PLACEHOLDER

Intelligent Robotics

Bernardo Hernandez Hernandez, 294561
Professor: José Ignacio Núñez Varela
Autonomous University of San Luis Potosí
School of Engineering, Computer Science Department

Introduction

Problem description

The task is to create a 3D map of a simple environment using a mobile robot and computer vision techniques. To do this, the robot will need to navigate through the environment, avoid obstacles, decide which area of the environment is yet to be explored and then move to it.

Goal

The project aims to develop a proof of concept of what could be done with a mobile robot and computer vision techniques, as well as create a workflow to ease the creation of future similar projects of the students of the Computer Science Department of the Autonomous University of San Luis Potosí.

Project characterization

Task description

The robot will need to navigate through the environment, avoid obstacles, decide which area of the environment is yet to be explored and then move to it. Using a Kinect, the robot will be able to detect the presence of obstacles and walls, saving the information to create a map of the environment. The robot will stop the exploration when it as no way to move towards the unexplored area, this could be because the robot has mapped the whole environment or because there is no way to move path wide enough to reach the unexplored area.

Environment description

The environment that the robot must map consists of a 3x3 meter space with box-like obstacles.

- Wall: A wall is a rectangular area of the environment that is not traversable, it defines the boundaries of 3x3 meter space. To create a wall we will use the plywood sheet available in the lab or the cardboard boxes we can gather. The approximate dimensions of the walls should be 3mx1mx1cm.
- Obstacle: An obstacle is a rectangular area of the environment that is not traversable, To create an obstacle we will use the cardboard boxes we can gather. All three dimensions of the obstacles should be between 20 cm and 40 cm.

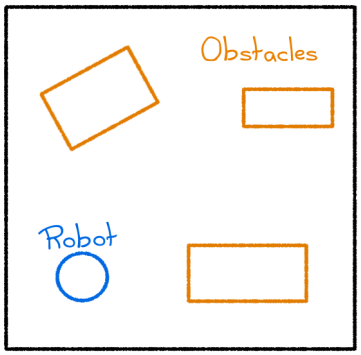


Figure 1: Environment

Robot description

Sensors

The robot will have a camera that will be used to detect the obstacles in the environment and the creation of the map of the environment.

Alternatively, the robot could use some other kind of sensors to detect the obstacles in the environment, like infrared sensors and bumpers integrated in the robot.

Actuators

The robot won't need to grasp any objects, the only interaction with the environment will navigation trough the environment.

Architecture

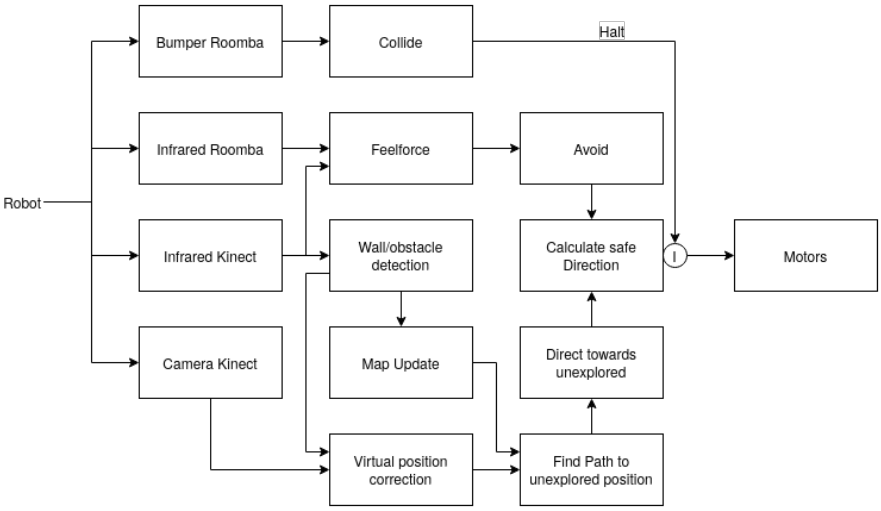


Figure 2: Robot Architecture

Modules description

Collide

Receives a collision message from the Roomba bumper sensor, and sends a message to the Robot to stop moving.

Feel force

Receives a force message from the Roomba's infrared sensors and the Kinect's depth sensor, using this info the Roomba will detect what direction will get it away from the walls.

Avoid

Determine if a wall is too close. Uses the map and environment information.

Wall/obstacle detection

Using the cloud point generate planes to represent walls and obstacles.

Map update

Add new detected wall and obstacles to the 3D map.

Virtual position correction

Using odometry techniques and landmarks gathered from the camera, correct the virtual position to better represent the real position of the robot.

Find path to unexplored

Find the closest unexplored reachable position.

Direct towards unexplored

Calculate the necessary moves to the reach the next position to explore.

