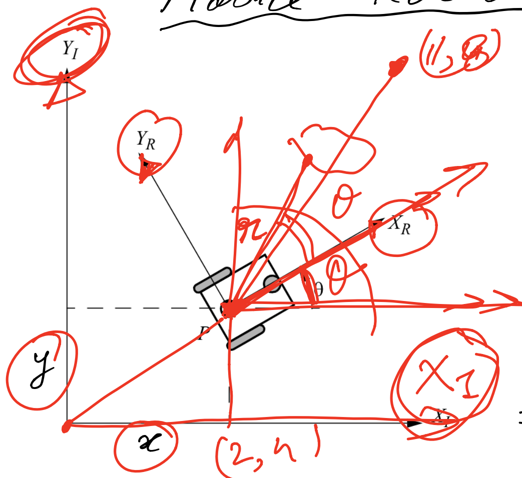
Mobile Robot Kinematics



X_I-Y_I — inertial reference frame

X_R-Y_R — robot reference frame / body frame

Most calculations happen in $X_I - Y_I$

Most measurements happen in $X_R - Y_R$

Pose of the robot

$$\xi_I = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \qquad \dot{\xi}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$$

$\dot{\xi}_R$ $\quad \dot{\xi}_R \quad \dot{\xi}_I$

$$\xi_R = [\ \ ]\ \xi_I$$

$n \times n$

$$\dot{\xi}_R = \underbrace{\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{R(\theta)} \dot{\xi}_I$$

$$\dot{\xi}_R = R(\theta)\,\dot{\xi}_I$$

$$\dot{\xi}_I = R(\theta)^{-1}\,\dot{\xi}_R$$

$R(\theta)$ — Rotation matrix

$R(\theta)$ is an orthogonal matrix: meaning that its rows and columns are orthogonal unit vectors

$$R(\theta)^T R(\theta) = R(\theta) R(\theta)^T = I$$

Useful consequence: $R(\theta)^T = R(\theta)^{-1}$

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

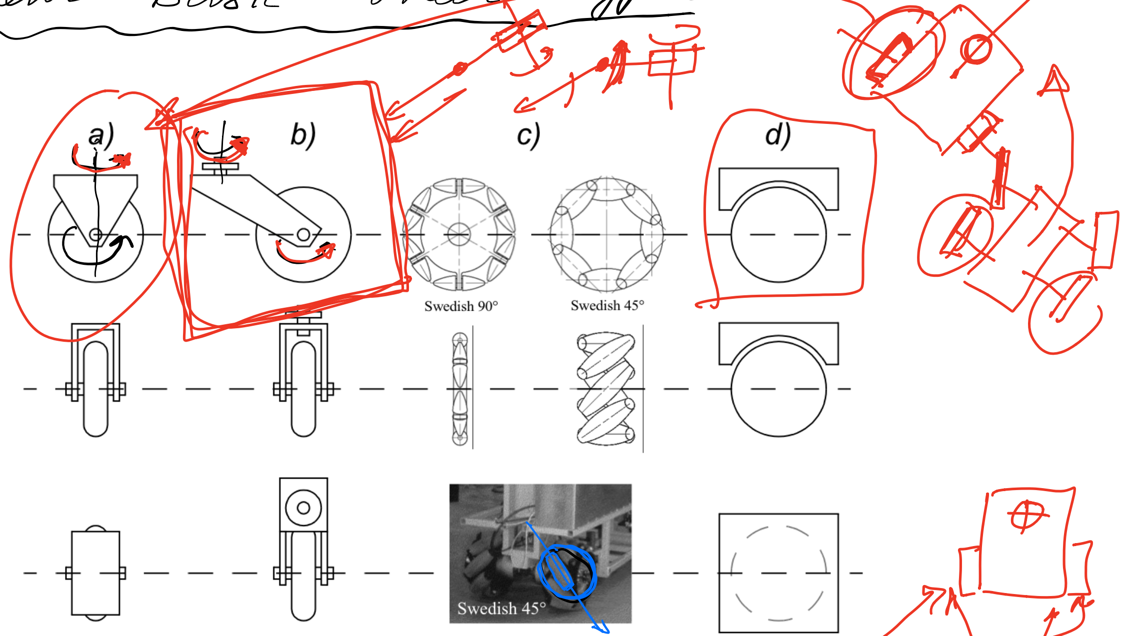
$$\Rightarrow \boxed{\dot{\xi}_I = R(\theta)^T \dot{\xi}_R}$$

