上海科技大学
ShanghaiTech University

CS283: Robotics Spring 2023: Perception & Robot Arms

Sören Schwertfeger / 师泽仁

ShanghaiTech University

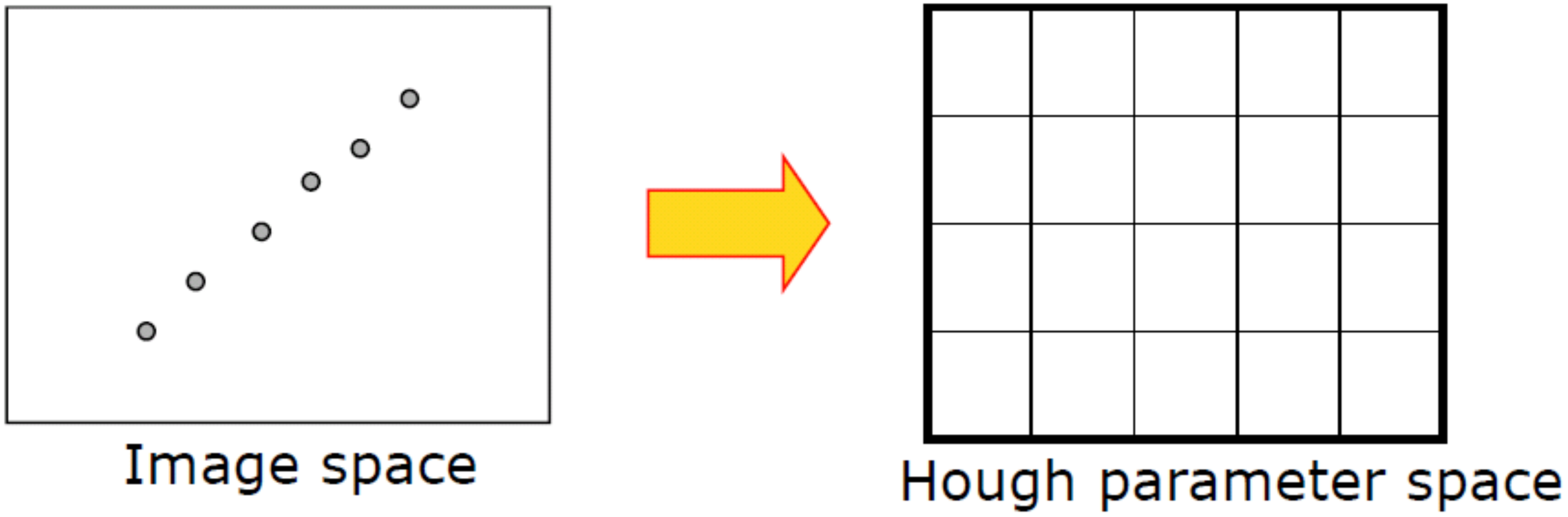
Outline

- Hough Transform
- Admin
- Arm links & joints
- Kinematics
- Arm Planning
- MoveIt & Grasping

RANSAC & HOUGH TRANSFORM

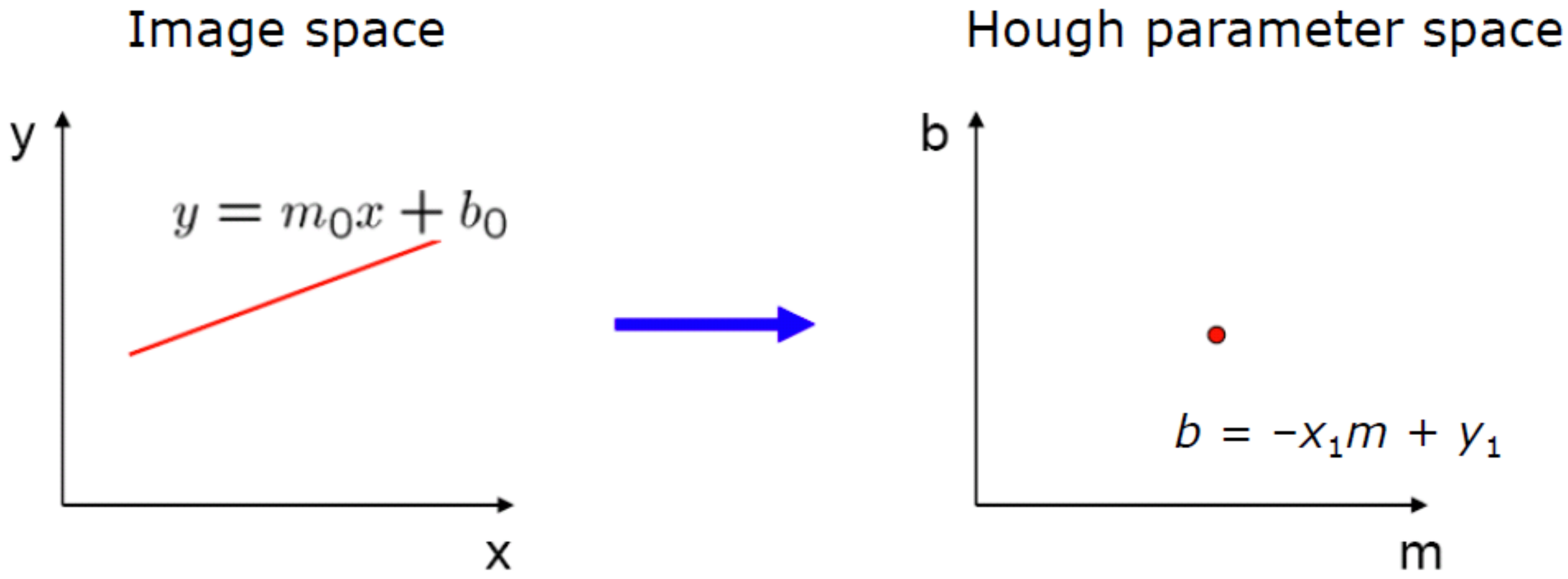
Algorithm 4: Hough-Transform

- Hough Transform uses a voting scheme



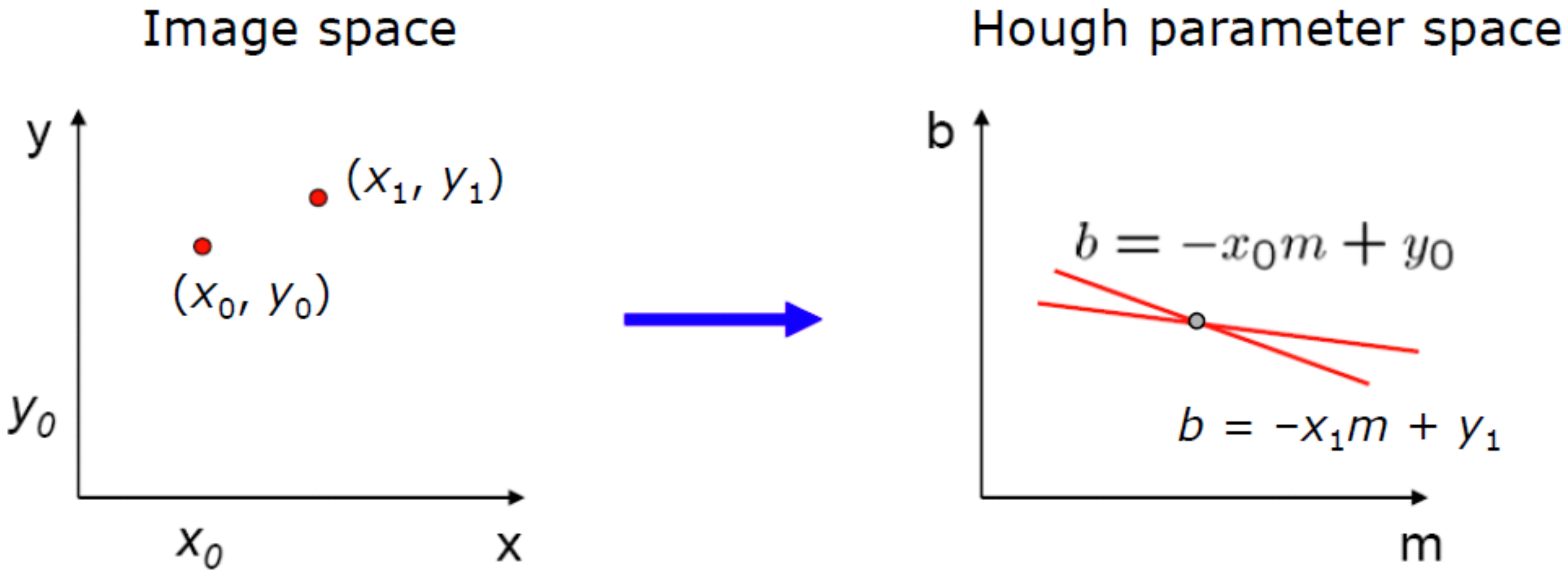
Algorithm 4: Hough-Transform

- A line in the image corresponds to a point in Hough space



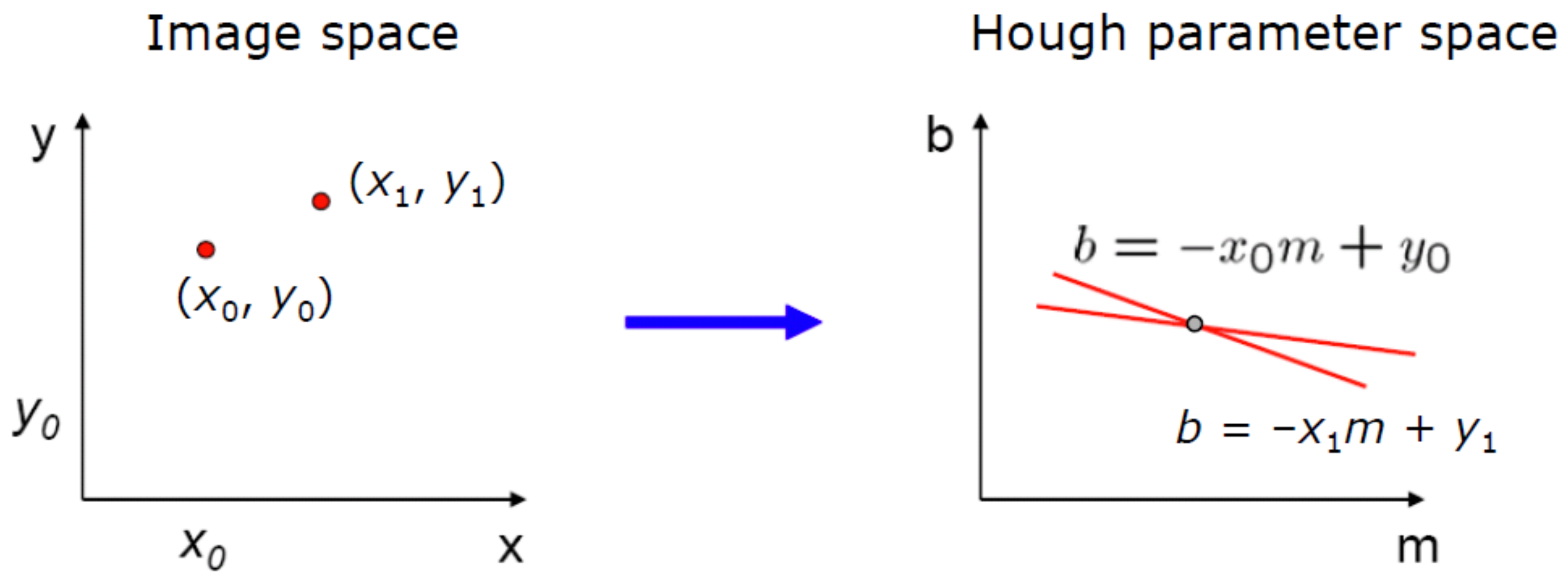
Algorithm 4: Hough-Transform

- What does a point (x_0, y_0) in the image space map to in the Hough space?



