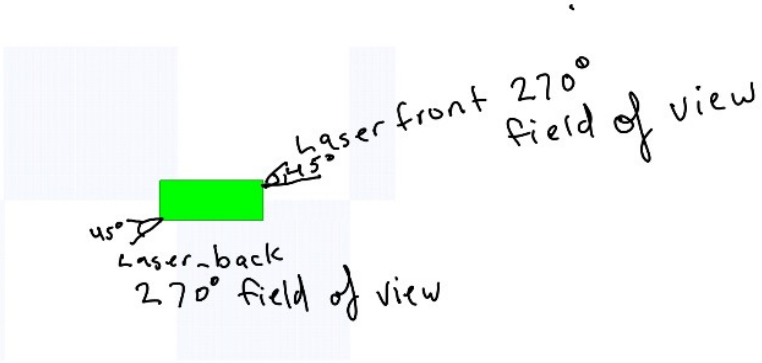
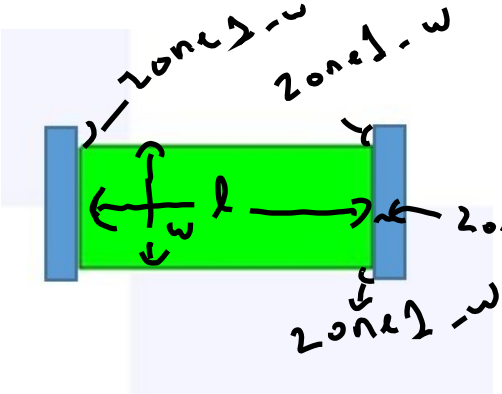
In the stage simulation there is a robot with two lasers mounted, see image below. The lasers have a 270 degree viewing angle and are mounted at a 45 degree angle from the vertical center line. The goal of this project is to write a c++ ros node that will use the laser data to limit the speed of the robot.



this should be input as a Ros parameter

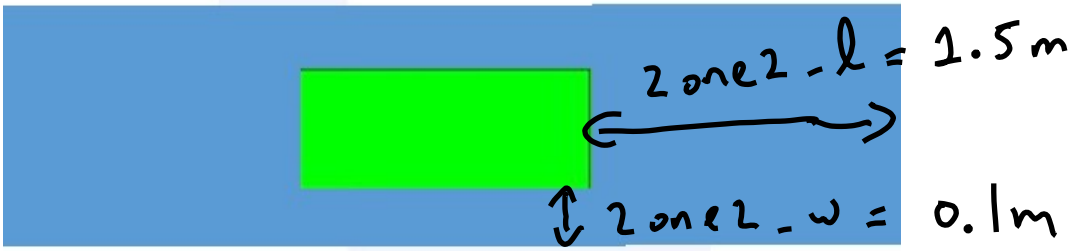
Let max speed be 2.0 m/s when no objects are present around the robot.

If there is an object in zone 1, see the blue area below then the speed of the robot should be 0.0 m/s should be a ros parameter

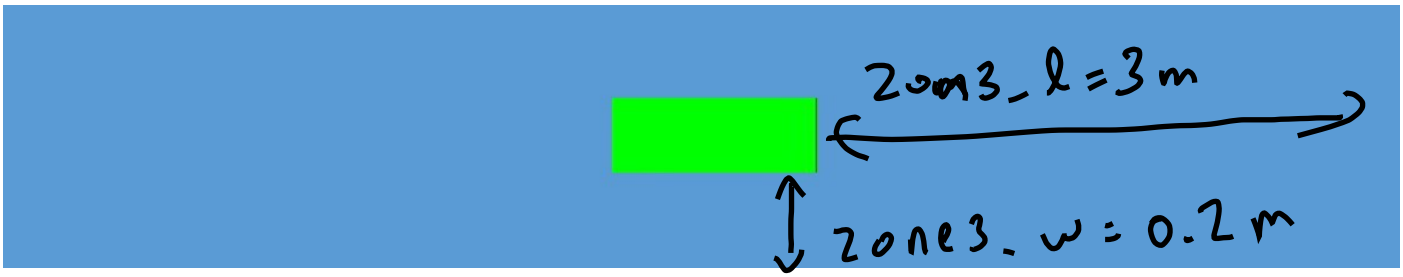


ros parameter
l, length = 0.6 m
w, width = 0.25 m
zone1_l = 0.1 m
zone1_w = 0.05 m

