F1TENTH Autonomous Racing

Pose Representation and Coordinate Transformation

Rahul Mangharam
University of Pennsylvania
rahulm@seas.upenn.edu

Penn Engineering

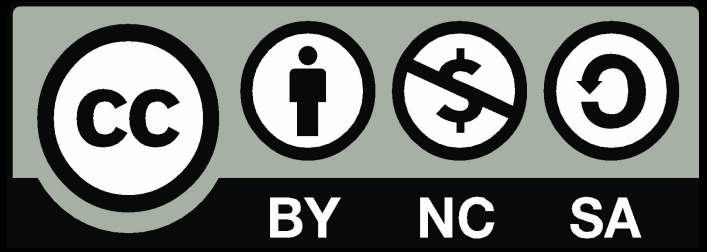
F1 TENTH

Safe Autonomy Lab
University of Pennsylvania

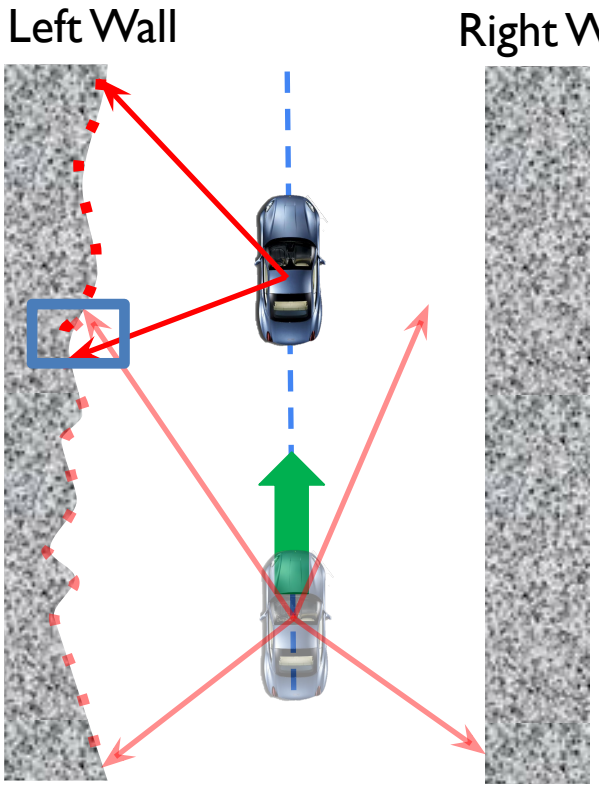
Acknowledgements

This course is a collaborative development with significant contributions from:
Hongrui Zheng (lead), Matthew O'Kelly (lead), Johannes Betz (lead), Joseph Auckley, Madhur Behl, Luca Carlone, Jack Harkins, Paril Jain, Kuk Jang, Sertac Karaman, Dhruv Karthik, Nischal KN, Thejas Kesari, Matthew Lebermann, Kim Luong, Yash Pant, Varundev Shukla, Nitesh Singh, Siddharth Singh, and many others.

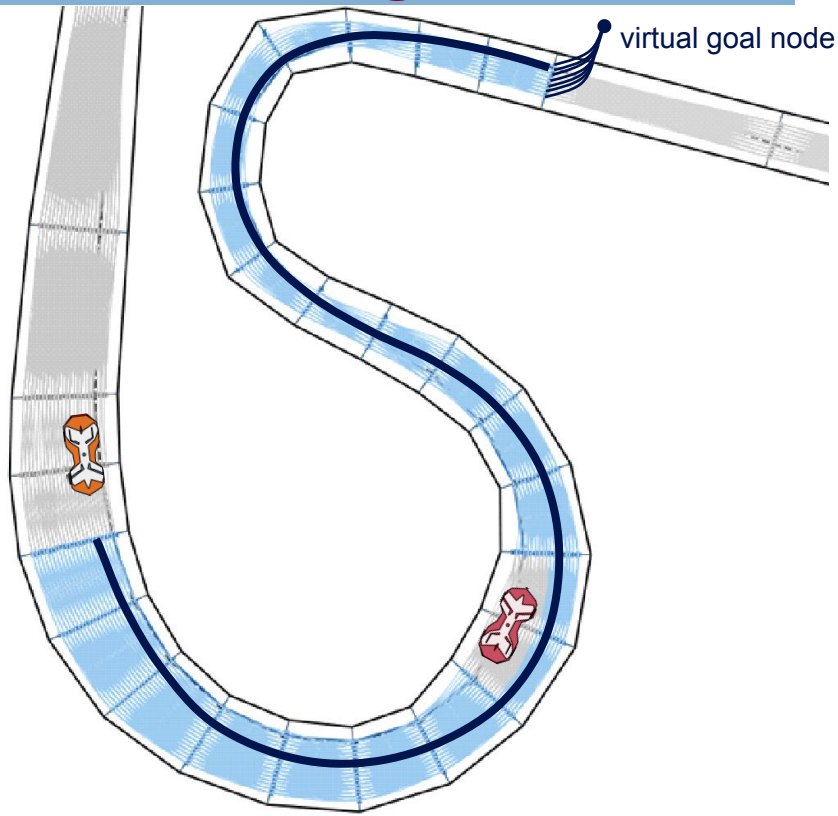
We are grateful for learning from each other



Where am I? Where do I want to go?



We want to know where the racecar is in space, and how it got there.



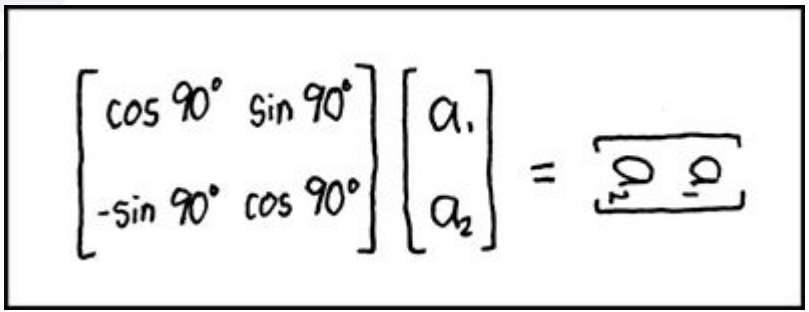
We want to plan how to get to a goal position through a sequence of movements.

Frames of Reference

It's just different ways of looking at the same point from the agent, sensor, actuator or the world.

What we'll cover today

1. Coordinate Frames and Transformations: why?
2. F1Tenth Reference Frames and Rigid Body Transformations
3. ROS-specific frames and transformations using tf2_ros



https://imgs.xkcd.com/comics/matrix_transform.png

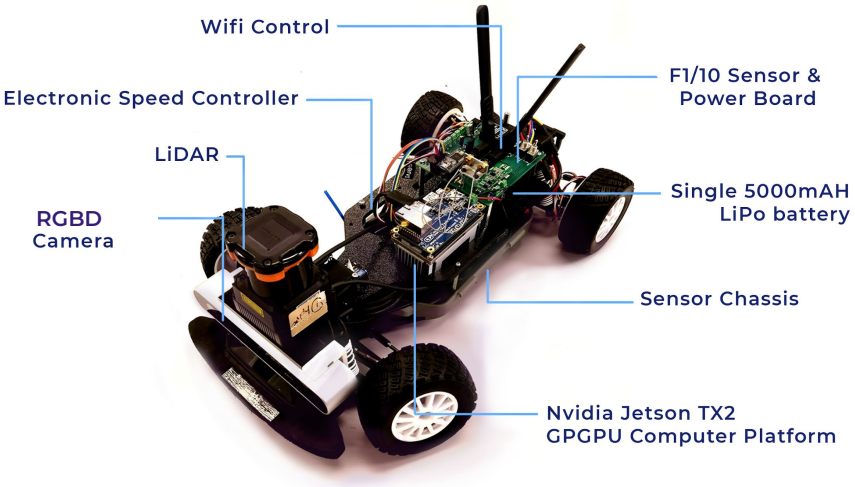
Frames and Transformations on Autonomous Vehicles

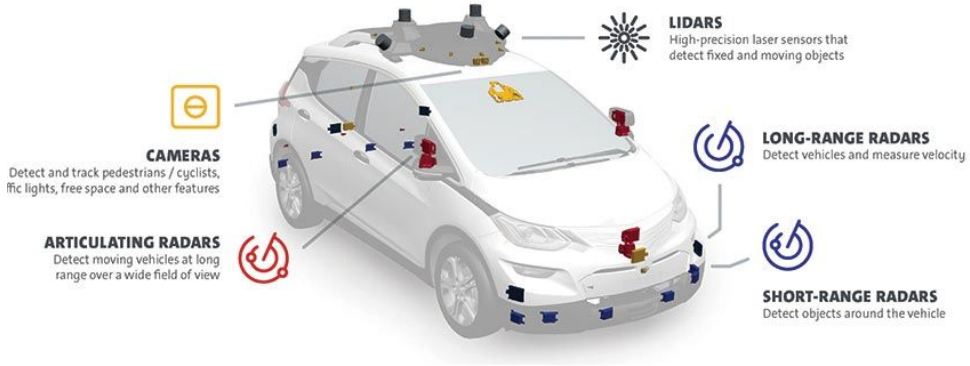
Penn Engineering × F1 TENTH

Why do we need Frame Transformations on AV (1)

Reason 1:

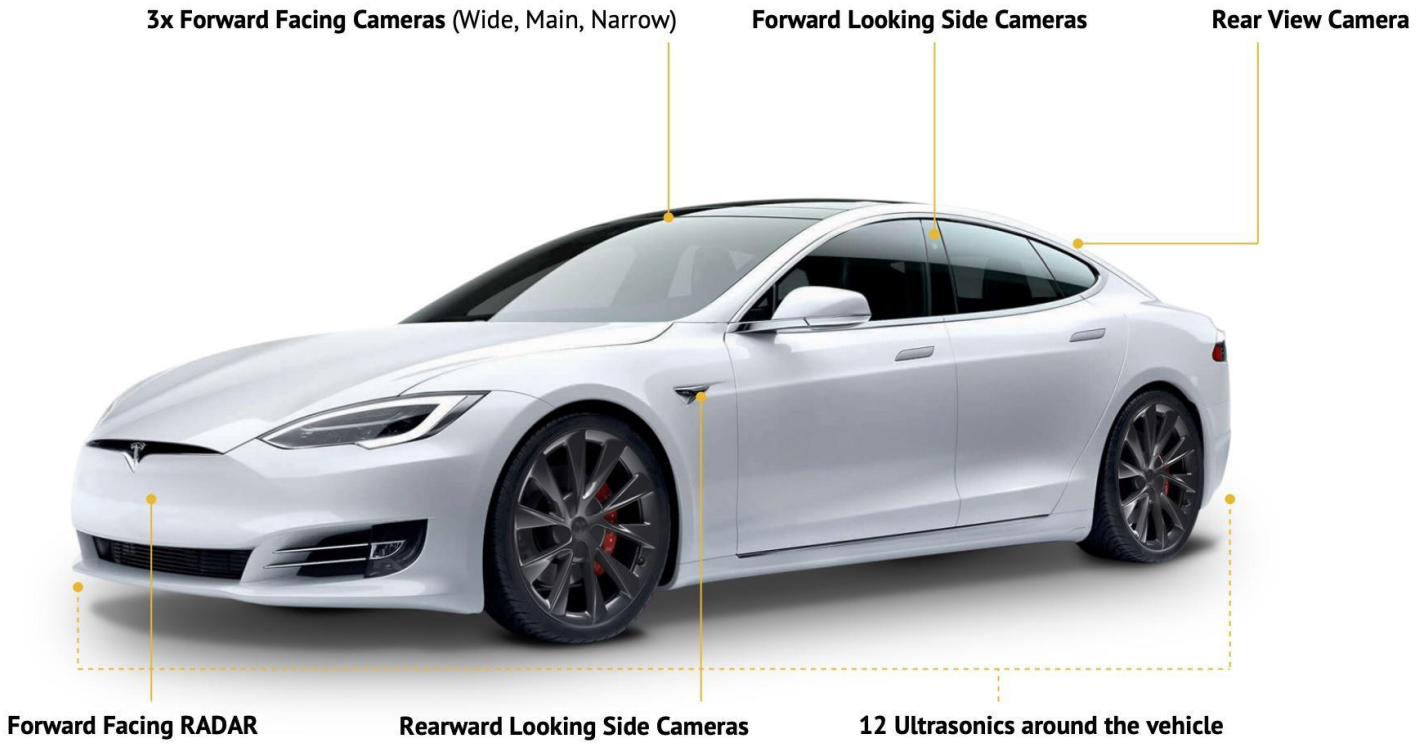
Sensors provides measurement in the frame of reference specific to that sensor





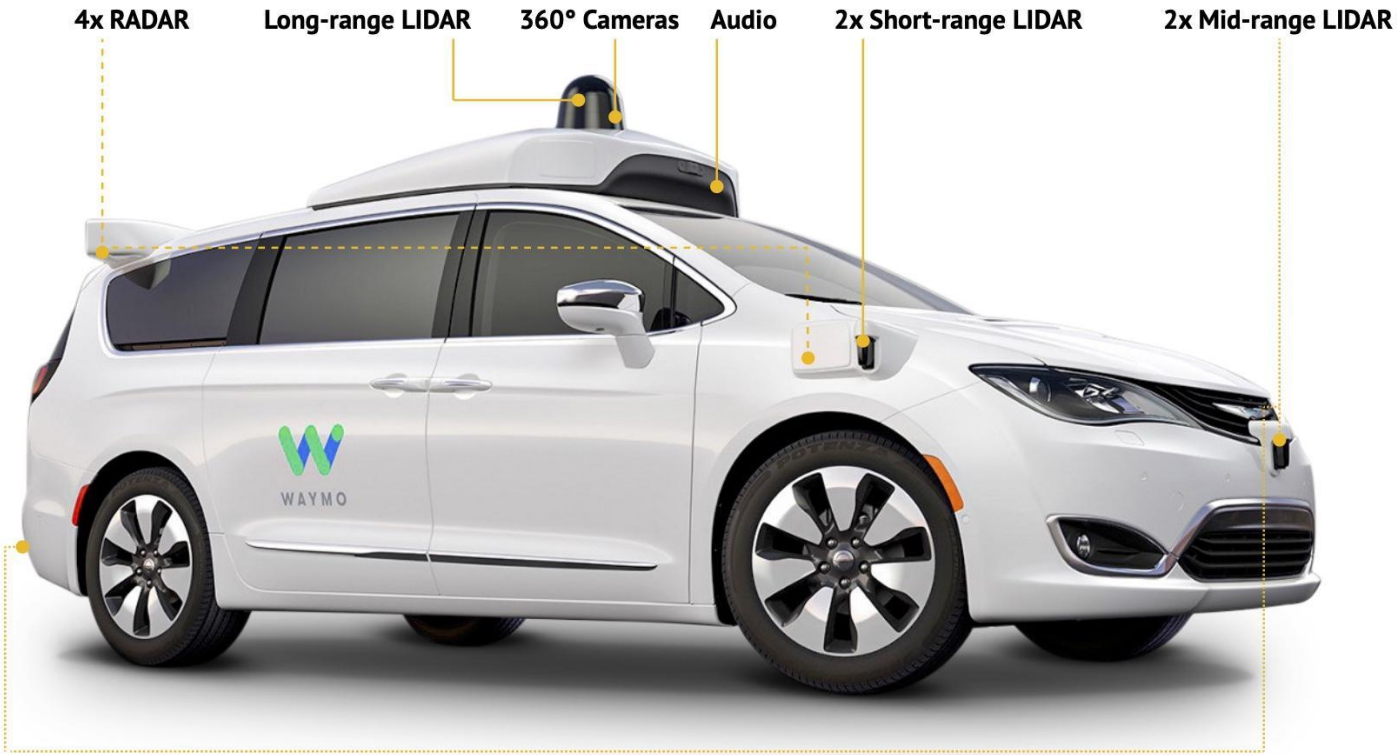
www.autonomousvehicletech.com

Tesla



Cameras! Lots of cameras!

Waymo



LIDARS! Lots of LIDARS!

Uber

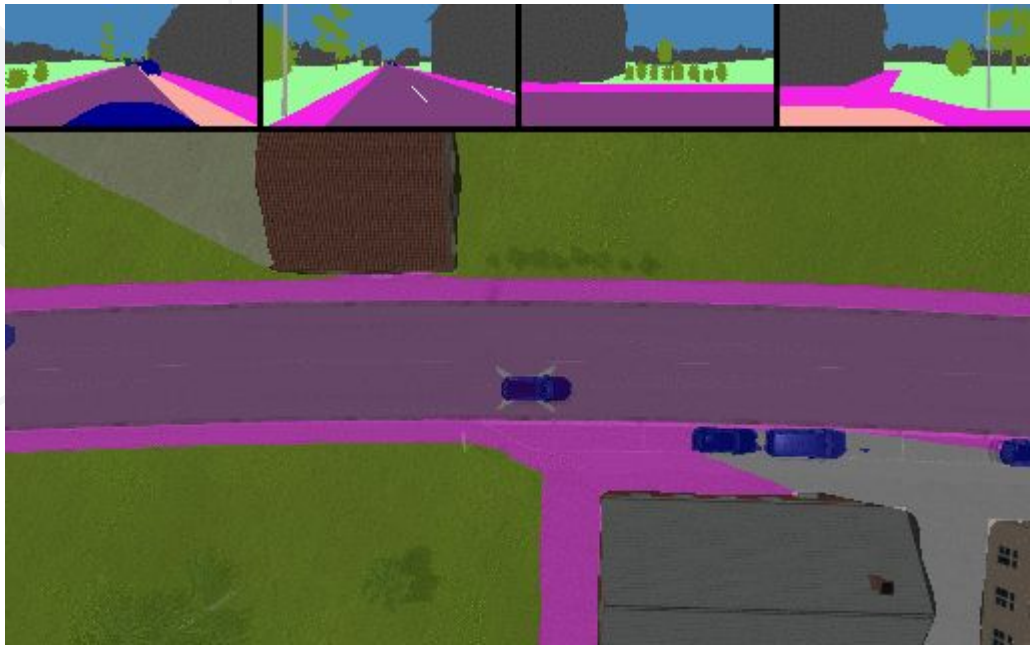
Cameras and LIDAR

Navigation requires fusing multiple sensor perspectives

Each sensor is processed individually

Navigation requires fusing multiple sensor perspectives

Multiple sensor outputs are combined to provide a combined Birds Eye View

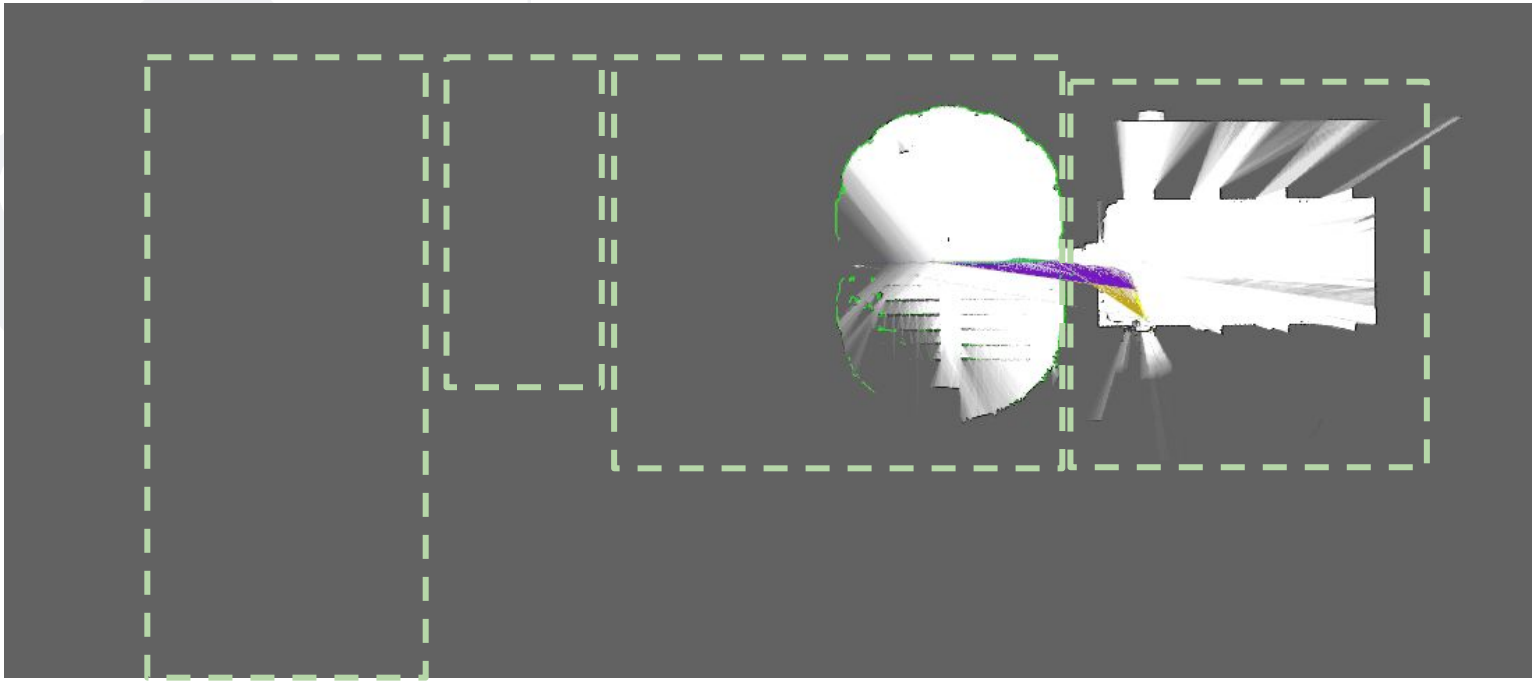


Input semantic segmentation images and predicted BEV map in Cam2BEV (source)

Why do we need Frame Transformations on AV (2)

Reason 2:

Mapping is usually done by tying submaps together



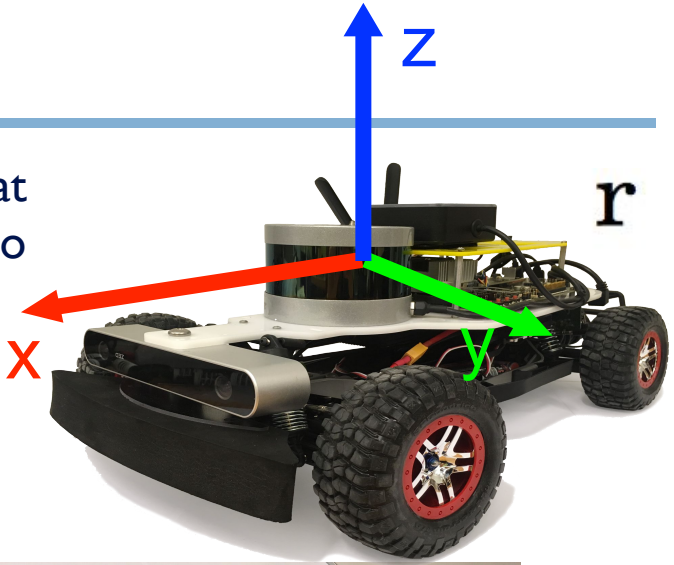
https://github.com/googlecartographer/cartographer_ros

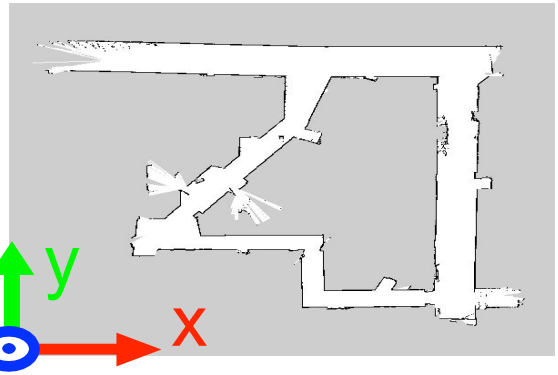
Coordinate Frames

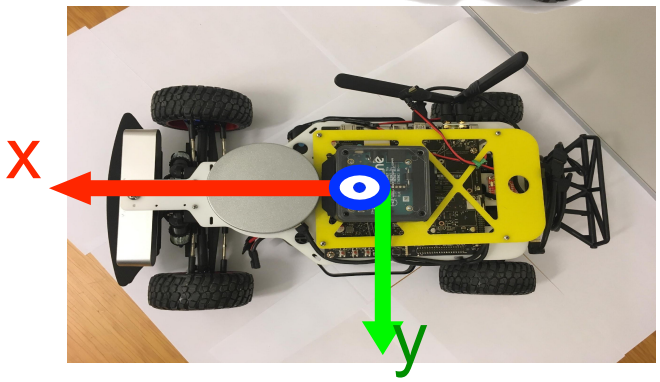
What is a Coordinate Frame?

