2.12 INTRODUCTION TO ROBOTICS
LECTURE III - PLANAR KINEMATICS
(by Alberto Rodriguez)

Kinematics studies the geometry of motion, without consideration (yet) for the forces that create it.

We have seen Kinematics before, specially in the context of particles and R.B.

- particles $\rightarrow \vec{p}, \vec{v}, \vec{a}$
- rigid bodies $\rightarrow \vec{p}, \vec{v}, \vec{a}, \vec{\omega}, \vec{\alpha}$

Find expressions of these as functions of generalized coordinates and their derivatives.

We will do the same, but applied to the Kinematic structures in robot manipulators.

So you might want to review concepts like generalized coordinates, degrees of freedom, constraints, instantaneous center of rotation, and formulas to describe rigid body motion in moving frames.

Today we will see examples of planar Kinematics, and start with a simple 2-DOF manipulator:

Most manipulators use two types of joints:

- PRISMATIC (displacement noted as "$d"$
- REVOLUTE (displacement noted as "$\theta"$

In general we write them like $\vec{q} = [q_1, ..., q_n]^T$
→ For serial links, the vector $\bar{q}$ of actuator states fully and uniquely determines the manipulator configuration $\rightarrow \bar{q}$ are **GENERALIZED COORDINATES**

**NOTE:** That is not always the case. Some systems are under-actuated, with more generalized coordinates than d.o.f.

- e.g. **DOUBLE PENDULUM $\rightarrow$** we cannot use $\theta$ to know where the mass is.

→ The manipulator configuration is usually specified by the position and orientation of its **END-EFFECTOR**

$$\bar{p}^o = (x, y, \theta) \rightarrow \text{This is what we ultimately want to control because it is where we specify the desired behavior.}$$

→ For this example, let's simplify and say that we only care about $(x, y)$

$$X = l_1 \cos \theta_1 + l_2 \cos (\theta_1 + \theta_2)$$

$$Y = l_1 \sin \theta_1 + l_2 \sin (\theta_1 + \theta_2)$$

This map is the **DIRECT KINEMATICS**. $$\bar{p} = f(\bar{q})$$

- If I tell you $(\theta_1, \theta_2)$ you can tell me where the robot will be $(x, y)$

→ The most useful information, however, is contained in the inverse of $f$.

- I tell you $(x, y)$ and you tell me $(\theta_1, \theta_2)$

This map is the **INVERSE KINEMATICS**.
Apply cosine laws at $\beta$ and $\delta$:

$$r^2 = x^2 + y^2 = l_1^2 + l_2^2 - 2l_1l_2 \cos \beta$$

$$l_2^2 = l_1^2 + r^2 - 2l_1r \cos \delta$$

Then:

$$\theta_2 = \pi - \beta = \pi - \cos^{-1} \left[ \frac{l_1^2 + l_2^2 - (x^2 + y^2)}{2l_1l_2} \right]$$

$$\theta_1 = \alpha - \gamma = \tan^{-1} \left[ \frac{y}{x} \right] - \cos^{-1} \left[ \frac{l_1^2 - l_2^2 + (x^2 + y^2)}{2l_1 \sqrt{x^2 + y^2}} \right]$$

→ It is simple enough that can be programmed real time in a simple microcontroller, like Arduino. FUN!!

→ Now if I give you $(x, y)$ can you tell me $(\theta_1, \theta_2)$? Not quite

1. **Multiplicity of solutions**

→ I also need to tell you which one I want.

→ The solutions in this case satisfy:

$$\begin{cases}
\theta_2' = -\theta_2 & \text{Simple in this case.} \\
\theta_1' = \theta_1 + 2\delta & \text{I only need to say a boolean to specify which one.}
\end{cases}$$

→ Gets more complicated with more D.O.F.s.

e.g. for a puma robot → Elbow UP/DOWN

\begin{align*}
\text{Shoulder RIGHT/LEFT} & \quad 8 \text{ solutions.} \\
\text{Forearm RIGHT/LEFT} & \\
\end{align*}

→ The extreme case is redundant manipulators (very common these days)

- Infinite solutions.
- We will talk more about them.
(2) **REACHABLE WORKSPACE:**

- Solutions do not always exist.

- **WORKSPACE**: Range of reachable manipulator configurations

- **DEXTEROUS WORKSPACE**: Range of reachable configurations while retaining freedom in the orientation of the end-effector.

(3) **COMPLICATED SOLUTIONS:**

- It gets complicated easily. Trigonometric algebra is not fun.
- The structure of the mechanism is designed to simplify it.

  e.g.:- coordinate robots from the past.
  - pneumatic robot: 3 DOF for position + 3 DOF at the wrist for orientation.

- Recently we see robots with less optimized kinematics, which is possible because computers are fast at finding numerical solutions.

(4) **TRAJECTORIES:**

- Let's suppose we have the mapping \((x, y) \rightarrow (\theta_1, \theta_2)\)
  how do we move the robot there?

