# NEATO\_LDS\_VIRTUAL\_BUMPER

Virtual bumper for the LDS

A virtual bumper was built to prevent the robot from making physical contact with other objects. For this project, a NEATO D5, a gen 2 LDS and an Arduino micro was placed inside of the robot. The Arduino is reading the LDS scan data directly off of the LDS by connecting the RX on the Arduino to test port 6 on the LDS interface board. The Arduino is then directly wired to the 4 bumpers on the D5 using a HE721C0510 Non-Latching relay to pull the switch either high or low. When the switch is receiving voltage the switch acts as if it was not pressed in. In virtualBumper.ino the left side bumper is set to pin 8 on the Arduino, right side is 9, left front is 10, right front is 11.

To parse the data, set the baud rate on the Arduino to 115200 and run the getpacket() function. Each packet received contains 4 degrees at a time and will be stored into rxbuff. Then, for each packet a for loop is used to access each degree in the packet. To acquire the distance, the dist() function is used.

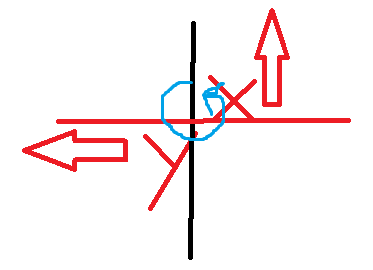
There is an increasing error in the angle as a point gets closer to the LDS (Azimuth Error due to parallax). To account for this error we take the arcsine of the distance from the center of rotation to the laser and divide that by the distance. This error changes the angle by about 0.4 to 0.7 degrees. The equation for this Azimuth error due to parallax is:

tempCorrection = asin (distanceFromCenterOfRotationTolazer / d) \* 180 / pi;

The virtual bumper is a square shape around the front of the LDS which can be seen in figure 1. The black box is the front of the D5. If a point were to be seen anywhere within the red box, a bumper will be triggered. Once a corrected angle is acquired the point is converted to X and Y coordinates using the following equations where tempa is the corrected angle and pi180 is pi / 180 to convert the angles to radians:

x = d \* cos(tempa \* pi180);

y = d \* sin (tempa \* pi180);

Positive X is up, positive y is to the left, increasing angle is clockwise and 0 starts on the vertical line. The origin is the center of the LDS.

After testing to see if the border is accurate, it seems as if all of the points are shifted by 4-8 degrees. A visual of this shift can be seen in figure 2. The border was measured using the output printing from the Arduino, a ruler, and a pencil. As the pencil is translated horizontally across the front of the robot towards the left, the y values increase and the x values decrease. As the pencil follows a line similar to what is shown in figure 2, the y values increase and the x values do not change which cause a lot of confusion. To fix this problem a rotation was applied to every point so that the measured border more closely matches figure 1. The equation to rotate each point is:

X = x\*cos(θ) - y\*sin(θ)

Y = x\*sin(θ) + y\*cos(θ)

This gives the location of a point rotated θ degrees about the origin (center of the LDS). After performing this rotation, a border similar to figure 3 is seen where the front border is now straight but the sides are moving inwards towards the LDS. This caused more confusion, and in an attempt to temporarily solve this issue a new border was made for the left and right sides of the robot. The code that should define the border original is:

where distance is the value set to change how far away the virtual bumper should be from the robot (in this example distance is set to 10).

if (y >= -168 - distance && y <= 168 + distance) {

if (x >= 160 && x <= 250 + distance) {. . .}}

The values for these points are acquired by the drawing sent from Mark Kelley which can be seen in figure 4. From previous tests using a LDS data visualizer it seemed as if the left and right side of the robot is not symmetrical. It looks like the left side of the robot is 10 millimeters longer than the left side, but from figure 4 it shows that it is symmetrical. After running tests measuring the sides of the robot using the LDS data visualizer, the virtual bumper tests, and going through the UART and running the getldsscan command it shows that the left side of the bumper is 10 millimeters larger than the value that is stated in figure 4. To account for that an extra 10 mm is added to the left side of the robot when calculating the virtual bumper locations. Then, to fix the issue with the angled sides, the if statement that determines the left and right borders have been changed to follow a non-straight line rather than a vertical line. The new border definition is:

if (y >= ((-1 / x) \* 100) + (-168 - distance - .5) && y <= (1 / x) \* 100 + (168 + distance + distance - .25)) {

if (x >= 160 && x <= 250 + distance) {. . .}}

There have been no calculations to determine whether or not (1/ x) \* 100 is the optimal curves to follow but it allows for a border that much more closely represents figure 1. The addition of -.5 and -.25 in the calculation for the y values correct for the error in distance that is produced by the inaccurate curve that defines the sides.