1. A set of orthogonal axes attached to a body that serves to describe position of points relative to that body

2. **axes** intersect at a point known as the **origin**



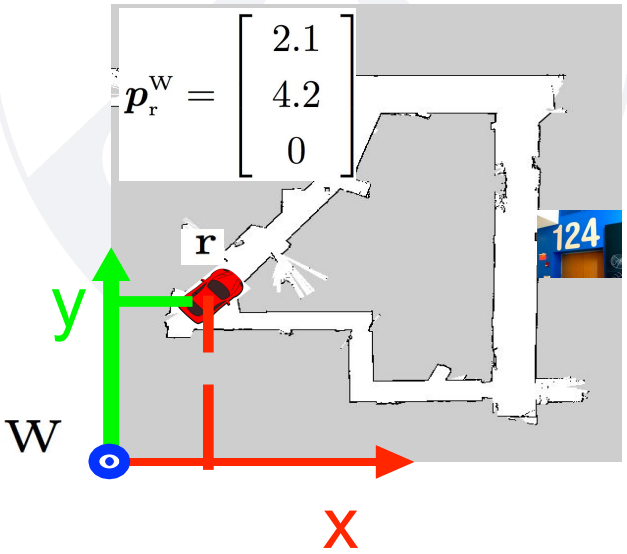


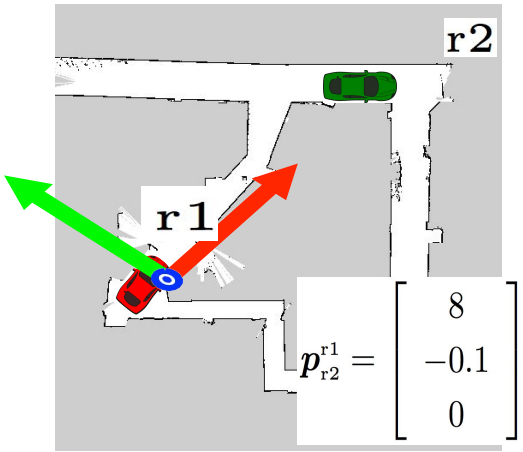


In many robotics problems,
the first step is to assign a coordinate frame to all objects of interest

Why do we need Coordinate Frames?

1. Compact representation of points
2. Compact representation of orientations



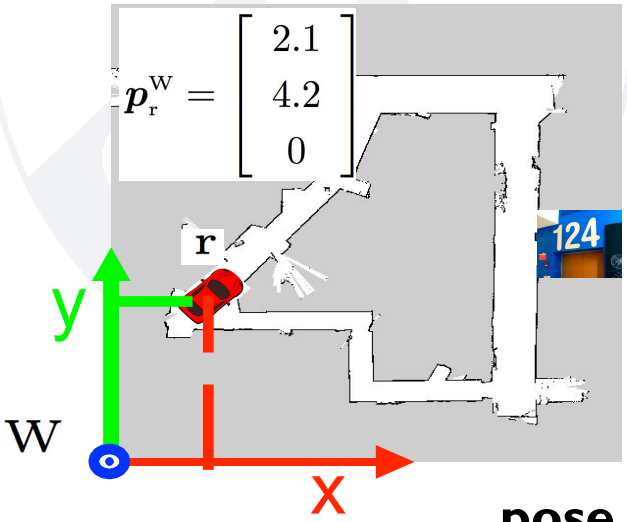


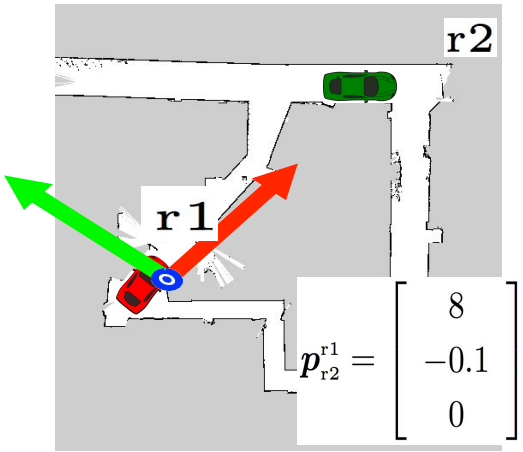
Absolute coordinates in World Frame

Relative coordinates between robots

Why do we need Coordinate Frames?

Coordinates have no meaning without specifying coordinate frame

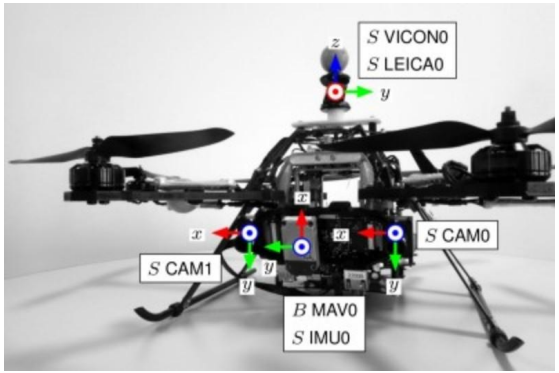


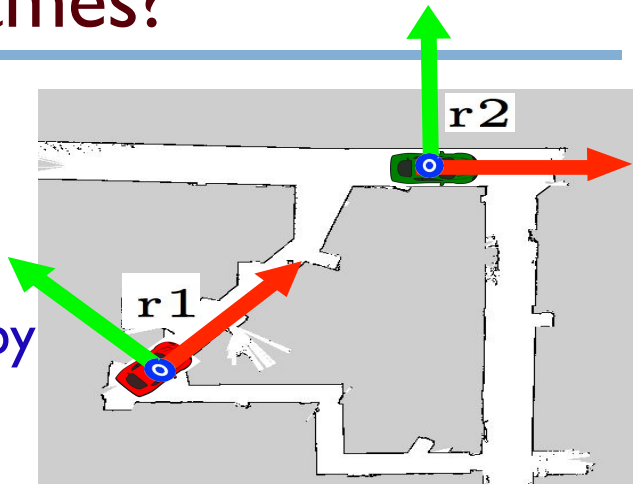


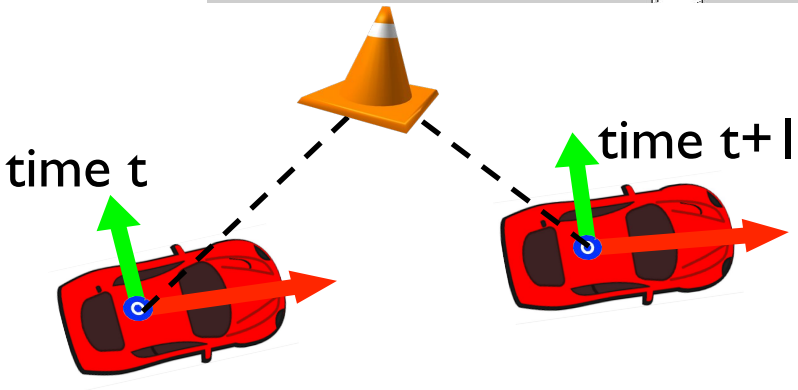
pose = position & rotation

Why do we need Coordinate Frames?

1. Compact representation of points
2. Compact representation of orientations
3. Understand and relate quantities observed by
   a. different robots and sensors
   b. at different time steps



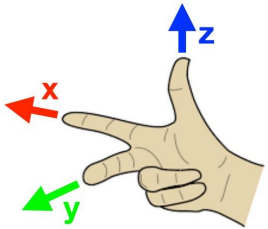


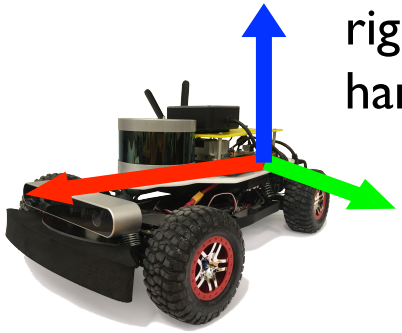


Rigid body transformations transform points from one frame to another

Right-handed Coordinate Frames

Direction of axes is important!

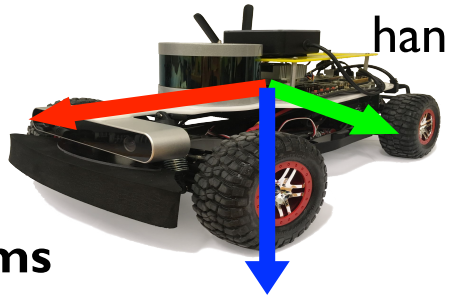




right-handed

There is a common mnemonic:

- positive **x axis** points along your **index** finger, positive **y axis** points along your **middle** finger.
- In so-called "**right-handed**" coordinate systems, positive z-axis points along the thumb of your right hand.
- In so-called "**left-handed**" coordinate systems, positive z-axis points along the thumb of your left hand.



left-handed

- **robotics always uses right-handed coordinate systems**
- left-handed systems are sometimes used in graphics

Transformations and Frames

Transformations and Frames: Map Frame

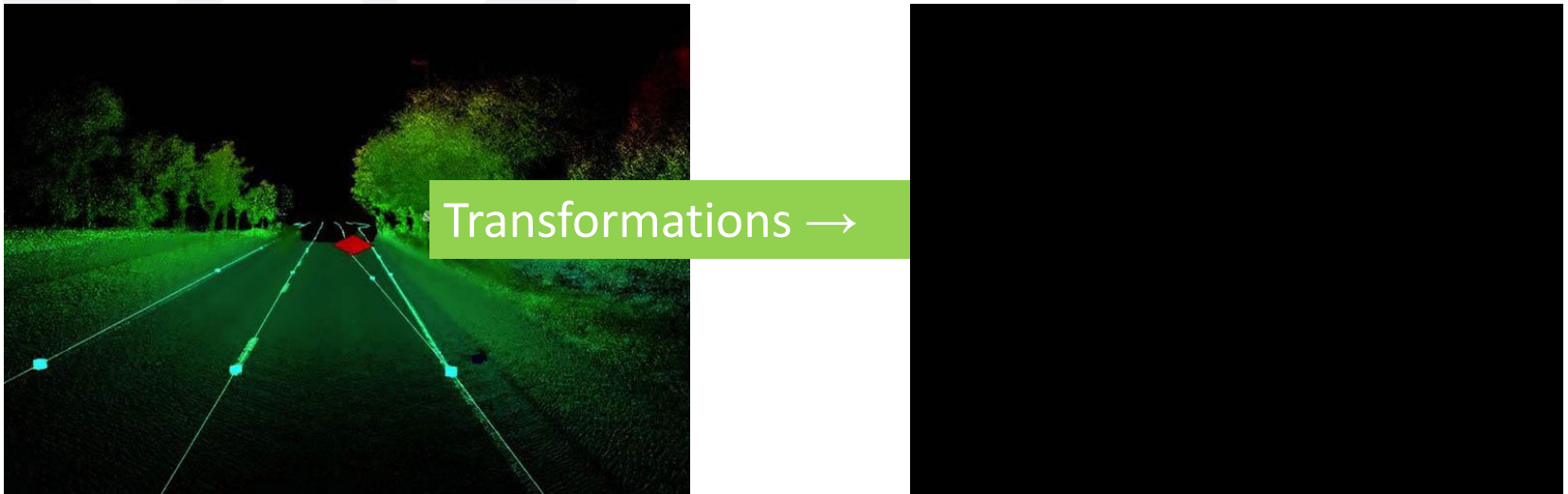
Map frame – where are you with respect to the map – coordinates from origin

Transformations and Frames: Sensor Frame

The Sensor frame – how does the world look from the sensor

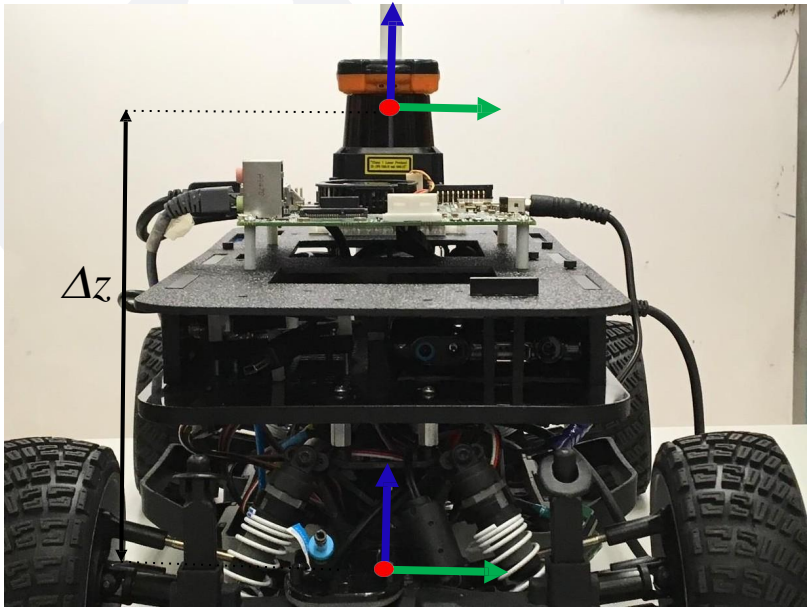
Does this tell you anything about where obstacles are in the map?
Does this tell you anything about where we are in the map?

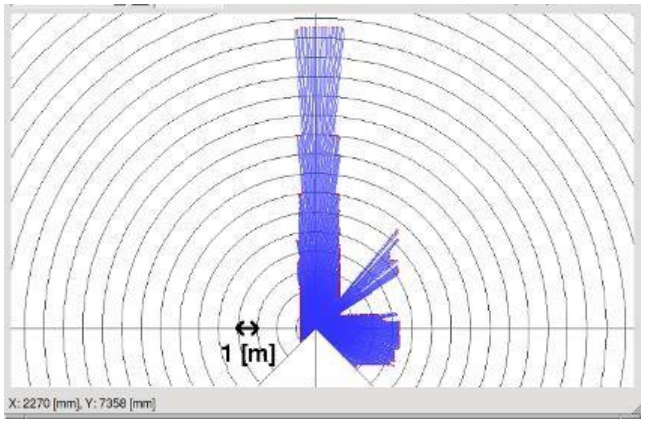
We must link frames together



Transformations and Frames

The frame of reference in which the measurement is taken

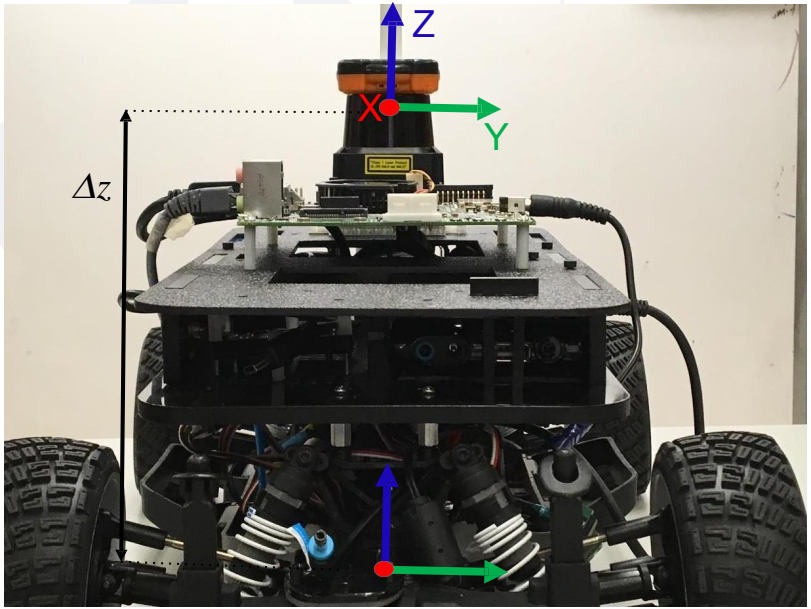


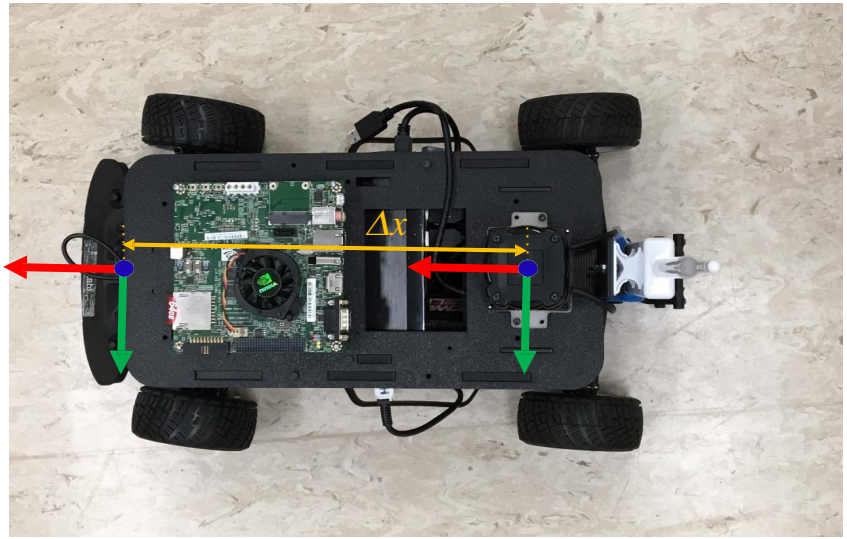


Distance measurements returned by LIDAR

Transformations and Frames

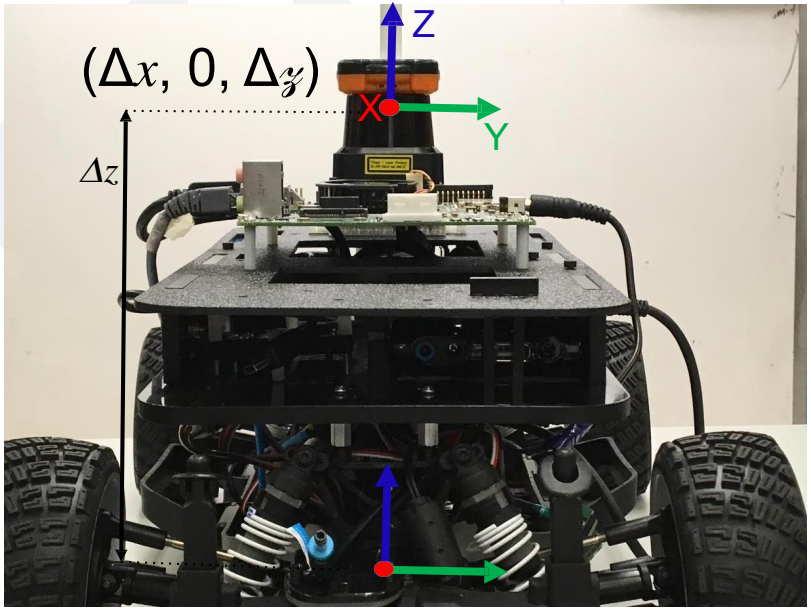
The scan values from the LIDAR will not tell us how far the obstacles are. We must take care of the offsets Δ_y and Δx from the middle of the car



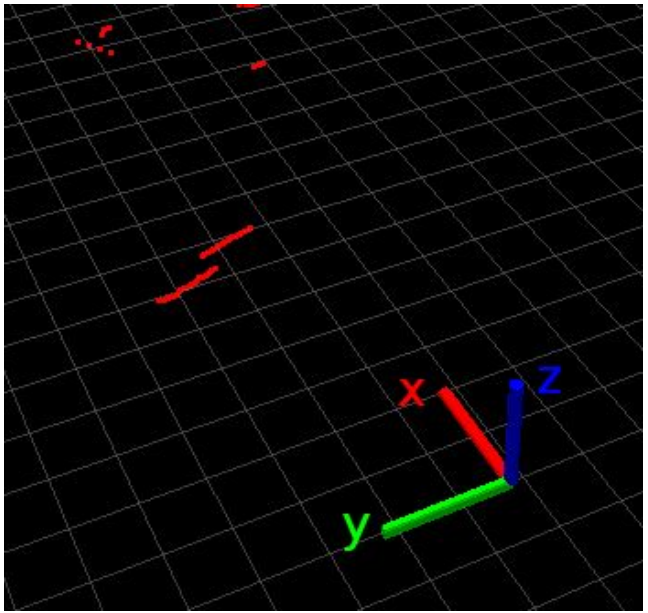


Transformations and Frames

The frame of reference in which the measurement is taken

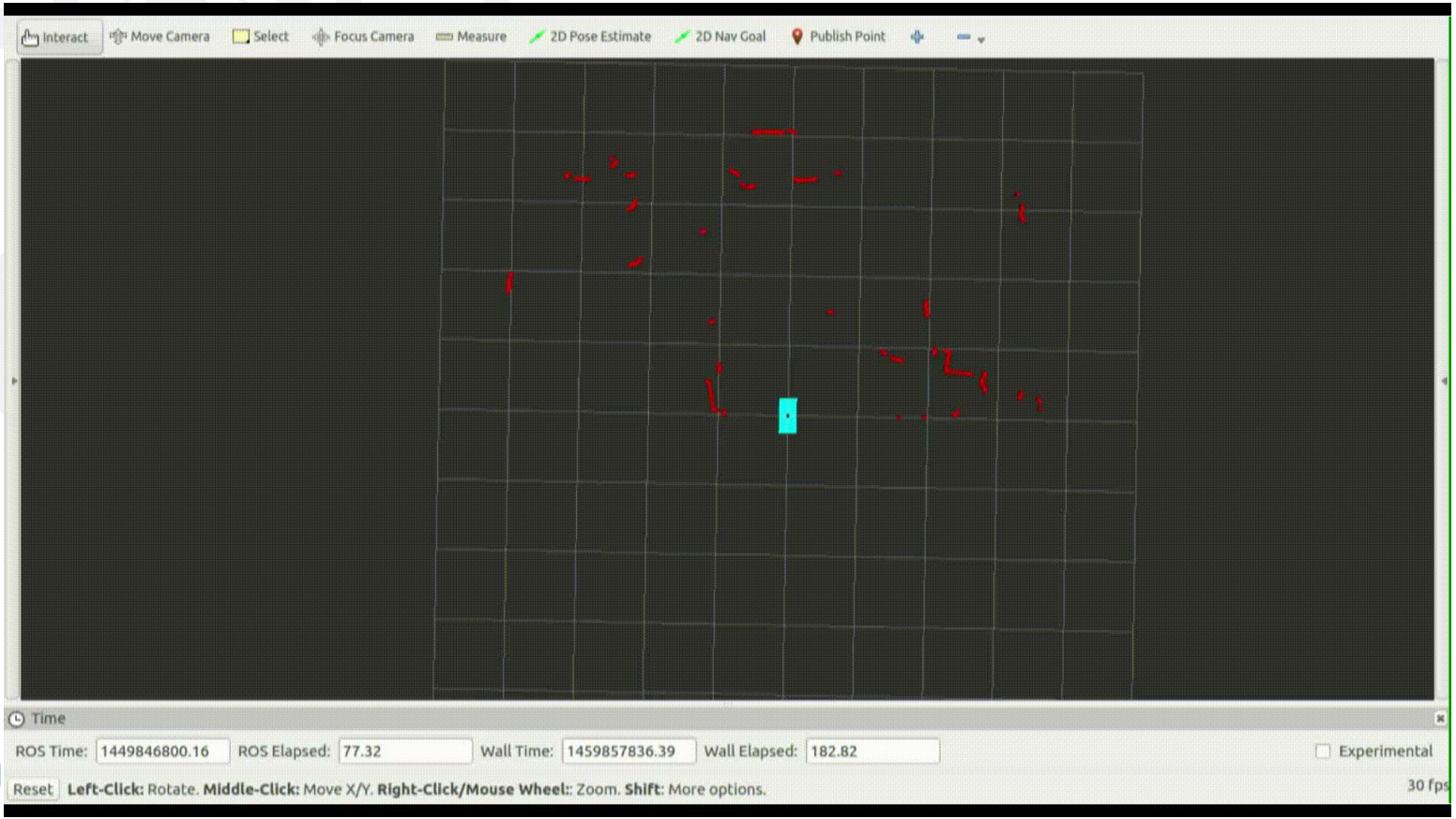


(0,0,0)