Algorithm 4: Hough-Transform

- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?
  - It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$



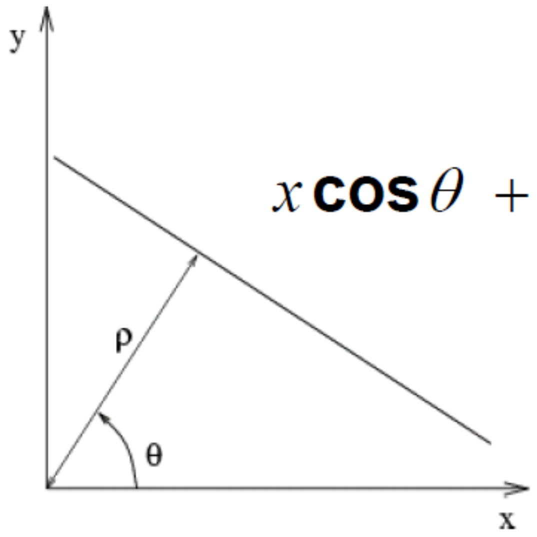
Algorithm 4: Hough-Transform

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m

Algorithm 4: Hough-Transform

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m

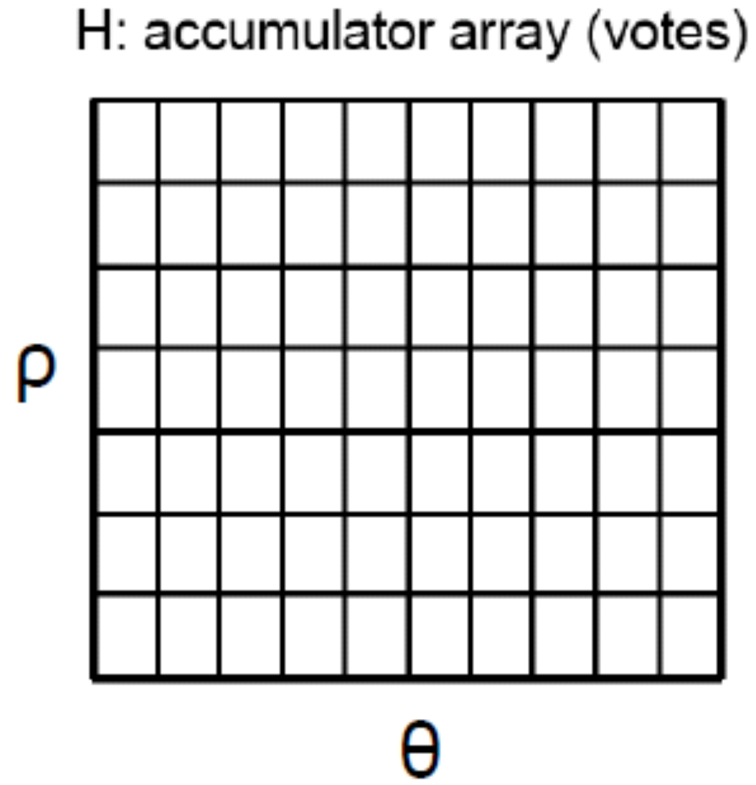
- Alternative: polar representation



$$x \cos\theta + y \sin\theta = \rho$$

Each point will add a sinusoid in the (θ,ρ) parameter space

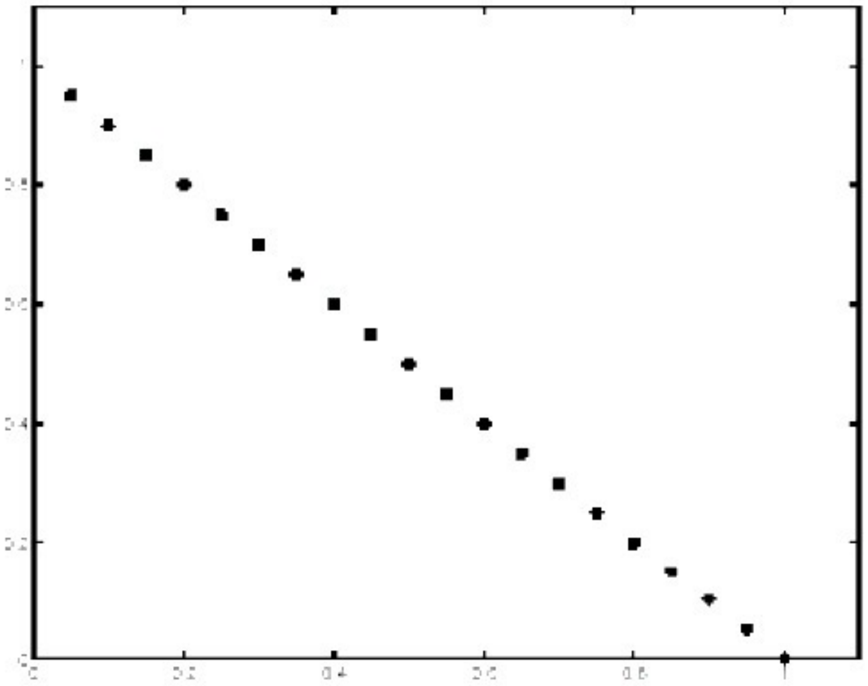
Algorithm 4: Hough-Transform



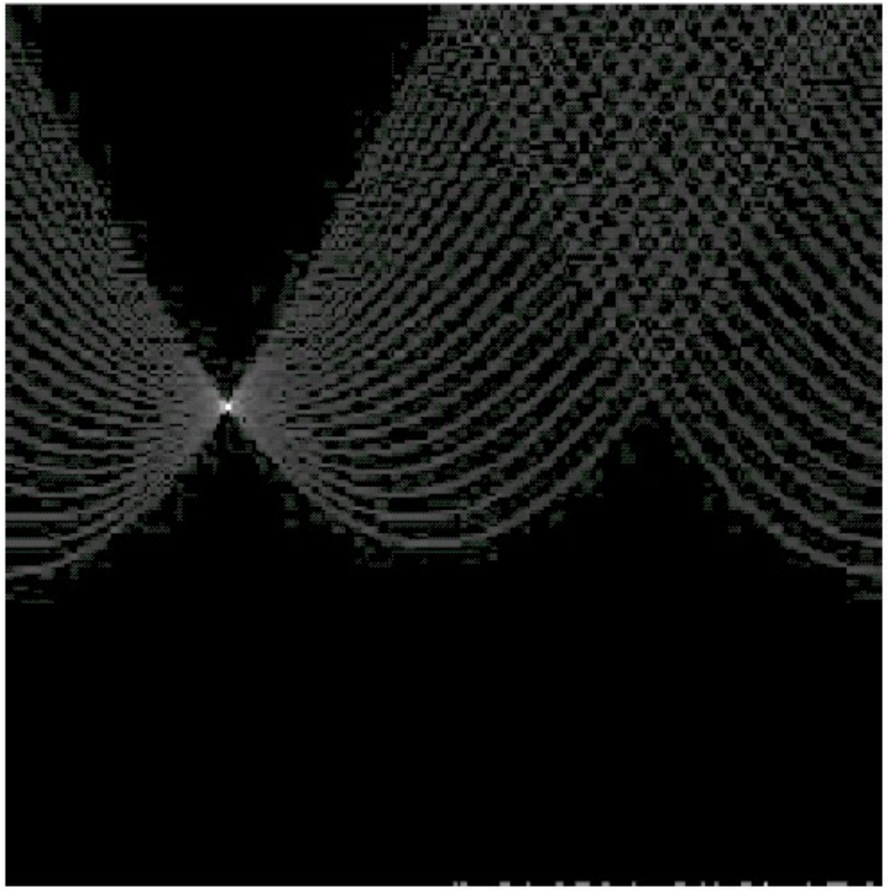
1. Initialize accumulator H to all zeros
2. For each edge point (x,y) in the image
   - For θ = 0 to 180 (with a step size of e.g. 18)
     - $\rho = x \cos\theta + y \sin\theta$
     - $H(\theta, \rho) = H(\theta, \rho) + 1$
   - end

   end
3. Find the values of (θ, ρ) where H(θ, ρ) is a local maximum
4. The detected line in the image is given by $\rho = x \cos\theta + y \sin\theta$