Q. What do you mean by non-zero velocity in body frame?

Q. Is the above calculation dependent on the robot type?

Q. How does a robot move given its geometry & speeds of individual wheels?

Four basic wheel types



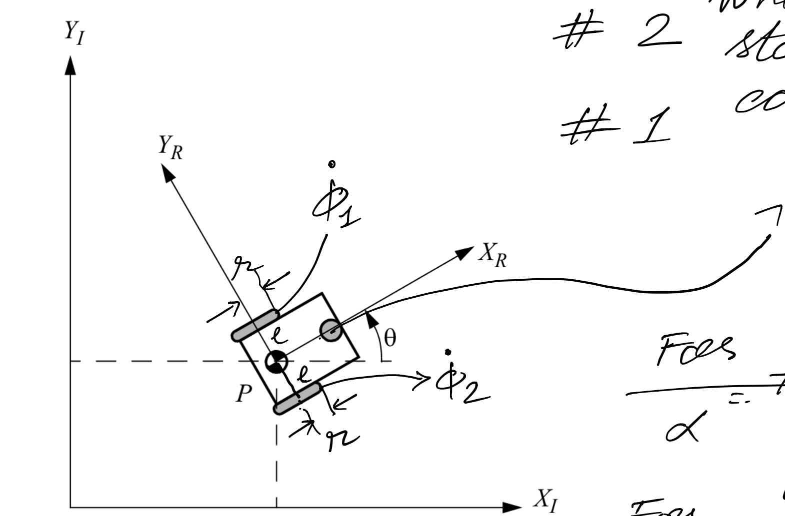
a) Standard Wheel — steered & Unsteered
(2 DOF) (1 DOF)

b) Caster wheel — 2 DOF

c) Swedish Wheel — 3 DOF

d) Ball / Spherical Wheel — difficult to implement

Differential - Drive Robot



\# 2 Wheels are fixed std wheels

\# 1 castor wheel

For wheel 1

$\alpha = \pi/2, \; \beta = 0$

For wheel 2

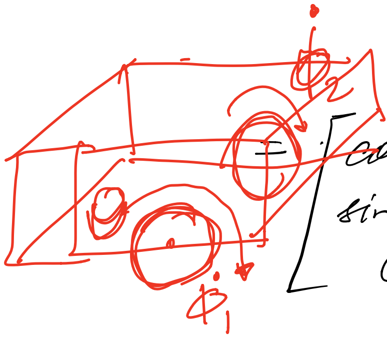
$\alpha = -\pi/2, \; \beta = \pi$

$$\begin{bmatrix} \begin{bmatrix} 1 & 0 & l \\ 1 & 0 & -l \end{bmatrix} \\ \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \end{bmatrix} R(\theta)\,\dot{\xi}_I = \begin{bmatrix} r\dot{\phi}_1 \\ r\dot{\phi}_2 \\ 0 \end{bmatrix}$$

Rolling

Sliding (Both are same)

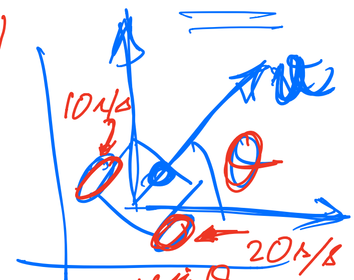
$$\dot{\xi}_I = R(\theta)^{-1}\begin{bmatrix}1 & 0 & l\\ 1 & 0 & -l\\ 0 & 1 & 0\end{bmatrix}^{-1}\begin{bmatrix}r\dot{\phi}_1\\ r\dot{\phi}_2\\ 0\end{bmatrix}$$

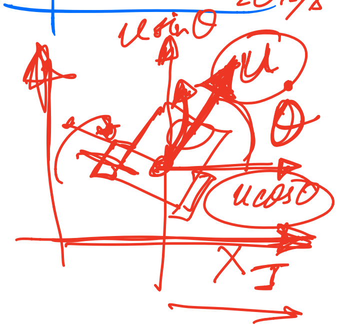


$$= \begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1/2 & 1/2 & 0\\ 0 & 0 & 1\\ \frac{1}{2l} & \frac{1}{2l} & 0\end{bmatrix}\begin{bmatrix}r\dot{\phi}_1\\ r\dot{\phi}_2\\ 0\end{bmatrix}$$

wheel angular velocity

$$\boxed{\begin{bmatrix}\dot{x}_I\\ \dot{y}_I\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\frac{r}{2}(\dot{\phi}_1 + \dot{\phi}_2)\cos\theta\\ \frac{r}{2}(\dot{\phi}_1 + \dot{\phi}_2)\sin\theta\\ \frac{r}{2l}(\dot{\phi}_1 - \dot{\phi}_2)\end{bmatrix}}$$