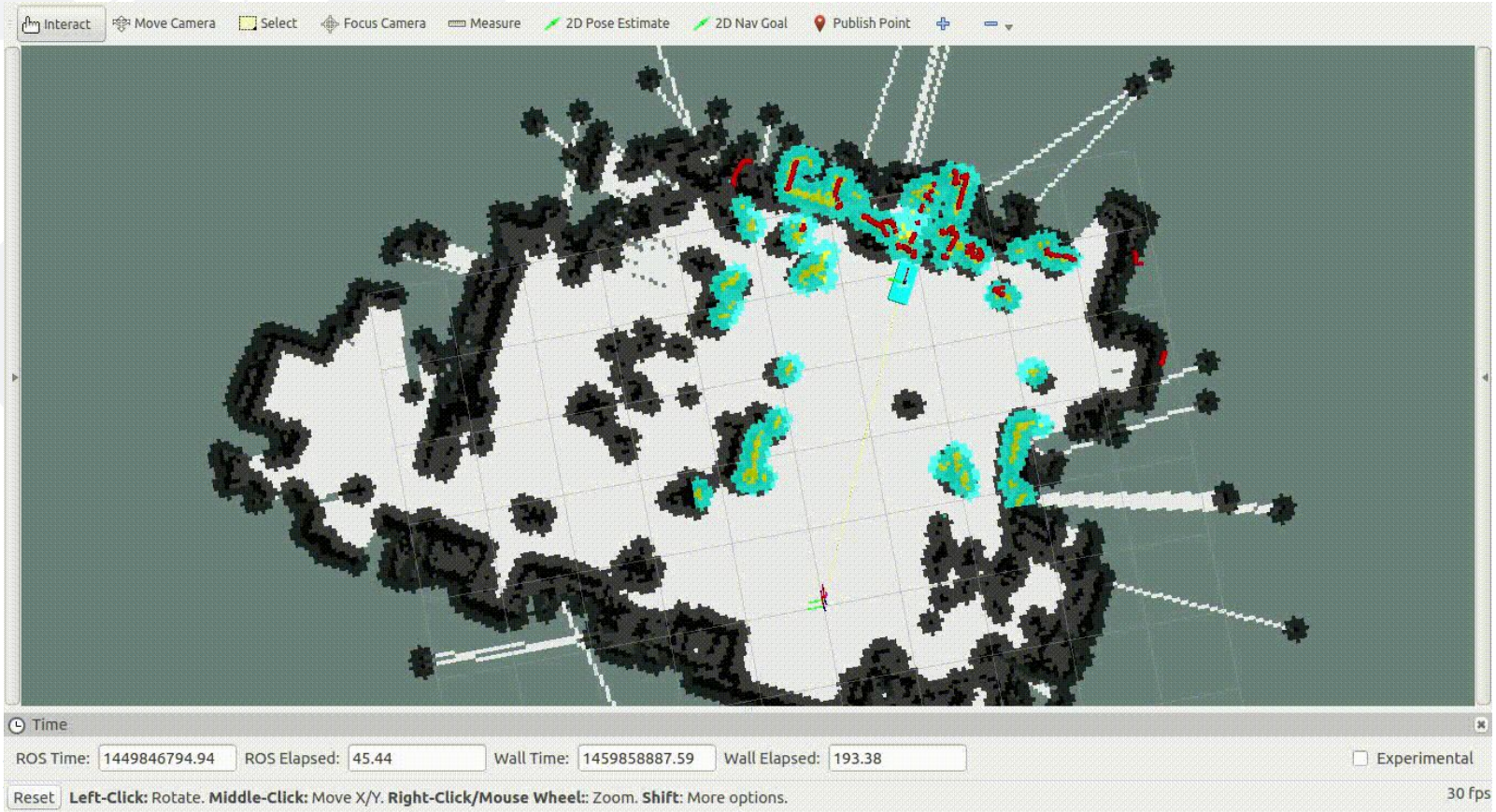


Distance measurements returned by LIDAR

A world without frames and transformations

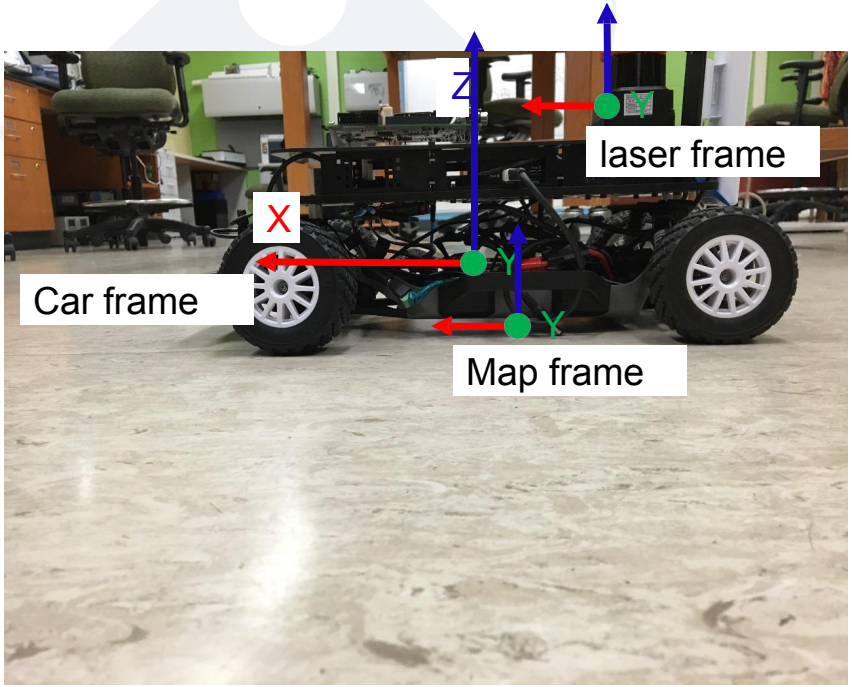


The actual motion of the car



Transformations and Frames

To convert measurements from one frame to another we need to perform transformations

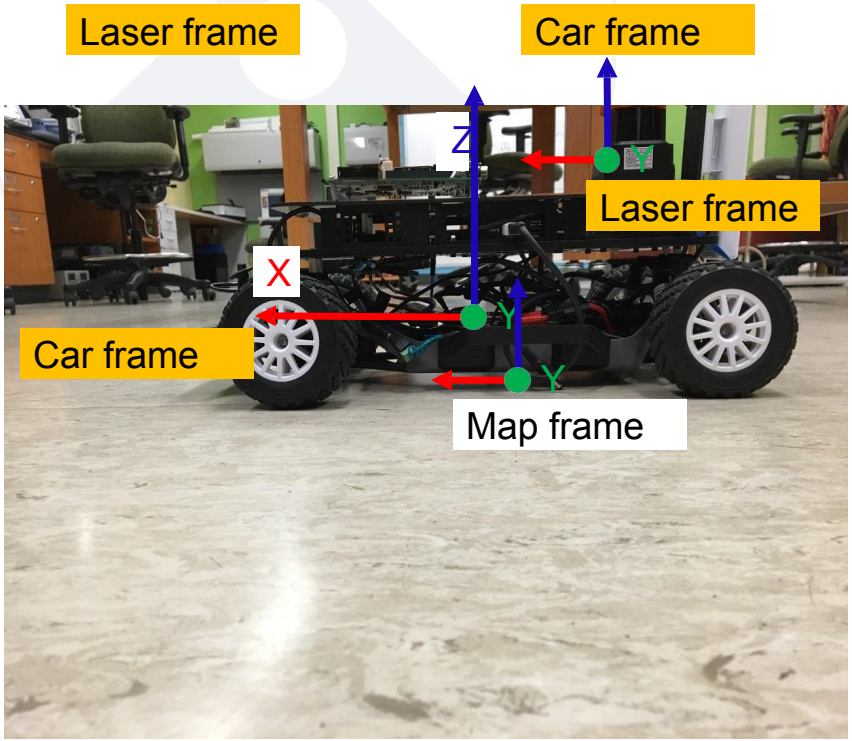


Between frames we need transformations that convert measurements from one frame to another

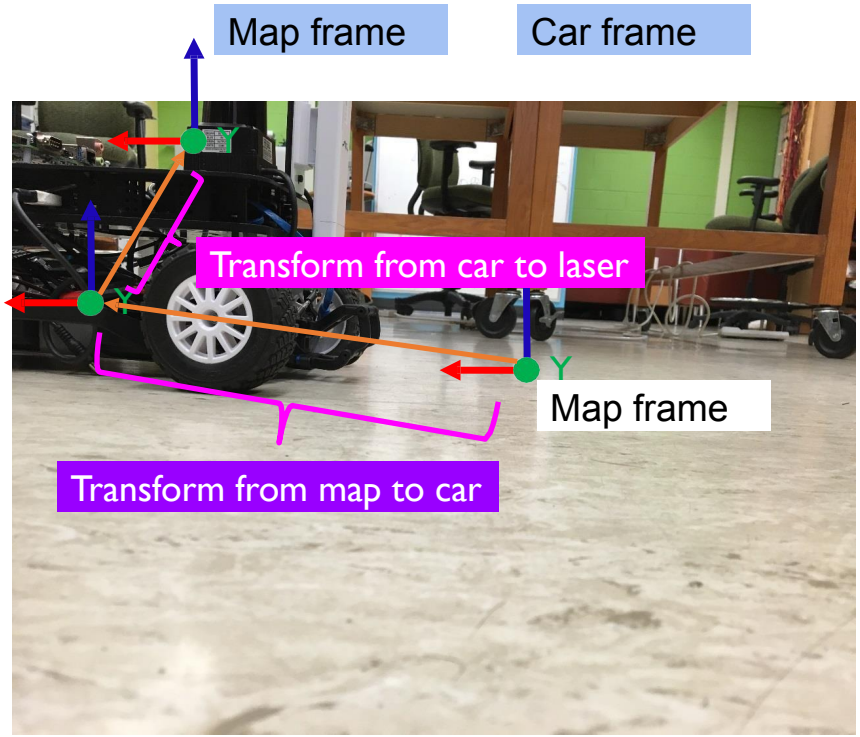
Transformations and Frames

To convert measurements from one frame to another we need to perform transformations

Static Transformations



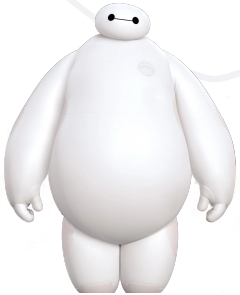
Dynamic Transformations



Rigid Body Transforms: Our Car is a Rigid Body

What's with it being Rigid?

Play-Doh: Not a rigid body



Baymax: Obviously, not a rigid body

Fixed structure: The distance between any two given points of a **rigid body** remains constant in time regardless of external forces exerted on it.

Rigid Body Transforms

- *Deformations* experienced by a rigid body are small relative to its motion.
- *Translation, rectilinear and curvilinear*: The motion of the body is completely specified by the motion of any point in the body. All points of the body have the same velocity and same acceleration.
- *Rotation about a fixed axis*: All particles move in circular paths about the axis of rotation. The motion of the body is completely determined by the angular velocity of the rotation.

Summary: why do we need transforms between frames?

1. Data is usually provided in the most convenient frame to the data source
2. We want to localize ourselves on a map
3. If an obstacle is detected in the *laser* frame, maybe we want to know where it is in the world, i.e. in the *map* frame
4. We need to know the transforms between frames of components on the car for more accurate actuation
5. If we had two disconnected maps (e.g. submaps), we might want to know their location w.r.t. Each other within a global world frame

Reference Frames on F1TENTH

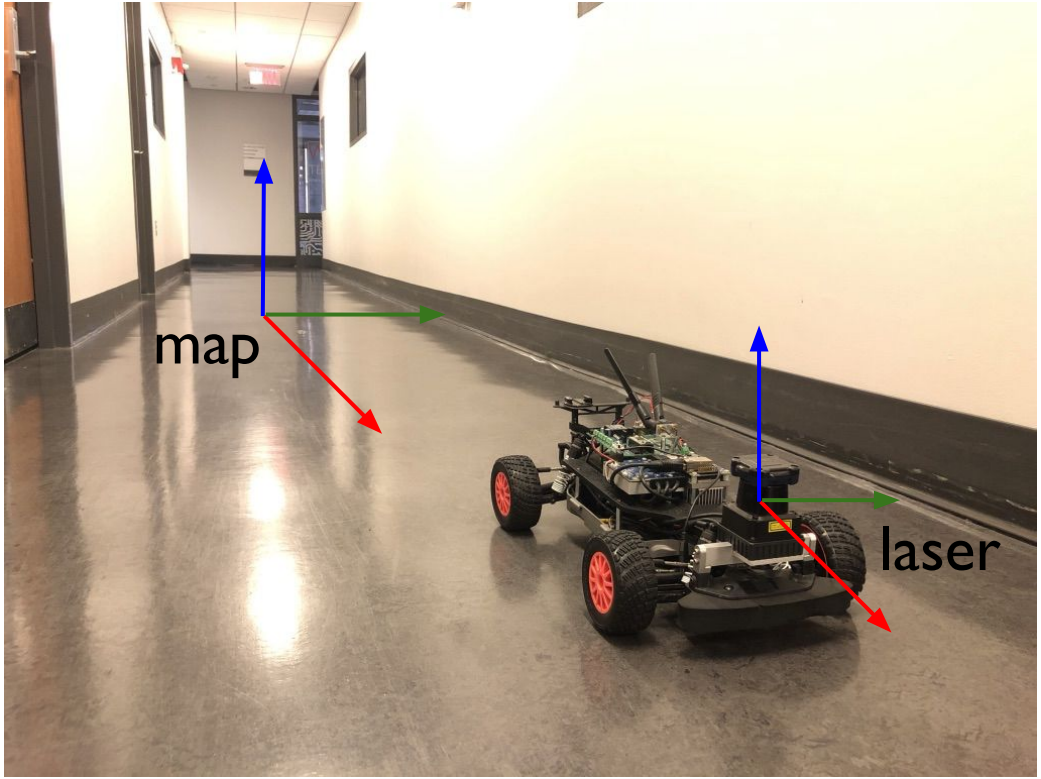
Multiple Reference Frames on FITENTH

Why?

We need to know where the robot is in the world.

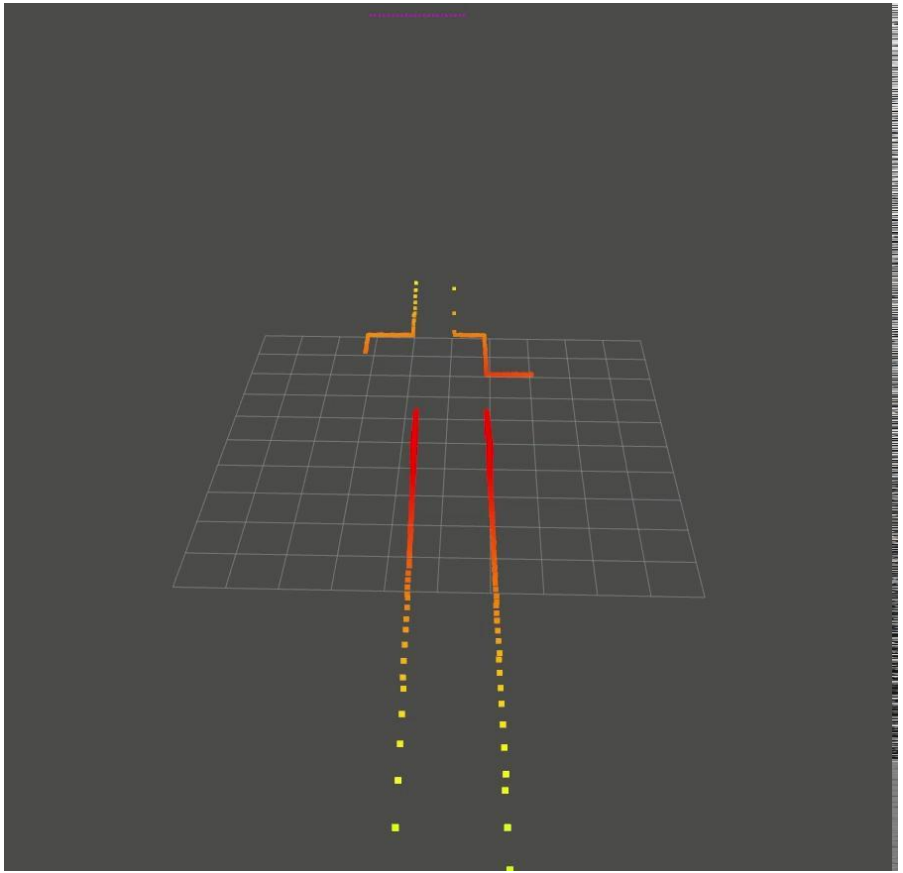
We need to know the relationship between the measurement frame and the actuation frame.

Also, if the lidar sees an obstacle, we might want to know the obstacle's position in the global frame.



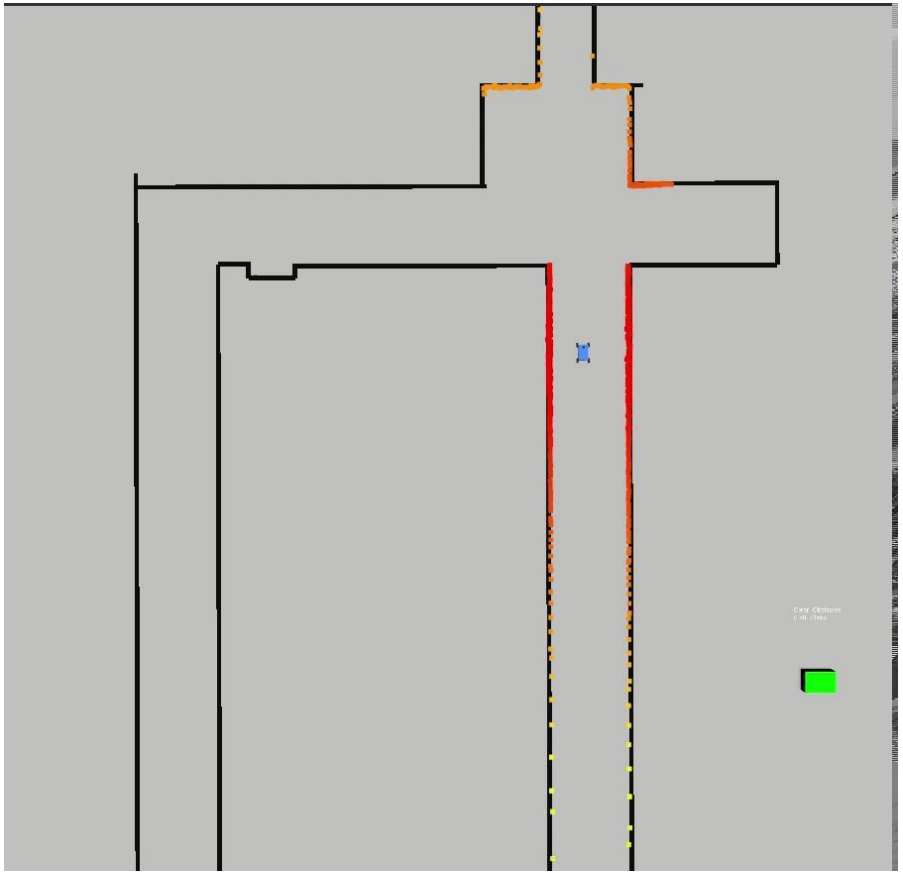
Measurement Frame

Measurement in **laser frame**: can see some information about the environment, and some information on the positional relationship between the robot and the world.



World Frame

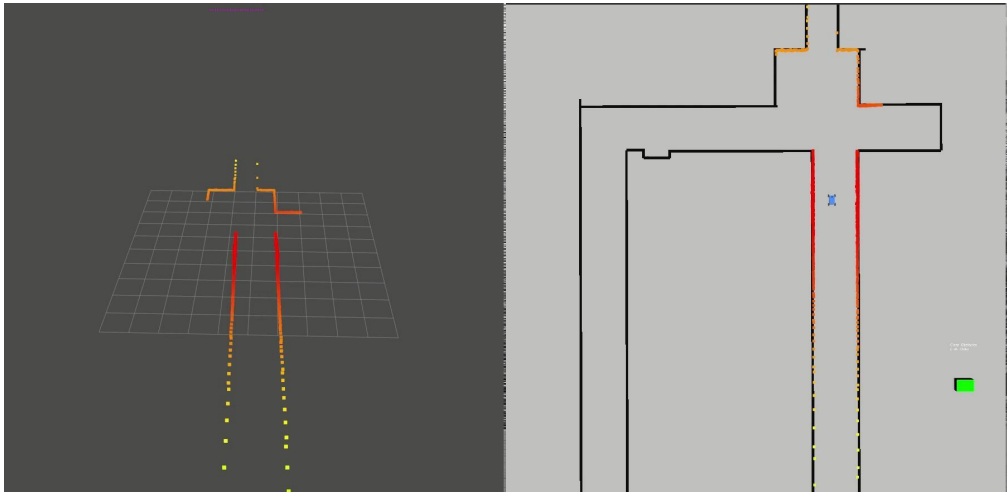
Measurement in **map frame**: the range measurement we see fully makes sense. The observation matches up with the environment. We know where the robot is in the environment.



