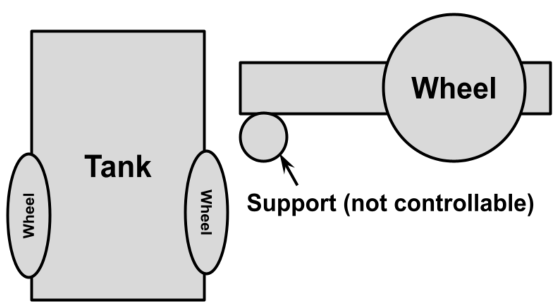
Determining 2D Position And Orientation Of A Tank-Like Vehicle While Minimizing External Measurements

Samiyel D Frazier
Aztec Robotics, Barstow High School
Barstow, California
samiyelfrazier@gmail.com

Abstract – This paper describes how and why to measure both the position and orientation of a vehicle with dual-wheel inputs in a global scope, with no outside measurements beyond the initial conditions. For the purposes of this paper, "Tank", "Tank-Like", and "Bot" will refer to any vehicle with exactly two independent and fixed wheels or treads; The wheels must have only one degree of freedom, allowing each to go forward/backward.



To achieve this, we use both an INS (Inertial Navigation System) developed by Charles Stark Draper and a CBDP (Cardioid-Based Delta Position) developed by students from the Aztec Robotics team. Effective use of both of these methods has varying results due to slippage of the wheels (partially accounted for by the INS), motors that lose effectiveness over time, and accuracy of initial measurements. The Aztec Robotics team was able to achieve a 3% average margin of error in initial testing without the use of an INS.

Index Terms - Autonomous robots, Motion analysis, Robot control, Robot motion, Simultaneous localization and mapping

I. INS Overview

An Inertial Navigation System (INS) Is a method of determining the position of a vessel via the use of an accelerometer and gyroscope. Developed by Charles Stark Draper before GPS was available.[1] In 1954, The military put this tech on nuclear submarines because it is hard to see underwater, and there were no maps to be of use even if they could see. submarines were inaccurate in their reported location, which the government did not like. To more effectively tell the location of their vessels, the navy used a gyroscope to tell the orientation (heading) of the ship, and an accelerometer to tell the velocity of their ship (via the change in velocity, and therefore change in position). This method has been adapted for use in Botball for years, as documented in the KIPR Archives.[2]

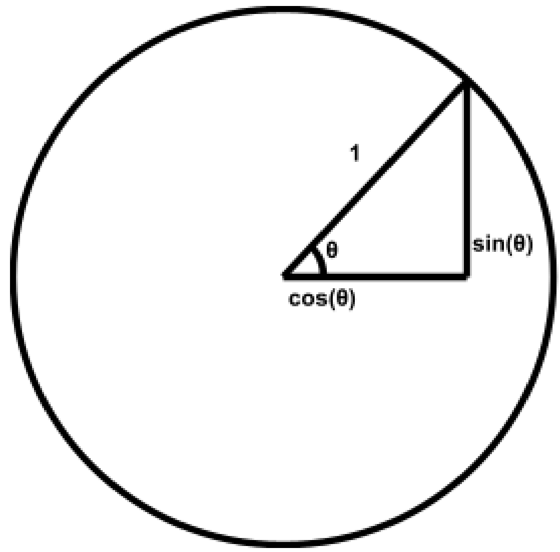
II. CBDP Overview

The Cardioid-Based Delta Position (CBDP) is a different solution to the same problem, this time developed by Aztec Robotics. The concept is as follows: If we can measure how far each wheel has traveled, we can measure how far the robot has traveled. This method can be broken down into 3 smaller, and somewhat computationally efficient methods.

A. Method 1: Count

In the scenario in which both wheels traveled an equal distance, one can assume the robot has had no change in orientation. When this is the case, you must only measure how far the wheel has traveled, and convert that angular measurement into a linear measurement of the robot. For example, if a wheel moving 10 degrees is equal to the bot moving 1 inch, a simple unit conversion can tell us where the bot is after any number of degrees change in wheels. You cannot simply add the linear value to the bot's coordinate though, as moving forward when facing north is not the same as moving forward when facing east.