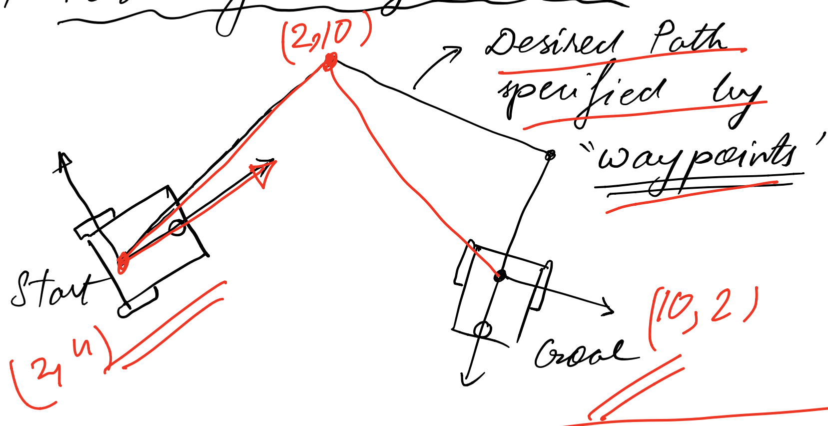
Defining,
$$u = \frac{r}{2}(\dot{\phi}_1 + \dot{\phi}_2)$$
$$\omega = \frac{r}{2l}(\dot{\phi}_1 - \dot{\phi}_2),$$

$$\dot{\xi}_I = \begin{bmatrix}\dot{x}_I\\ \dot{y}_I\\ \dot{\theta}_I\end{bmatrix} = \begin{bmatrix}u\cos\theta\\ u\sin\theta\\ \omega\end{bmatrix}$$ ← familiar unicycle model.

One can solve for $\dot{\phi}_1$ & $\dot{\phi}_2$,

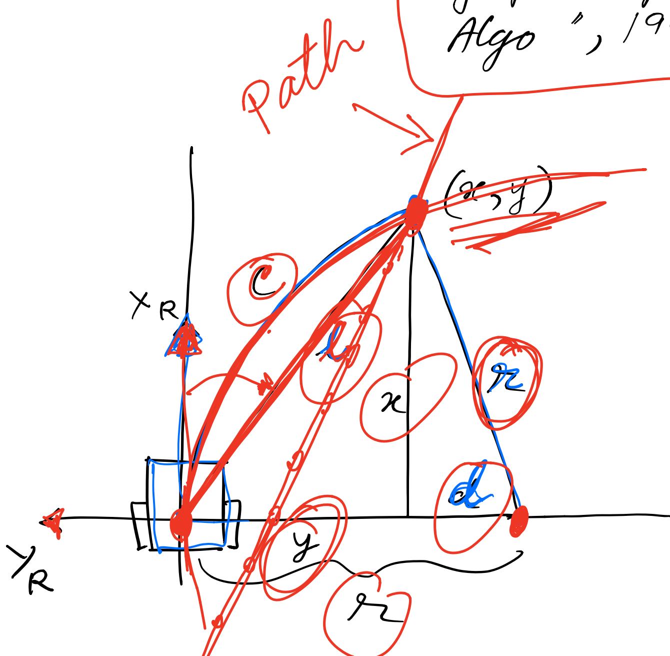
$$\left.\begin{aligned}\dot{\phi}_1 &= \frac{u + \omega l}{r}\\ \dot{\phi}_2 &= \frac{u - \omega l}{r}\end{aligned}\right\}$$

Path following control



Assume path "vector of way-points" is known.

