Principles of Robot Autonomy I

Models: Environment maps, Robot geometry models,
Coordinate transforms

Principles of Robot Autonomy I

Announcements:

- Lab section 0 this week – set up your computer for ROS2 (on your own)
- HW 1 out today, due Thurs Oct 9
- Slides uploaded to canvas before lecture
- Total enrolled: 167, total waitlist: 135

Lecture 2:

- Environment maps
- Robot geometry models
- Coordinate transforms, SO(2)/SO(3), SE(2)/SE(3)

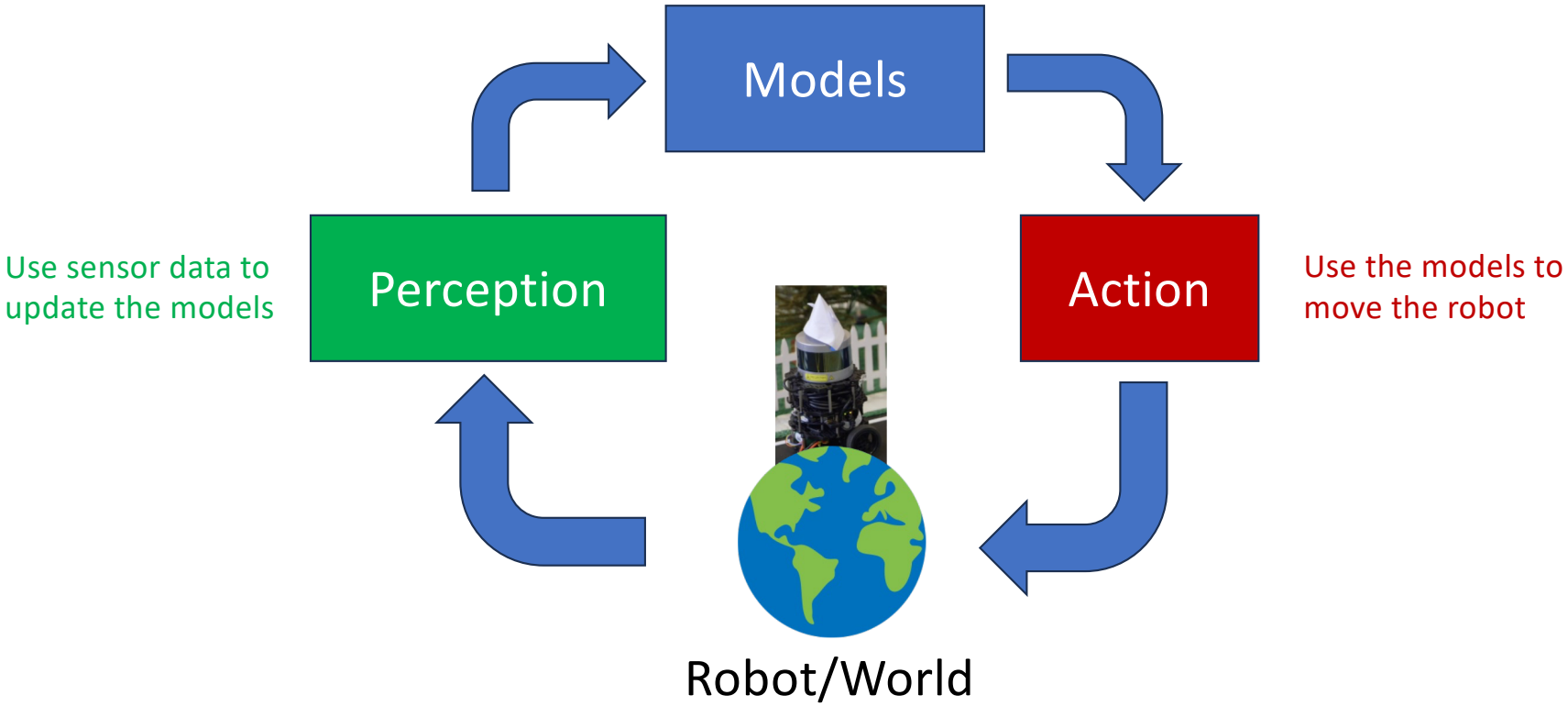
Stanford University

MSL