Algorithm 4: Hough-Transform



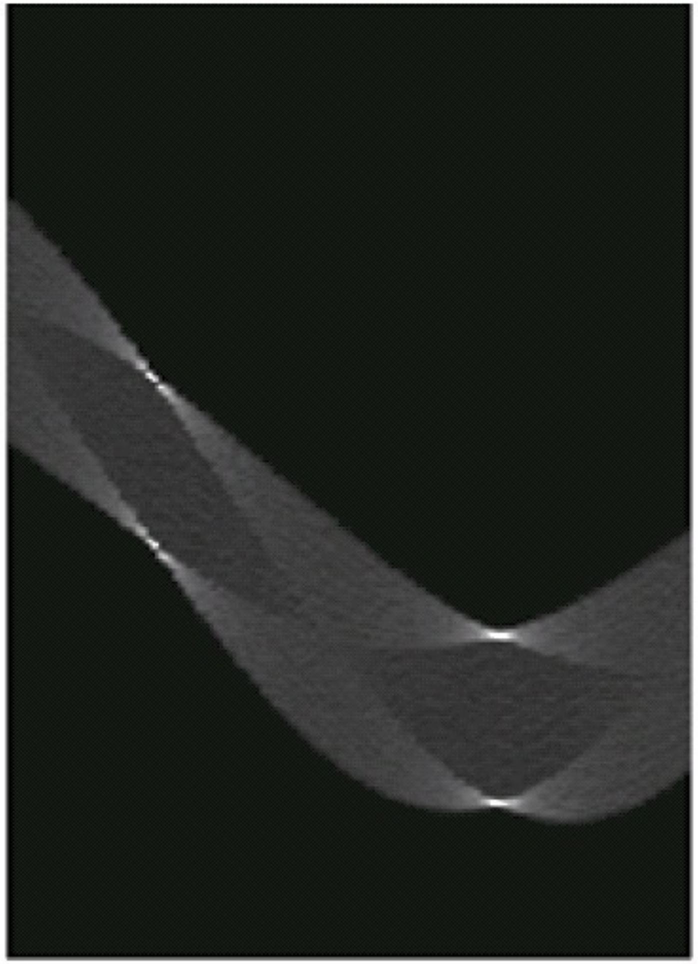
features



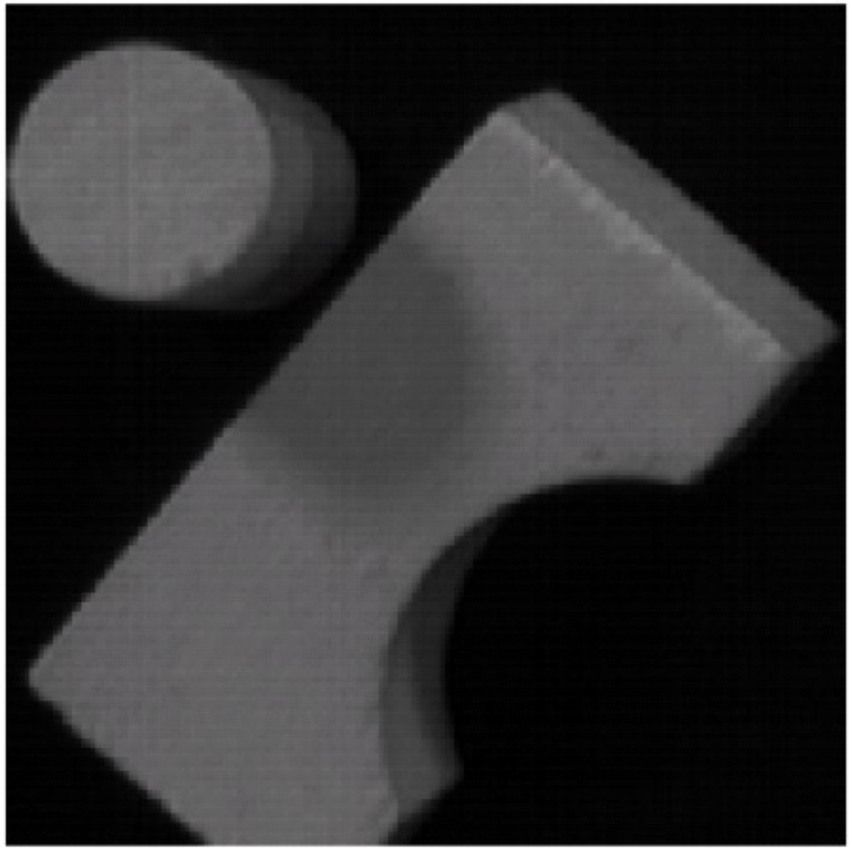
votes

Algorithm 4: Hough-Transform

Square

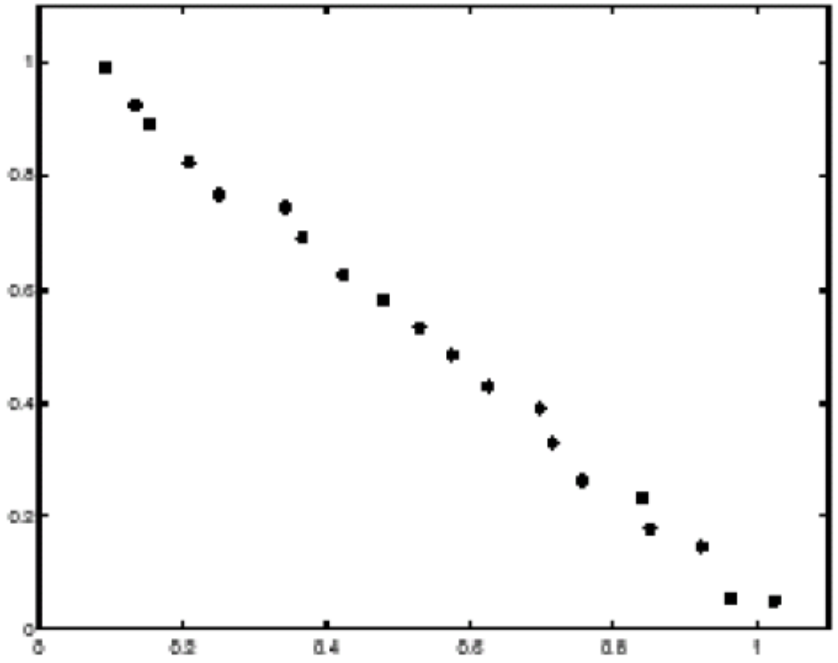


Algorithm 4: Hough-Transform

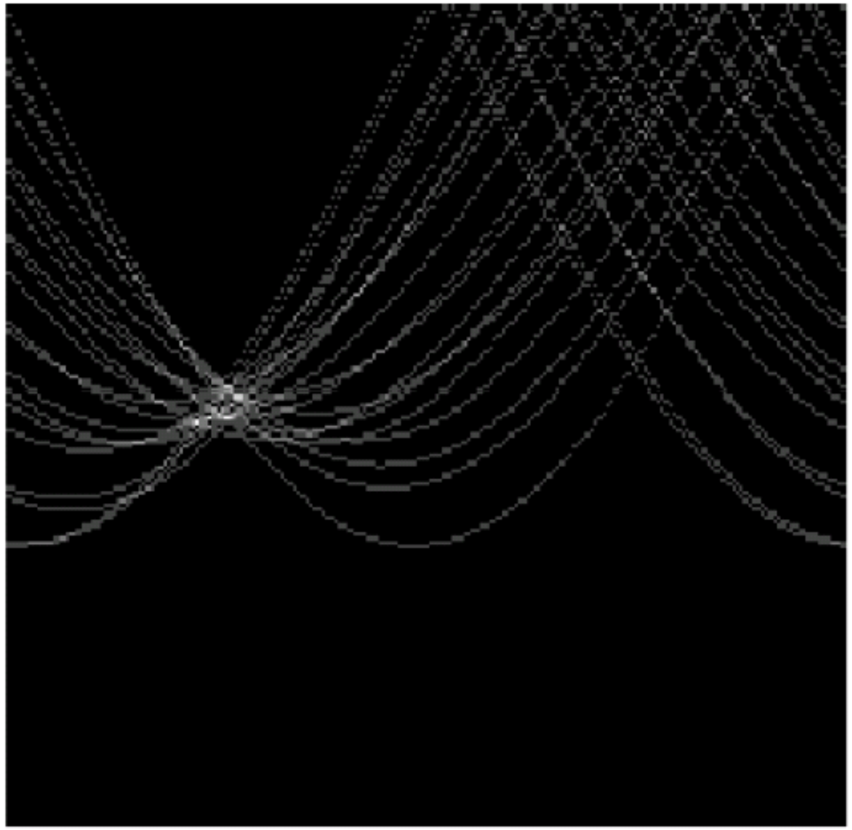


Algorithm 4: Hough-Transform

Effect of Noise



features

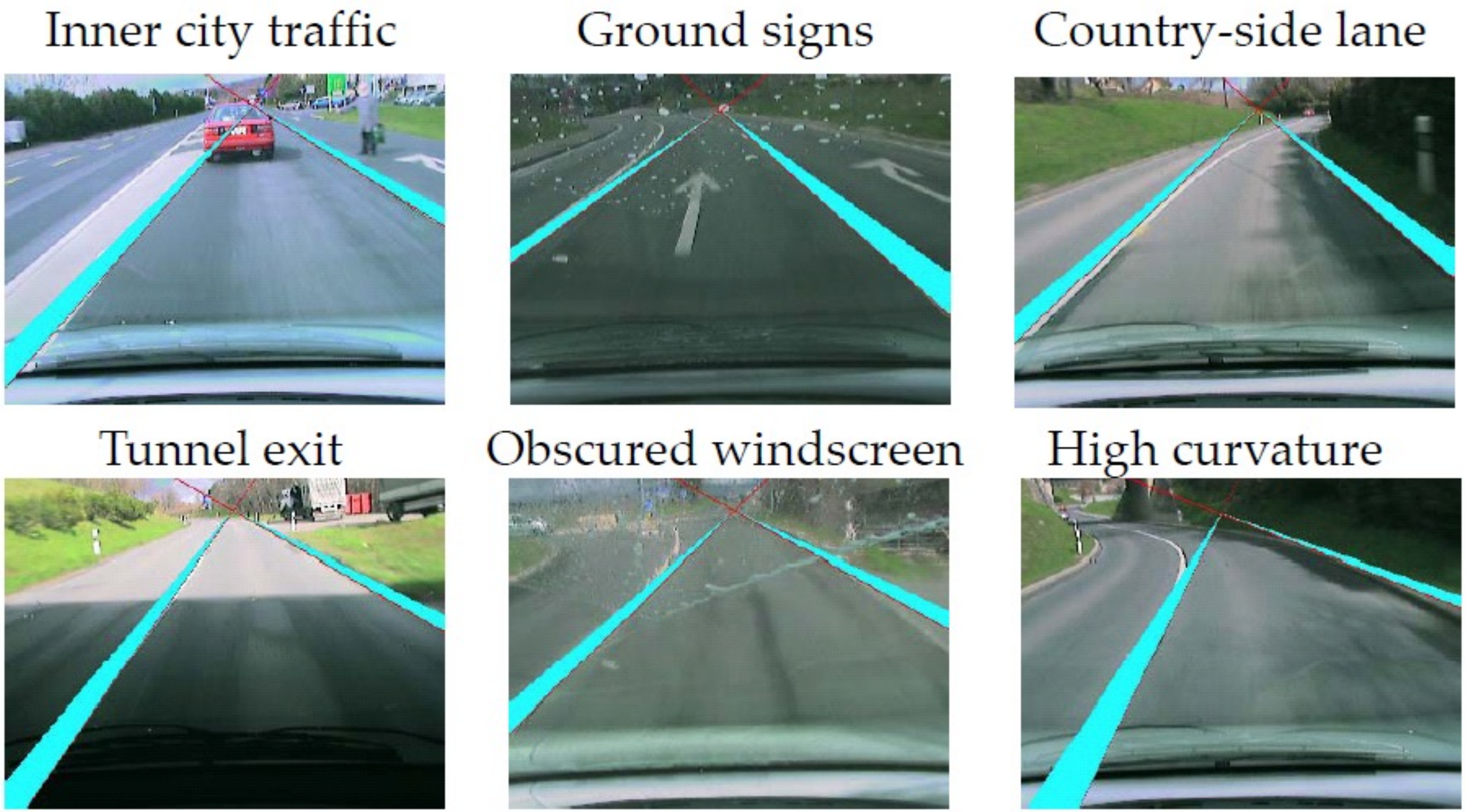


votes

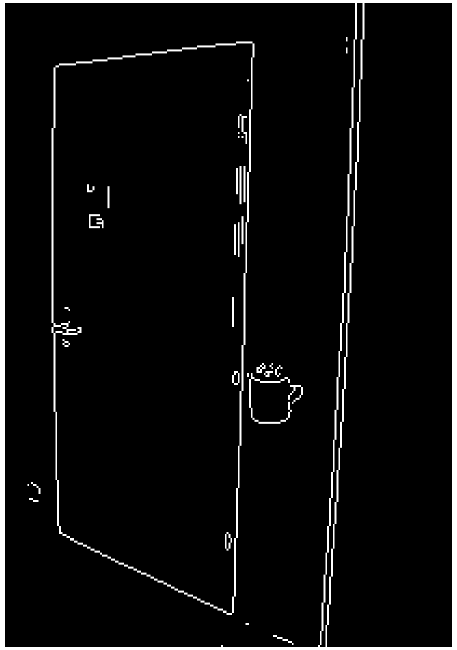
- Peak gets fuzzy and hard to locate

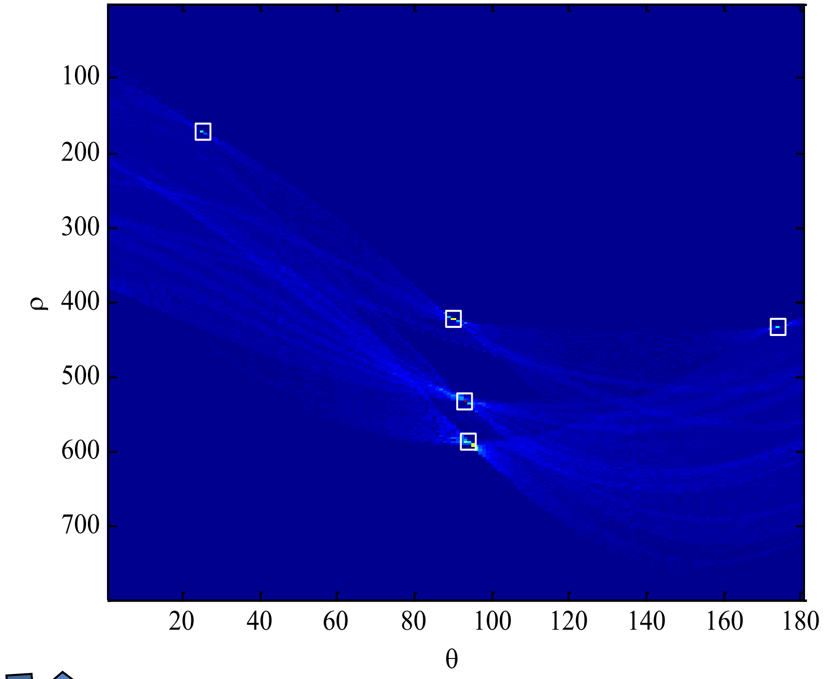
Algorithm 4: Hough-Transform

Application: Lane detection

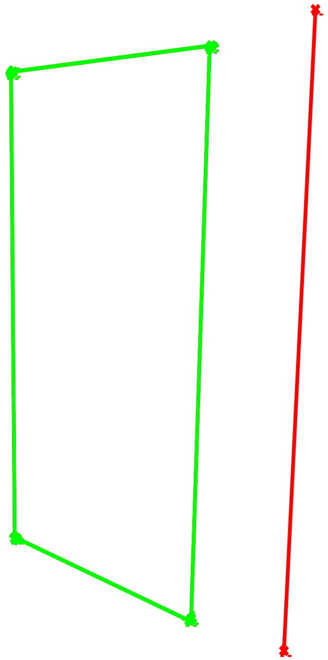


Example – Door detection using Hough Transform





Hough Transform



Hough Transform: other features

Lines:

$$p = (d, \upsilon)$$

$$g(x,y,p) := x \cdot \cos(\upsilon) + y \cdot \sin(\upsilon) - d$$

Circles:

$$p = (x_0, y_0, r)$$

$$g(x,y,p) := (x - x_0)^2 + (y - y_0)^2 - r^2$$

Ellipses:

$$p = (x_0, y_0, a, b, \psi)$$

$$g(x,y,p) := \frac{\left[(x - x_0)\cdot\cos(\psi) + (y - y_0)\cdot\sin(\psi)\right]^2}{a^2} + \frac{\left[(y - y_0)\cdot\cos(\psi) - (x - x_0)\cdot\sin(\psi)\right]^2}{b^2} - 1$$