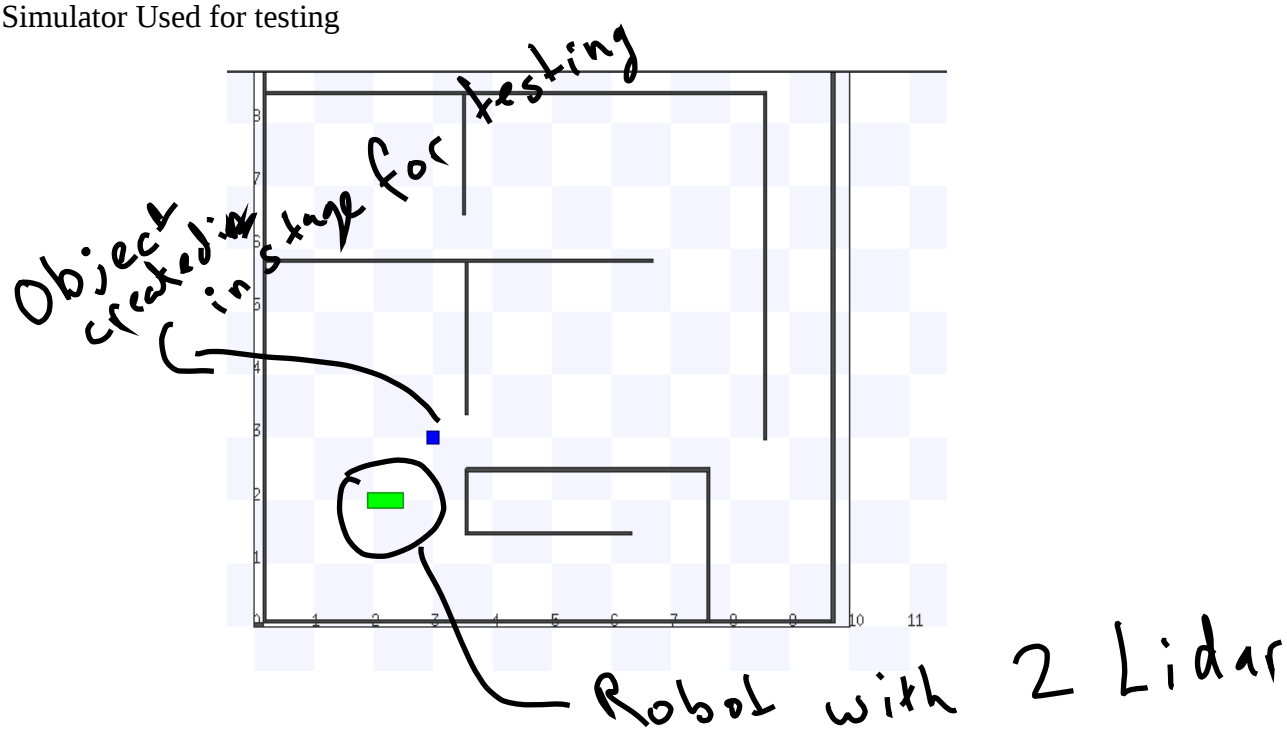
If there is an object in zone 2, see the blue area below then the max speed of the robot should be 0.5 m/s if the object is in front of the robot and -0.5 m/s if the object is behind the robot. If there is an object to the side then the speed should be between -0.5 m/s and 0.5 m/s