Multiple Reference Frames

- The world makes more sense if we put the laser scans in the global map frame instead of the **laser frame.**

- More information available when planning in a **global frame** instead of extracting information in the observation frame.

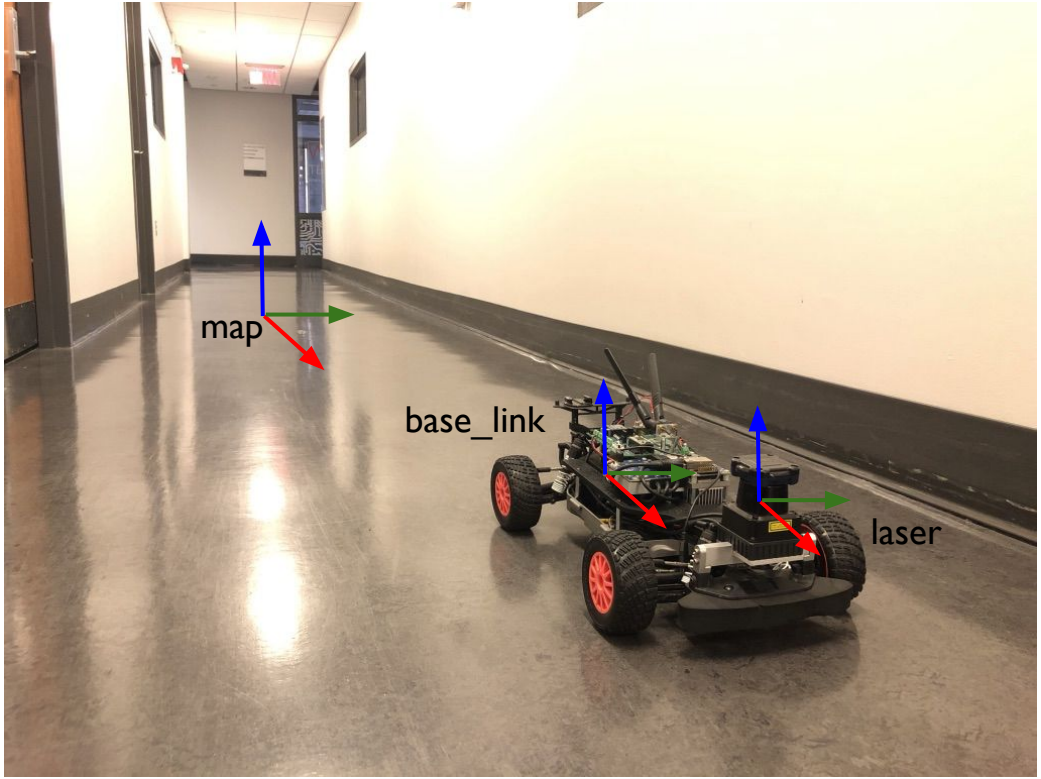


Laser frame

Map frame

Frames we will use

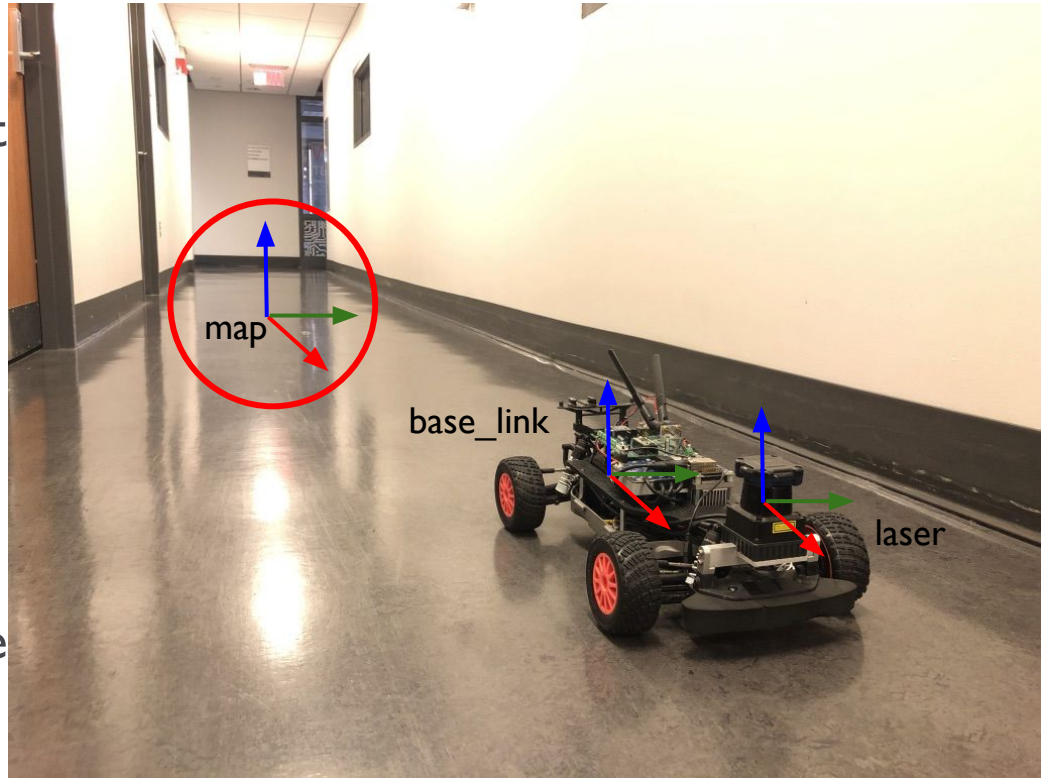
1. map
2. base_link
3. laser



P.S. In ROS, RGB → XYZ

Frames we will use: Environment

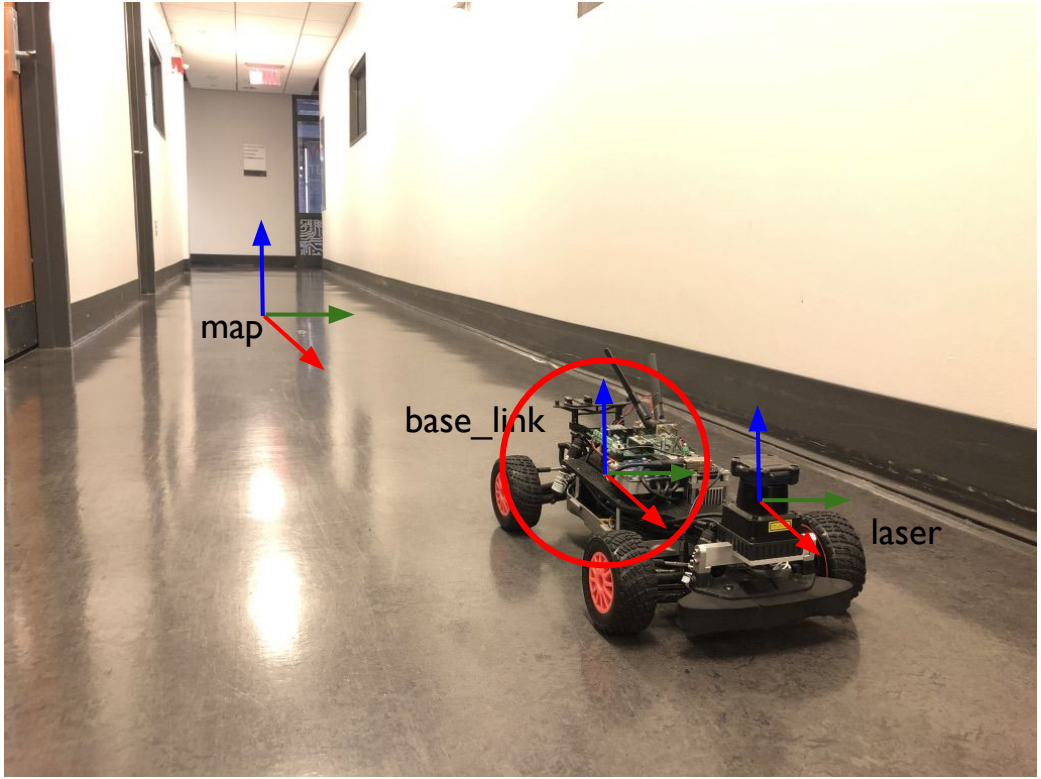
1. **Map - Global Frame**
   - Represents the environment around the robot
   - Map frame origin can be set arbitrarily when we make the map
   - You have to set an origin when you're first making the map and keep it consistent throughout your navigation task



Frames we will use: Actuation

2. base_link

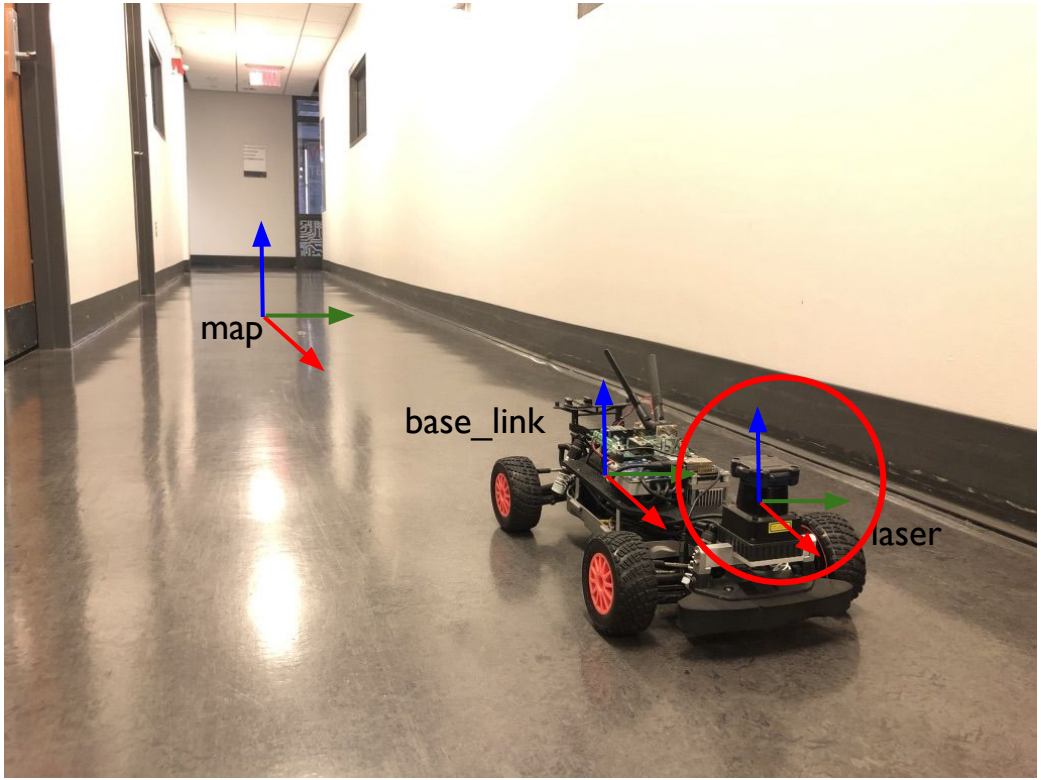
- Defined as the center of the car's rear axle
- Moves with the car relative to the map frame



Frames we will use: Measurement

3. laser

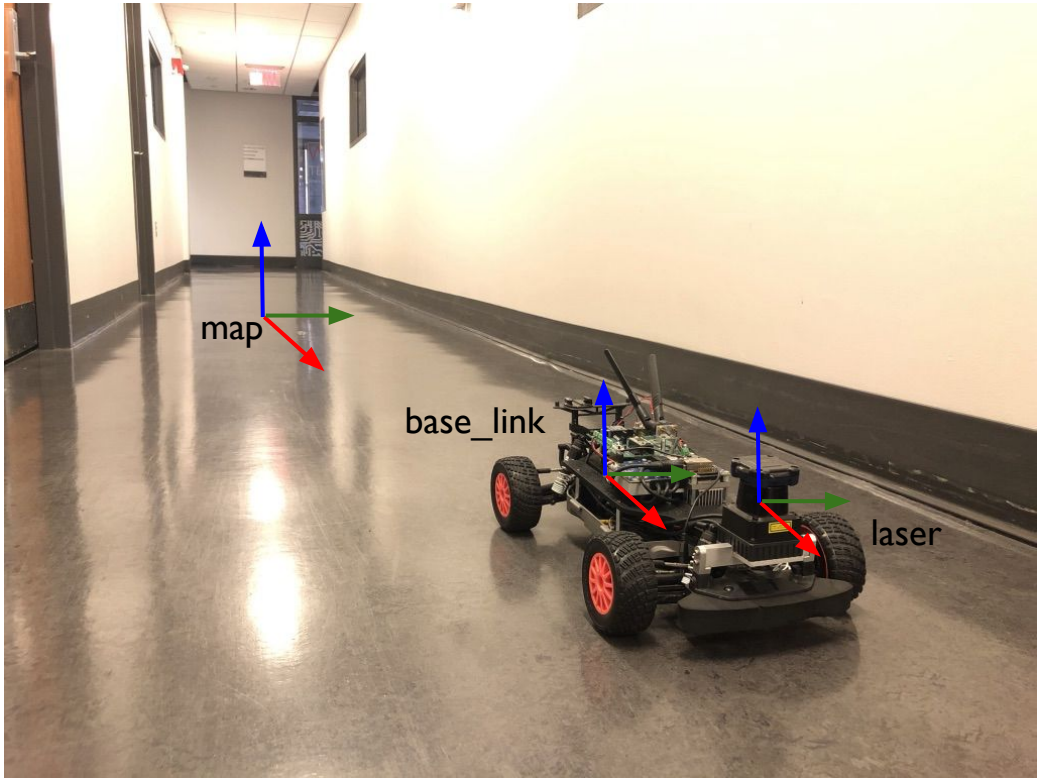
- The frame in which the laser scan measurement is taken
- Also moves with the car relative to the map frame
- The car itself is a rigid body so we can assume that the transformation between laser and base link is static



Odometry

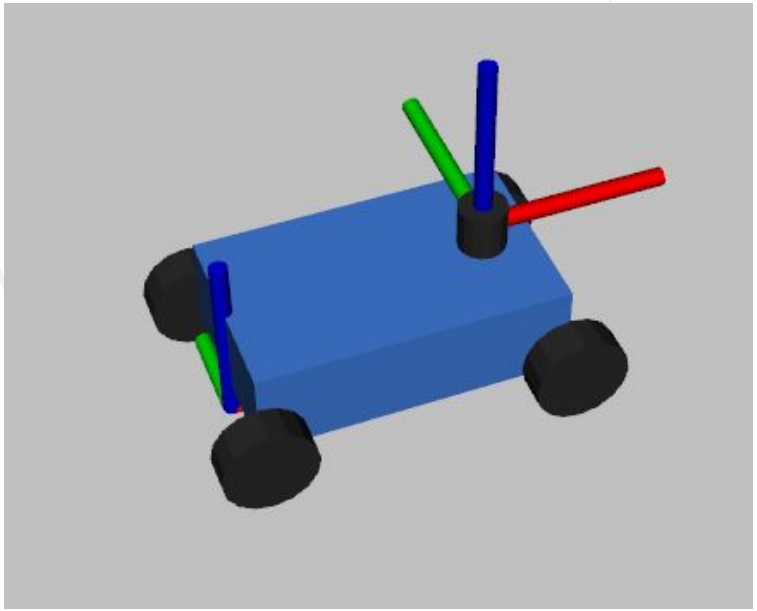
odom

- Not exactly a frame
- History describing the relationship between two frames - base-link and map
- The Odometry message in ROS provides a pose estimate of the car. The pose is relative to the odometry frame.



Poses & Transformations

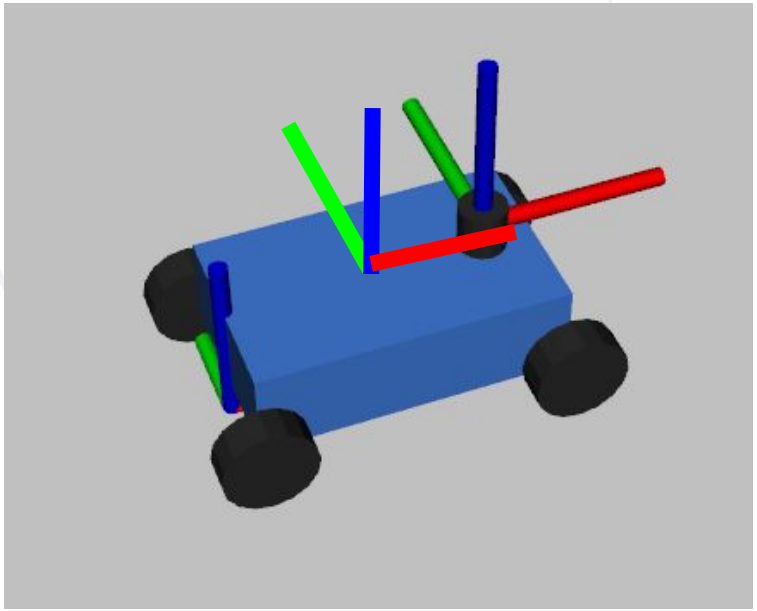
An Example



- In the safety lab, when you define only one Time to Collision (TTC) threshold for emergency braking, the distance between the wall and the car after the car is stopped is different depending on the traveling direction.

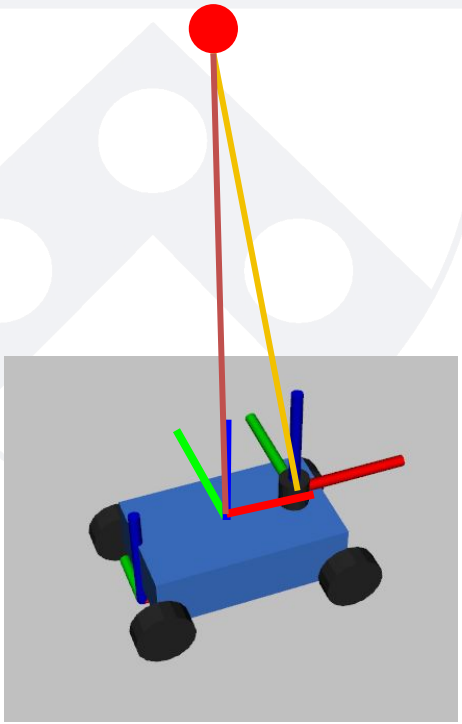
- Caused by range measurements taken in the laser frame, which is toward the front of the vehicle.

An Example



Could be remedied by adding a frame in the center of the chassis and calculated the TTC from there.

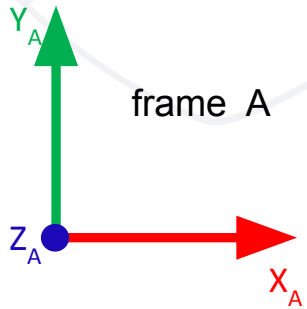
An Example



How do we transform the measurements in the original frame into the new frame that we just added?

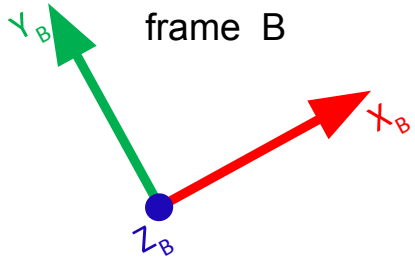
Rigid Body Transforms

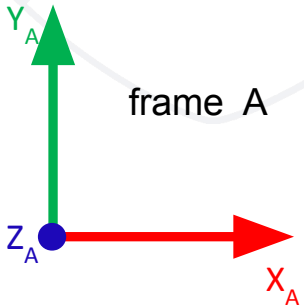
Lets formalize the problem statement at hand
Let's say we have a frame A



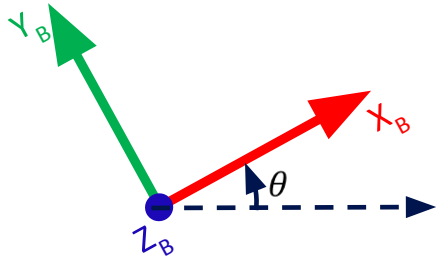
Rigid Body Transforms

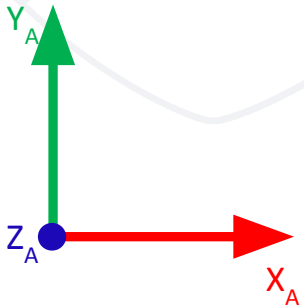
And we have a frame B





Rigid Body Transforms

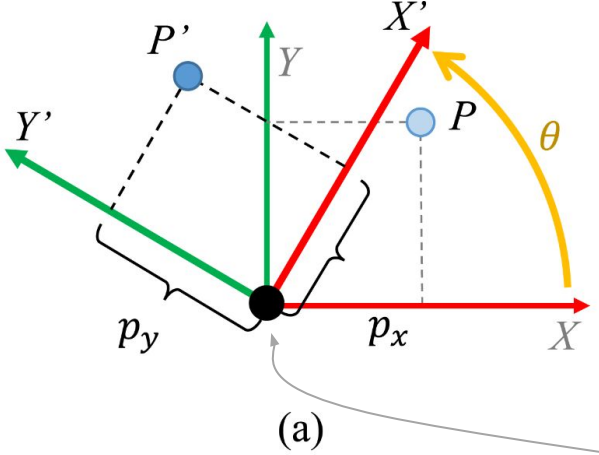




Frame B is rotated by some theta w.r.t the X-axis of Frame A

We want to express the coordinates of Frame B in terms of Frame A

Rotating a point in 2D

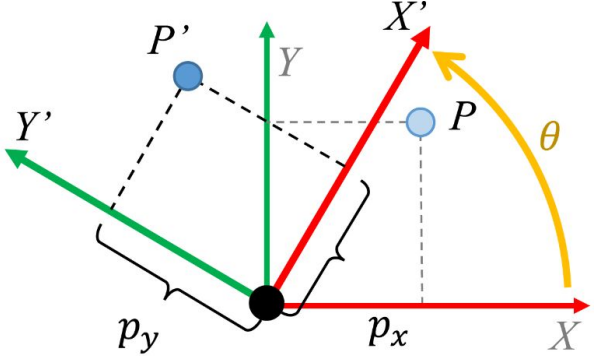


Let's start by assuming that both the frames have the same origin.
So we deal with rotation only for now, no translation.