Pure pursuit: (see Craig Coulter "Implementation of pure pursuit Path Tracking Algo", 1992)



Three inputs to the algo
a) linear velocity
b) mean angular velocity
c) look-ahead distance → (l)
d) Vector of ordered waypoints (path)

All calculations in the Robot frame

Step 1: Determine current location

Step 2: Find path point closest to vehicle

Step 3: Find a goal point (x, y) on the path at a distance 'l' from the current location.

lookahead pt

$x_R - y_R$

Step 4: Transform goal point to robot frame

Step 5: Find the curvature of such a curve C s.t. l is the chord-length of C.

$$x^2 + y^2 = l^2$$

$$y + d = r$$

$$d = r - y$$

$$(r-y)^2 + x^2 = r^2$$

$$r^2 - 2ry + y^2 + x^2 = r^2$$

$$2ry = x^2 + y^2 = l^2$$

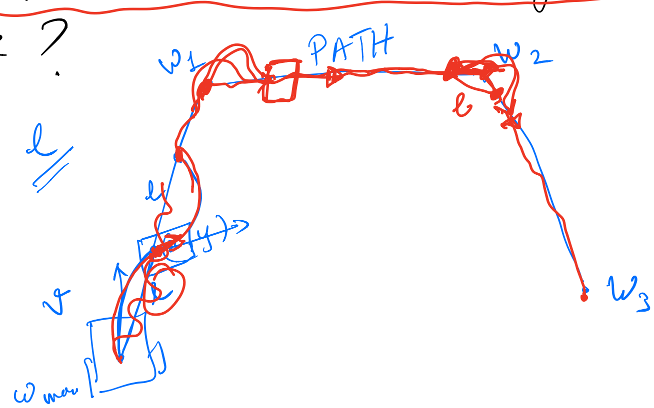
$$\Rightarrow \boxed{r = \frac{l^2}{2x}} ; \boxed{\gamma = \frac{1}{r} = \frac{2x}{l^2}}$$

6) Set steering to the calculated curvature

Exercise

Q. How to drive a diff. drive robot along a given curvature

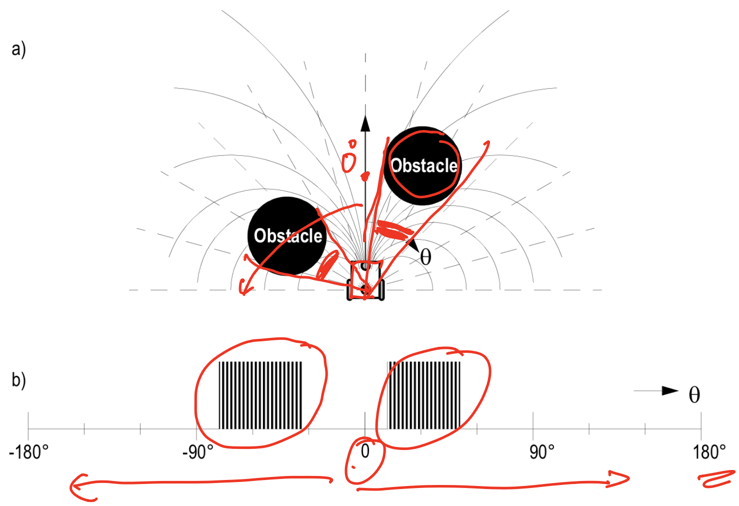
Q. What if the robot cannot follow that curvature?

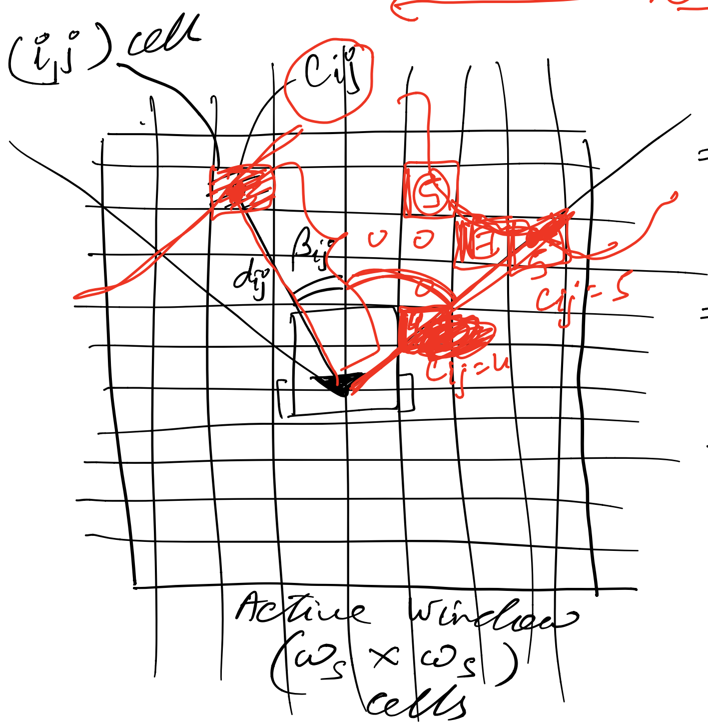


2) Vector Field Histogram

(Borenstein, Koren, IEEE TRA 1991)