Hough Transform

- Advantages
  - Noise and background clutter do not impair detection of local maxima
  - Partial occlusion and varying contrast are minimized
- Negatives
  - Requires time and space storage that increases exponentially with the dimensions of the parameter space

Comparison **Line Detection**

- Deterministic methods perform better with laser scans
  - Split-and-merge, Line-Regression, Hough transform
  - Make use of the sequencing property of scan points.
- Nondeterministic methods can produce high False Positives
  - RANSAC
  - Does not use the sequencing property
  - But it can cope with outliers
- Overall:
  - Split-and-merge is the fastest, best real-time application

ADMIN

Project Meetings

- Make an appointment with your student advisor, Prof. and all members to meet next week!
  - Failure goes directly to “meetings” part of project grade.

- Afterwards meet every week with your advisor
  - There should be some progress every week
  - Notes will be made in a meeting.txt every week by your advisor.

Presentation

- Choose one paper from ICRA or IROS which is relevant to your project!
  - ICRA: https://ieeexplore.ieee.org/xpl/conhome/1000639/all-proceedings (any year!)
  - IROS: https://ieeexplore.ieee.org/xpl/conhome/1000393/all-proceedings (any year!)
  - Only full papers (6 or more pages) are allowed; no workshop papers

- Present the paper as if it were your own work!
- Front page: Name of the Paper; Full citation of the paper; **Your name in Pinyin; Your email address**
- Last slide: **ONE** slide about how this paper is relevant to your project.
- Your presentation has to be professional – not cute…
- Submit pdf or ppt to the paper repository till Wednesday, March 15 22:00 to repo! Late submissions (or if you come with the ppt/ pdf to the presentation time) will receive a flat 33% loss of points!
- Presentations in 4 slots – during lecture slots March 21 & 23.
- 10 minute presentation plus 1 minute project relevance plus 4 minutes questions
  - Do not rush your presentation! Better present less items more slowly!
  - 10 minute presentation => 8 – max. 12 slides
  - Maybe have a slide towards the end that you can skip if you run out of time.
  - Give a test presentation to your friends beforehand!
- Finish early for practicing – don’t learn by heart.

Grading of the Presentation

- 10 %: Your basic understanding/ knowledge about the paper you present
- 20 %: Presentation timing (plus or minus one minute is ok) – no rushing – good speed!
- 10 %: Correct written English in presentation:
  - No complete sentences, no grammatical or spelling mistakes
- 10 %: Good structure of presentation:
  - Depends on the type of paper, how much time you have, how long you need to present the main achievement.
  - For example: outline, introduction/ motivation, **problem statement**, state of the art, **approach**, experiments, **results**, **conclusion**, outlook
- 20 %: Clarity of written presentation
- 10 %: Good presentation style:
  - Interact with audience: look at the whole room (not just your slides, notes, or the back of the room)
  - Present the paper – do not read (or repeat the learned) speech from a prepared text
  - Use the presentation as visual aid – not as your tele-prompter to read from
  - Move your body – do not stand frozen at one place
- 10 %: Answering the questions
  - Questions have to be asked and answered in English – Chinese can be used for clarification
- 10 %: Asking questions to other students!
- **Not** scored: Your English skill

Project Proposal (1)

- **Title:** Find a nice, catchy title for your project
- **Abstract**: A short abstract/ summary what the project is about
- **Introduction**: general description & Motivation
- **State of the Art**: Literature & open-source-ROS packages
- Per team member:
  - present and cite three papers with just three or four sentences
  - present in more detail one further paper relevant to your project. Describe it with at least 1/3rd of a page.
  - present in detail one open source ROS package relevant to your project. At least 1/3rd of a page
  - => about one page per team member – => 3 pages for 3 person team

Project Proposal (2)

- **System Description**
- **System Evaluation**: Describe how you want to test your system.
  - Experiments & how to measure their success
- **Work Plan**: Define some mile stones.
  - Possible phases: Algorithm design, implementation, testing, evaluation, documentation – some of those things can also happen in a loop (iteration).
  - Deliverables of Project:
    - Proposal (this document)
    - Mid-term report
    - Final demo
    - Final Report
    - Website
- **Conclusions**: Short summary and conclusions

Project Proposal (3)

- Important dates:
- **March 10, 22:00: due date for the proposal**
- **April 24: due date for mid-term report**
- **May 25 – June 6 (tbd): due date for the final report.**

- Parts of proposal go into the final project report.
- Please don’t forget to take **pictures and videos** when testing your system!

- In English! Using LaTeX!
- Put sources and PDF in git.

- **Additional task:** In glit/ gitlab: “Readme.txt” with:
  - Team Name and Members; email addresses
  - Documentation and how to’s regarding your project.

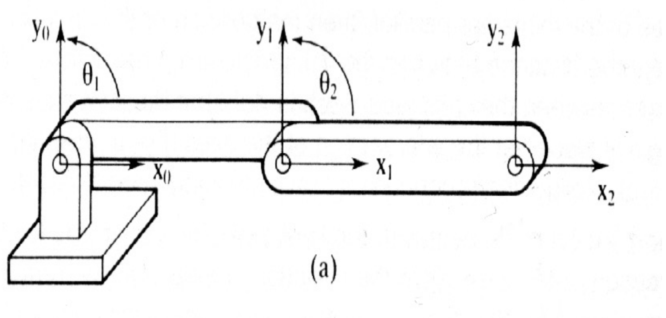
ROBOT ARMS

Robot Arm

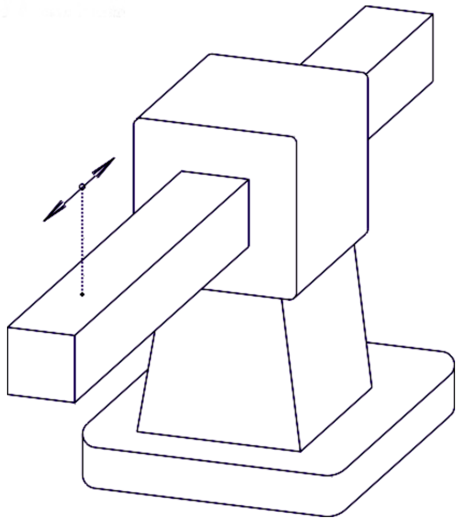
- Consists of Joints and Links …
- and a Base and a Tool (or End-Effector or Tip)

Joints

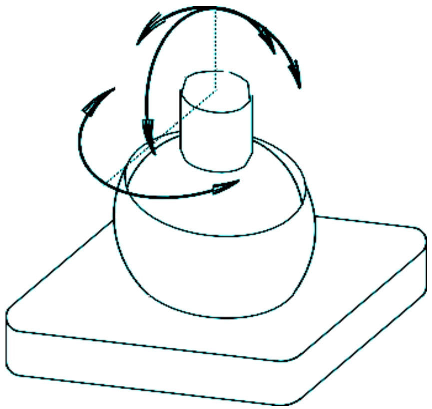
- Revolute Joint: 1DOF



- Prismatic Joint/ Linear Joint: 1DOF



- Spherical Joint: 3DOF



Note on Joints

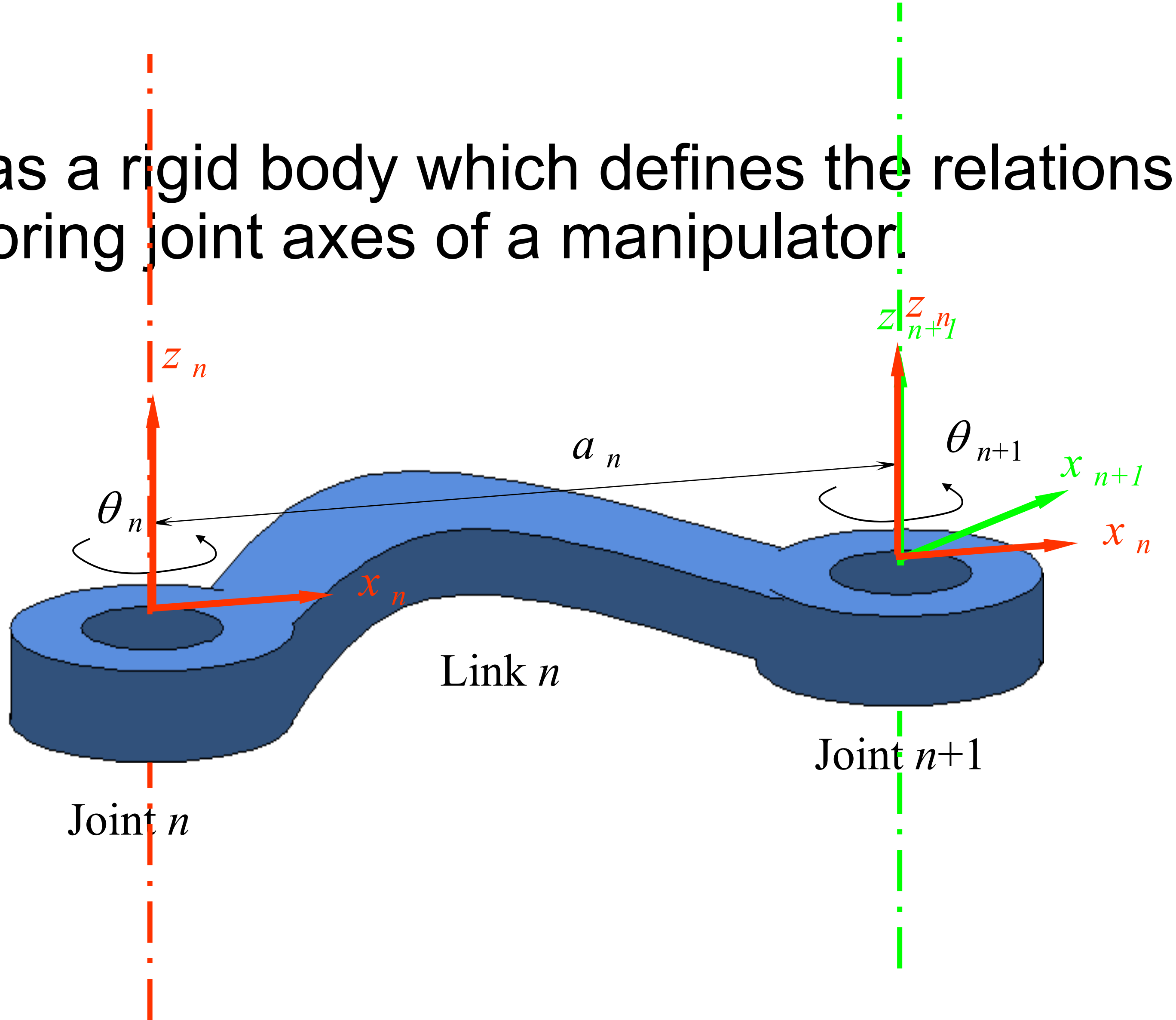
- Without loss of generality, we will consider only manipulators which have joints with a single degree of freedom.

