RLSC Homework

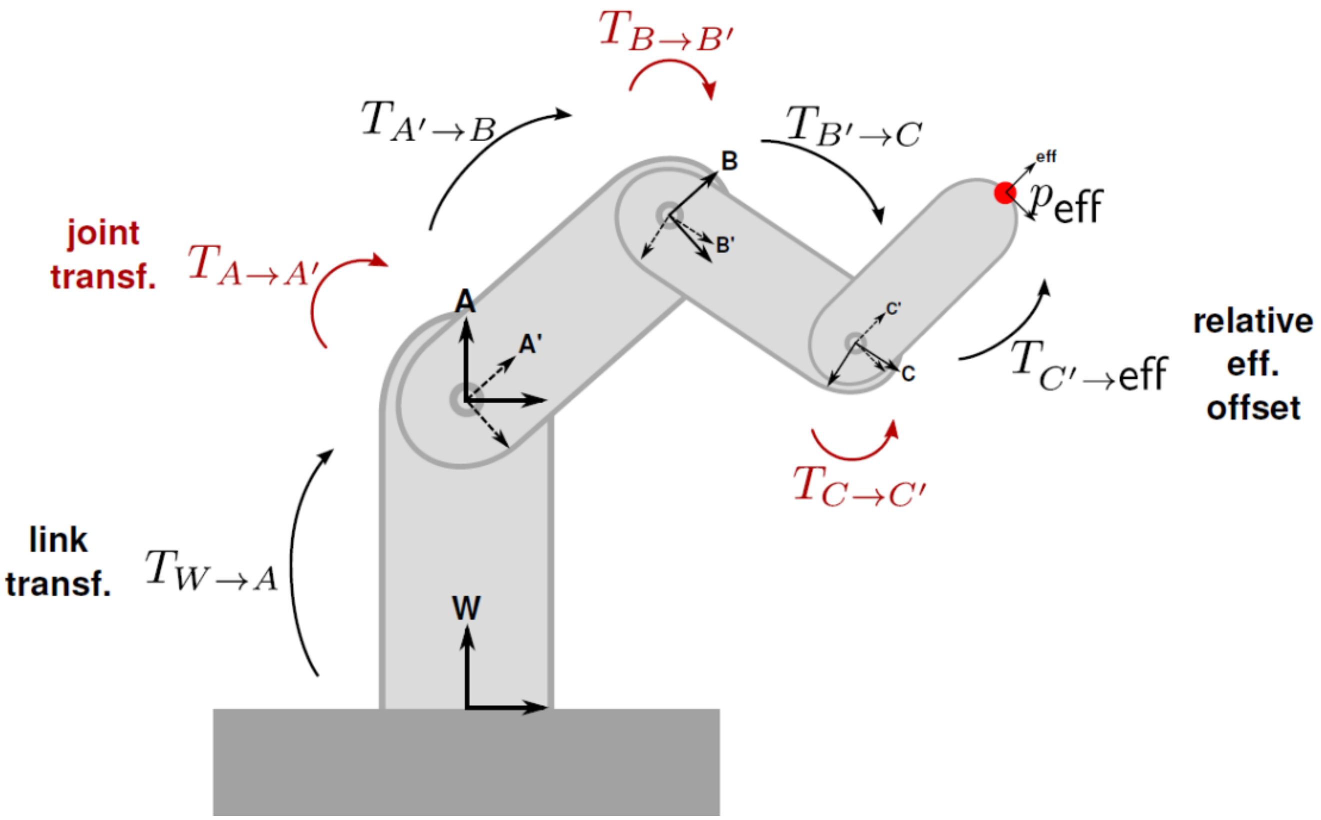
- Kinematics recap.
- PCA
- Baxter robot
- Example code