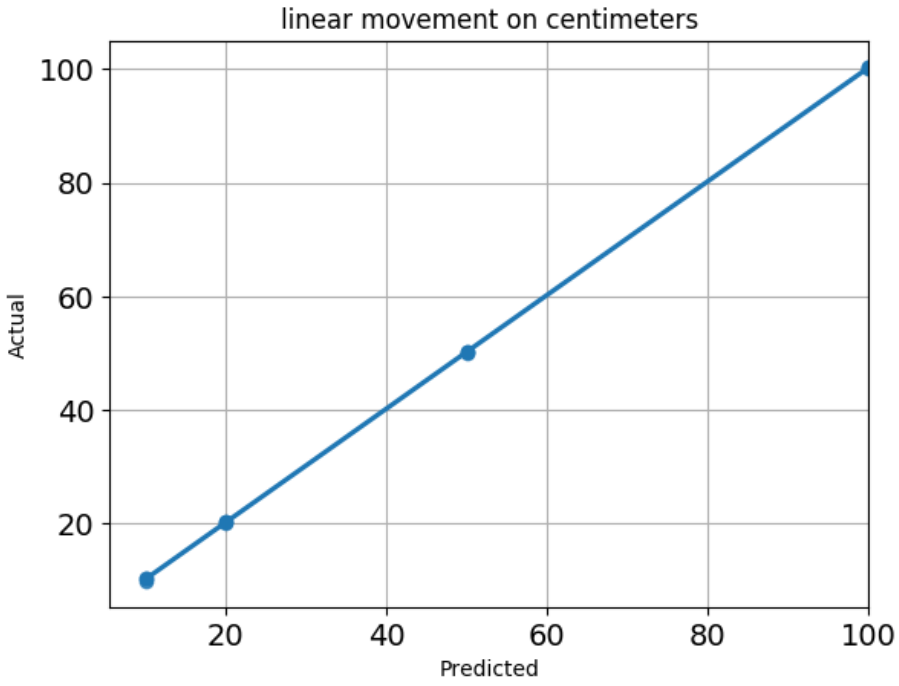
Calculate safe direction

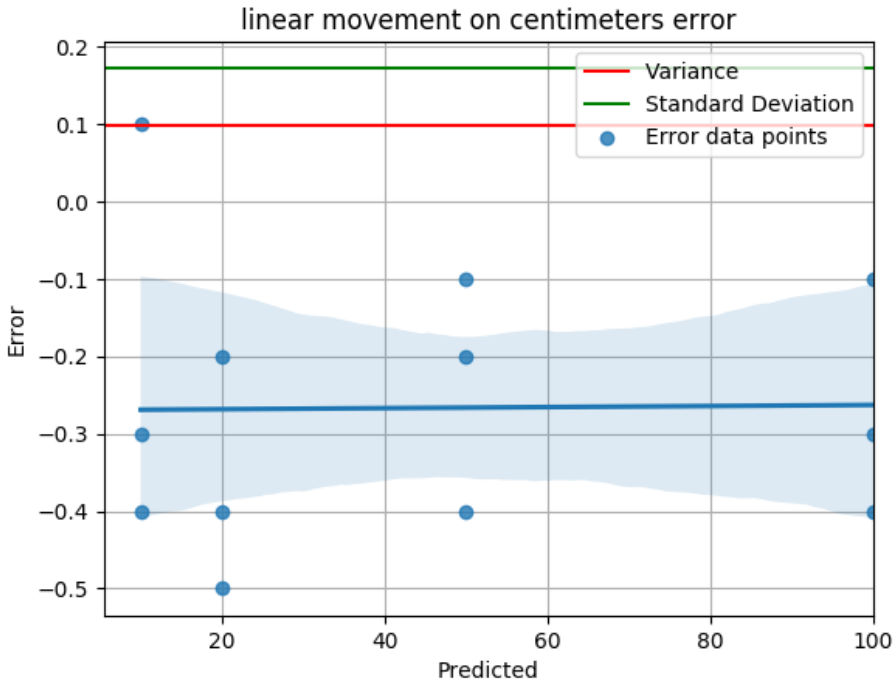
Using the info from the "Direct towards unexplored" and "Avoid" modules determine the value the motors need to have.

Linear Movement

Using the move interface we developed

Prediction (cm)	Real (cm)
10	09.9
10	10.4
10	10.3
20	20.2
20	20.4
20	20.5
50	50.4
50	50.1
50	50.2
100	100.1
100	100.3
100	100.4





Conclusion

There is only a small error, and it is somewhat constant among all distances, this is error is probably caused from a combination of the start and stop of the movement form the motors and human error during the measures.

Move module

```
#!/usr/bin/env python
import rospy
from geometry_msgs.msg import Twist

def set_speed(linear, angular, duration):

    """
    Set the speed of the robot.
    linear: float
        Linear speed in m/s.
    angular: float
        Angular speed in rad/s.
    duration: float
        Duration of the movement in seconds.
    """

    assert(-0.5 <= linear <= 0.5)
    assert(-4.25 <= angular <= 4.25)


    pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)

    rospy.init_node('move', anonymous=True)

    move_cmd = Twist()
    move_cmd.linear.x = linear
    move_cmd.angular.z = angular

    now = rospy.Time.now()
    rate = rospy.Rate(10)

    while rospy.Time.now() < now + rospy.Duration.from_sec(duration):
        pub.publish(move_cmd)
        rate.sleep()

def linear_move(meters):
    """
    Move the robot forward for a certain distance.
    meters: float
        Distance in meters.
    """
    linear_speed = 0.2
    if meters < 0:
```

```
        linear_speed = -linear_speed
    ANGULAR_SPEED = 0.0
    duration = abs(meters / linear_speed)

    set_speed(linear_speed, ANGULAR_SPEED, duration)


def main():
    try:
        linear_move(1.0)
    except rospy.ROSInterruptException:
        pass

if __name__ == '__main__':
    main()
```