- A joint having n degrees of freedom can be modeled as n joints of one degree of freedom connected with n-1 links of zero length.

- We could use 6-DoF Transforms to describe their 3D relation – but we can do better (fewer variables):

Link

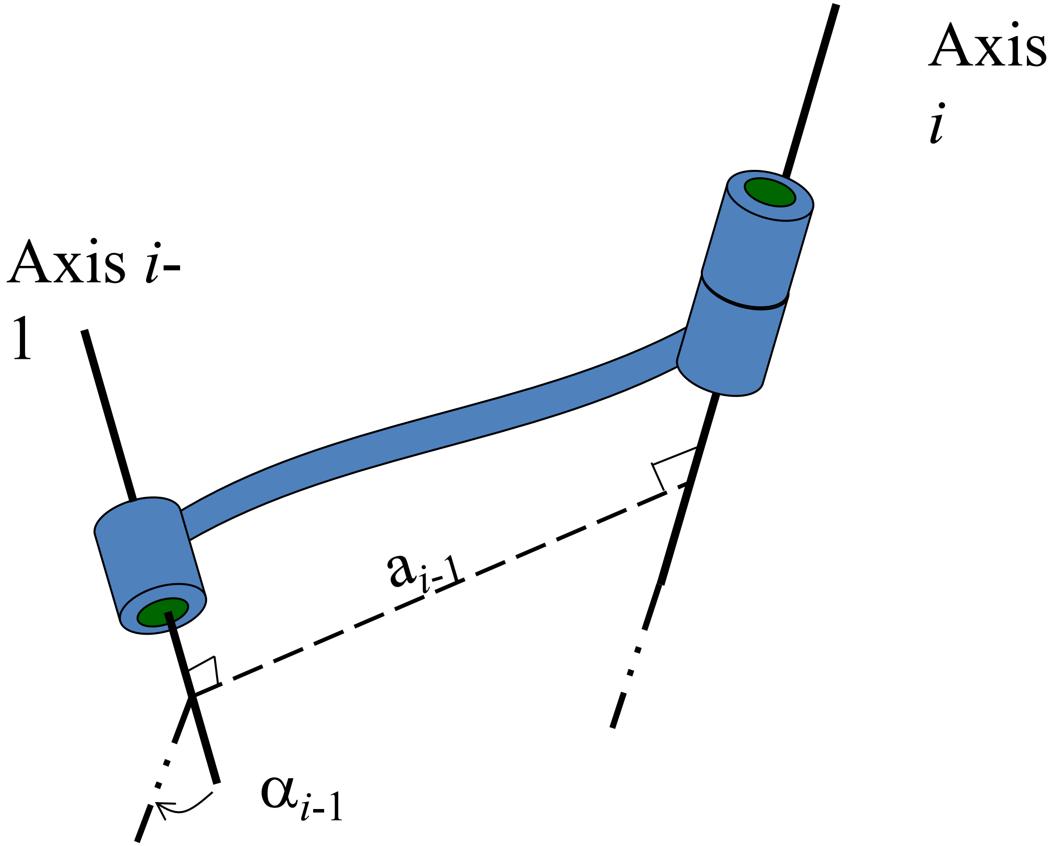
- A link is considered as a rigid body which defines the relationship between two neighboring joint axes of a manipulator.



The Kinematics Function of a Link

- The kinematics function of a link is to maintain a fixed relationship between the two joint axes it supports.

- This relationship can be described with two parameters: the link length a, the link twist a



Link Length

- Is measured along a line which is **mutually perpendicular** to **both** axes.

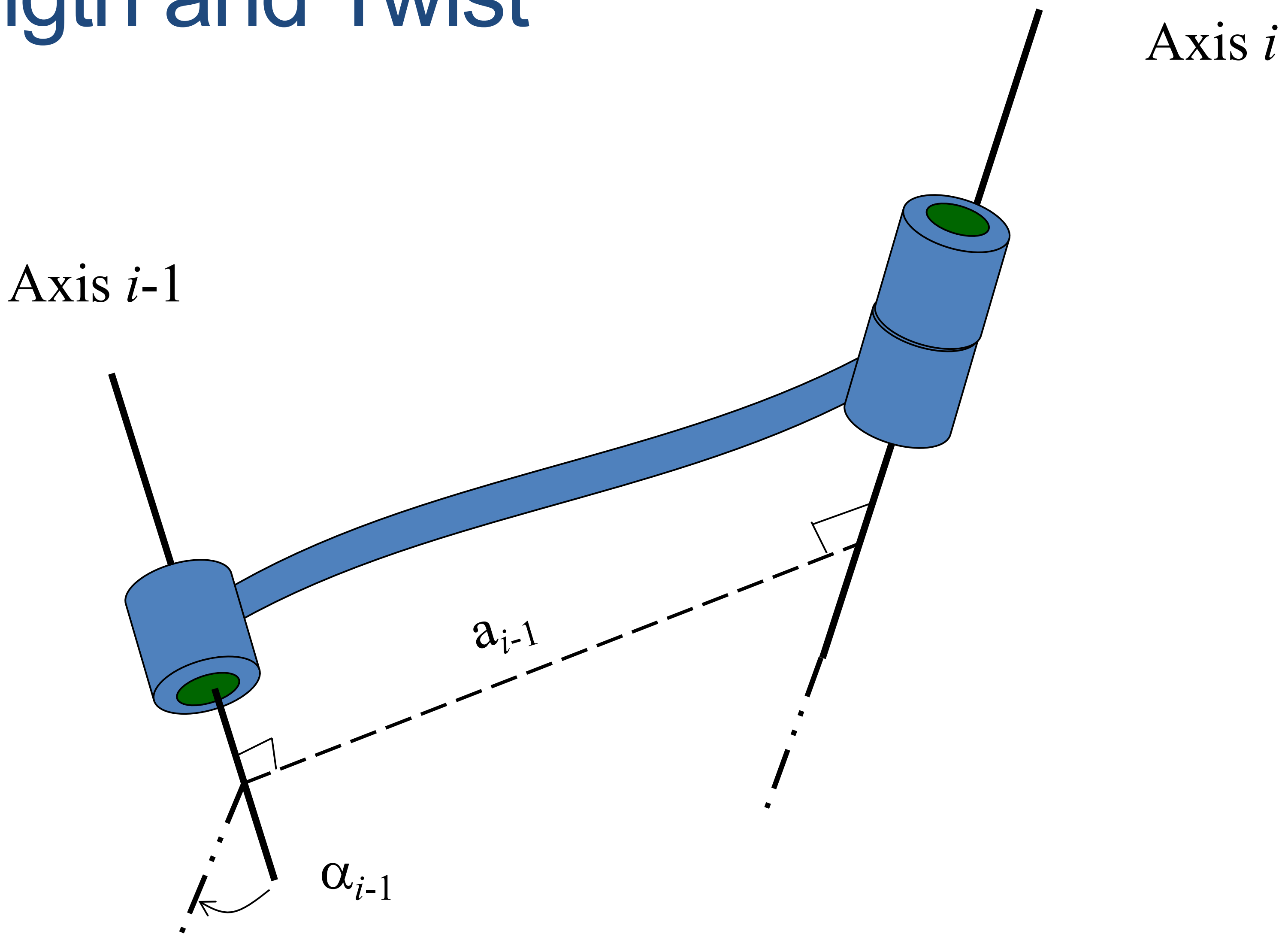
- The mutually perpendicular always exists and is unique except when both axes are parallel.

Link Twist

- Project both axes i-1 and i onto the plane whose normal is the mutually perpendicular line, and measure the angle between them

- Right-hand coordinate system

Link Length and Twist

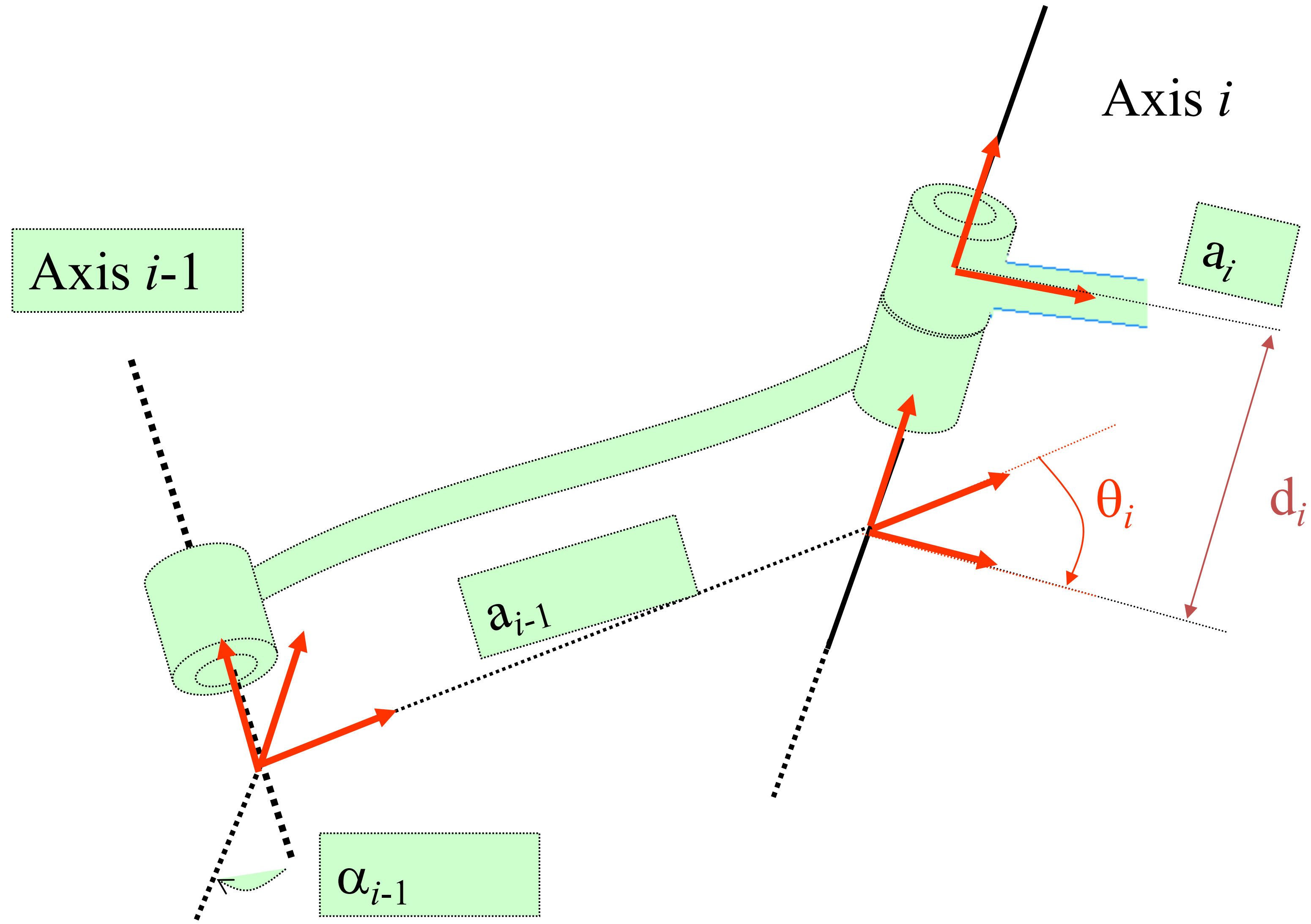


Joint Parameters (the Denavit-Hartenberg (DH) Parameters)

A joint axis is established at the connection of two links. This joint will have two normals connected to it - one for each of the links.