IPRL

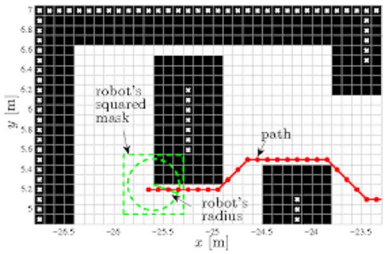
ASL Autonomous Systems Lab

Full Stack Autonomy: the Perception-Action Loop

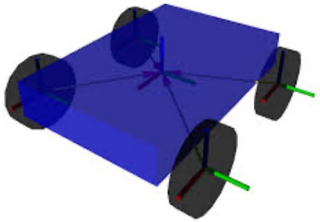


Models: Environment and Robot Representations

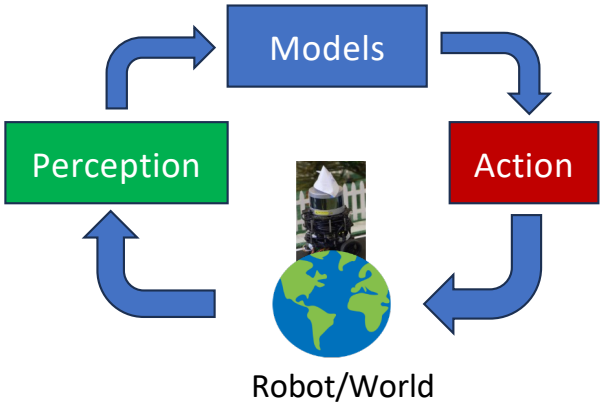
Maps:



Robot models
or simulators:



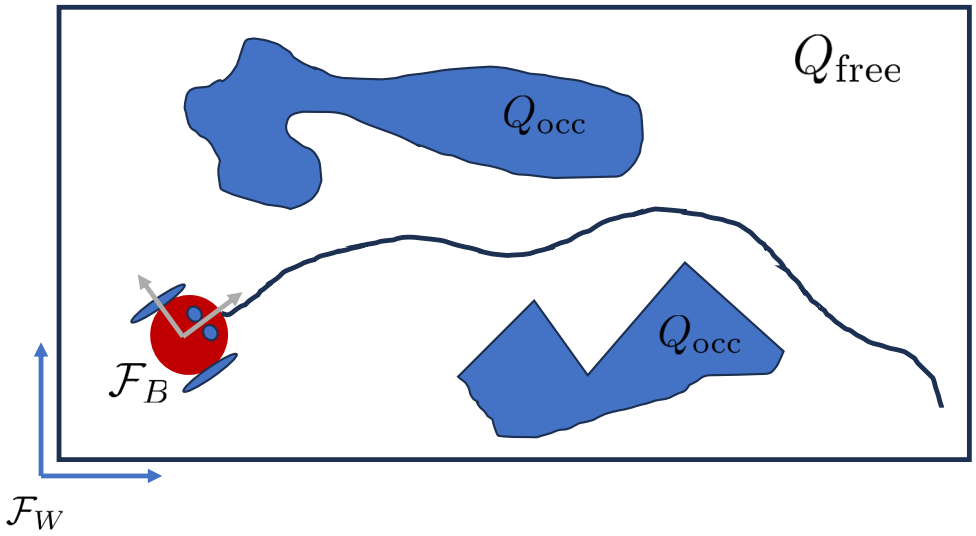
Agent models or
simulators:



Maps

Purpose: To represent free and occupied space in a *fixed world frame*. Used for collision avoidance and path planning in robot navigation.