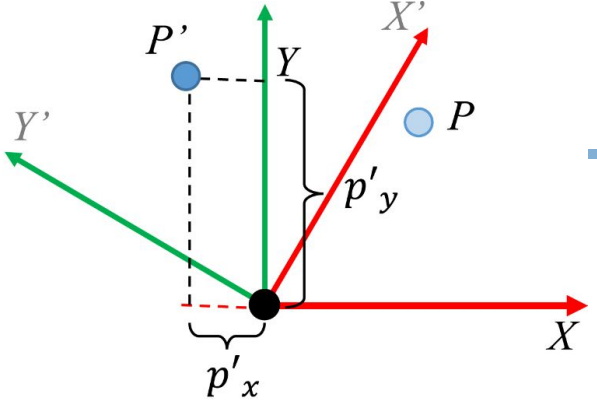
Rotations about the origin by angle θ can be defined as linear transformations.

Consider two reference frames with a common origin O,
the pre-rotation axes X and Y, and the post-rotation axes X' and Y'.

Rotating a point in 2D



(a)



(b)

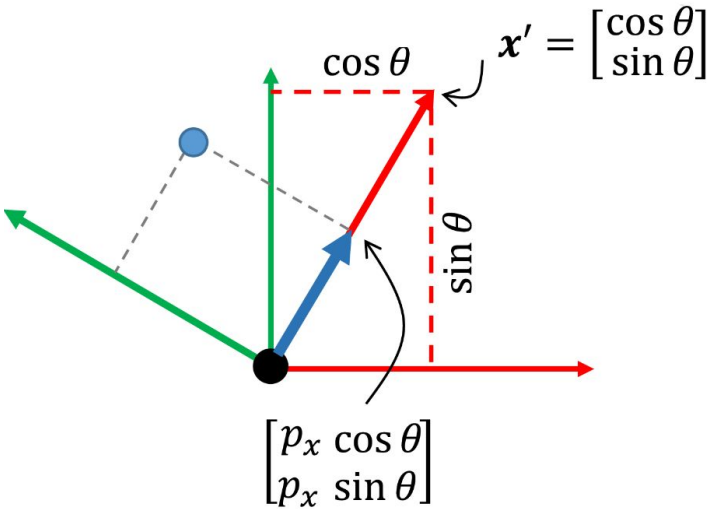
$$\mathbf{p}' = p_x\mathbf{x}' + p_y\mathbf{y}' = \begin{bmatrix} p_x\cos\theta - p_y\sin\theta \\ p_x\sin\theta + p_y\cos\theta \end{bmatrix}$$

Now consider that along with the coordinate frames, a point P was rotated to P'.
We will derive how to determine its new coordinates relative to the original reference frame.

Notice that P' still has coordinates (px,py) relative to the post-rotation frame X', Y', since distances do not shrink or grow when objects are rotated.

Rotating a point in 2D

Specifically, P' is obtained by walking px units from the origin in the direction of X', and then py units in the direction of Y'.
Hence, to determine its coordinates in the original reference frame, we can use the fact that the coordinates of X' and Y' are known



(c)

$$\mathbf{p}' = p_x\mathbf{x}' + p_y\mathbf{y}' = \begin{bmatrix} p_x\cos\theta - p_y\sin\theta \\ p_x\sin\theta + p_y\cos\theta \end{bmatrix}$$

$$\mathbf{p}' = R(\theta)\mathbf{p}$$

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

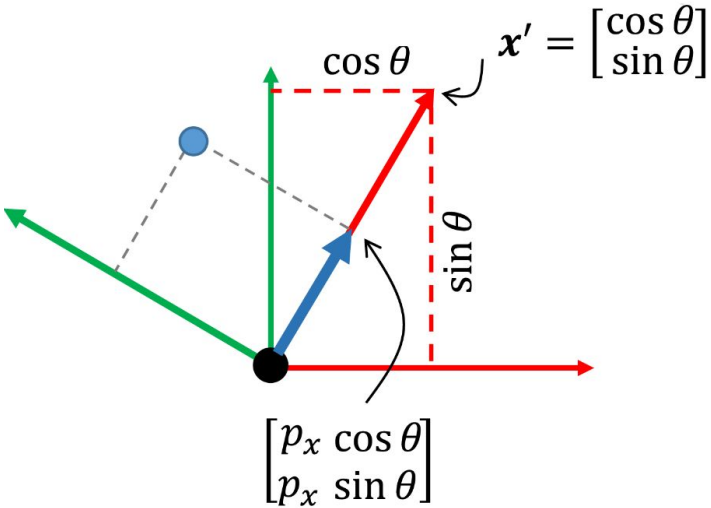
Rotating a point in 2D

Similarly we calculate **y**' by walking py distance along Y' axis

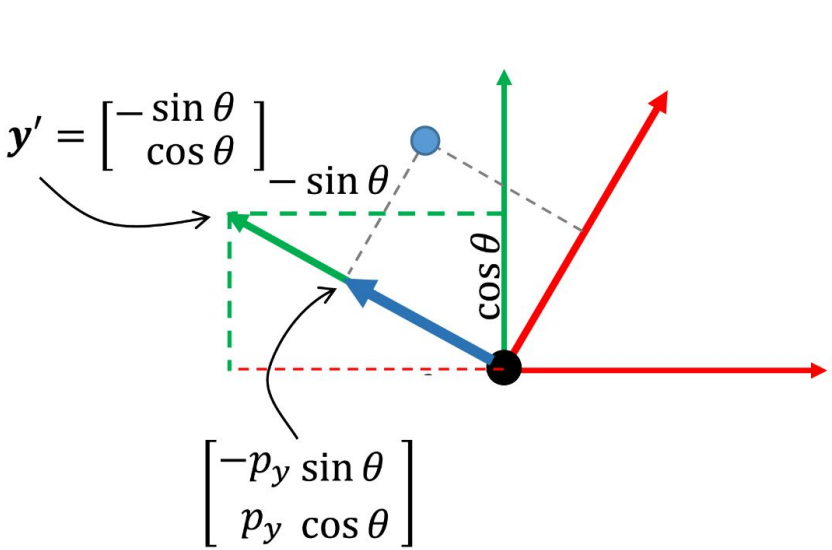
Using a little trigonometry, we shall see that X' has coordinates

x' =(cos θ,sin θ)

y'=(−sin θ,cos θ)



(c)

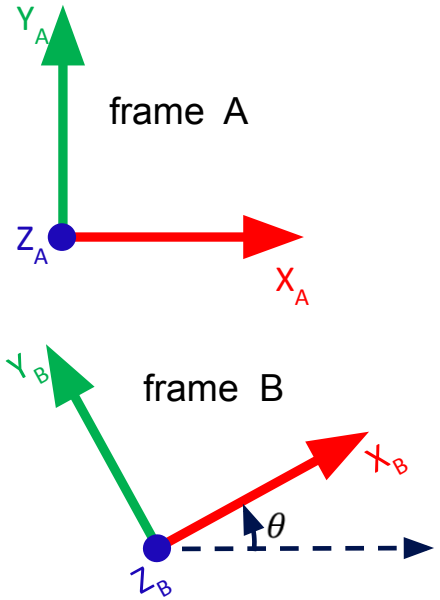


(d)

Rigid Body Transforms: Rotation Matrices

Special type of matrices called Rotation matrices

$$\widehat{X_B} = R_{11}\widehat{X_A} + R_{21}\widehat{Y_A} +$$
$$\widehat{Y_B} = R_{12}\widehat{X_A} + R_{22}\widehat{Y_A} +$$
$$\widehat{Z_B} = R_{13}\widehat{X_A} + R_{23}\widehat{Y_A} +$$



Let's look at the rotation in the form of Rotation Matrices.
We need to represent the orientation of axes of the frame B in frame A

XBCAP, YBCAP and ZBCAP are the unit vectors of the respective axes, represented using the unit vectors of the axes of frame A

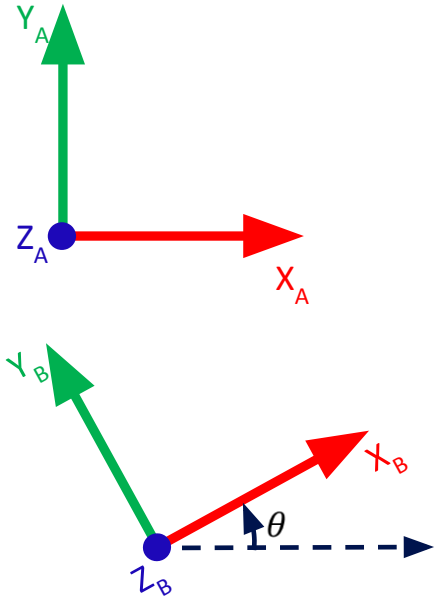
Rigid Body Transforms: Rotation Matrices

We use these expressions to form a rotation matrix R .
This Rotation matrix gives us the orientation of points in frame B w.r.t frame A

$$\widehat{X_B} = R_{11}\widehat{X_A} + R_{21}\widehat{Y_A} + R_{31}\widehat{Z_A}$$
$$\widehat{Y_B} = R_{12}\widehat{X_A} + R_{22}\widehat{Y_A} + R_{32}\widehat{Z_A}$$
$$\widehat{Z_B} = R_{13}\widehat{X_A} + R_{23}\widehat{Y_A} + R_{33}\widehat{Z_A}$$

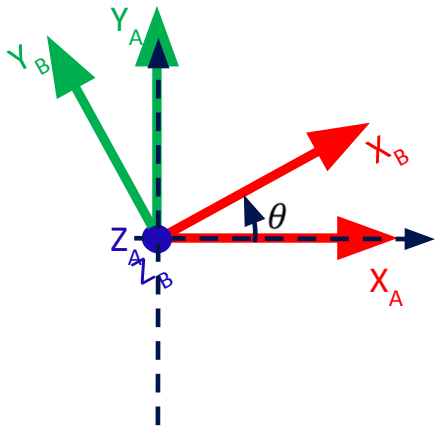
$${}^{A}R_{B} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

Takes points in frame B and represents their orientation in frame A



Rigid Body Transforms: Rotation Matrices

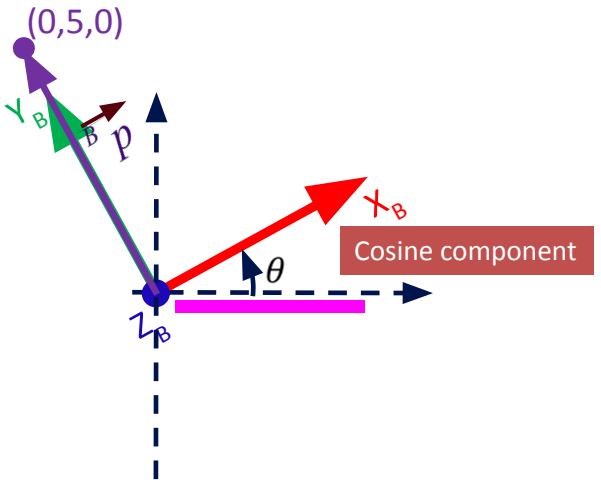
$$\widehat{X_B} = R_{11}\widehat{X_A} + R_{21}\widehat{Y_A} +$$
$$\widehat{Y_B} = R_{12}\widehat{X_A} + R_{22}\widehat{Y_A} +$$
$$\widehat{Z_B} = R_{13}\widehat{X_A} + R_{23}\widehat{Y_A} +$$



Rigid Body Transforms: Rotation Matrices

Now let's take a point P expressed in frame B. This point is at (0,5,0)
Let's first represent XBCAP w.r.t to the axes of Frame A.
XB has the cosine component of XA,

$$\widehat{X_B} = R_{11}\widehat{X_A} + R_{21}\widehat{Y_A} +$$
$$\widehat{Y_B} = R_{12}\widehat{X_A} + R_{22}\widehat{Y_A} +$$
$$\widehat{Z_B} = R_{13}\widehat{X_A} + R_{23}\widehat{Y_A} +$$



$$\widehat{X_B} = \cos(\theta) \times \widehat{X_A}$$

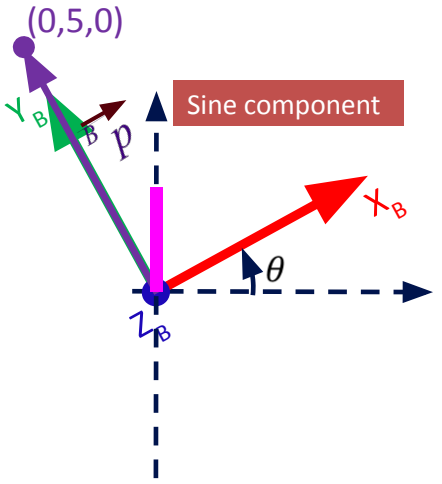
$$R_{11}$$

Penn Engineering

Rigid Body Transforms: Rotation Matrices

And the sine component of Ya

$$\widehat{X_B} = R_{11}\widehat{X_A} + R_{21}\widehat{Y_A} +$$
$$\widehat{Y_B} = R_{12}\widehat{X_A} + R_{22}\widehat{Y_A} +$$
$$\widehat{Z_B} = R_{13}\widehat{X_A} + R_{23}\widehat{Y_A} +$$



$$\widehat{X_B} = \cos(\theta) \times \widehat{X_A} + \sin(\theta) \times \widehat{Y_A}$$

$\cos(\theta) = R_{11}$ $\quad$ $\sin(\theta) = R_{21}$

Rigid Body Transforms: Rotation Matrices

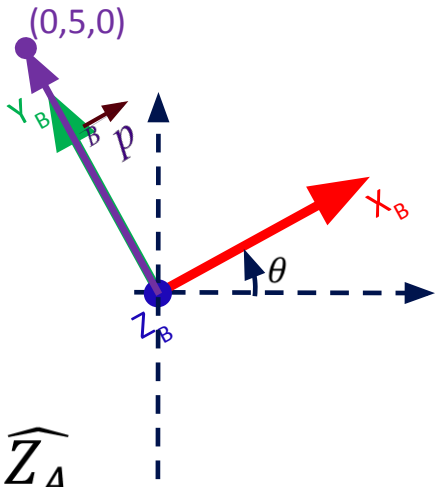
And no component of Za

What we have effectively done is found out the first column of the Rotation matrix in question

$$\widehat{X_B} = R_{11}\widehat{X_A} + R_{21}\widehat{Y_A} +$$

$$\widehat{Y_B} = R_{12}\widehat{X_A} + R_{22}\widehat{Y_A} +$$

$$\widehat{Z_B} = R_{13}\widehat{X_A} + R_{23}\widehat{Y_A} +$$



$$\widehat{X_B} = \underbrace{\cos(\theta)}_{R_{11}} \times \widehat{X_A} + \underbrace{\sin(\theta)}_{R_{21}} \times \widehat{Y_A} + \underbrace{0}_{R_{31}} \times \widehat{Z_A}$$

Rigid Body Transforms: Rotation Matrices

Our rotation matrix now looks like this where we filled the first column using the equation
Notice that the axes are unit vectors

$$
{}^{A}R_{B} = \begin{bmatrix} \boxed{\begin{matrix}\mathrm{Cos}(\theta)\\ \mathrm{Sin}(\theta)\\ 0\end{matrix}} & \boxed{\begin{matrix}R_{12}\\ R_{22}\\ R_{32}\end{matrix}} & \boxed{\begin{matrix}R_{13}\\ R_{23}\\ R_{33}\end{matrix}} \end{bmatrix} \begin{matrix}\widehat{X_A}\\ \widehat{Y_A}\\ \widehat{Z_A}\end{matrix}
$$

$\widehat{X_B}$ $\widehat{Y_B}$ $\widehat{Z_B}$

$$
{}^{A}R_{B} = \begin{bmatrix} C_\theta & -S_\theta & 0 \\ S_\theta & C_\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

$$
C_\theta = \mathrm{Cos}(\theta)
$$
$$
S_\theta = \mathrm{Sin}(\theta)
$$

The rows of this matrix signify the axes of frame A
We similarly find for the remaining columns in the rotation matrix.

We have the Rotation Matrix, so now what?

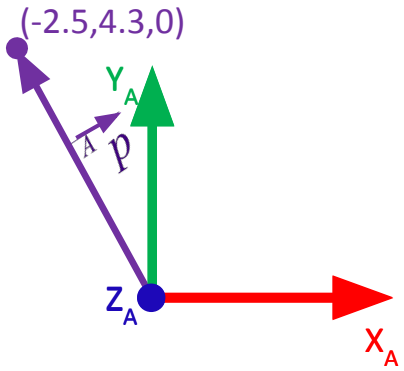
$$^{A}\vec{p} = {}^{A}\vec{R}_{B} \times {}^{B}\vec{p}$$

Known Known

We now have the point P as referenced in frame A

For example

$$\theta = \pi/6 \Rightarrow {}^{A}\vec{p} = (-2.5, 4.3, 0)$$



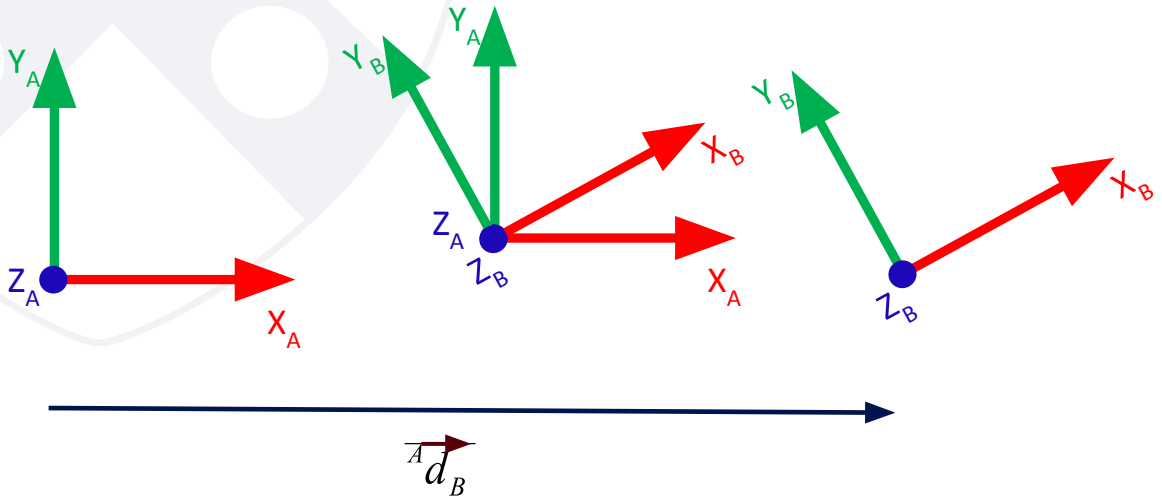
Now that we have R, we can represent the point p in the frame of A
Assume theta = pi/6 as an example
p in the frame of A is the Rotation matrix in the intermediate step

$$R = \begin{bmatrix} 0.86 & -0.5 & 0 \\ 0.5 & 0.86 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Important point to remember

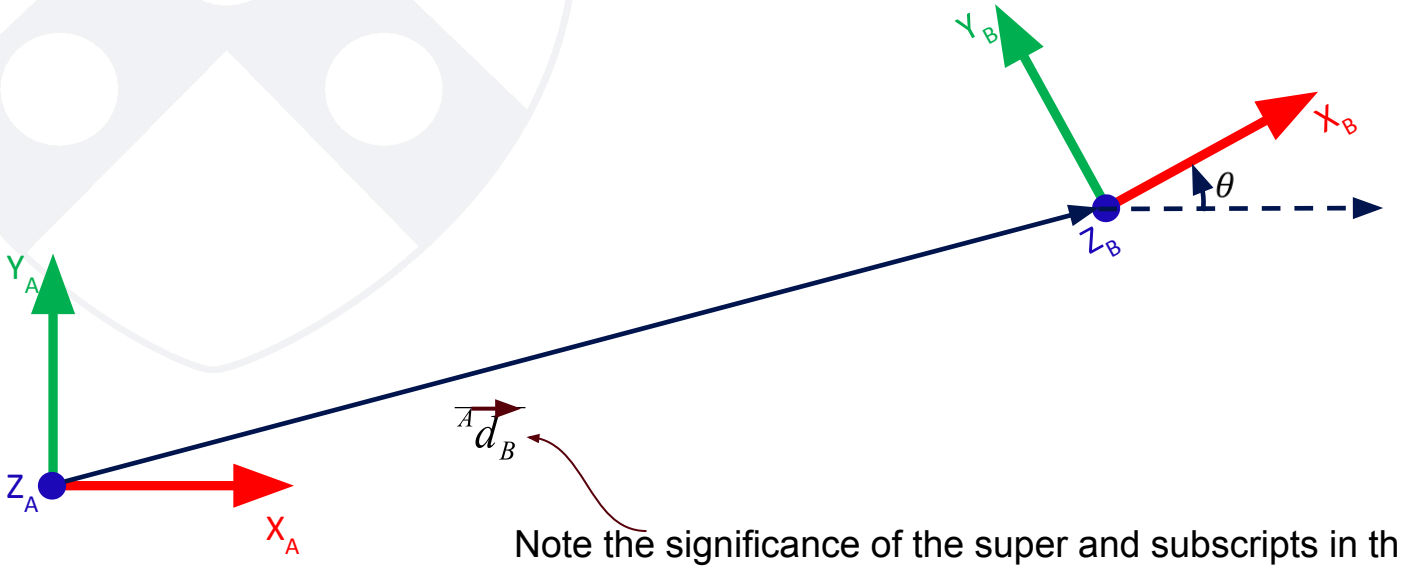
The rotation matrix will take care of perspectives of orientation, what about displacement?

Origins of both the frames are at the same location



Rigid Body Transforms

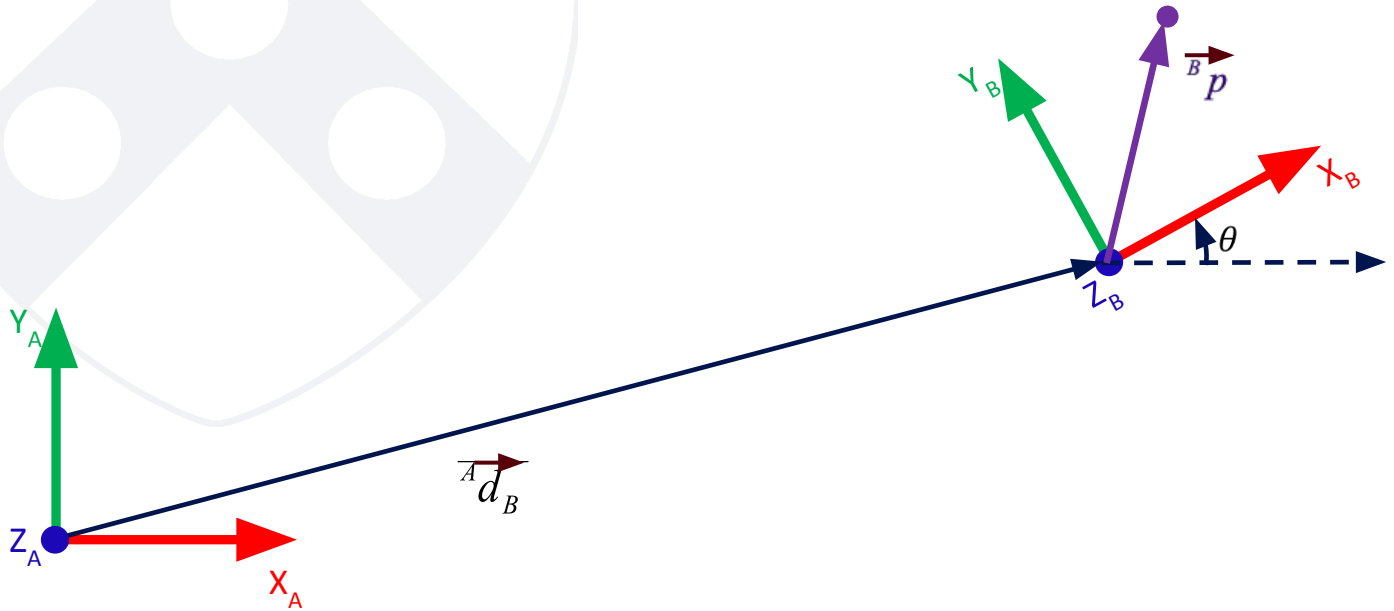
Let frame B be a distance d from the origin of frame A.