- The relative position of two links is called *link offset* d_n which is the distance between the links (the displacement, along the joint axes between the links).
- The *joint angle* θ_n between the normals is measured in a plane normal to the joint axis.

Link and Joint Parameters



Link and Joint Parameters

4 parameters are associated with each link. You can align the two axis using these parameters.

- Link parameters:

a_n the length of the link.

α_n the twist angle between the joint axes.

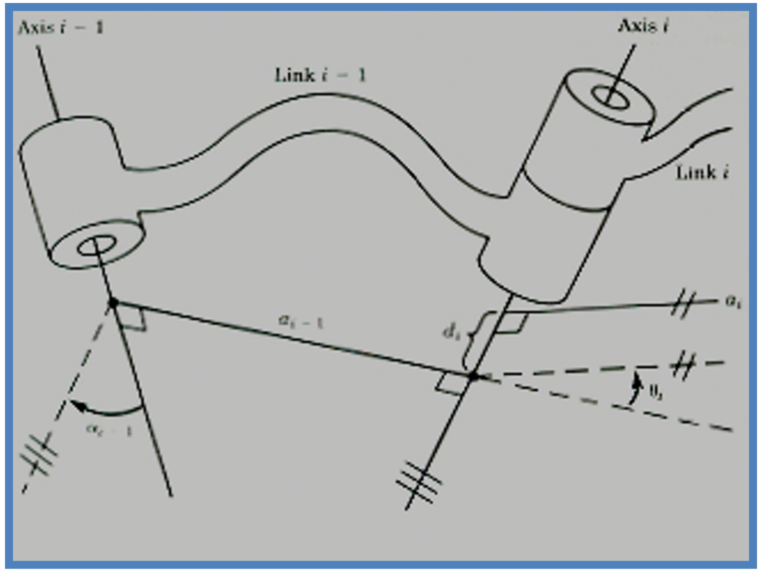
- Joint parameters:

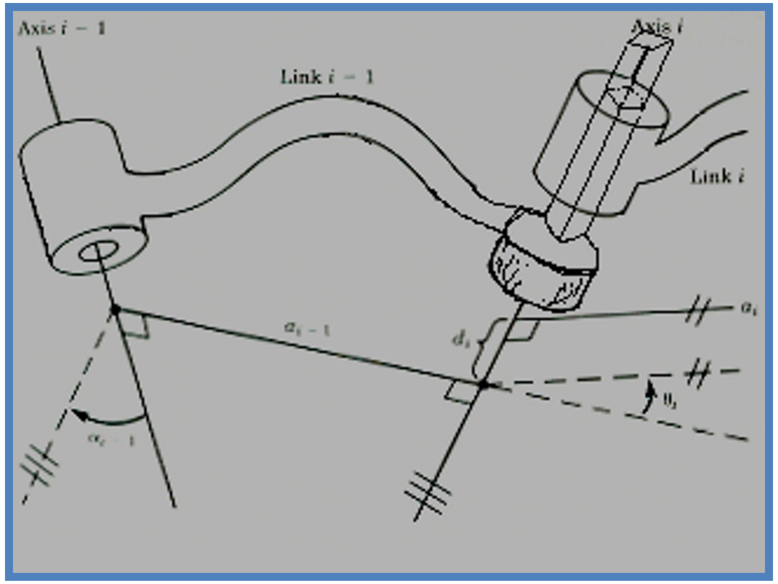
θ_n the angle between the links.

d_n the distance between the links

Link Connection Description:

For Revolute Joints: a, α, and d. are all fixed, then "θ_i" is the. Joint Variable.



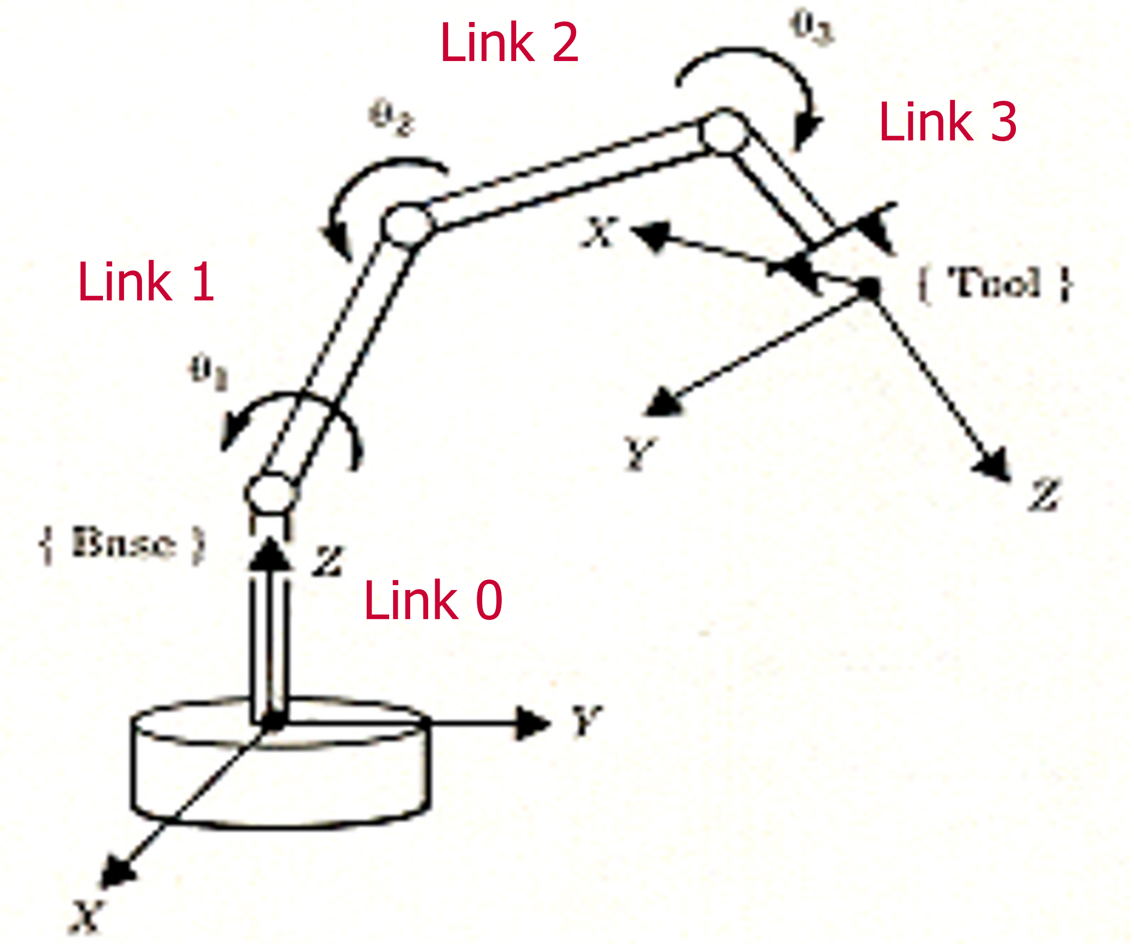


For Prismatic Joints: a, α, and θ. are all fixed, then "d_i" is the. Joint Variable.

These four parameters: (Link-Length a_{i-1}), (Link-Twist α_{i-1}), (Link-Offset d_i), (Joint-Angle θ_i) are known as the *Denavit-Hartenberg Link Parameters*.

Links Numbering Convention

Base of the arm:	**Link-0**
1st moving link:	**Link-1**
.	**.**
.	**.**
.	**.**
Last moving link:	**Link-n**



A 3-DOF Manipulator Arm

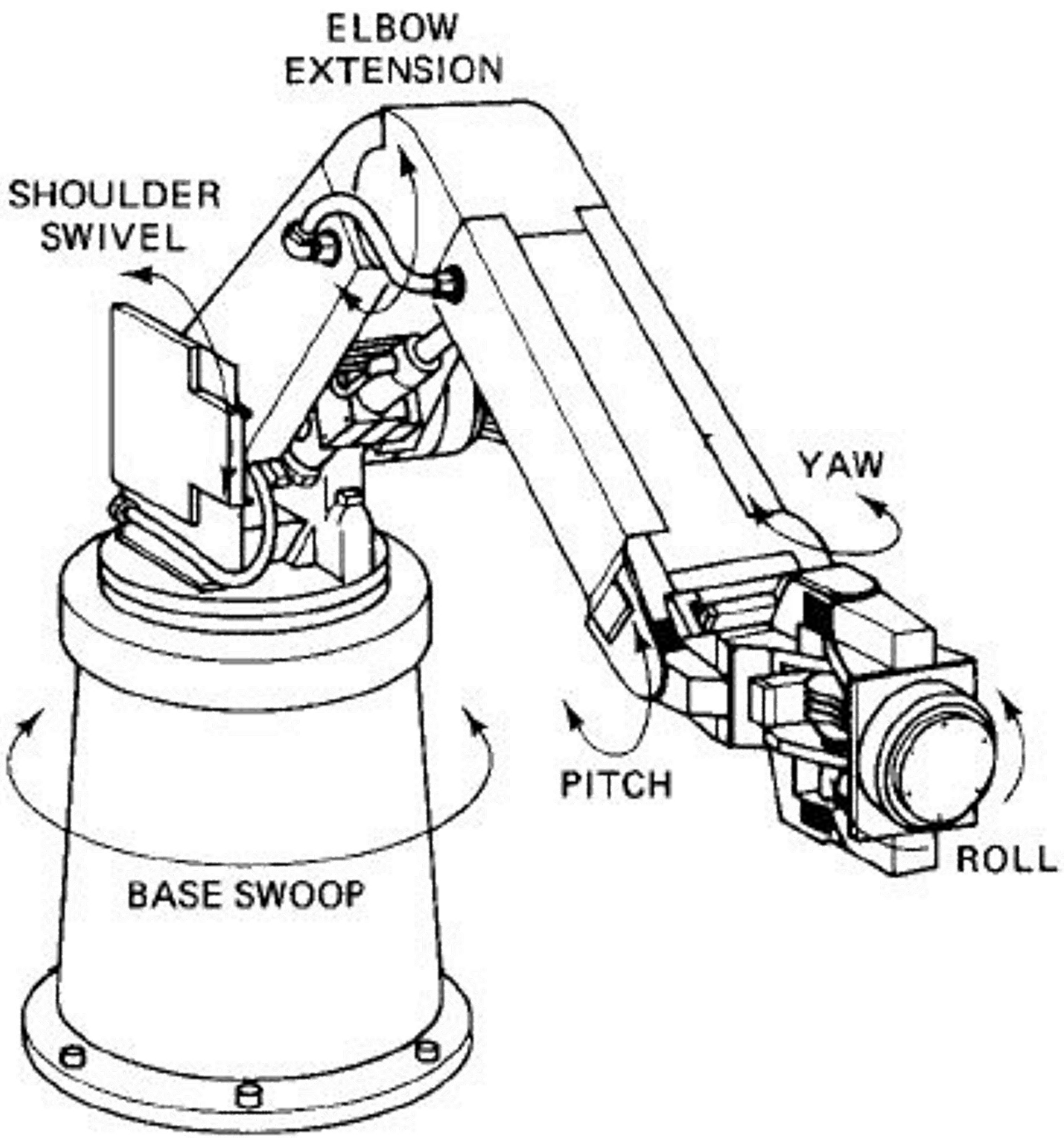
First and Last Links in the Chain

- $a_0 = \alpha_{n.} = 0$
- $\alpha_0 = \alpha_{n.} = 0$
- If joint 1 is revolute: d_0 is fixed and θ_1 is arbitrary
- If joint 1 is prismatic: d_0= arbitrary and θ_1 is fixed