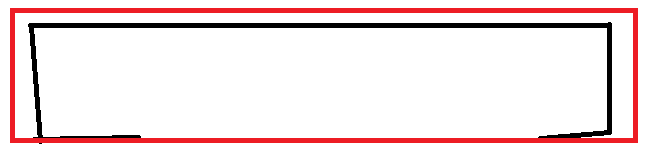


Figure 1Ideal border

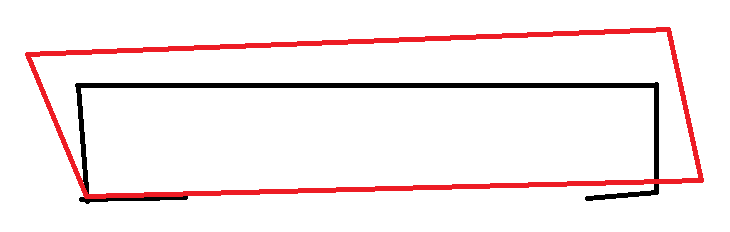


Figure 2actual border

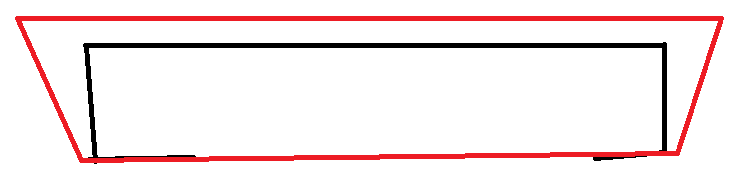


Figure 3border after rotation

The code that determines the angles that trigger the bumpers are:

if (a > 6 && a < 25) {

/\*Left front bumper\*/}

} else if ( a > 34 && a <= 90 ) {

/\*Side left bumper\*/}

} else if (a > 335 && a < 354) {

/\*right front bumper\*/}

} else if (a >= 270 && a <= 330) {

/\*right side bumper\*/ }

else if (a >= 354 || a <= 6) {

/\*both front bumper\*/ }

else if (a >= 25 && a <= 34) {

/\*left corner bumpers\*/ }

else if (a > 330 && a <= 335) {

/\*right corner bumpers\*/ }

There are 4 bumpers and 3 areas where multiple bumpers need to be triggered. These angles can be changed until the performance of the robot matches what is expected.

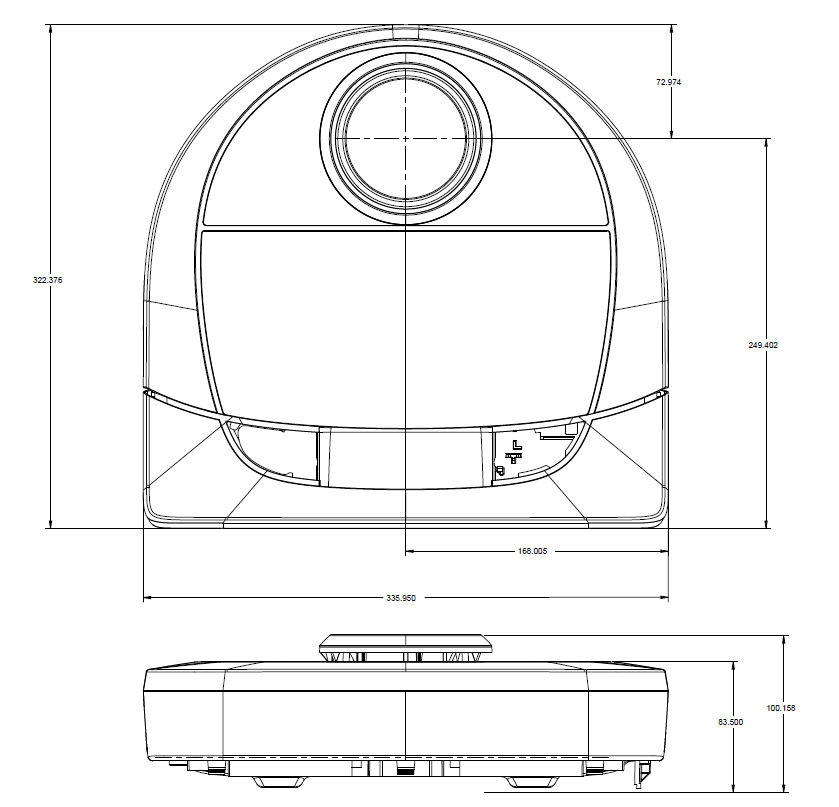


Figure 4Drawing of D5

Future work on this project would be to figure out exactly why there is such a large shift in the points and to figure out what is needed to correct for that shift so that a more accurate border can be generated. With the current version of virtualBumper.ino the border is approximately .75 centimeters away from the front of the robot WITHOUT the bumper attached.