Note the significance of the super and subscripts in this notation - distance of frame B w.r.t frame A

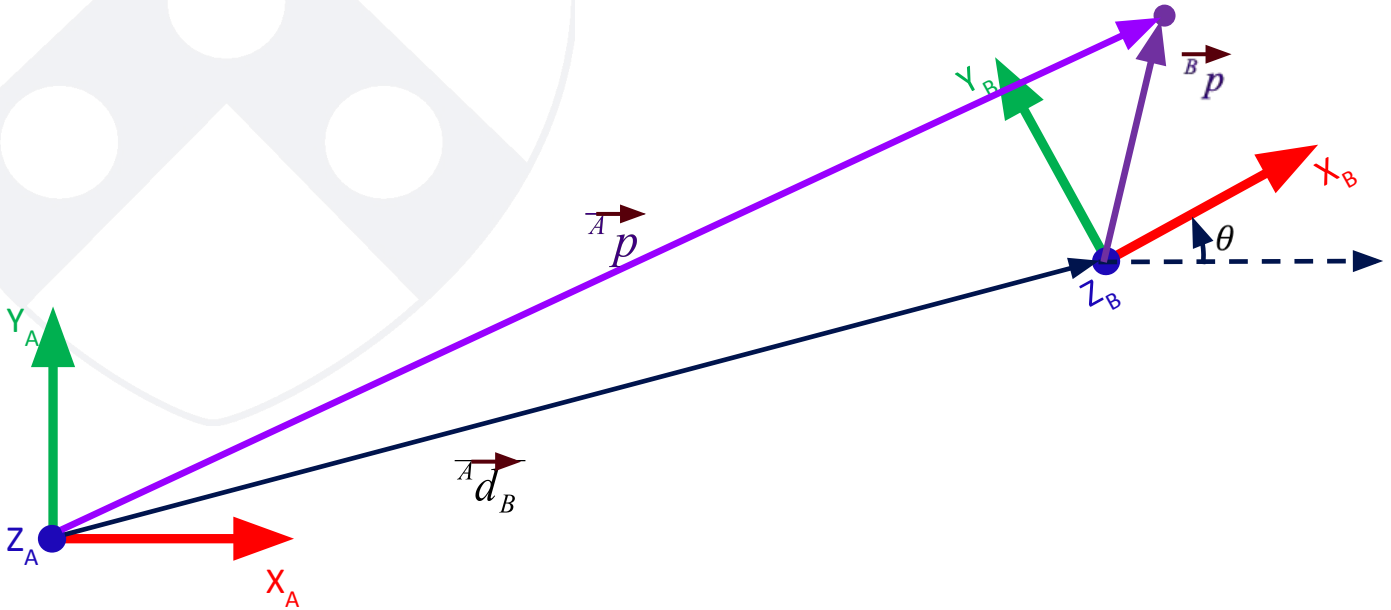
Rigid Body Transforms

Let there be a point P as shown with a known location in frame B



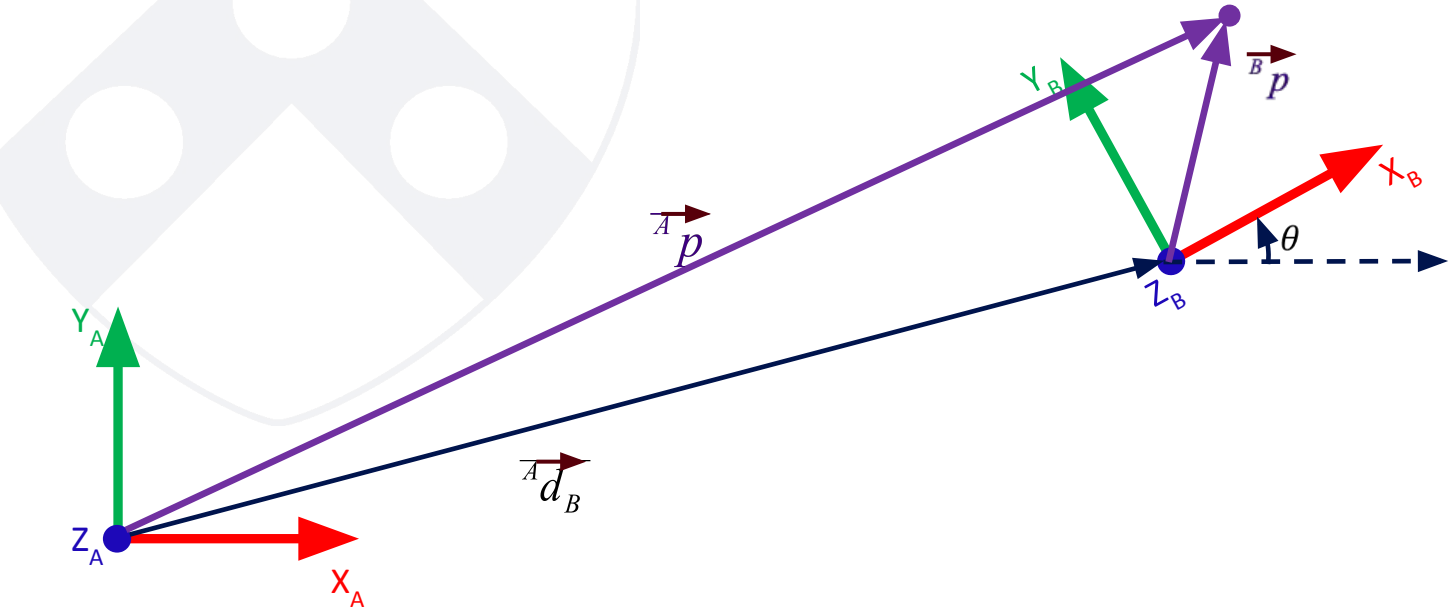
Rigid Body Transforms

The problem statement now is to find the representation of Point P in frame A.



Rigid Body Transforms

What we need is Point **p** with respect to Frame A, given its pose in Frame B

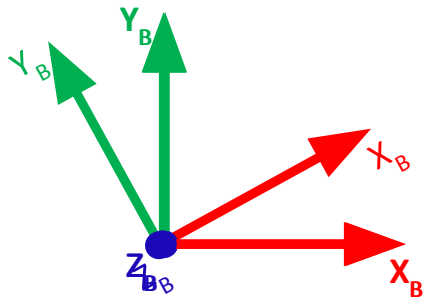


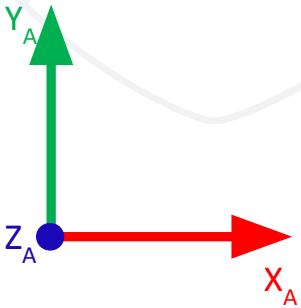
Penn Engineering

Rigid Body Transforms:
And we are back to the Future

With translation, we need to add the displacement between the origins of the frames
The updated formula is as shown.

$$\vec{{}^{A}p} = {}^{A}R_{B} \times {}^{B}p + {}^{A}d_{B}$$





We can now compensate for any rotation or translation between frames

Rigid Body Transforms:
Homogeneous transformation matrix H

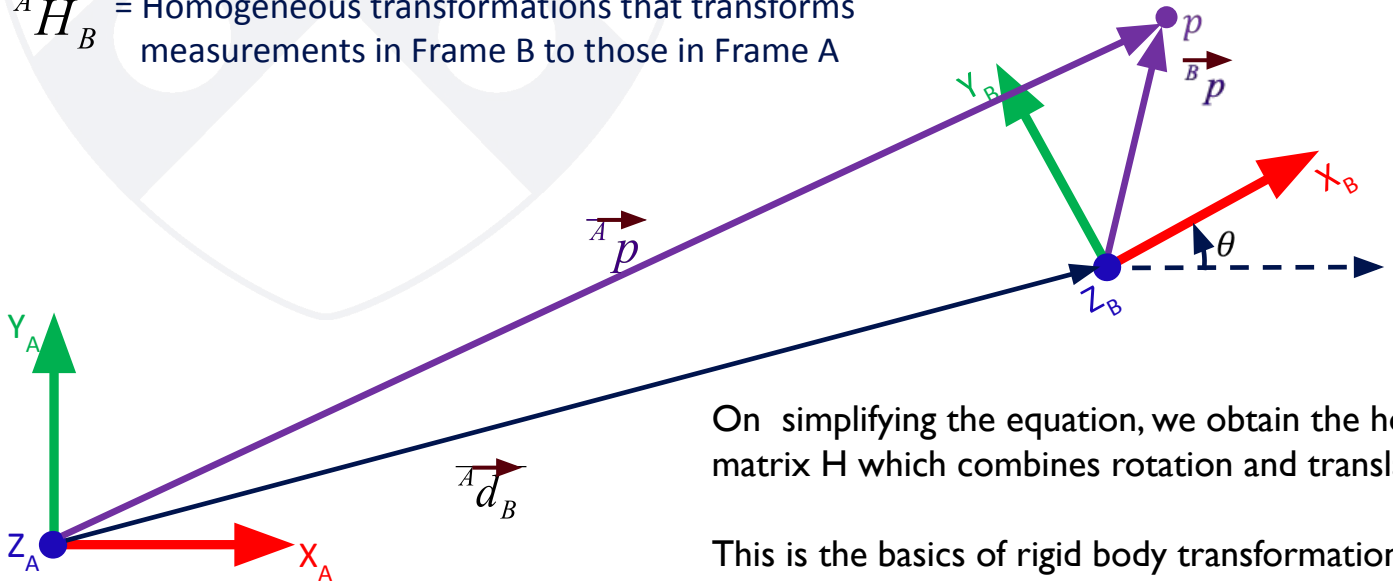
What we need is Point p with respect to Frame A, given its pose in Frame B

$$\vec{{}^{A}p} = {}^{A}H_{B} \times \vec{{}^{B}p}$$

$${}^{A}H_{B} = \begin{bmatrix} {}^{A}R_{B} & {}^{A}d_{B} \\ 0 & 1 \end{bmatrix}$$

$${}^{A}R_{B} = \begin{bmatrix} C_{\theta} & -S_{\theta} & 0 \\ S_{\theta} & C_{\theta} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

${}^{A}H_{B}$ = Homogeneous transformations that transforms measurements in Frame B to those in Frame A



On simplifying the equation, we obtain the homogeneous transformation matrix H which combines rotation and translation in one matrix

This is the basics of rigid body transformations in robotic systems.

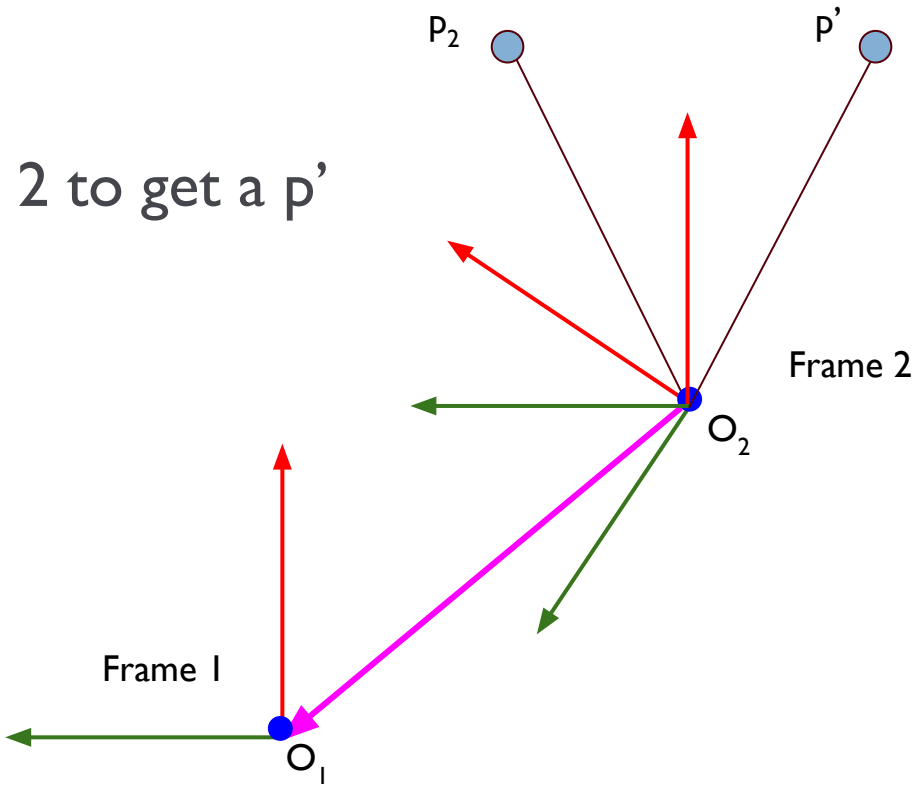
Recap:

Rotation & Translation in a slightly different example

First we apply the rotation

we rotate the reference frame 2 to get a p'

$$p' = \mathbf{R}_2^1 p_2$$



Putting everything together (with translation)

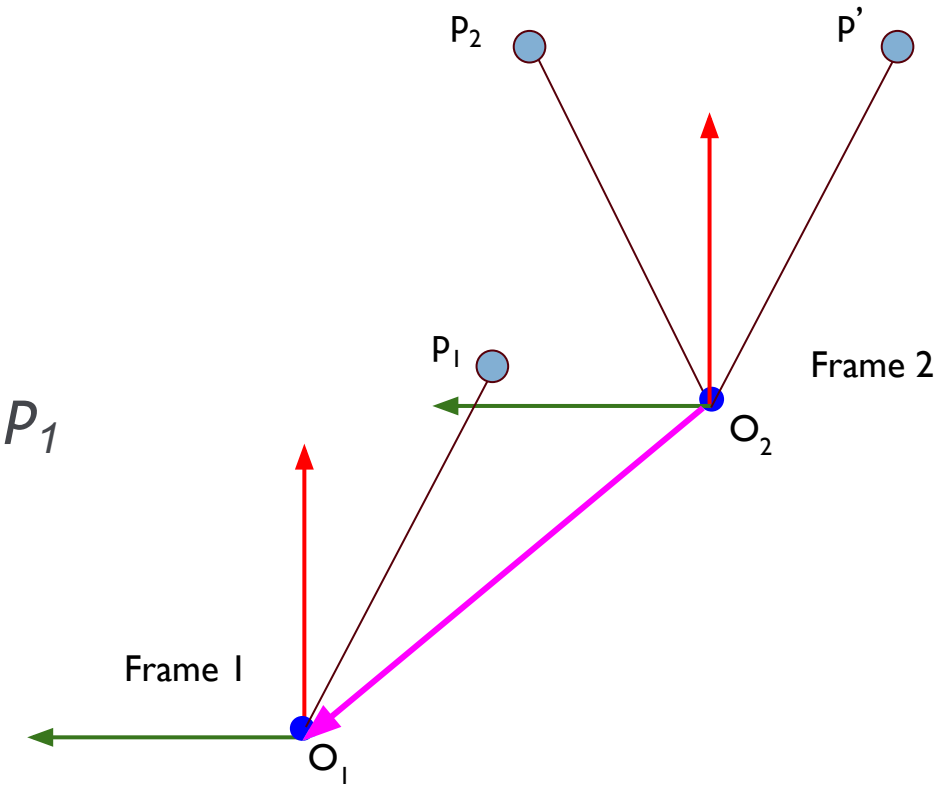
$$p' = \mathbf{R}_2^1 p_2$$

Then we apply the translation

p' is moved to the position of p_1

$$p_1 = \mathbf{R}_2^1 p_2 + O_{21}$$

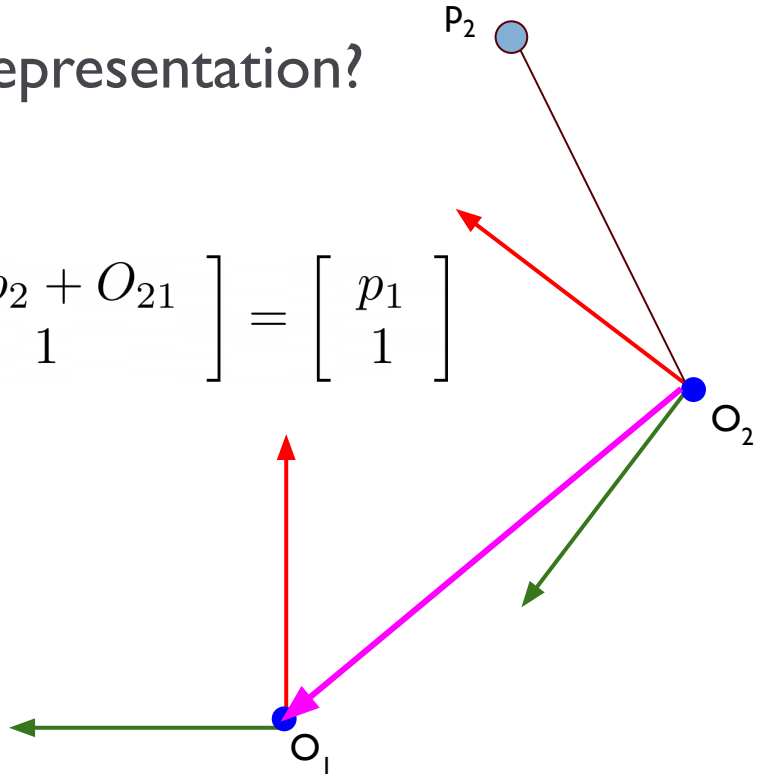
Multiplication is non-linear. Addition is linear



Putting everything together

What if we want a more compact representation?

$$p_1 = \mathbf{H}_2^1 p_2 = \begin{bmatrix} \mathbf{R}_2^1 & O_{21} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} p_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_2^1 p_2 + O_{21} \\ 1 \end{bmatrix} = \begin{bmatrix} p_1 \\ 1 \end{bmatrix}$$



Homogeneous Transformations

- A Homogeneous Transformation is a matrix representation of a rigid body transformation.

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix}$$

- Here R is a 2x2 (2D) / 3x3 (3D) rotation matrix and d is a 2x1 (2D) / 3x1 (3D) displacement vector.
- The Homogeneous Representation of a vector is:

$$\mathbf{P} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

- Here p is a 2x1 (2D) / 3x1 (3D) vector

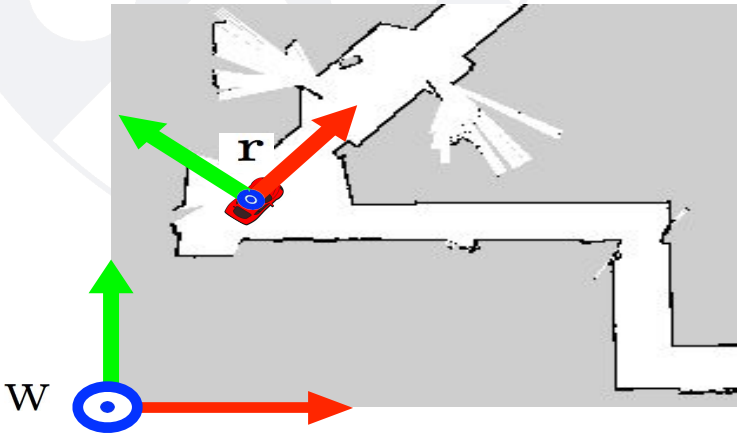
Padding with 1

Summary: Poses & Transformations

Points & Positions

How to express orientation of a coordinate frame with respect to another?

e.g., What is the orientation of the robot frame in the world frame?



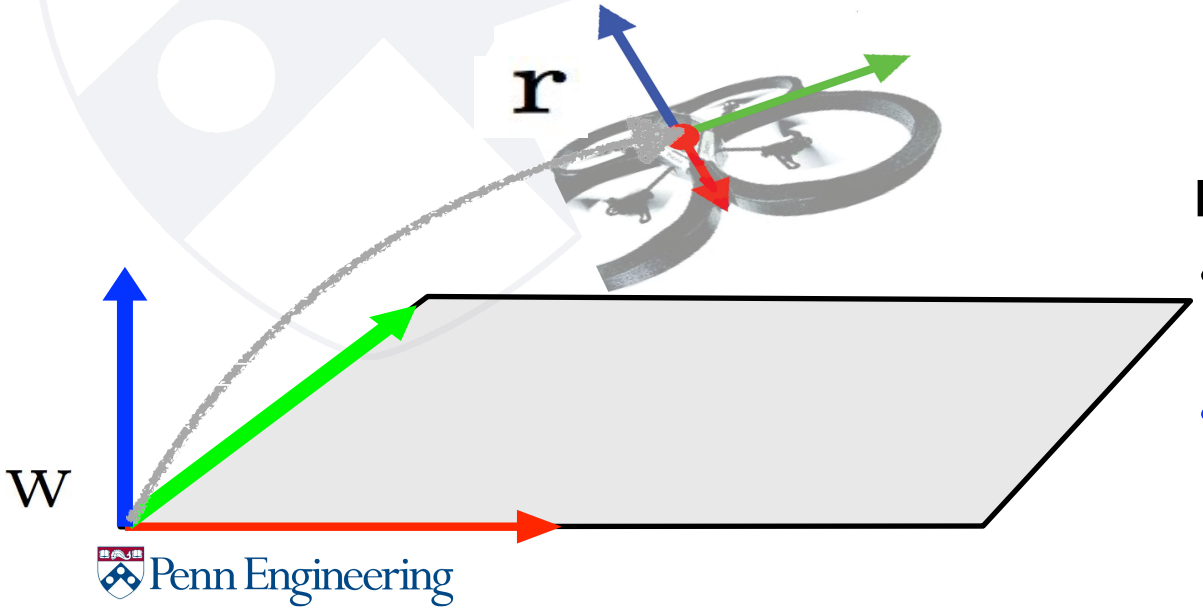
$$\boldsymbol{p}_{\mathrm{r}}^{\mathrm{W}} = \begin{bmatrix} 2.1 \\ 4.2 \end{bmatrix}$$ **2D**

$$\boldsymbol{p}_{\mathrm{r}}^{\mathrm{W}} = \begin{bmatrix} 2.1 \\ 4.2 \\ 0 \end{bmatrix}$$ **3D**

Poses & Transformations

Pose = position & rotation

$$T_r^W \qquad p_r^W \qquad R_r^W$$

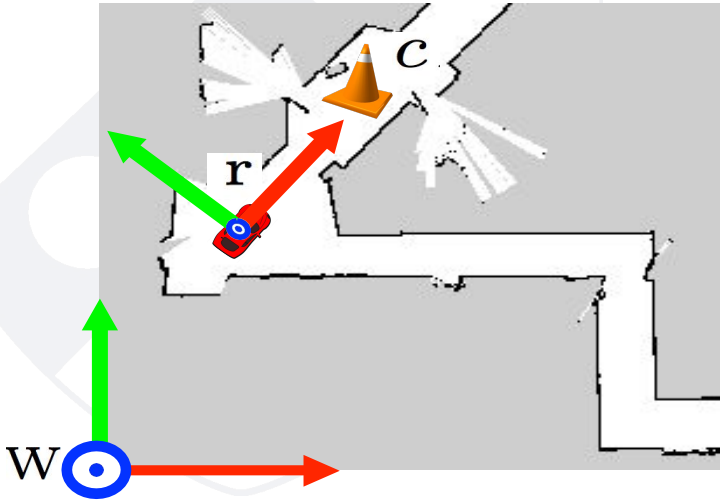


Poses are used to:

- express relations between coordinate frames
- transform points from one coordinate frame to another

Rigid-body Transformations

How to transform points from one coordinate frame to another?