At each instant, robot generates (using sensor range readings) a polar histogram





Active window at each instant discretized into grid ($\omega_s \times \omega_s$ cells)

each grid contains a certainty value c_{ij}

c_{ij} indicates the measure of confidence that an obstacle exists within the cell

$c_{ij} = (0,1)$

Algo: 1) Direction of cell is calculated as

$$\boxed{\beta_{ij} = \tan^{-1}\frac{y_j^o - y_0}{x_i^o - x_0}}$$

lidar → range → orient (β_{ij})

(x_0, y_0) — vehicle center, (x_i^o, y_j^o) → active cell coordinate

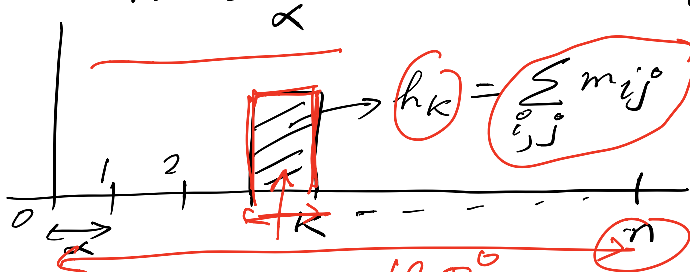
1 ← 98 ← 100 − 1 (distance) 2   99

2) $m_{ij} = (c_{ij})^2 \left(a - b\sqrt{(x_i - x_0)^2 + (y_j - y_0)^2}\right)$

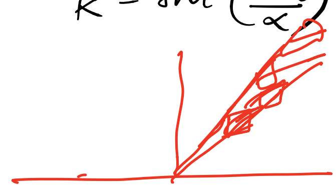
a, b: constants to be tuned

3) Create angular histogram H with α (to be chosen) resolution, s.t.

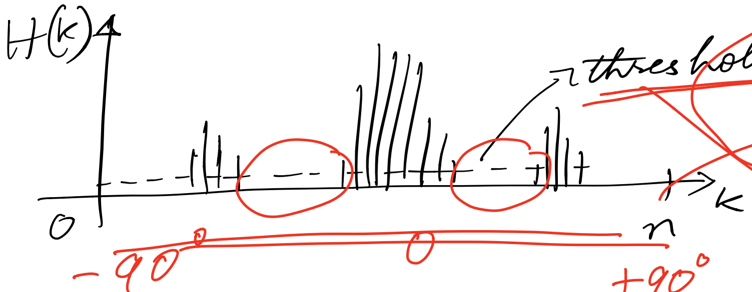
$n = \frac{\cancel{360}\ 180}{\alpha}$ is an integer.

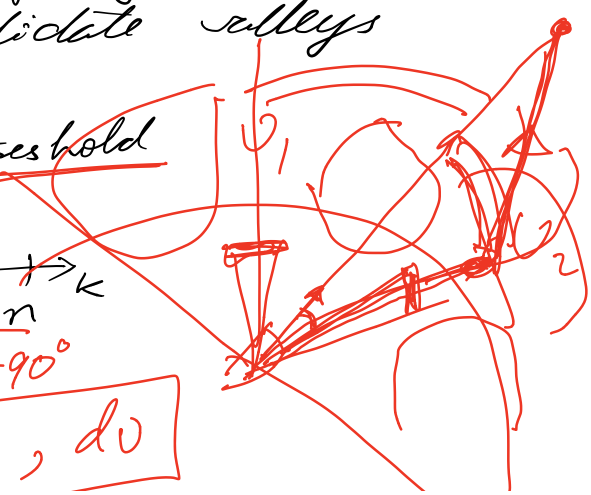


$h_k = \sum_{i,j} m_{ij}^o$ $k = \mathrm{Int}\left(\frac{\beta_{ij}}{\alpha}\right)$



A threshold (user specified) is used to detect the candidate valleys





d_w, d_v

Robot chooses the candidate valley aligned most with target.

Steers through the middle of the chosen valley.