  - PID loop on each axis

- But the trajectory might be weird in cartesian space.

- What if we want it to go in a straight line?
Differential Motion,

→ Kinematics also concerned with \( \dot{\mathbf{r}} \) and \( \ddot{\mathbf{r}} \)
   
   * for control in cartesian space
   * for dynamics

→ Recall from the 2-dof manipulator:

\[
\begin{align*}
    x &= l_1 \cos \theta_1 + l_2 \cos (\theta_1 + \theta_2) = x(\theta_1, \theta_2) \\
    y &= l_1 \sin \theta_1 + l_2 \sin (\theta_1 + \theta_2) = y(\theta_1, \theta_2)
\end{align*}
\]

→ Down-converting \( \rightarrow \) Differentiating

\[
\begin{align*}
    \frac{dx}{dt} &= \frac{2x}{\partial \theta_1} \frac{d\theta_1}{dt} + \frac{2x}{\partial \theta_2} \frac{d\theta_2}{dt} \\
    \frac{dy}{dt} &= \frac{2y}{\partial \theta_1} \frac{d\theta_1}{dt} + \frac{2y}{\partial \theta_2} \frac{d\theta_2}{dt}
\end{align*}
\]

which we can express in matrix form as:

\[
\begin{bmatrix}
    \frac{dx}{dt} \\
    \frac{dy}{dt}
\end{bmatrix} =
\begin{bmatrix}
    \frac{2x}{\partial \theta_1} & \frac{2x}{\partial \theta_2} \\
    \frac{2y}{\partial \theta_1} & \frac{2y}{\partial \theta_2}
\end{bmatrix}
\begin{bmatrix}
    \frac{d\theta_1}{dt} \\
    \frac{d\theta_2}{dt}
\end{bmatrix}
\]

(\text{or} \quad \dot{\mathbf{r}} = \mathbf{J} \cdot \dot{\mathbf{q}})

→ Linear map between velocities in joints and velocity in cartesian space

* We note it as \( \mathbf{J} \), the Jacobian Matrix.

→ In our case

\[
\mathbf{J} =
\begin{bmatrix}
    -l_1 \sin \theta_1 - l_2 \sin (\theta_1 + \theta_2) & -l_2 \sin (\theta_1 + \theta_2) \\
    l_1 \cos \theta_1 + l_2 \cos (\theta_1 + \theta_2) & l_2 \cos (\theta_1 + \theta_2)
\end{bmatrix}
\]

NOTES:

1. \( \mathbf{J} \) depends on the configuration \( \mathbf{J} = \mathbf{J}(\theta_1, \theta_2) = \mathbf{J}(q) \)

2. Each column of \( \mathbf{J} \) points us to a different freedom
- We can move the robot along any linear combination of $\mathbf{J}_1$ and $\mathbf{J}_2$.
- Now, what can we do if we want to move along a given $\dot{\mathbf{y}}$?

$$\dot{\mathbf{q}} = \mathbf{J}^{-1} \dot{\mathbf{y}} \quad \text{YES, but NO}$$

- $\mathbf{J}^{-1}$ might not exist, and if it does, it might have other problems.
- We will go more in depth into differential inverse kinematics later.

**Non-holonomic systems**

- These are systems for which there is no kinematic mapping.
- We mentioned the case of the inverted pendulum. We will see now the case of a 2-DOF planar vehicle like the one in the lab.

$$\vec{p} = (x, y, \theta) \rightarrow 3 \text{ D.O.F.}$$
$$\vec{q} = (\theta_L, \theta_R) \rightarrow 2 \text{ D.O.F.}$$

Under-actuated system.
→ Suppose we drive the car with these actuator trajectories:

![Graphs showing actuator trajectories with time t and angles θR and θL.](image)

**Outcome:**

![Diagram showing motion from one point to another.](image)

→ In both cases, the final configuration of the actuators is \((θ_R, θ_L) = (90°, 90°)\), but the resultant trajectory is very different.

→ We cannot hope to find direct kinematic expressions, unless, for example, if we keep track of all the trajectories of the actuators.

\[
θ_R(t), θ_R'(t), θ_L(t), θ_L'(t) \forall 0 ≤ t ≤ T \implies (x(t), y(t))
\]

DEAD-RECKONING.