Robot Specifications

Number of axes

- Major axes, (1-3) => position the wrist
- Minor axes, (4-6) => orient the tool
- Redundant, (7-n) => reaching around obstacles, avoiding undesirable configuration



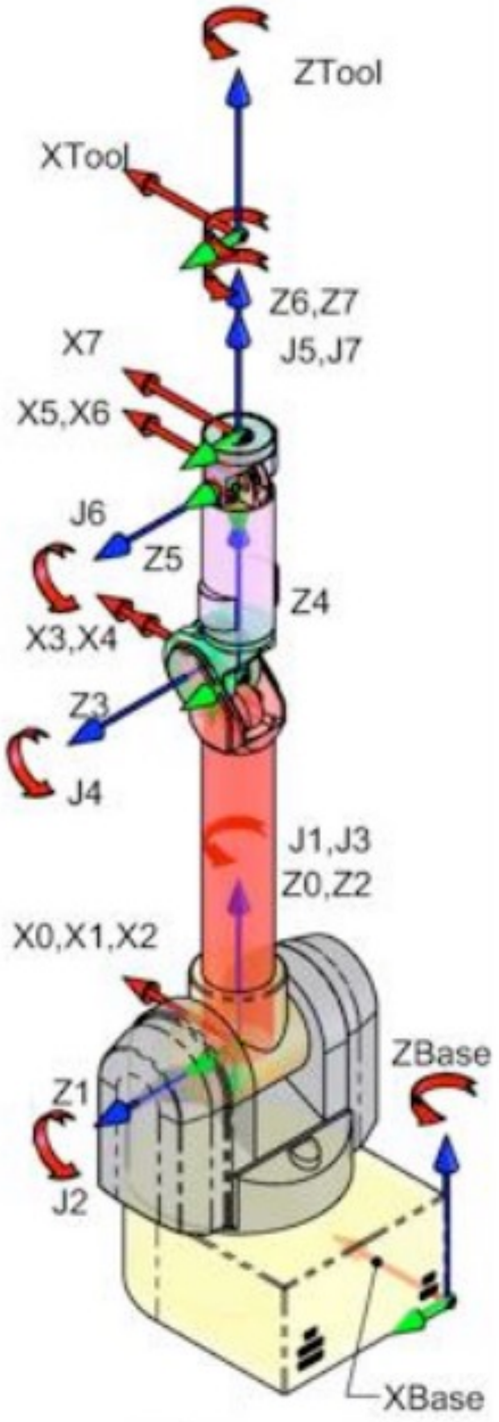
Example: Puma 500

Unimate Puma 500
Oussama Khatib | Used with permission

θ_j	d_j	a_j	α_j
q1	0.0000	0.0000	$\pi/2$
q2	0.0000	0.4318	0
q3	0.1500	0.0203	$-\pi/2$
q4	0.4318	0.0000	$\pi/2$
q5	0.0000	0.0000	$-\pi/2$
q6	0.0000	0.0000	0

Frames

- Choose the base and tool coordinate frame
  - Make your life easy!
- Several conventions
  - Denavit Hartenberg (DH), modified DH, Hayati, etc.



KINEMATICS

Kinematics

Forward Kinematics (angles to pose) (it is straight-forward -> easy)

What you are given: The constant arm parameters (e.g. DH parameters)
The angle of each joint

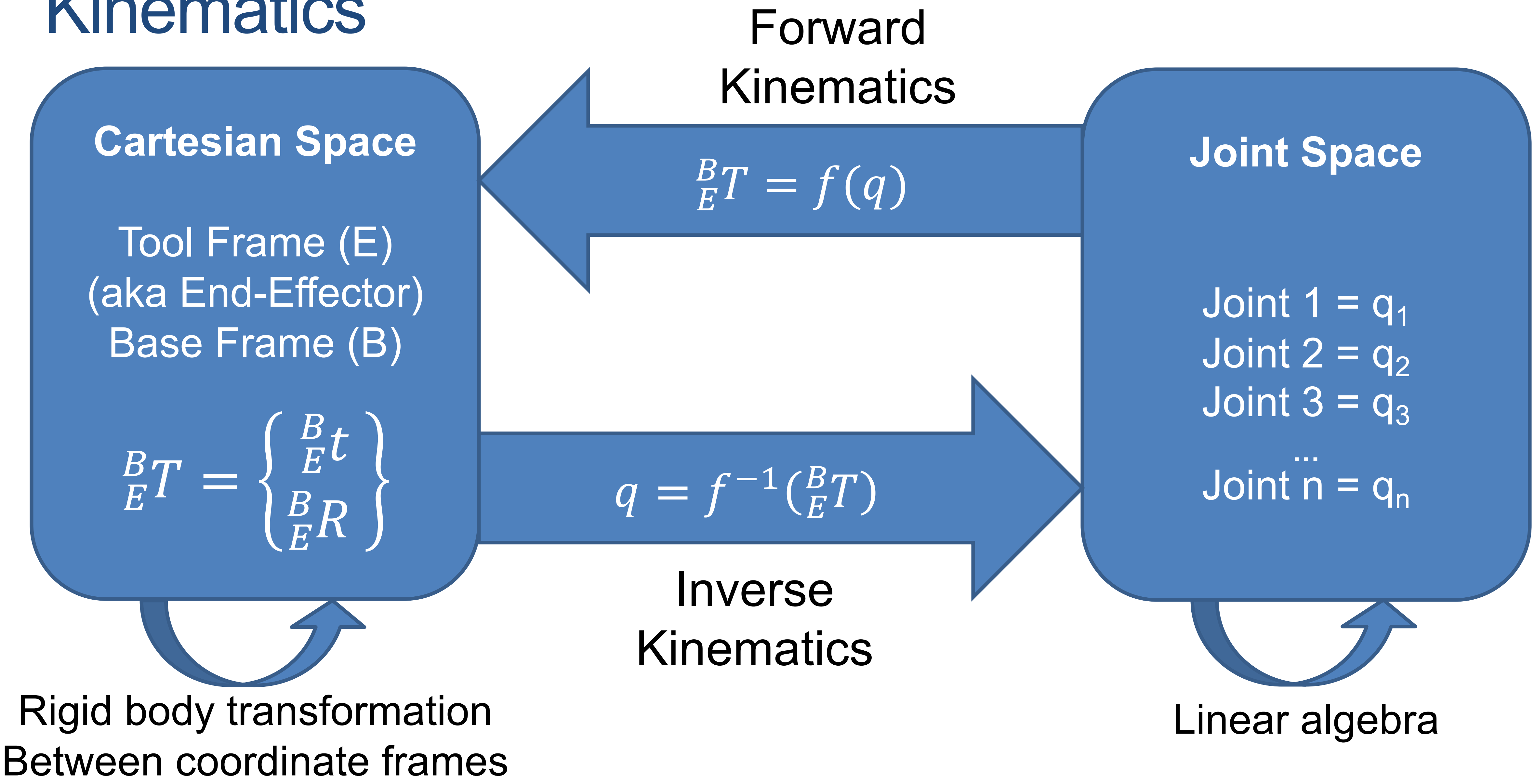
What you can find: The pose of any point (i.e. it's (x, y, z) coordinates & 3D orientation)

Inverse Kinematics (pose to angles) (more difficult)

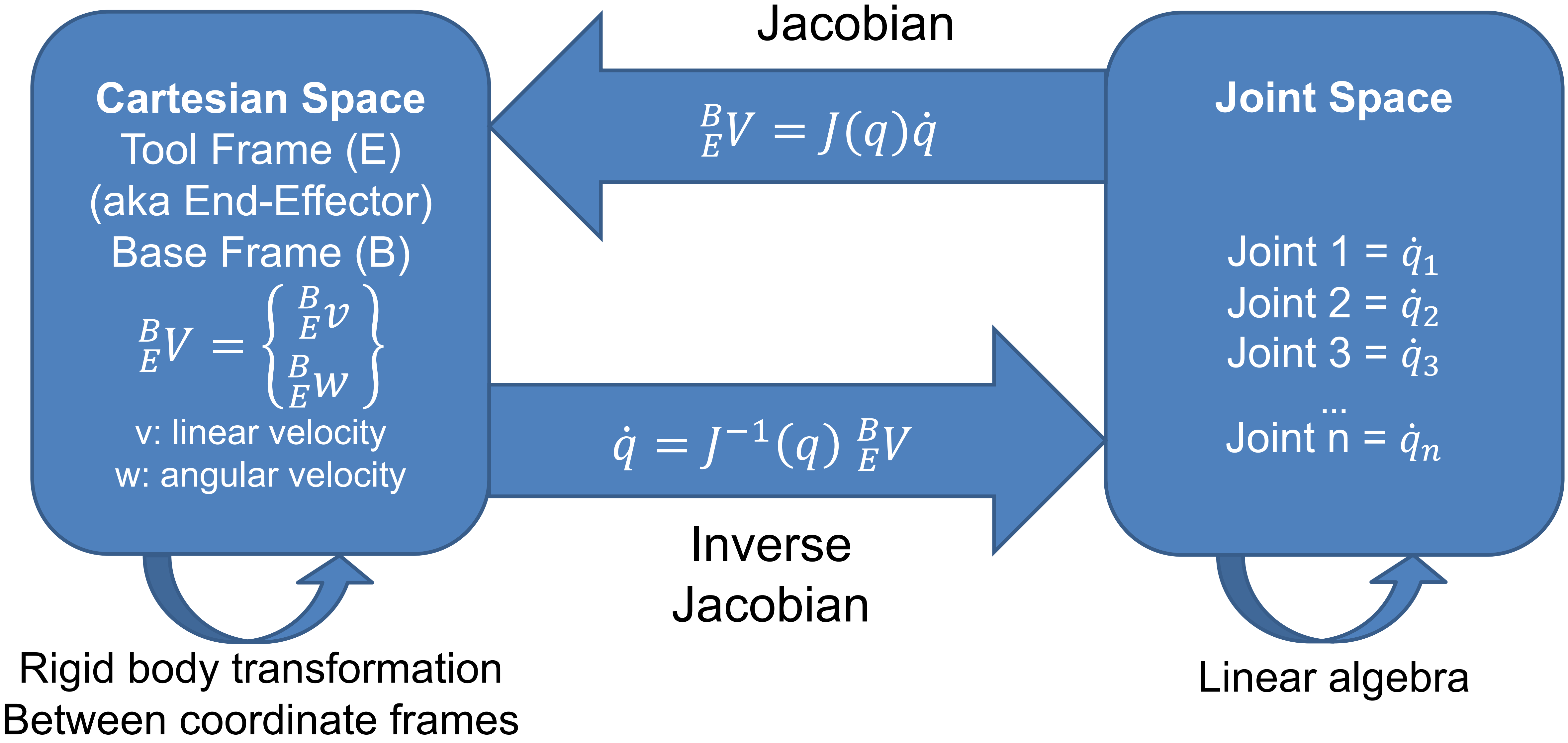
What you are given: The constant arm parameters (e.g. DH parameters)
The pose of some point on the robot

What you can find: The angles of each joint needed to obtain that pose

Kinematics


Forward
Kinematics
$^{B}_{E}T = f(q)$
Cartesian Space
Tool Frame (E)
(aka End-Effector)
Base Frame (B)
$^{B}_{E}T = \begin{Bmatrix} ^{B}_{E}t \\ ^{B}_{E}R \end{Bmatrix}$
Joint Space
Joint 1 = q_1
Joint 2 = q_2
Joint 3 = q_3
...
Joint n = q_n
$q = f^{-1}(^{B}_{E}T)$
Inverse
Kinematics
Rigid body transformation
Between coordinate frames
Linear algebra