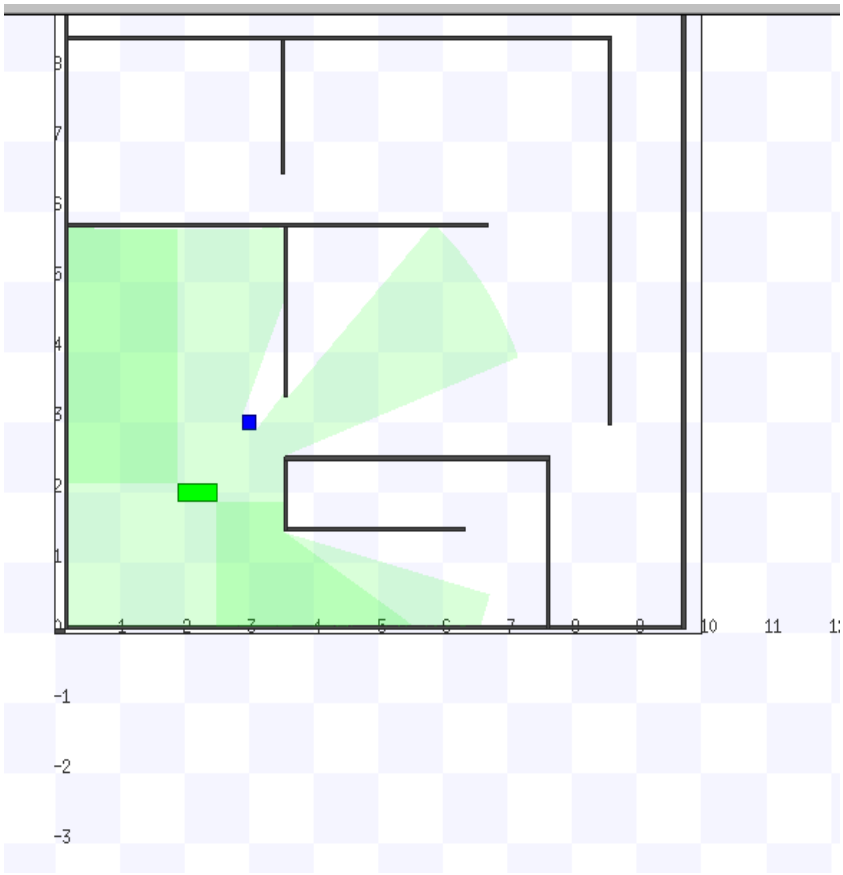
If there is an object in zone 3, see the blue area below then the max speed of the robot should be 1 m/s if the object is in front of the robot and -1 m/s if the object is behind the robot. If there is an object to the side then the speed should be between -1 m/s and 1 m/s



Stage Simulator Used for testing



To view the laser data in stage click view and then data. See below



Sample code to process laser data:

```
MIN_SCAN_ANGLE -60.0/180*M_PI  // angle range to check object
MAX_SCAN_ANGLE +60.0/180*M_PI  // angle range to check object
MIN_PROXIMITY_RANGE 0.5  // if distance less than this robot stops

void scanCallback(const sensor_msgs::LaserScan::ConstPtr& scan)
{
		// Find the closest range between the defined minimum and maximum angles
		int minIndex = ceil((MIN_SCAN_ANGLE - scan->angle_min) / scan->angle_increment);
		int maxIndex = floor((MAX_SCAN_ANGLE - scan->angle_min) / scan-
>angle_increment);

		float closestRange = scan->ranges[minIndex];
		for (int currIndex = minIndex + 1; currIndex <= maxIndex; currIndex++) {
		  if (scan->ranges[currIndex] < closestRange) {
			closestRange = scan->ranges[currIndex];
		  }
		}

		//ROS_INFO_STREAM("Closest range: " << closestRange);

		if (closestRange < MIN_PROXIMITY_RANGE) {
		  keepMoving = false;
		}
}
```