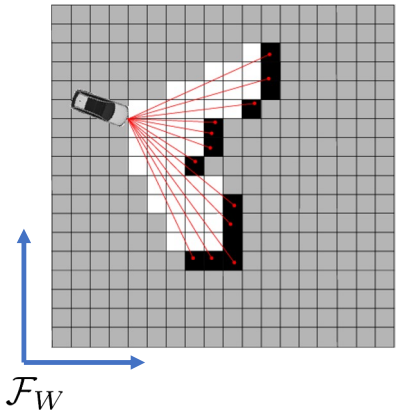
$$Q = Q_{\text{free}} \cup Q_{\text{occ}} \quad \text{so} \quad Q_{\text{free}} = Q_{\text{occ}}^{C}$$



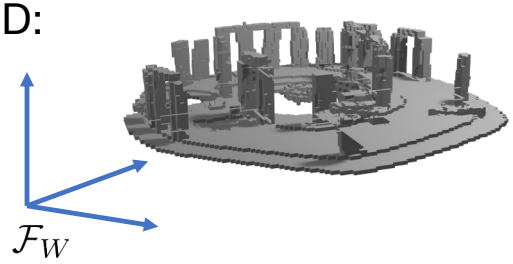
How to mathematically represent these sets?
How to update them from robot sensor measurements as the robot moves?

Occupancy Grid Map

2D:



3D:



Grid cell i:

Location $q_i = (q_i^x, q_i^y)$

State $x_i \in \{0, 1\}$

$$Q = Q_{\text{free}} \cup Q_{\text{occ}}$$

$$Q_{\text{free}} = \{q_i \mid x_i = 0\}$$

$$Q_{\text{occ}} = \{q_i \mid x_i = 1\}$$

Probabilistic occupancy grid:

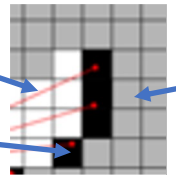
$$p_i = p(x_i = 1 \mid \text{sensor measurements})$$

$$p_i \in [0, 1]$$

Probability threshold:

$p_i \leq 0.05$

$p_i \geq 0.95$



$0.05 \leq p_i \leq 0.95$

Log-Odds:

$$l_i = \log(\frac{p_i}{1 - p_i}) \in [-\infty, \infty]$$

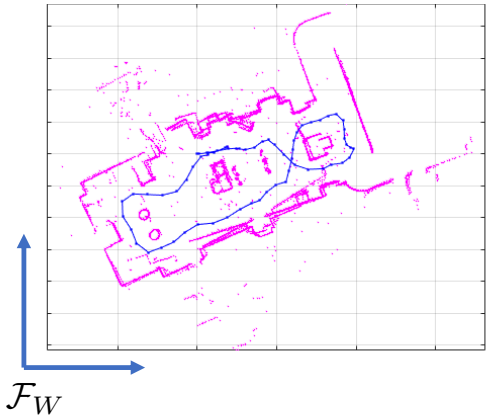
$$l_i = 0 \iff p_i = 0.5$$

Makes updating with Bayes' rule very simple (we'll see later)

"Mapping": Update all p_i or l_i with Bayes' rule in real time using robot sensor measurements.

Point Cloud Map

2D:



Points represent occupied space:

$$q_i \in \mathbb{R}^2 \text{ or } \mathbb{R}^3$$

Native output of a lidar sensor.

$$Q_{\text{occ}} = \{q_i\}_{i=1}^{N}$$

Still need to transform from robot body to world frame.

Probabilistic point cloud map:

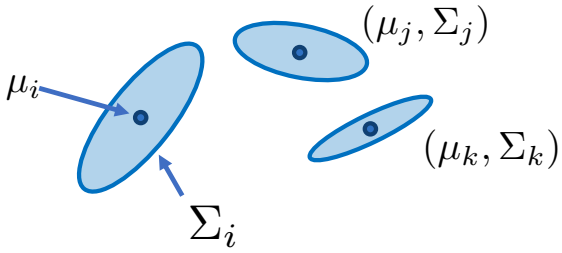
Multivariate Gaussian

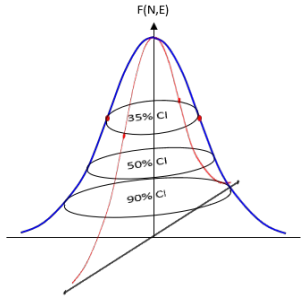
$$p(q_i \mid \text{robot measurements}) = \mathcal{N}(\mu_i, \Sigma_i)$$

Mean vector

Covariance matrix

3D:

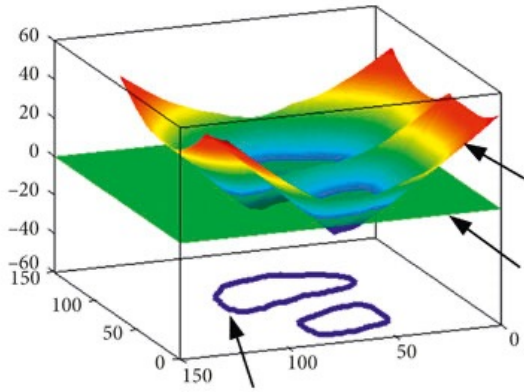




"Mapping": Update (μ_i, Σ_i) with a Kalman filter or a factor graph optimization solver.

Analytic Representations

Signed distance function (SDF) level set: $g(q) \leq 0$



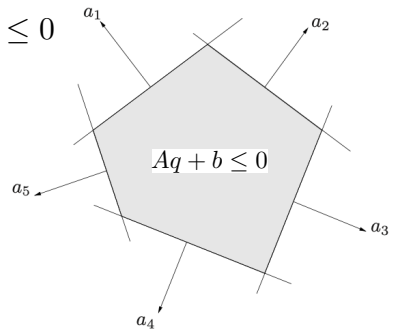
$y = g(q)$

$y = 0$

$g(q) \leq 0$

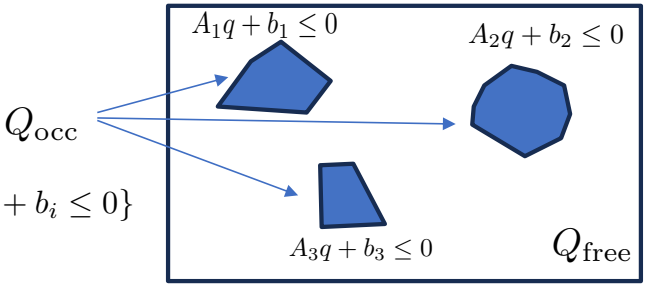
Convex polytopes: $Aq + b \leq 0$

$a_1^T q + b_1 \leq 0$



$$A = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_5^T \end{bmatrix} \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_5 \end{bmatrix}$$

Map with convex obstacles

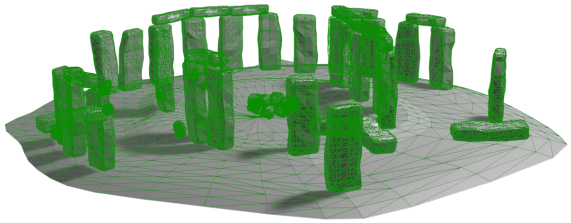


$Q_{\text{occ}} = \{q \mid \exists i \text{ s.t. } A_i q + b_i \leq 0\}$

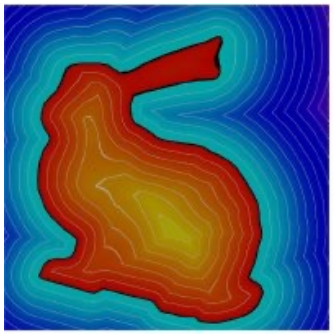
Other Map Representations

Neural Representations (NeRF, 3DGS):

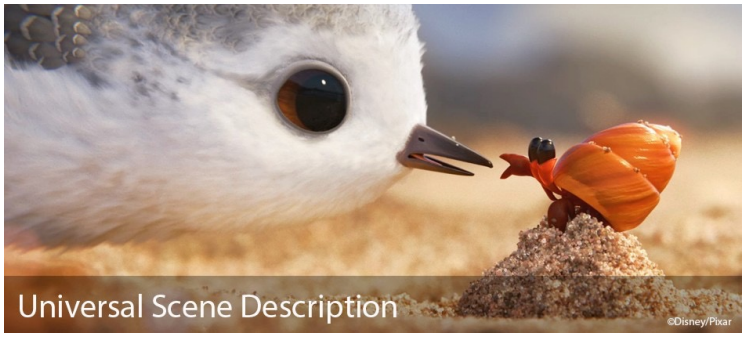
Triangular Mesh (CV/graphics,
Standard Triangle Language (STL)):