Kinematics: Velocities



FORWARD KINEMATICS

Forward Kinematics

- DH Parameters to pose:

$$A = R_z(\Theta_j)\, t_z(d_j)\, t_x(a_j)\, R_x(\alpha_j) \quad =>$$

$${}^{B}_{E}T = R_z(\Theta_1)\, t_z(d_1)\, t_x(a_1)\, R_x(\alpha_1) \quad R_z(\Theta_2)\, t_z(d_2)\, t_x(a_2)\, R_x(\alpha_2) \ \cdots R_z(\Theta_N)\, t_z(d_N)\, t_x(a_N)\, R_x(\alpha_N)$$

- For revolute joints: only Θ varies – other three are constant for that robot
- For prismatic joints: only d varies – other three are constant for that robot

- For non-holonomic (mobile) robots: Chain the time-varying motion: Odometry (HW2)

INVERSE KINEMATICS (IK)

Inverse Kinematics (IK)

- Given end effector pose, compute required joint angles
- In simple case, analytic solution exists
  - Use trig, geometry, and algebra to solve

- Possible Problems of Inverse Kinematics
  - Multiple solutions
  - Infinitely many solutions
  - No solutions
  - No closed-form (analytical solution)

- Generally (more DOF) difficult
  - Use Newton’s method

- Analytic solution of 2-link inverse kinematics

$$x^2 + y^2 = a_1^2 + a_2^2 - 2a_1 a_2 \cos(\pi - \theta_2)$$

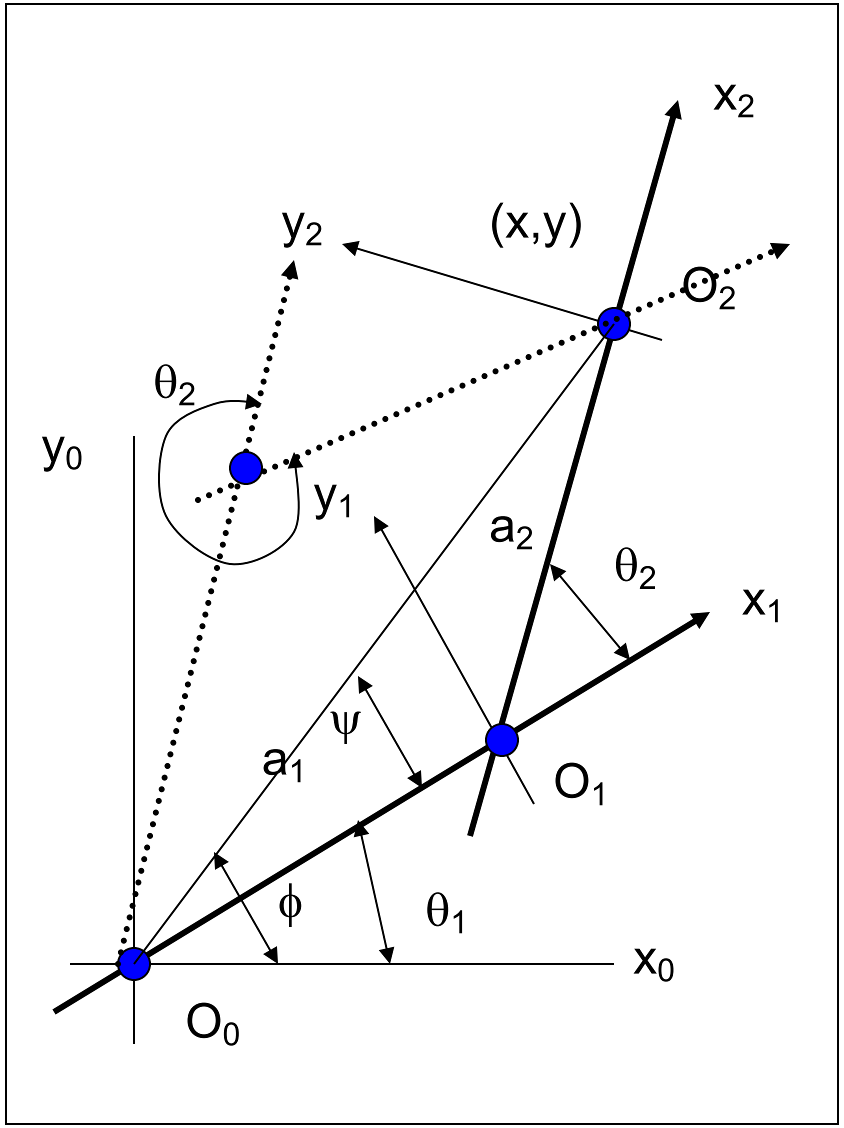
$$\cos\theta_2 = \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1 a_2}$$

for greater accuracy

$$\tan^2\frac{\theta_2}{2} = \frac{1-\cos\theta}{1+\cos\theta} = \frac{2a_1a_2 - x^2 - y^2 + a_1^2 + a_2^2}{2a_1a_2 + x^2 + y^2 - a_1^2 - a_2^2}$$

$$= \frac{\left(a_1^2 + a_2^2\right)^2 - \left(x^2 + y^2\right)}{\left(x^2 + y^2\right) - \left(a_1^2 - a_2^2\right)^2}$$

$$\theta_2 = \pm 2\tan^{-1}\sqrt{\frac{\left(a_1^2 + a_2^2\right)^2 - \left(x^2 + y^2\right)}{\left(x^2 + y^2\right) - \left(a_1^2 - a_2^2\right)^2}}$$



- Two solutions: elbow up & elbow down

Iterative IK Solutions

- Often analytic solution is infeasible
- Use Jacobian
- Derivative of function output relative to each of its inputs
- If y is function of three inputs and one output

$$y = f(x_1, x_2, x_3)$$

$$\delta y = \frac{\delta f}{\partial x_1} \cdot \delta x_1 + \frac{\delta f}{\partial x_2} \cdot \delta x_2 + \frac{\delta f}{\partial x_3} \cdot \delta x_3$$

- Represent Jacobian J(X) as a 1x3 matrix of partial derivatives

Jacobian

- In another situation, end effector has 6 DOFs and robotic arm has 6 DOFs
- $f(x_1, \ldots, x_6) = (x, y, z, r, p, y)$
- Therefore $J(X) = 6x6$ matrix

$$\begin{bmatrix} \frac{\partial f_x}{\partial x_1} & \frac{\partial f_y}{\partial x_1} & \frac{\partial f_z}{\partial x_1} & \frac{\partial f_r}{\partial x_1} & \frac{\partial f_p}{\partial x_1} & \frac{\partial f_y}{\partial x_1} \\ \frac{\partial f_x}{\partial x_2} & & & & & \\ \frac{\partial f_x}{\partial x_3} & & & & & \\ \frac{\partial f_x}{\partial x_4} & & & & & \\ \frac{\partial f_x}{\partial x_5} & & & & & \\ \frac{\partial f_x}{\partial x_6} & & & & & \end{bmatrix}$$

Jacobian Transpose Method

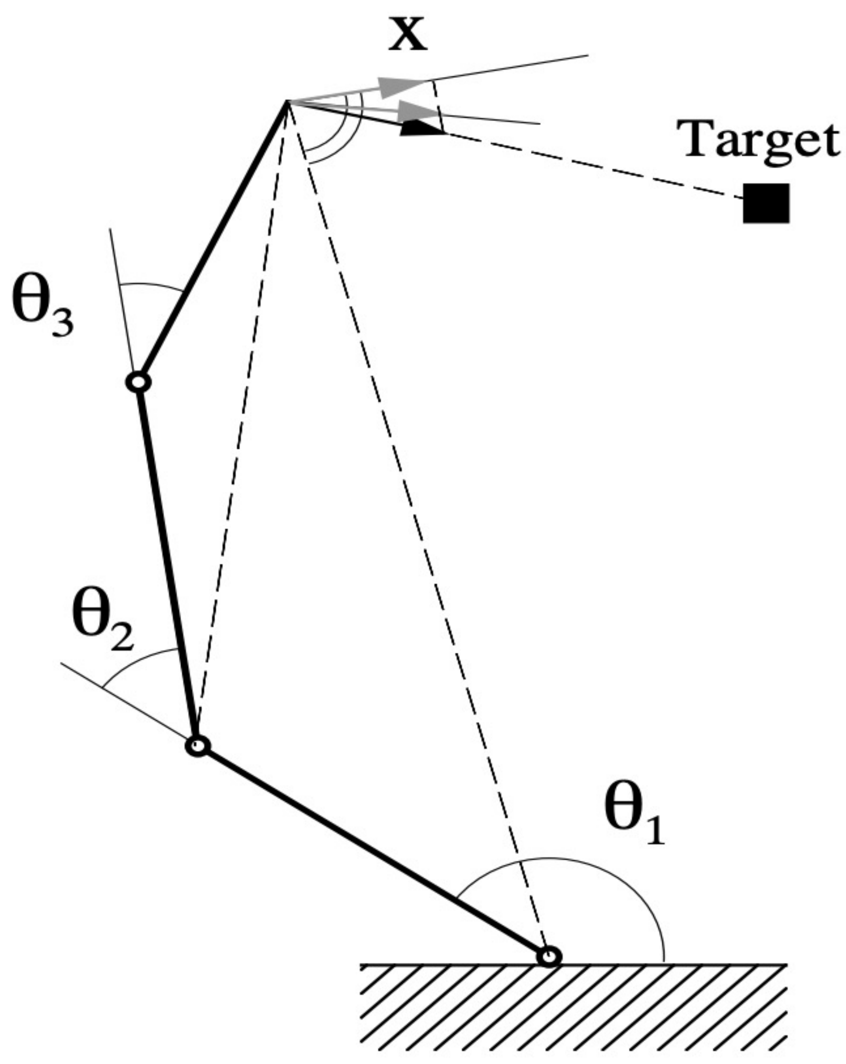
- Relates velocities in parameter space to velocities of outputs

$$\dot{Y} = J(X) \cdot \dot{X}$$

- If we know $Y_{current}$ and $Y_{desired}$, then we subtract to compute $\dot{Y}$
- Invert Jacobian and solve for $\dot{X}$:

$$\dot{X} = \alpha J^T(X) \cdot \Delta Y$$

- Projects difference vector ΔY to those dimensions which can reduce it the most
- Disadvantages:
  - Needs many iterations until convergence in certain configurations (e.g., Jacobian has very small coefficients)
  - Unpredictable joint configurations



Stefan Schaal; USC, Max Planck

Iterative Solution to Inverse Kinematics

- Only holds for high sampling rates or low Cartesian velocities
- “a local solution” that may be “globally” inappropriate
- Problems with singular postures
- Can be used in two ways:
  - As an instantaneous solutions of “which way to take “
  - As an “batch” iteration method to find the correct configuration at a target

PLANNING

Kinematic Problems for Manipulation

- Reliably position the tip - go from one position to another position
- Don't hit anything, avoid obstacles
- Make smooth motions
  - at reasonable speeds and
  - at reasonable accelerations
- Adjust to changing conditions -
  - i.e. when something is picked up *respond to the change in weight*

Planning Problem

- (Arm) Pose: Set of join values
- (Arm) Trajectory:
  - Given a start pose and an end pose
  - A list of intermediate poses
  - That should be reached one after the other
  - With associated (desired) velocities and accelerations (maxima)
    - Without time (without velocity and acceleration): path! So:
    - Path: poses;   Trajectory: poses with speeds (and maybe accelerations)
- Constrains:
  - Don’t collide with yourself
  - Don’t collide with anything else (except: fingers with the object to manipulate!!!)
  - Additional possible constrains:
    - Maximum joint velocities or accelerations
    - Keep global orientation of a joint (often end-effector) within certain boundaries