To effectively measure this, you must measure the sin (y value for any given angle) and cosine (x value for any given angle),



and multiply both by the distance traveled. The resulting equations should look as follows

$Inches(\theta) = 0.1 \cdot \theta$

$Distance = Inches(\Delta degrees)$

$\Delta x = Distance \cdot cos(orientation)$

$\Delta y = Distance \cdot sin(orientation)$

B. Method 2: Pivot

In the scenario in which the Δdegrees of both wheels are equal opposites (i.e. -10, +10), the robot will have turned on the center point of an imaginary "axel", being the midpoint of both wheels. The robot will have stayed at the same coordinate, as if spinning on its heel. To measure the robot's change in orientation via the wheel's change in angle, we use the measurement of a cardioid. A cardioid is a graph drawn by measuring a point on a circle as it rotates around the circumference of a larger circle with no slippage.[3] A visual intuition can be understood via https://www.desmos.com/calculator/7ytn9gpo1j.

By measuring a "turn rate" (being the angular offset of the center circle as compared to the angular offset of the rotating circle), two equations emerge.

$\tau(turn\,rate) = \frac{Axle\,Circumference}{Wheel\,Circumference} + 1$ [4]

$\Delta orientation = \frac{\Delta degrees}{\tau}$

C. Method 3:

When neither of these scenarios apply, we must use a less computationally efficient method. First, we must calculate the curvature (as a radius) the robot experienced.

$L = Inches(\Delta degrees)$

$R = Inches(\Delta degrees)$

$C(curvature\ as\ a\ radius) = \frac{(L+R)}{(L-R)}$

Then, we must calculate the length of the arc, or distance the robot traveled.

$\alpha = \frac{(L+R)}{2}$

Next, we calculate the angle that the arc makes given the radius

$\theta = \frac{\alpha}{|C|}$ (Curvatures to the left and right give different radii, which approach infinity as the path becomes straighter.)

After that, we calculate the xy coordinate that the robot started at as if it was rotating around (0,0).

$G = (|C|, 0)$ (a point along the circle of curvature)

$x_1 = (G.x \cdot cos(\theta)) - (G.y \cdot sin(\theta))$

$y_1 = (G.x \cdot sin(\theta)) + (G.y \cdot cos(\theta))$

Which can be simplified to

$x_1 = |C| \cdot cos(\theta)$

$y_1 = |C| \cdot sin(\theta)$

(x_1, y_1) is the new position around the center of rotation as though the center of rotation is (0,0).

We simply add (x_1, y_1) to the already stored coordinate of the previous position, and we can find the new position of the robot. To find the new orientation, we similarly add θ to the previous orientation of the robot.

III. Implementation:

To bring all of this together, we first make a function that determines the method to use.

If the values (L, R) are equal: use method 1.

If the values are not equal, and the absolute values of the values are equal (equal opposites): use method 2. Otherwise: use method 3.

The beauty of this is that it measures everything relative to the robot's previous position, and therefore relative to its initial position. If we simply lie to the computer and say it started at a different position, it will be none the wiser. This means that if we take an initial measurement of where the robot is at boot-up, we can consistently get a global position, rather than a relative one.

The CBDP is not without its faults, though. Wheel slippage, motor inaccuracies, and other unaccounted-for variables can harm the program's accuracy, which compounds over time. Incorporation of an INS can make inaccuracies less fatal to the program.

Acknowledgements:

I would like to thank Timothy Brinkley, the Aztec Robotics coach, for his continued support in this endeavor. allowing me the time, creative freedom, and trust to research, develop, and test these methods made this possible. These results would not have been achieved if it weren't for his expert guidance. I would also like to thank Edgar Moran for his assistance when I was first writing and testing the math you see above. His knowledge of math and organizational skills played a very large part in the success of this project.

[1] Unknown, in *Developing Inertial Navigation,* Washington D.C., USA, Smithsonian

[2] Yusen Hu, in *Utilizing the Inertial Gyroscope for Reliable Inertial Navigation*, Beijing, China, Dulwich College, 2019

[3]Yao Liu, in *Introducing Calculus: The Cardioid*, Ithaca, New York, USA, Cornell, pp. 4

[4]Unknown, in *Error Found in S.A.T. Question*, New York, New York, USA, New York Times, 1982