Neural Signed Distance Function (SDF):

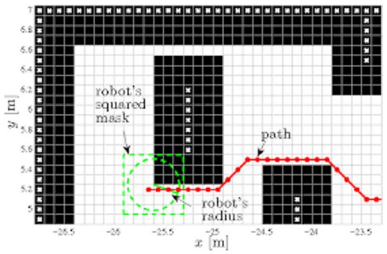


Simulator and Animation representations
(e.g. Universal Scene Description (USD) format)

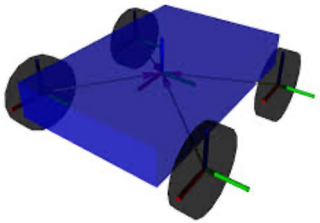


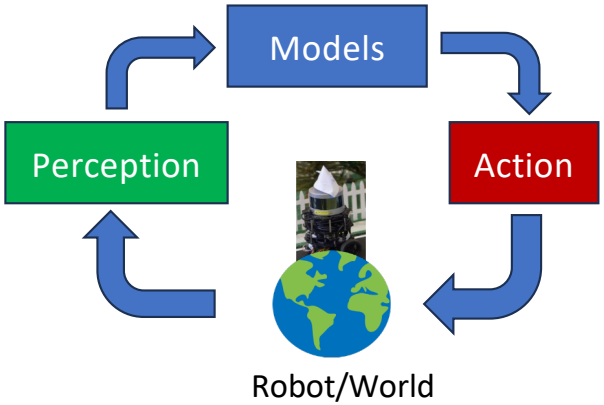
Models: Environment and Robot Representations

Maps:



Robot models
or simulators:





Models: Robot Geometry

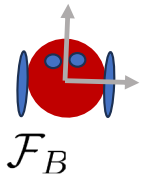
Purpose: To represent the space occupied by the robot body as the robot moves around.

In robot navigation, typically robot is modeled as a rigid body.
Manipulators usually modeled as multi-link chains of rigid bodies.
Both represented as Universal Robot Description File (URDF) in ROS2.

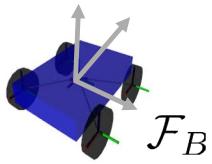
Robot body is a set $B \subset \mathbb{R}^2$ or $\mathbb{R}^3$ relative to a body-fixed coordinate system.

URDF uses simple geometric primitives (spheres, cylinders, capsules):

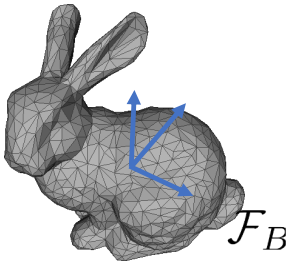
2D:



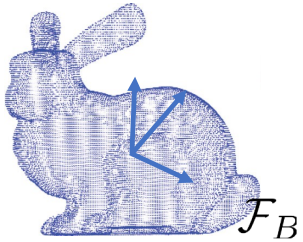
3D:



Triangular mesh, represented as Standard Triangle Language (STL) file:



Point cloud:



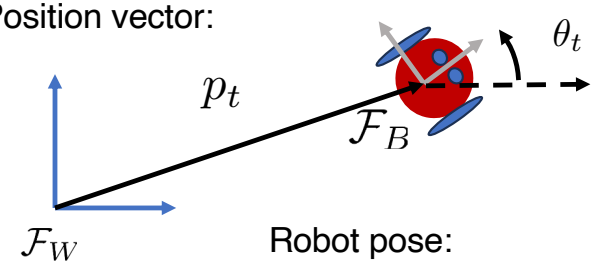
We need this for planning, to determine collision with occupied space in the map!
Also needed for rendering the robot motion.

