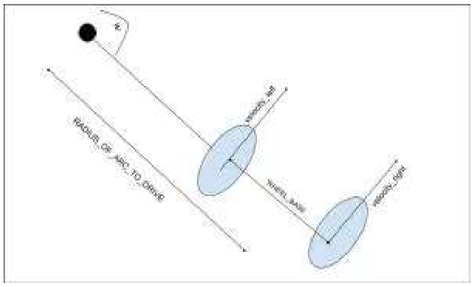
Drive Kinematics: Skid Steer & Mecanum (ROS Twist included)

by David Kohanbash *on* June 22, 2016



Hi all
I am often in need of the basic kinematic motion equations for skid steer vehicles. I have also recently been working with mecanum wheeled vehicles. The skid steer equations are fairly simple and easy to find, however I will include it in different versions and include a ROS approach. The mecanum wheel equations are harder to find and there are different versions floating around. The first version I found had a lot of trig and mostly worked. The version I present here is easier to intuitively understand and seems to work better (I don't need a random scale factor for this version), I also include a ROS approach for them.

Skid Steer / Differential Drive

Here is some math for 2 and 4 wheel differential drive vehicles, 2 wheels and a castor, or skid steer tracked vehicles.

Arc based commands

The basic skid steer equations are:

> *velocity_right = w(RADIUS_OF_ARC_TO_DRIVE + WHEEL_BASE/2)*
> *velocity_left = w(RADIUS_OF_ARC_TO_DRIVE – WHEEL_BASE/2)*

Where w is the angular rotation, RADIUS_OF_ARC_TO_DRIVE is the arc radius that the robot should drive, and the WHEEL_BASE is the distance from the center of the left wheel to the center of the right wheel (See image above).

This can also be written as:

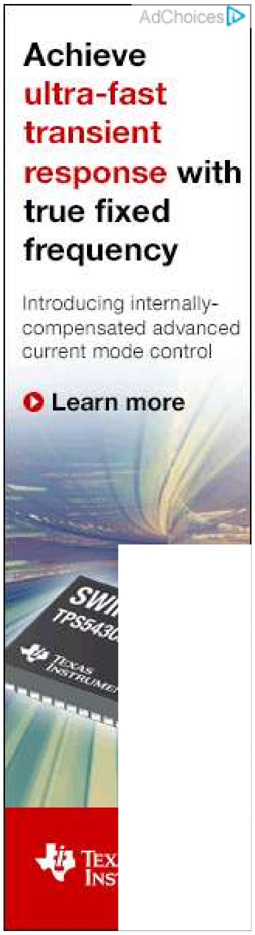
Robotics Blog for the Professional Roboticist

Subscribe to Blog via Email

Enter your email address to subscribe to this blog and receive notifications of new posts by email.

Email Address

Subscribe



```
w = (velocity_right-velocity_left)/WHEEL_BASE
```

There are two special cases:

```
IF velocity_right == velocity_left :
THEN the radius of the arc is infinite so the robot will drive straight.

IF velocity_right == -velocity_left :
THEN the radius of the arc is 0, and the robot rotates in place (ie. point turn)
```

Linear & Angular Velocity Commands for ROS

In ROS if using the Twist topic (which is the default for drive messages) (message name is often cmd_vel) you will often set linear_velocity in the linear.x field and angular_velocity in the angular.z field.

```
velocity_left_cmd = (linear_velocity – angular_velocity * WHEEL_BASE / 2.0)/WHEEL_RADIUS;

velocity_right_cmd = (linear_velocity + angular_velocity * WHEEL_BASE / 2.0)/WHEEL_RADIUS;
```

Mecanum Wheel Math

Mecanum wheels from Andymark

In ROS if using the Twist message you will often set the linear.x, linear.y and angular.z fields. One unrelated note is that if you are operating on uneven terrain then doing mecanum type motions will fail and have a lot of slip. Skid steer type motions will often work better (using the mecanum wheels).

```
WHEEL_SEPARATION_WIDTH = DISTANCE_LEFT_TO_RIGHT_WHEEL / 2

WHEEL_SEPARATION_LENGTH = DISTANCE_FRONT_TO_REAR_WHEEL / 2
```

Forward kinematics

Wheel commands units are in rad/s

```
wheel_front_left = (1/WHEEL_RADIUS) * (linear.x – linear.y – (WHEEL_SEPARATION_WIDTH + WHEEL_SEPARATION_LENGTH)*angular.z);
```

Your support is appreciated

BECOME A PATRON

Top Posts & Pages

- Upcoming Robotics Conferences & Events List
- Kalman Filtering - A Practical Implementation Guide (with code!)
- PID Control (with code), Verification, and Scheduling
- LIDAR vs RADAR: A Detailed Comparison
- Motors and Feedback (encoders)
- Robotics for Kids (and Adults) - Getting Started and How to Progress
- Selecting and Sizing Your Motor (and gear boxes)
- LIDAR (Laser Scanner) Fundamentals

Categories

- Computing & Networking
- Electrical & Electronics
- Mechanical
- Misc
- Research
- Robots
- Software & Algorithms
- Systems & Sensors
- Testing
- Uncategorized

Upcoming Events

MON
08
JAN
2018

ISR 2018 Abstracts Due

TUE
09

CES 2018

> wheel_front_right = (1/WHEEL_RADIUS) * (linear.x + linear.y + (WHEEL_SEPARATION_WIDTH + WHEEL_SEPARATION_LENGTH)*angular.z);
>
> wheel_rear_left = (1/WHEEL_RADIUS) * (linear.x + linear.y – (WHEEL_SEPARATION_WIDTH + WHEEL_SEPARATION_LENGTH)*angular.z);
>
> wheel_rear_right = (1/WHEEL_RADIUS) * (linear.x – linear.y + (WHEEL_SEPARATION_WIDTH + WHEEL_SEPARATION_LENGTH)*angular.z);

To drive a robot you will probably need to also invert one side since the motors are mounted opposite the other side. For example:

> wheel_front_right = -1 * wheel_front_right
>
> wheel_rear_right = -1 * wheel_rear_right

Also this gives an output in rad/s. If your motor controller is operating with encoder counts as the unit you will need to convert the units.

