Comp1421 Distance Sensors

C.B.Price January 2021

Purpose	To understand some ePuck sensors: (i) Time-of-flight sensor, (ii) Infra-red distance Sensors.
Files Required	Webots project folders on website. **CBP_1421_Sensorium_1.wbt** world
ILO Contribution	3
Send to Me	If you are working online, send me a movie-clip of your solution
Homework	

Activities

1 Reading the code

(a) Open the world CBP_1421_Sensorium_1.wbt and make sure the controller CBP_1421_Sensorium_1.c is selected and appears in the Text Editor.

(b) Spend some time reading the code, the **main(…)** function. In particular look for the following lines which initialize the simulator and the motors and sensors

```
wb_robot_init();
cbp_motors_init();
cbp_sensors_init(TIME_STEP);
```

Make sure you understand the purpose of these lines.

(c) Now look for the while loop

```
while (wb_robot_step(TIME_STEP) != -1) {

}
```

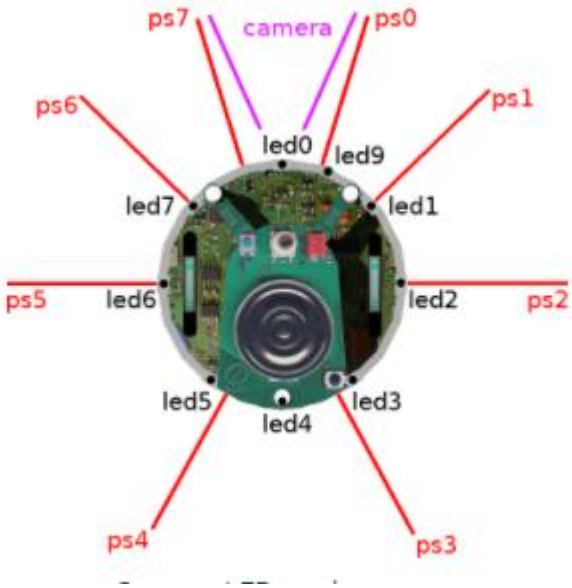
which runs for ever. At each loop, it makes a call to **wb_robot_step(TIME_STEP)**

(d) Look inside the while loop and find the line of code **cbp_get_distance_vals(dist_vals);** This returns an array **dist_vals** of the distance sensor readings. Look for the declaration of the array before the **main(…)** function.

(e) Now see how the distance array values are printed out to the console:

```
printf("Distance ");
for(int i=0;i<8;i++) {
  printf("%+5.3f  ",dist_vals[i]);
}
printf("\n");
```

Here is the arrangement of the distance sensors



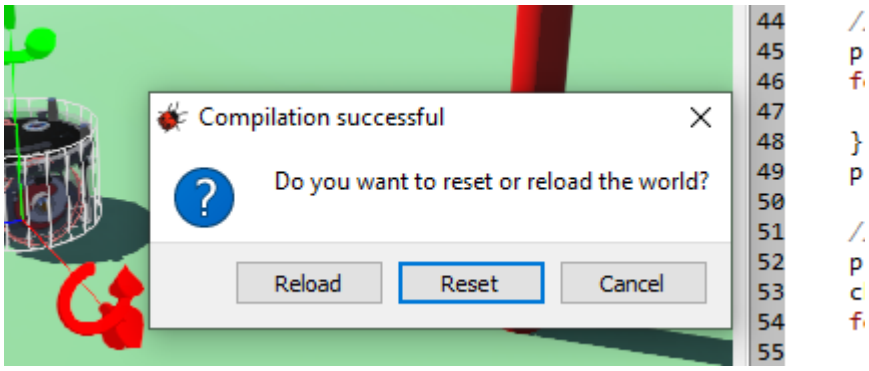
(f) Now work down the code and find out how the light sensor values are read and printed out to the console.

(g) Finally look for the code that prints out the time-of-flight long-range distance sensor value.

2 **Investigating the infra-red distance sensors**

(a) In the code (text editor) add an extra blank line. This will 'touch' the code, so the system knows it needs re-compiling. Now recompile by pressing the following button

If all is well, you will get the following dialogue box



Hit Reload to reload the world. Now we are ready to experiment with the sensors

(b) Run the simulation by pressing  and you will get a list of sensor readings in the console. These will be updated every 500 milliseconds. You should see the sensor 'whiskers' coming out of the robot. If not, select **View > Optional Rendering > Show DistanceSensor Rays**.

(c) In the Scene Tree, left clock on **DEF OBSTACLE1 Solid** to select the tall red cylinder. You

	can move this using its colored gizmos: (d) Move the cylinder in front of the robot and look at the values of TOF. Move the cylinder closer and further from the robot and note how this changes. (e) Now move the cylinder very close to the robot until it hits distance sensor 0 (‘ps0’ in the diagram above). You should see how the associated distance sensor value changes. Try and make sense of these readings. The characteristics of the distance sensors are provided below	
	Time of Flight Sensor This appears to have no limits. So when there is no obstacle in front of it, it will detect the RectangeArena walls **Infra-red Distance sensors** These can measure from 0.0m to 0.07m (7cms) so only work very close to the robot. If a call to the associated function is made, and the object is out of range, the sensor should return **-1**.	