Given point "c" in coordinate frame "r" what is the position of the point in frame "w"?

$$\boldsymbol{p}_{\mathrm{c}}^{\mathrm{W}} = \boldsymbol{R}_{\mathrm{r}}^{\mathrm{W}} \boldsymbol{p}_{\mathrm{c}}^{\mathrm{r}} + \boldsymbol{p}_{\mathrm{r}}^{\mathrm{W}}$$

Remark about notation

Positions

$$\boldsymbol{p}_{\mathrm{r}}^{\mathrm{W}} \ , \ {}^{\mathrm{W}}\boldsymbol{p}_{\mathrm{r}} \ , \ \boldsymbol{p}$$

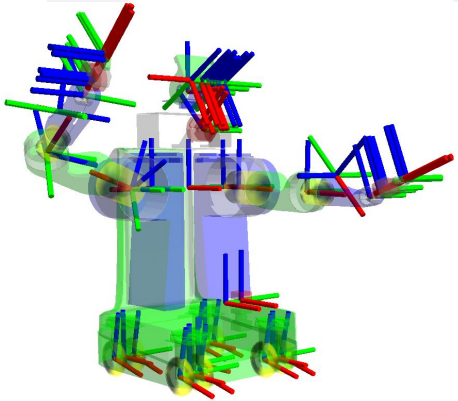
Rotations

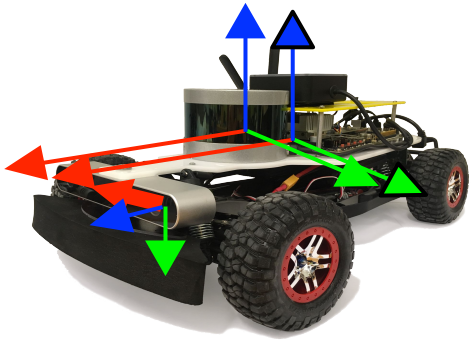
$$\boldsymbol{R}_{\mathrm{r}}^{\mathrm{W}} \ , \ {}^{\mathrm{W}}\boldsymbol{R}_{\mathrm{r}} \ , \ \boldsymbol{R}$$

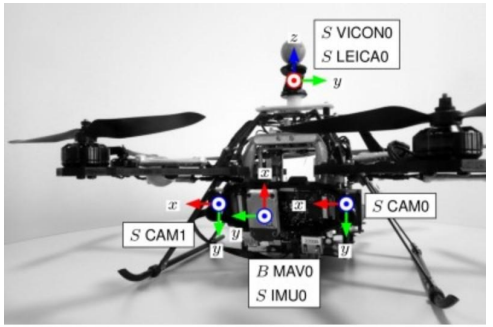
Poses

$$\boldsymbol{T}_{\mathrm{r}}^{\mathrm{W}} \ , \ {}^{\mathrm{W}}\boldsymbol{T}_{\mathrm{r}} \ , \ \boldsymbol{T}$$

- potential for confusion
- choice of notation depends on balance of simplicity / informativeness
- using consistent notation helps doing math right







Composition of Transforms

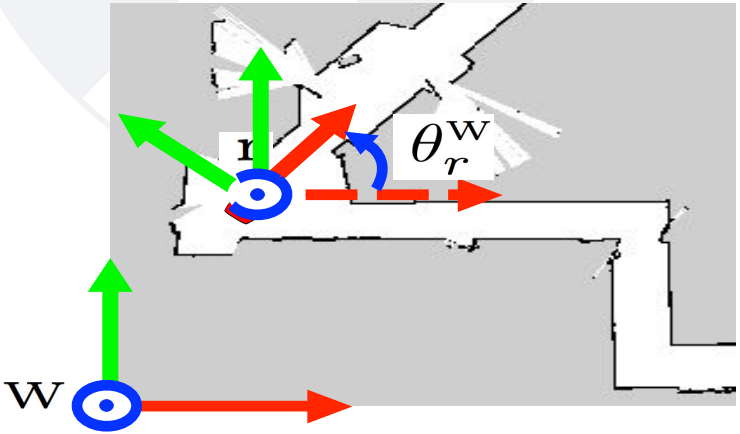
We now discuss how to express a sequence of movements of a robot

Orientation & Rotations: 2D Case

How to express orientation of a coordinate frame with respect to another?

e.g., What is the orientation of the robot frame in the world frame?

- **let's start from the 2D case**



$$\theta_r^{\mathrm{W}} = 47^\circ = 0.82\mathrm{rad}$$

$$\theta_r^{\mathrm{W}} \in (-\pi, +\pi]$$

wait! is angle positive or negative?

$$\boldsymbol{R}_r^{\mathrm{W}} = \begin{bmatrix} \cos(\theta_{\mathrm{r}}^{\mathrm{W}}) & -\sin(\theta_{\mathrm{r}}^{\mathrm{W}}) \\ \sin(\theta_{\mathrm{r}}^{\mathrm{W}}) & \cos(\theta_{\mathrm{r}}^{\mathrm{W}}) \end{bmatrix}$$

x y

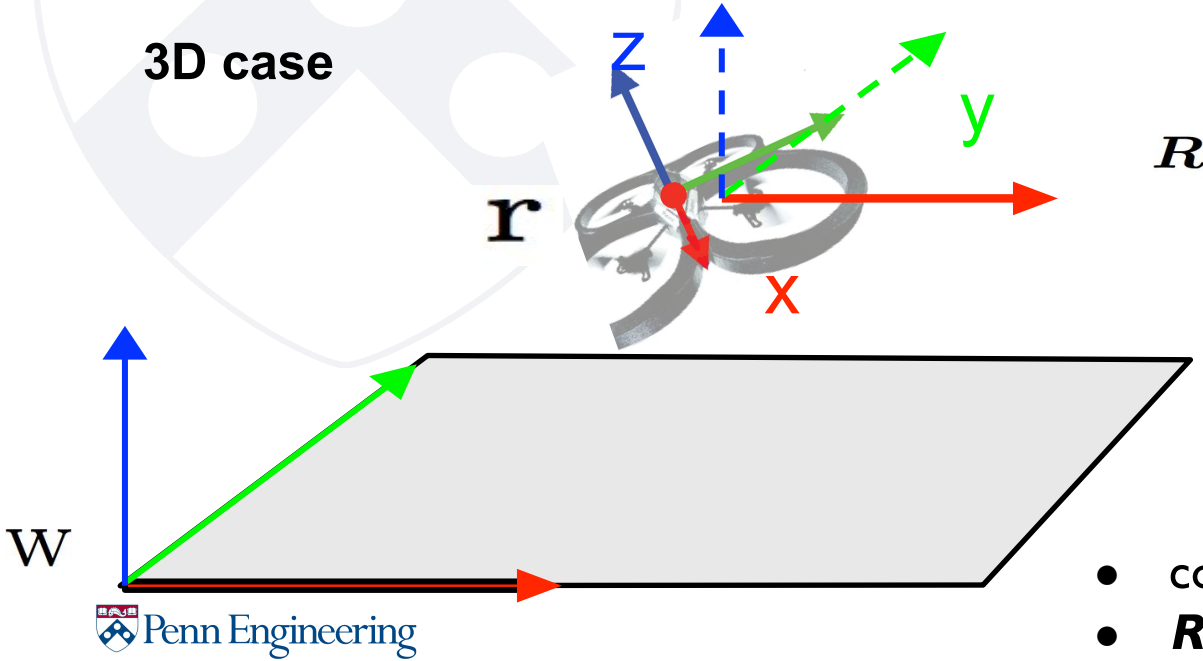
2D rotation matrix

Orientation & Rotations: 3D Case

How to express orientation of a coordinate frame with respect to another?

e.g., What is the orientation of the robot frame in the world frame?

3D case



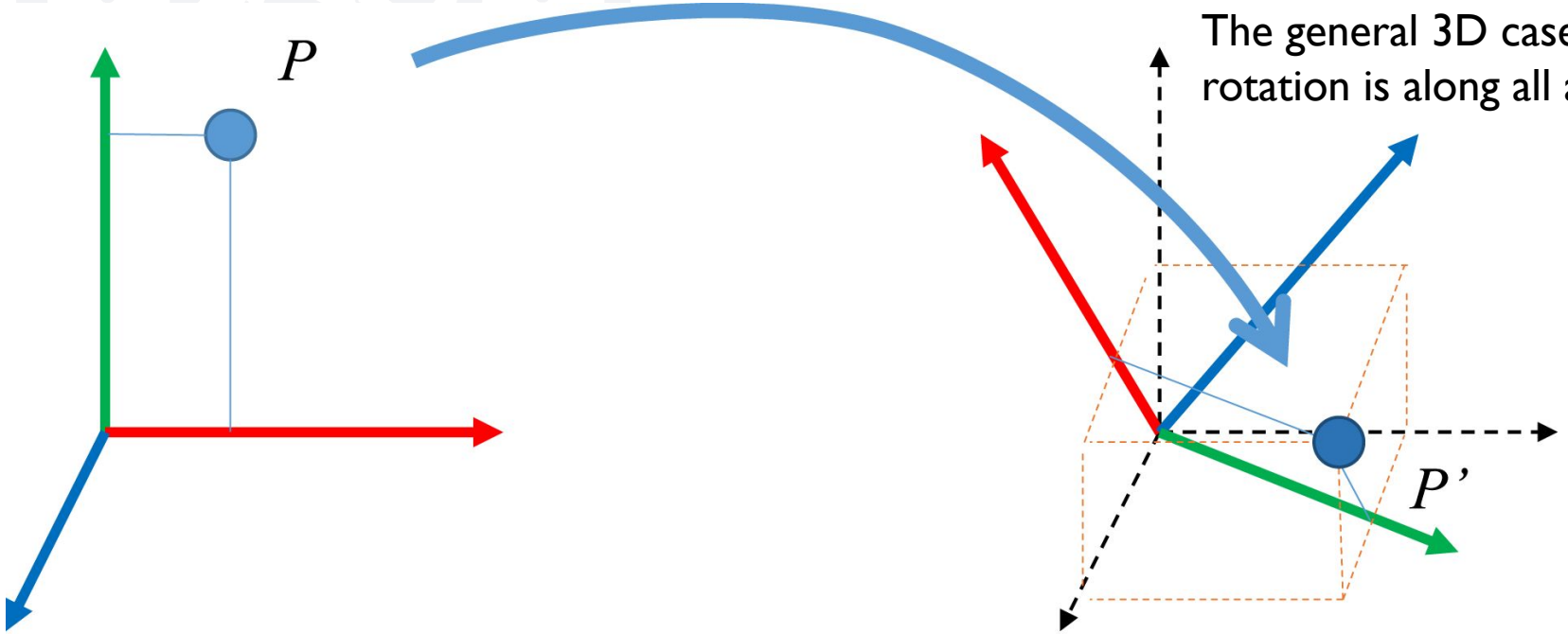
$$\boldsymbol{R}_r^{\mathrm{W}} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

x y z

3D rotation matrix

- columns: unit norm, orthogonal
- ***R* is an orthonormal matrix**

3D Rotation Matrix

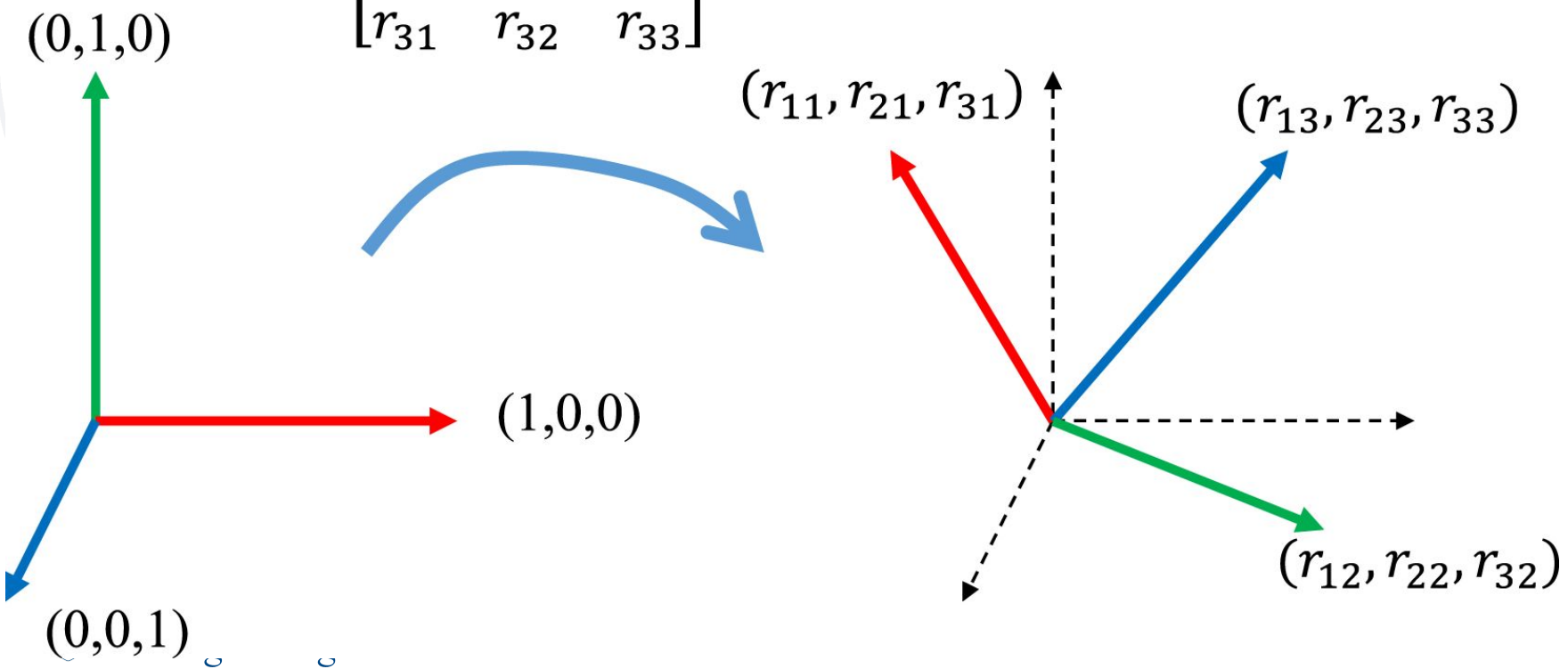


$$\boldsymbol{p} = (x, y, z)$$

$$\boldsymbol{p}' = R\boldsymbol{p} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

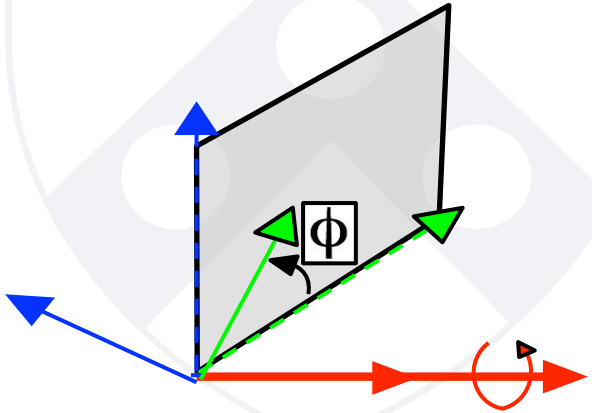
3D Rotation Matrix

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$



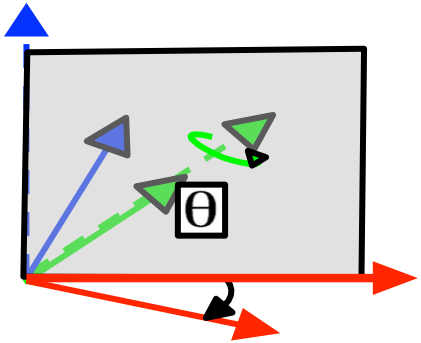
Elementary Rotations

Rotation around the **x** axis



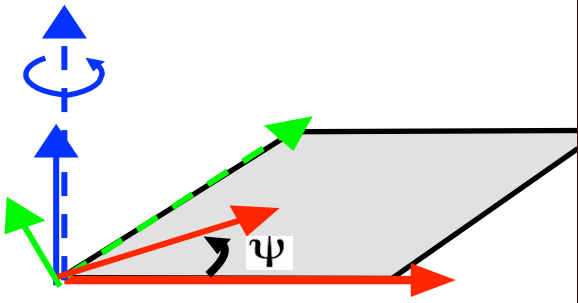
$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

Rotation around the **y** axis



$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

Rotation around the **z** axis



$$R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

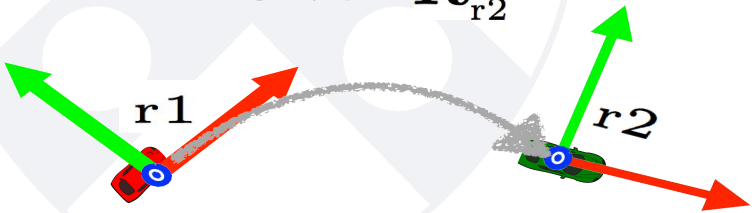
Inverse of Rotations

If we re given a rotation from from r1 to r2

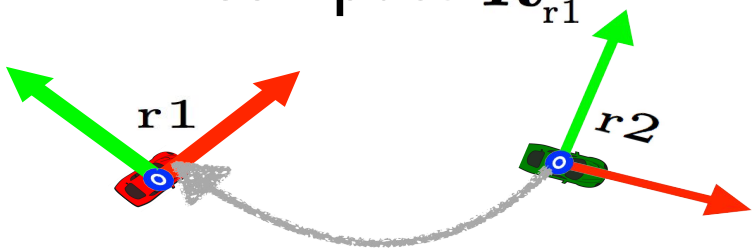
We can compute the rotation from r2 to r1 by taking the **inverse** of the given rotation

This is also equal to the **transpose** of the given rotation matrix

Given $\boldsymbol{R}_{\mathrm{r2}}^{\mathrm{r1}}$



compute $\boldsymbol{R}_{\mathrm{r1}}^{\mathrm{r2}}$

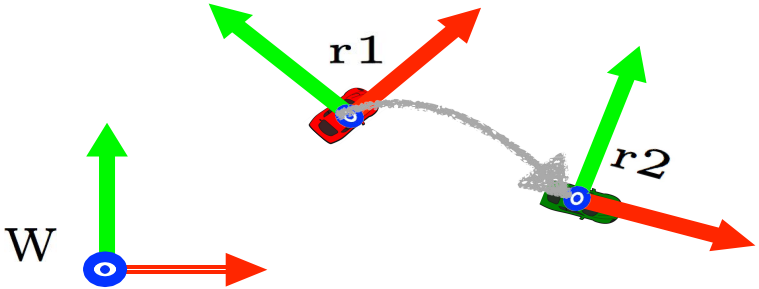


	Rotation Matrices	Euler Angles	Quaternions
Inverse	$\boldsymbol{R}_{\mathrm{r1}}^{\mathrm{r2}} = (\boldsymbol{R}_{\mathrm{r2}}^{\mathrm{r1}})^{-1} = (\boldsymbol{R}_{\mathrm{r2}}^{\mathrm{r1}})^{\mathsf{T}}$	☹	$\boldsymbol{q}_{\mathrm{r1}}^{\mathrm{r2}} = [s_{\mathrm{r2}}^{\mathrm{r1}} \quad -\boldsymbol{v}_{\mathrm{r2}}^{\mathrm{r1}}]$

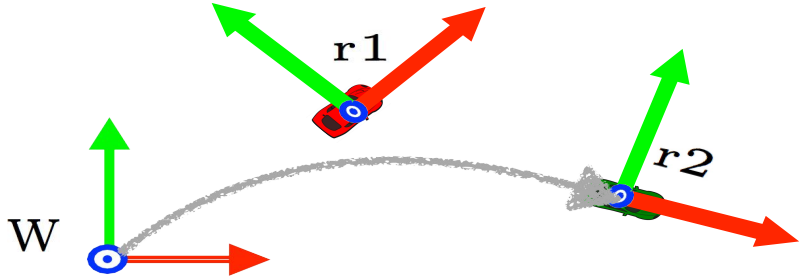
Composition of Rotations

We can compose a sequence of rotations by multiplying individual rotations.
If we have a rotation from W to r1 and then r1 to r2,
we can multiply the individual rotations to capture the rotation from the origin to r2

Given $\boldsymbol{R}_{\mathrm{r1}}^{\mathrm{W}}$ and $\boldsymbol{R}_{\mathrm{r2}}^{\mathrm{r1}}$



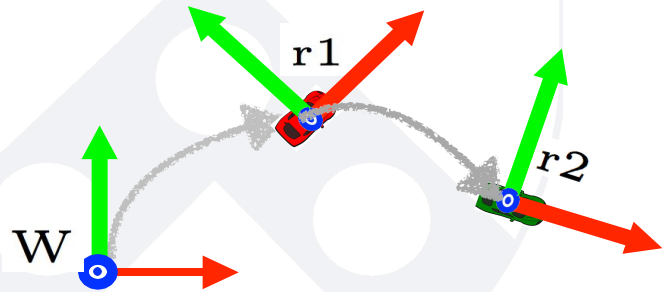
compute $\boldsymbol{R}_{\mathrm{r2}}^{\mathrm{W}}$



	Rotation Matrices	Euler Angles	Quaternions
Composition	$\boldsymbol{R}_{\mathrm{r2}}^{\mathrm{W}} = \boldsymbol{R}_{\mathrm{r1}}^{\mathrm{W}} \cdot \boldsymbol{R}_{\mathrm{r2}}^{\mathrm{r1}}$		$\boldsymbol{q}_{\mathrm{r2}}^{\mathrm{W}} = [s_{\mathrm{r1}}^{\mathrm{W}} s_{\mathrm{r2}}^{\mathrm{r1}} - \boldsymbol{v}_{\mathrm{r1}}^{\mathrm{W}} \cdot \boldsymbol{v}_{\mathrm{r2}}^{\mathrm{r1}}$ $s_{\mathrm{r1}}^{\mathrm{W}} \boldsymbol{v}_{\mathrm{r2}}^{\mathrm{r1}} + s_{\mathrm{r2}}^{\mathrm{r1}} \boldsymbol{v}_{\mathrm{r1}}^{\mathrm{W}} + \boldsymbol{v}_{\mathrm{r1}}^{\mathrm{W}} \times \boldsymbol{v}_{\mathrm{r2}}^{\mathrm{r1}}]$ (quaternion product)

Composition and Inverse of Transformations

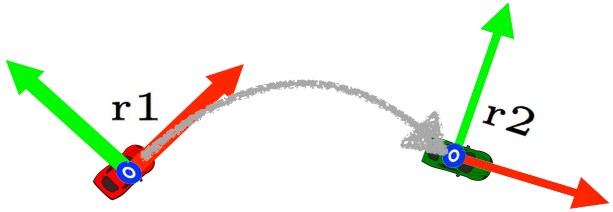
Composition



$$T_{r2}^{W} = T_{r1}^{W} \cdot T_{r2}^{r1}$$

This works for Homogeneous Transforms too
So we can apply composition and inverse transforms by chaining the sequence of individual transforms

Inverse



$$T_{r1}^{r2} = \begin{bmatrix} R_{r1}^{r2} & p_{r1}^{r2} \\ 0 & 1 \end{bmatrix}$$

$$T_{r1}^{r2} = (T_{r2}^{r1})^{-1} =$$

$$= \begin{bmatrix} (R_{r1}^{r2})^{T} & -(R_{r1}^{r2})^{T} p_{r1}^{r2} \\ 0 & 1 \end{bmatrix}$$

Other representations of rotations

- Quaternions: (ROS standard)
  - Supplementary materials in LaValle's Virtual Reality lectures.
- Euler angles
- Axis angle

Euler Angles

Euler's rotation theorem (1.0): Any rotation can be written as the product of no more than 3 elementary rotations (no two consecutive rotations along the same axis).

$$\boldsymbol{R} = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha)$$

$$\boldsymbol{R} = \boldsymbol{R}_y(\gamma)\boldsymbol{R}_z(\beta)\boldsymbol{R}_y(\alpha)$$

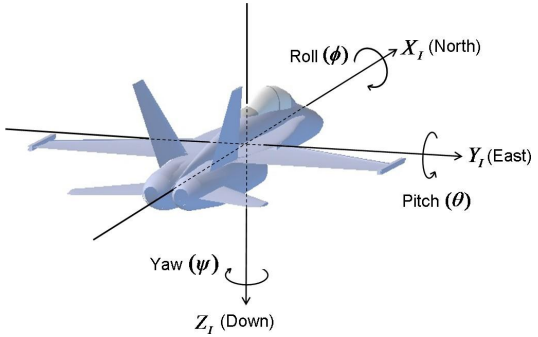
- why not x-y-z, y-x-z, x-x-z, …?
- α, β, γ are generally called **Euler angles**

$$\boldsymbol{R} = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha)$$

roll

pitch

yaw



- We only need 3 numbers to represent a rotation!

http://petercorke.com/wordpress/toolboxes/robotics-toolbox

Where will you use this?

