the proximity sensors
    VL53L0X_start(); //


//    left_motor_set_speed(0); //left motor speed
//    right_motor_set_speed(0);  //left motor speed



  //  int get_prox(unsigned int sensor_number); // sensor_number 0-7

    int threshold = 300;   //set threshold for proximity(distance)
    int count = 0;
    /* Infinite loop. */
    while (1)
    {
      //waits 1 second
        //chThdSleepMilliseconds(1000);


       int proximity_reading[8] = { 0,1,2,3,4,5,6,7 };
              for (int i = 0; i < 8; i++)
              {
                  proximity_reading[i] = get_calibrated_prox(i);
              }
                //without distance sensor

              if (proximity_reading[0] > threshold || proximity_reading[1] > threshold || proximity_reading[2] > 500 )
              {
                  // chThdSleepMilliseconds(1000);
                  left_motor_set_speed(-800);
                  right_motor_set_speed(800); //if sensor 0 or1 or 2 detected obstacle.robot turn left
                  set_led(LED7, 1);
                  set_led(LED1, 0);
                  set_led(LED3, 0);
                  set_led(LED5, 0);
              }

                // turnning while dis_sensor and pro_sensor approaching at the same time

              else if (proximity_reading[7] > threshold || proximity_reading[6] > threshold || proximity_reading[5] >
     500)
              {
```

```
                    left_motor_set_speed(800);
                    right_motor_set_speed(-800);//if sensor 7 or 6 or5 detected obstacle.robot turn right

                    set_led(LED3, 1);
                    set_led(LED1, 0);
                    set_led(LED5, 0);
                    set_led(LED7, 0);
                }

                else
                {

                    left_motor_set_speed(800);
                    right_motor_set_speed(800);//sensor 0.1.2.3.4.7.6.5 don't detected obstacle rotot so straight
              set_led(LED1, 1);
              set_led(LED3, 0);
              set_led(LED5, 0);
              set_led(LED7, 0);
                 }

          if ( proximity_reading[0] > threshold && proximity_reading[1] > threshold && proximity_reading[2] > threshold
     && proximity_reading[7] > threshold && proximity_reading[6] > threshold && proximity_reading[5] > threshold)
                {
                    left_motor_set_speed(800);
                    right_motor_set_speed(-800);//If entering a dead end,  proximity sensor 012567 detect an obstacle at
     the same time, the robot turn right
                    chThdSleepMilliseconds(1000);//wait 1s;
                    set_led(LED5, 1);
                    set_led(LED1, 0);
                    set_led(LED3, 0);
                    set_led(LED7, 0);
                }


              }



#define STACK_CHK_GUARD 0xe2dee396
uintptr_t __stack_chk_guard = STACK_CHK_GUARD;

void __stack_chk_fail(void)
{
    chSysHalt("Stack smashing detected");
}


****************                 END OF TASK 1                    ***************



 ************************   Task 2: Trailing an object   ******************************



#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#include "ch.h"
#include "hal.h"
#include "memory_protection.h"
#include <main.h>


#include "leds.h"
#include "spi_comm.h"
#include "sensors/proximity.h"
#include "motors.h"
#include "sensors/VL53L0X/VL53L0X.h"


messagebus_t bus;
MUTEX_DECL(bus_lock);
CONDVAR_DECL(bus_condvar);

//declare function
//int max(int num1, int num2, int num3, int num4, int num5, int num6, int num7, int num8);

int main(void)
{

    halInit();
    chSysInit();
```

```
    mpu_init();
   /* Inits the Inter Process Communication bus. */
  messagebus_init(&bus, &bus_lock, &bus_condvar);
  clear_leds();
  set_body_led(0);
  set_front_led(0);
    motors_init(); //motors initial
    proximity_start(); //Start the proximity measurement module
    calibrate_ir(); //Calibrate the proximity sensors
    VL53L0X_start(); //

  //  int get_prox(unsigned int sensor_number); // sensor_number 0-7

    int threshold = 200;   //set theshold for proximity(distance)
    int threshold_1 = 100;

    int needs_run = 1;

    /* Infinite loop. */
    while (1)
    {
      //waits 1 second
        //chThdSleepMilliseconds(1000);


      int proximity_reading[8] = {0,1,2,3,4,5,6,7};
      for(int i = 0; i < 8; i++)
      {
        proximity_reading[i] = get_calibrated_prox(i);
      }

      int threshold_2 = 60;

    if(proximity_reading[0] < 50) //55  54  50
    {
      left_motor_set_speed(600);  // go straight
      right_motor_set_speed(600);
      set_led(LED1, 1);
      set_led(LED3, 0);
      set_led(LED5, 0);
      set_led(LED7, 0);
      if(proximity_reading[0] < 50  && proximity_reading[0] > 10)
      {
        left_motor_set_speed(0);  // stop> 50
        right_motor_set_speed(0);
      }
    }
    if(proximity_reading[0] > 50)
    {
      left_motor_set_speed(-600); // move back
      right_motor_set_speed(-600);
      set_led(LED1, 0);
      set_led(LED3, 0);
      set_led(LED5, 1);
      set_led(LED7, 0);

      if(proximity_reading[0] < 50  && proximity_reading[0] > 10)
      {
        left_motor_set_speed(0);  // stop
        right_motor_set_speed(0);
      }
    }
    if(proximity_reading[1] > threshold_2 || proximity_reading[2] > threshold_2 || proximity_reading[3] > threshold_2)
    {
      left_motor_set_speed(800);  // turn right
      right_motor_set_speed(-800);
      set_led(LED1, 0);
      set_led(LED3, 1);
      set_led(LED5, 0);
      set_led(LED7, 0);
    }

    if(proximity_reading[4] > threshold_2 || proximity_reading[5] > threshold_2 || proximity_reading[6] > threshold_2)
    {
      left_motor_set_speed(-800); // turn left
      right_motor_set_speed(800);
      set_led(LED1, 0);
      set_led(LED3, 0);
      set_led(LED5, 1);
      set_led(LED7, 0);
    }
    }
}

```

```
251
252 #define STACK_CHK_GUARD 0xe2dee396
253 uintptr_t __stack_chk_guard = STACK_CHK_GUARD;
254
255 void __stack_chk_fail(void)
256 {
257     chSysHalt("Stack smashing detected");
258 }
```