Inverse Kinematics

> linear.x = (wheel_front_left + wheel_front_right + wheel_rear_left + wheel_rear_right) * (WHEEL_RADIUS/4)
>
> linear.y = (-wheel_front_left + wheel_front_right + wheel_rear_left – wheel_rear_right) * (WHEEL_RADIUS/4)
>
> angular.z = (-wheel_front_left + wheel_front_right – wheel_rear_left + wheel_rear_right) * (WHEEL_RADIUS/(4 * (WHEEL_SEPARATION_WIDTH + WHEEL_SEPARATION_LENGTH)))

Source for mecanum wheel math: here. There are other versions of how to compute the wheel velocities but this is the one I like best.

I know this post was terser than most of my posts but I hope this math helps you.

Liked it? Take a second to support David Kohanbash on Patreon!

BECOME A PATRON

Share this:

Facebook 2 | Google | LinkedIn 9 | Twitter | Reddit | Email | More

Related

JAN
2018

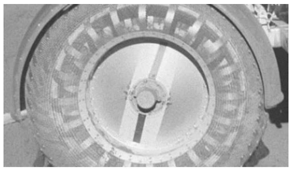
THU
01
FEB
2018
RSS 2018 Papers Due

SUN
04
FEB
2018
AAAI 2018 Conference

MON
26
FEB
2018
Autonomous Vehicles 2018 Summit

See All Events

Drive Selection - Wheels, tracks and more
July 25, 2016
In "Mechanical"



Wheel & Grouser Design
September 29, 2014
In "Mechanical"

Book Review: Mastering ROS for Robotics Programming by Lentin Joseph
March 17, 2016
In "Misc"

Tags: differential, drive, kinematics, mecanum, motion, ROS, skid, steer, twist, wheels

Categories: Software & Algorithms

Comments

Joseph Salmon

June 22, 2016 at 11:20 am

Please keep me informed.

Reply

David Kohanbash

June 22, 2016 at 11:24 am

Informed about what?

Reply

Drive Selection - Wheels, tracks and more - Robots For Roboticists

July 25, 2016 at 11:02 am

[…] the past we have looked at wheel design and the kinematics of skid steer and mecanum wheels. In this post we will take a quick look at different types of mobility types (ie. wheels, tracks, […]

Reply

Kevin Mulligan

October 12, 2016 at 4:57 pm

Thank you very much for the kinematics code. I have one comment on the content, I'm fairly certain that the terms forward and inverse have been swapped around.

Tweets by @DavidKohanbash

David Kohanbash
@DavidKohanbash

Lithium, Lithium-ion, LiPo & LiFePO4 Battery Safety and Fire Handling
robotsforroboticists.com/lithium-ion-ba…

Lithium, Lithium-ion,…
Lithium battery safety …
robotsforroboticists.com

Nov 14, 2017

David Kohanbash
@DavidKohanbash

Just helped to save a robot: How to Rescue a Non-booting GRUB 2 on Linux
linux.com/learn/how-resc…

How to Rescue a No…
Once upon a time we …
linux.com

Nov 14, 2017

David Kohanbash
@DavidKohanbash

Pulurobotics: Why don't we use ROS?!
pulurobotics.fi/blog/pulurobot…

Why don't we use R…
pulurobotics.fi

Nov 13, 2017

David Kohanbash
@DavidKohanbash

UEFI: Building a Better BIOS
logicsupply.com/explore/io-hub…

UEFI: Building a Better BIOS
The basic input/output system (BIO…
logicsupply.com

Nov 10, 2017

Embed View on Twitter

RSS Feeds

My understanding is that inverse kinematics is when you have a desired outcome in physical space and want to know what to set the motors to. While forward kinematics is when you know what the motors are doing and would like to calculate where the robot will go.

Reply

Kevin Mulligan

October 12, 2016 at 4:58 pm

I forgot to enable notifications on my original comment.

Reply

David Kohanbash

October 30, 2016 at 8:21 am

How I phrased it above is how I (and others) talk about drive motions for wheels. It is not used the same as it is used with arms.

Technically you may be correct…

Reply

Leave a Reply

Enter your comment here...

RSS - Posts

Links

- Autonomous Robots Blog
- eevblog
- hizook
- robots.net
- Robots Pod Casts
- Robotics Online
- Society of Robots
- The Robot Report

The information contained in this website is for general information purposes only. The information is provided freely by Robots for Roboticists and while we endeavor to keep the information up to date and correct, we make no representations or *warranties of any kind, express or implied, about the completeness, accuracy, reliability, suitability or availability with respect to the website or the information, products, services, or related graphics contained on the website for any purpose. Any reliance you place on such information is therefore strictly at your own risk.*

Privacy Policy|Log In

Built with WordPress & Slipstream Theme by ThemeLab