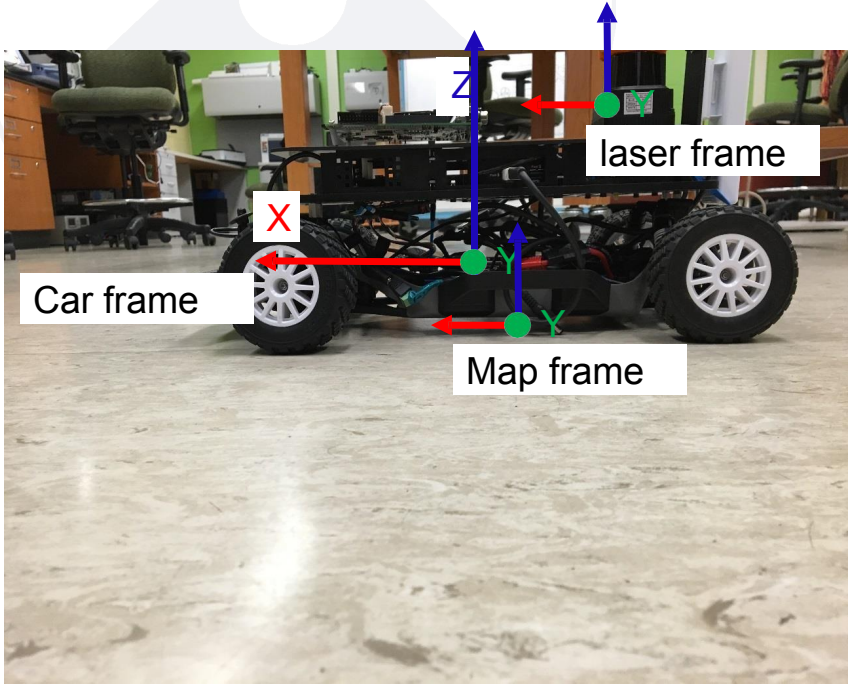
Labs 2, 3: Automatic Emergency Braking,
Reactive Methods for navigation & planning in local frame

Later labs: Map based methods, need transforms to perform planning in global frame.

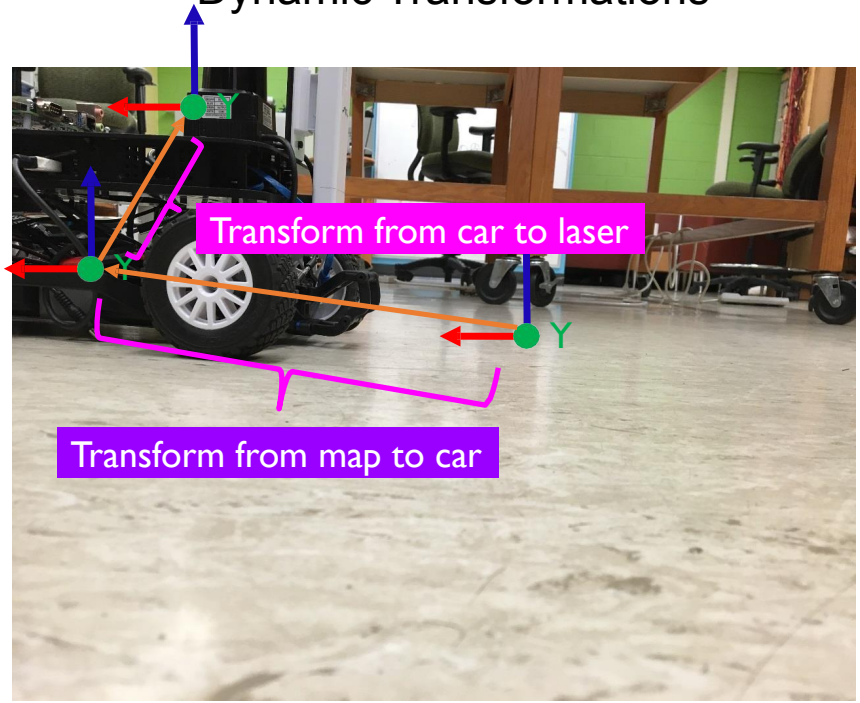
Next: How do we do this in ROS?

To convert measurements from one frame to another we need to perform transformations

Static Transformations



Dynamic Transformations



Key Takeaways

1. Coordinates are numerical representations of geometric concepts, like points, directions, frames of reference, and movement.

2. Points, directions, and displacements in n -dimensional space are represented by n -dimensional vectors, while rotations and scalings are represented by $n \times n$ matrices.

3. Rigid transformations consist of a rotation followed by a translation. They represent both rigid body movement and changes of coordinate frame.

4. Homogeneous coordinates represent rigid transforms using matrix multiplication in an $n+1$ dimensional space where the last coordinate is either 0 or 1.

5. When working with coordinates it is easy to make mistakes. Having clear assumptions, clear notation and/or using coordinate management software can reduce the risk of error.

Questions?

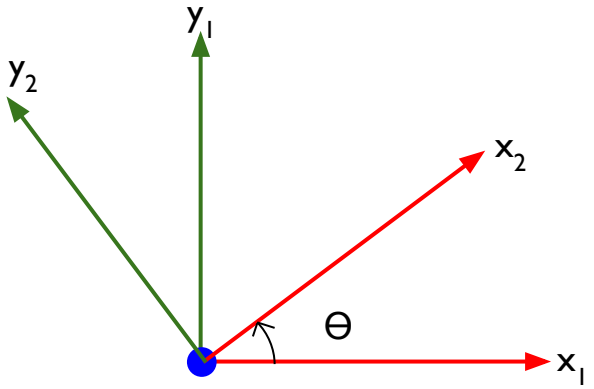
Extra slides

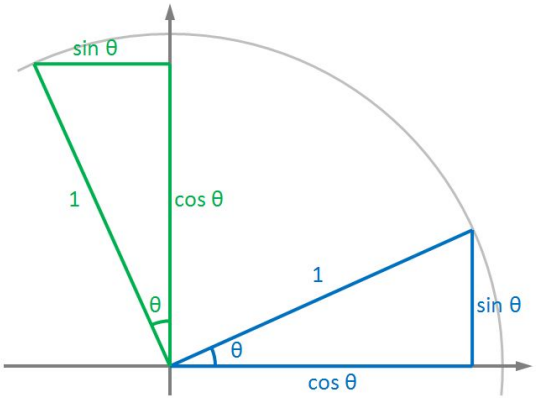
Rotation Matrix

$$\mathbf{x}_2 = cos(\theta)\mathbf{x}_1 + sin(\theta)\mathbf{y}_1$$
$$\mathbf{y}_2 = -sin(\theta)\mathbf{x}_1 + cos(\theta)\mathbf{y}_1$$

$$\mathbf{R} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

x y



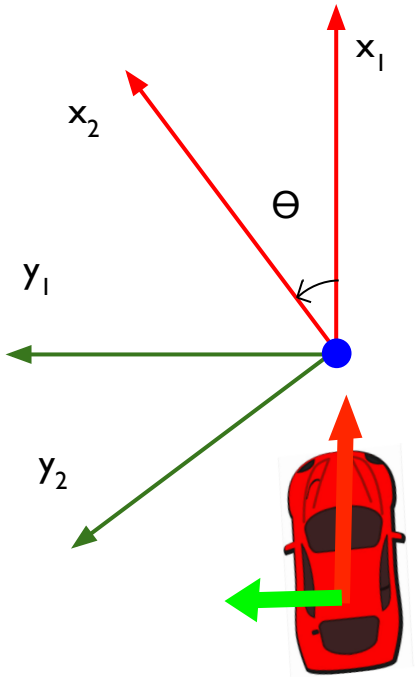


Rotation Matrix - from the racecar's perspective

$$\mathbf{x}_2 = cos(\theta)\mathbf{x}_1 + sin(\theta)\mathbf{y}_1$$
$$\mathbf{y}_2 = -sin(\theta)\mathbf{x}_1 + cos(\theta)\mathbf{y}_1$$

$$\mathbf{R} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

x y

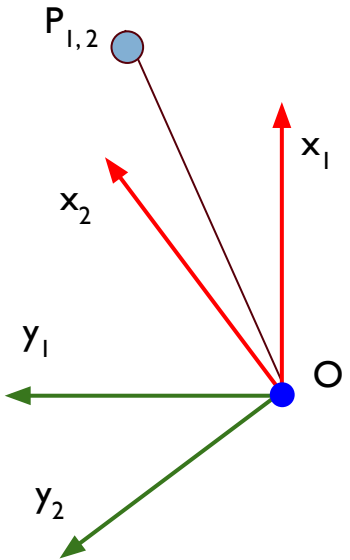


Rotation Matrix

Recall we represented the basis vectors of one frame as linear combinations of the other frame's basis vector. We can do the same here.

$$OP_1 = p_{x1}x_1 + p_{y1}y_1$$

$$OP_2 = p_{x2}x_2 + p_{y2}y_2$$

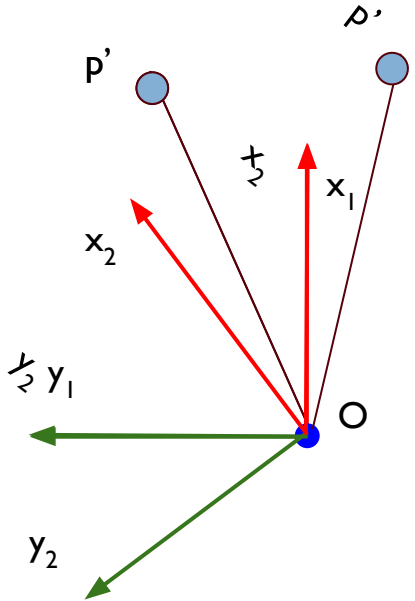


Rotation Matrix

How do we relate the two?

$$OP_1 = p_{x1}x_1 + p_{y1}y_1$$

$$OP_2 = p_{x2}x_2 + p_{y2}y_2$$

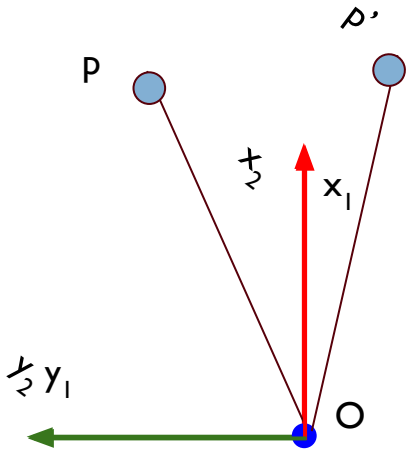


Rotation Matrix

Instead of using two frames, we can represent these two vectors in the same basis vector now.

$$OP_1 = p_{x1}x_1 + p_{y1}y_1$$

$$OP_2 = p_{x2}x_2 + p_{y2}y_2$$



Homogeneous Transformations

- How to transform points from one coordinate frame to another?

$$\boldsymbol{p}_{\mathrm{c}}^{\mathrm{W}} = \boldsymbol{R}_{\mathrm{r}}^{\mathrm{W}} \boldsymbol{p}_{\mathrm{c}}^{\mathrm{r}} + \boldsymbol{p}_{\mathrm{r}}^{\mathrm{W}}$$

- Homogeneous coordinates:

$$\tilde{\boldsymbol{p}}_{\mathrm{c}}^{\mathrm{W}} = \begin{bmatrix} \boldsymbol{p}_{\mathrm{c}}^{\mathrm{W}} \\ 1 \end{bmatrix} \qquad \tilde{\boldsymbol{p}}_{\mathrm{c}}^{\mathrm{r}} = \begin{bmatrix} \boldsymbol{p}_{\mathrm{c}}^{\mathrm{r}} \\ 1 \end{bmatrix}$$

- More elegant, matrix formulation:

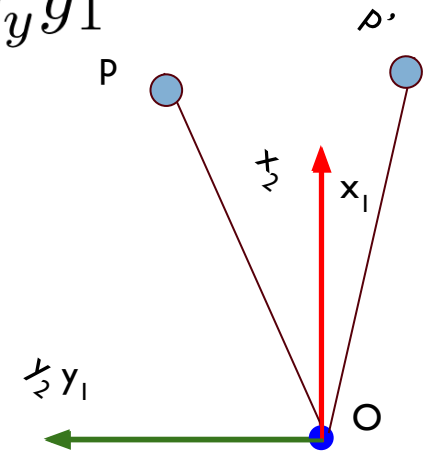
$$\tilde{\boldsymbol{p}}_{\mathrm{c}}^{\mathrm{W}} = \underbrace{\begin{bmatrix} \boldsymbol{R}_{\mathrm{r}}^{\mathrm{W}} & \boldsymbol{p}_{\mathrm{r}}^{\mathrm{W}} \\ \mathbf{0} & 1 \end{bmatrix}}_{\boldsymbol{T}_{\mathrm{r}}^{\mathrm{W}}} \tilde{\boldsymbol{p}}_{\mathrm{c}}^{\mathrm{r}}$$

Pose & rigid body transformation

Rotation Matrix

$$OP = p_x x_1 + p_y y_1 \quad OP' = p'_x x_1 + p'_y y_1$$

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{bmatrix} p'_x \\ p'_y \end{bmatrix}$$



$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{bmatrix} \begin{bmatrix} p'_x \\ p'_y \end{bmatrix}$$

???

Example: Rotations in 3D

Rotation matrices in 3D around each axis. See something similar?

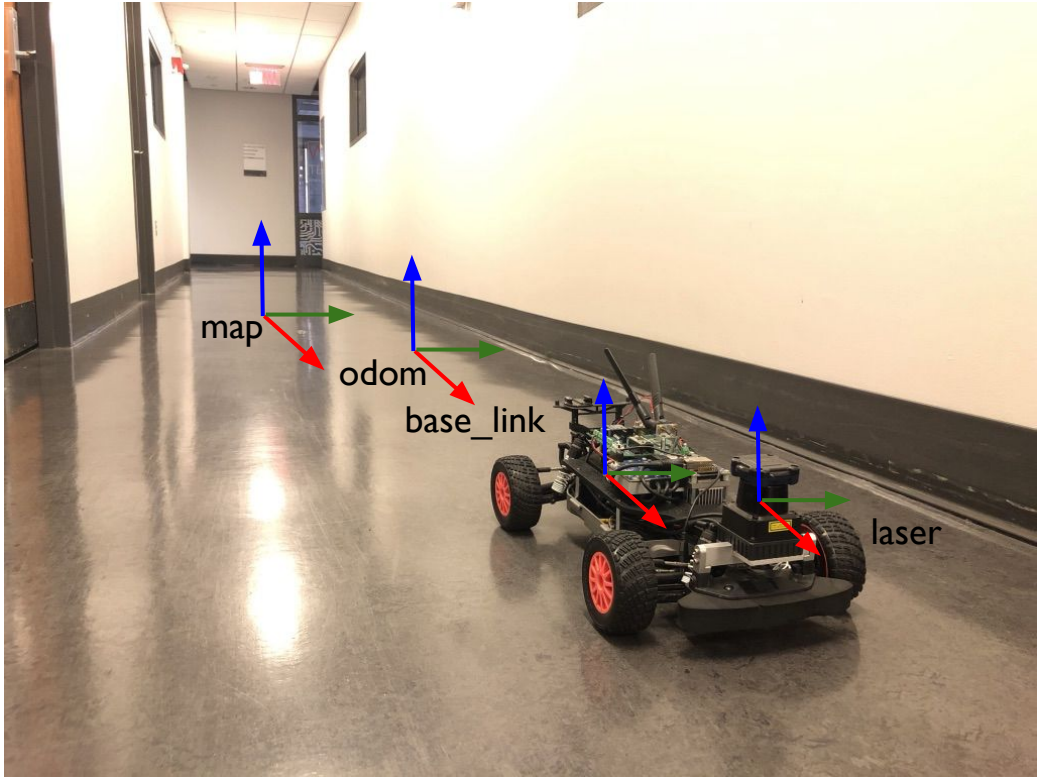
$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

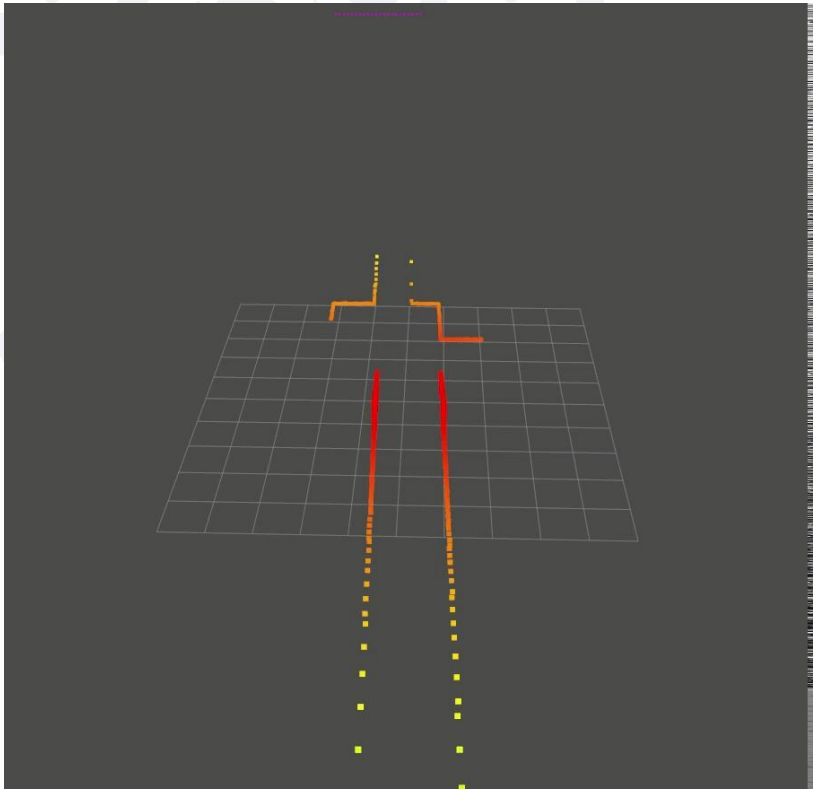
$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rigid Body Pose

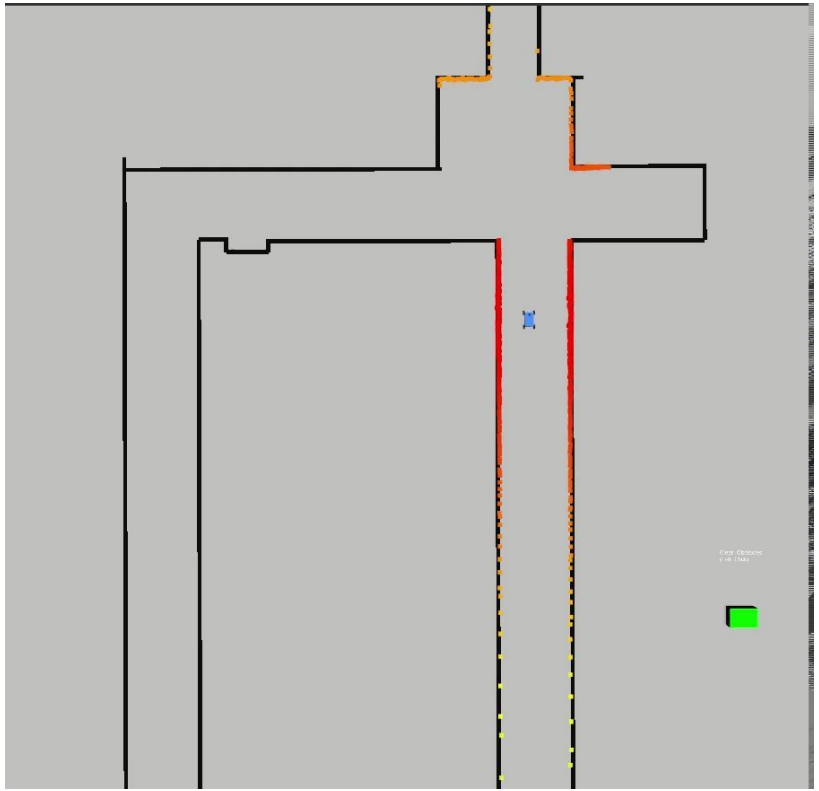
Once chosen a frame, a rigid body is described in space by its position and orientation with respect to that reference frame.



Multiple Reference Frames



Penn Engineering Laser frame



Map frame