Coordinate Frames and Transforms

Robot body in body-fixed frame: $B \subset \mathbb{R}^2$ or $\mathbb{R}^3$

Position vector:

Heading angle:

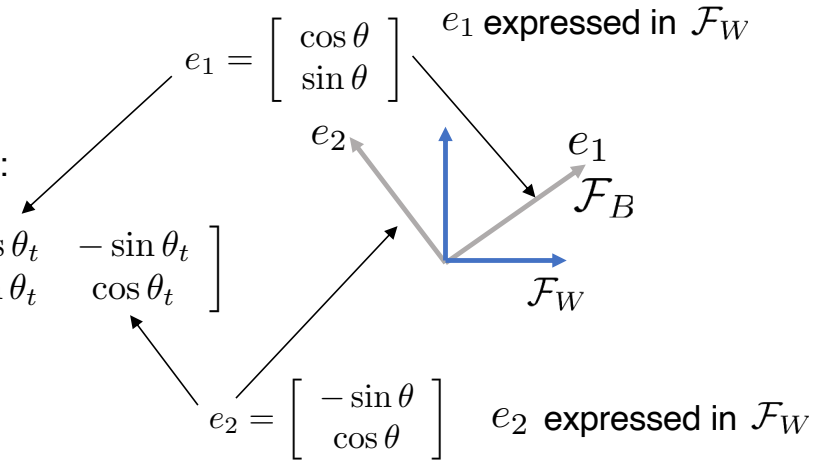


Robot pose:

$$(p_t, \theta_t) \text{ or } (p_t, R_t)$$

Rotation matrix:

$$R_t = \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix}$$



Transform vector from body to world:

$$v_W = R_t v_B + p_t \quad \text{or} \quad \begin{bmatrix} v_W \\ 1 \end{bmatrix} = \begin{bmatrix} R_t & p_t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_B \\ 1 \end{bmatrix}$$

Robot body in world frame:

$$B(p_t, R_t) = \{R_t b + p_t \mid b \in B\}$$

Recall, special orthogonal matrix:

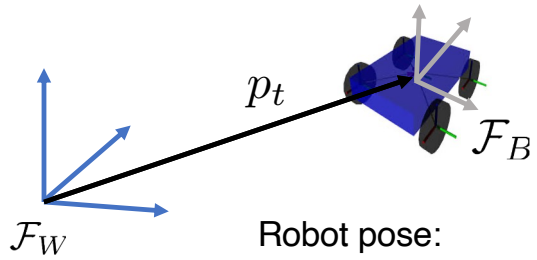
$$R_t \in \mathbb{SO}(2) \text{ or } \mathbb{SO}(3)$$

Definition: $R_t^T R_t = I \text{ and } \det(R_t) = +1$

All columns mutually orthogonal unit vectors.

More detail on 3D

Robot body in body-fixed frame: $B \subset \mathbb{R}^2$ or $\mathbb{R}^3$



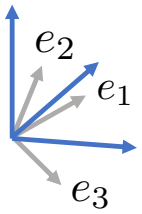
Robot pose:

(p_t, R_t)

Rotation matrix:

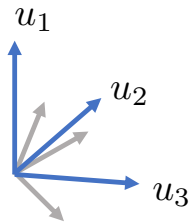
$$R_t = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}$$

e_i expressed in $\mathcal{F}_W$



Inverse transform:

$$R_t = \begin{bmatrix} u_1^T \\ u_2^T \\ u_3^T \end{bmatrix}$$



$$R_t^T = R_t^{-1} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}$$

Transform vector from body to world:

$$v_W = R_t v_B + p_t \quad \text{or} \quad \begin{bmatrix} v_W \\ 1 \end{bmatrix} = \begin{bmatrix} R_t & p_t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_B \\ 1 \end{bmatrix}$$

Robot body in world frame:

$$B(p_t, R_t) = \{R_t b + p_t \mid b \in B\}$$

Recall, special orthogonal matrix:

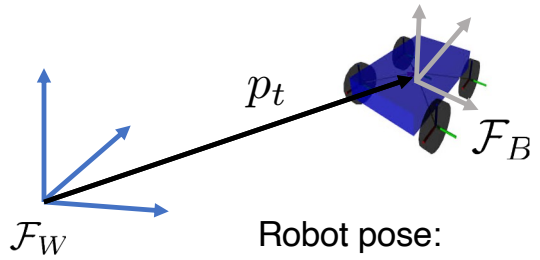
$$R_t \in \mathbb{SO}(2) \text{ or } \mathbb{SO}(3)$$

Definition: $R_t^T R_t = I$ and $\det(R_t) = +1$

All columns mutually orthogonal unit vectors.