→ Let's try to build first the Jacobian:

If we fix \(θ_L (θ, > 0)\) and change \(θ_R\) for a small change \(Δθ_R\):

\[
Δx = \frac{1}{2} r Δθ_R, \quad Δy = 0
\]

\[
\tan Δy = \frac{r Δθ_R}{2b} \approx \sin Δy \sim Δy
\]
→ Then \[
\begin{bmatrix}
\Delta x \\
\Delta y \\
\Delta \psi
\end{bmatrix} = \begin{bmatrix}
\frac{r}{2} \\
0 \\
-\frac{v}{2b}
\end{bmatrix} \Delta \theta_R
\]
similarly \[
\begin{bmatrix}
\Delta x \\
\Delta y \\
\Delta \psi
\end{bmatrix} = \begin{bmatrix}
\frac{r}{2} \\
0 \\
-\frac{v}{2b}
\end{bmatrix} \Delta \theta_L
\]
\[
\theta \text{ R fixed }
\]
\[
\theta \text{ L fixed }
\]

→ If we divide by \( \Delta t \) and combine them:
\[
\begin{bmatrix}
x \\
y \\
\psi
\end{bmatrix} = \begin{bmatrix}
\frac{r}{2} & 0 \\
0 & 0 \\
\frac{v}{2b} & -\frac{v}{2b}
\end{bmatrix} \begin{bmatrix}
\dot{\theta}_R \\
\dot{\theta}_L
\end{bmatrix}
\]

Jacobian Matrix \( J_T \)

→ Note that there are two interesting actuator velocities:
\[
q = [\theta_R \theta_L]^T = (1, 1)^T \rightarrow \text{MOVES FORWARD} \quad \dot{p} = (r, 0, 0)^T
\]
\[
= (1, -1)^T \rightarrow \text{ROTATES in PLACE} \quad \dot{p} = (0, 0, r_b)^T
\]

→ How do we recover the pose of the vehicle?

- Define inertial reference frame \( \mathcal{A} \) \((x, y, \psi)\)
- Moving body frame \( \mathcal{B} \) \((x, y, \psi)\)

\[
\begin{align*}
\text{In body frame} & \quad \begin{cases}
x(t) = \frac{r}{2} (\dot{\theta}_R(t) + \dot{\theta}_L(t)) \\
\dot{y}(t) = 0 \\
\dot{\psi}(t) = \frac{r}{2b} (\dot{\theta}_R(t) - \dot{\theta}_L(t))
\end{cases}
\end{align*}
\]

→ We want to recover the values \((x(T), y(T), \psi(T))\)

→ When the vehicle has followed a known trajectory in actuator space

\[
\begin{align*}
\text{In body frame} & \quad \begin{cases}
x(t) = \frac{r}{2} (\dot{\theta}_R(t) + \dot{\theta}_L(t)) \\
\dot{y}(t) = 0 \\
\dot{\psi}(t) = \frac{r}{2b} (\dot{\theta}_R(t) - \dot{\theta}_L(t))
\end{cases}
\end{align*}
\]
• In the inertial frame:

\[ \begin{align*}
\dot{X}(t) &= \dot{x}(t) \cos \psi(t) = \frac{r}{2} \left( \dot{\theta}_R(t) + \dot{\theta}_L(t) \right) \cos \psi(t) \\
\dot{Y}(t) &= \dot{x}(t) \sin \psi(t) = \frac{r}{2} \left( \dot{\theta}_R(t) + \dot{\theta}_L(t) \right) \sin \psi(t) \\
\dot{\psi}(t) &= \dot{\psi}(t) = \frac{r}{2b} \left[ \dot{\theta}_R(t) - \dot{\theta}_L(t) \right]
\end{align*} \]

System of Differential Equations.

• We know the initial condition \( X(t=0) = Y(t=0) = \psi(t=0) = 0 \)

then:

\[ \begin{align*}
X(T) &= \int_0^T \dot{X}(t) \, dt = \frac{r}{2} \int_0^T \left( \dot{\theta}_R(t) + \dot{\theta}_L(t) \right) \cos \psi(t) \, dt \\
Y(T) &= \int_0^T \dot{Y}(t) \, dt = \frac{r}{2} \int_0^T \left( \dot{\theta}_R(t) + \dot{\theta}_L(t) \right) \sin \psi(t) \, dt \\
\psi(T) &= \int_0^T \dot{\psi}(t) \, dt = \frac{r}{2b} \int_0^T \left( \dot{\theta}_R(t) - \dot{\theta}_L(t) \right) \, dt = \\
&= \frac{r}{2b} \left[ \int_0^T \dot{\theta}_R(t) \, dt - \int_0^T \dot{\theta}_L(t) \, dt \right]
\end{align*} \]

→ The orientation can be fully determined from the current state of the actuators. → KEY PROPERTY

→ It allows to solve the set of differential equations without effort.

→ Then:

\[ \begin{align*}
X(T) &= \frac{r}{2} \int_0^T \left( \dot{\theta}_R(t) + \dot{\theta}_L(t) \right) \cos \left( \frac{r}{2b} \left( \theta_R(t) - \theta_L(t) \right) \right) \, dt \\
Y(T) &= \frac{r}{2} \int_0^T \left( \dot{\theta}_R(t) + \dot{\theta}_L(t) \right) \sin \left( \frac{r}{2b} \left( \theta_R(t) - \theta_L(t) \right) \right) \, dt
\end{align*} \]

→ Can be computed numerically based on a history of wheel trajectories.