Rigid Body Transforms

Robot body in body-fixed frame: $B \subset \mathbb{R}^2$ or $\mathbb{R}^3$



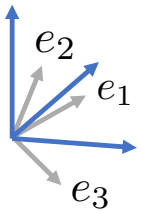
Robot pose:

(p_t, R_t)

Rotation matrix:

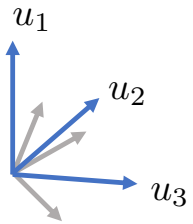
$$R_t = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}$$

e_i expressed in $\mathcal{F}_W$



Inverse transform:

$$R_t = \begin{bmatrix} u_1^T \\ u_2^T \\ u_3^T \end{bmatrix}$$



$$R_t^T = R_t^{-1} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}$$

Transform vector from body to world:

$$v_W = R_t v_B + p_t \quad \text{or} \quad \begin{bmatrix} v_W \\ 1 \end{bmatrix} = \overbrace{\begin{bmatrix} R_t & p_t \\ 0 & 1 \end{bmatrix}}^{T_t} \begin{bmatrix} v_B \\ 1 \end{bmatrix}$$

Inverse transform:

$$v_B = R_t^T v_W - R_t^T p_t \quad \text{or} \quad \begin{bmatrix} v_B \\ 1 \end{bmatrix} = \overbrace{\begin{bmatrix} R_t^T & -R_t^T p_t \\ 0 & 1 \end{bmatrix}}^{T_t^{-1}} \begin{bmatrix} v_W \\ 1 \end{bmatrix}$$

Special Euclidean group (rigid body transforms):

$$(p_t, R_t) \in \mathbb{SE}(2) \text{ or } \mathbb{SE}(3)$$

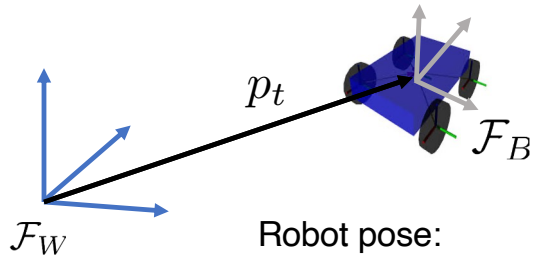
and $(-R_t^T p_t, R_t^T) \in \mathbb{SE}(2)$ or $\mathbb{SE}(3)$

we can also say: $T_t \in \mathbb{SE}(2)$ or $\mathbb{SE}(3)$

and $T_t^{-1} \in \mathbb{SE}(2)$ or $\mathbb{SE}(3)$

Rotation Matrix Parameterizations

Robot body in body-fixed frame: $B \subset \mathbb{R}^2$ or $\mathbb{R}^3$



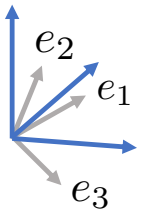
Robot pose:

(p_t, R_t)

Rotation matrix:

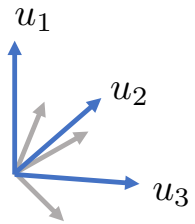
$$R_t = \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}$$

e_i expressed in $\mathcal{F}_W$



Inverse transform:

$$R_t = \begin{bmatrix} u_1^T \\ u_2^T \\ u_3^T \end{bmatrix}$$



Parameterizations:

- Rotation matrix R_t
- Euler angles (ϕ, ψ, θ)
- Quaternions (q_1, q_2, q_3, q_4)
- Angle-axis $(\omega_1, \omega_2, \omega_3)$

All have 3 degrees of freedom.

Must have at least 4 parameters) to represent $\mathbb{SO}(3)$ without singularities.

Rotation matrix:

R_t 9 parameters

$R_t^T R_t = I$ 6 constraints

= 3 degrees of freedom

Robot-Map Collisions

Main point: Use robot pose to represent robot body in world frame to check for **collisions** and to plan **collision free motion.**

Robot body in world frame:

$$B(p_t, R_t) = \{R_t b + p_t \mid b \in B\}$$