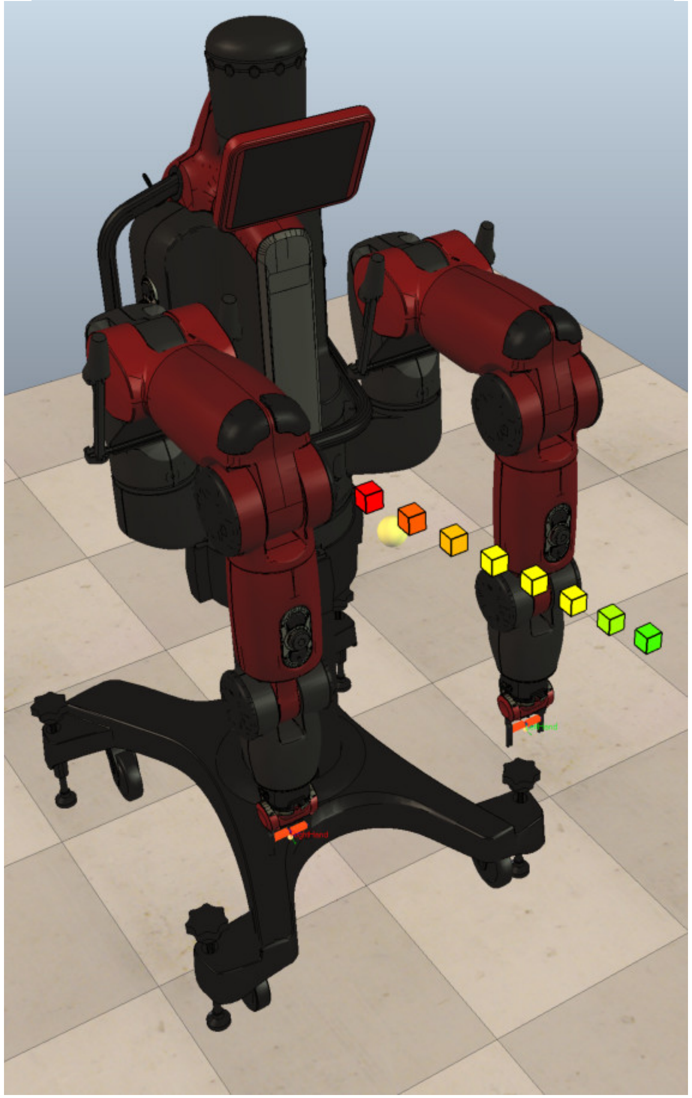
Kinematics - notation

$q \in \mathbb{R}^n$	vector of joint angles (robot configuration)
$\dot{q} \in \mathbb{R}^n$	vector of joint angular velocities
$\delta q \in \mathbb{R}^n$	small step in joint angles
$y \in \mathbb{R}^d$	some "endeffector(s) feature(s)" e.g. position $\in \mathbb{R}^3$ or vector $\in \mathbb{R}^3$
$\phi : \; q \mapsto y$	kinematic map
$J(q) = \frac{\partial \phi}{\partial q} \in \mathbb{R}^{d \times n}$	Jacobian
$\lVert v \rVert_W^2 = v^\top W v$	squared norm of v w.r.t. metric W

Kinematic Structure

- Provided through KDL





Kinematic Map and Jacobian

For any joint angle vector $q \in \mathbb{R}^n$ we can compute $T_{W\to\text{eff}}(q)$ by *forward chaining* of transformations

$T_{W\to\text{eff}}(q)$ gives us the *pose* of the endeffector

$$\phi_{\text{pos}}(q) = T_{W\to\text{eff}}(q).\text{translation} \quad \in \mathbb{R}^3$$

Given the kinematic map $y = \phi(q)$
what is the Jacobian $J(q) = \frac{\partial}{\partial q}\phi(q)$?

$$J(q) = \frac{\partial}{\partial q}\phi(q) = \begin{pmatrix} \frac{\partial \phi_1(q)}{\partial q_1} & \frac{\partial \phi_1(q)}{\partial q_2} & \dots & \frac{\partial \phi_1(q)}{\partial q_n} \\ \frac{\partial \phi_2(q)}{\partial q_1} & \frac{\partial \phi_2(q)}{\partial q_2} & \dots & \frac{\partial \phi_2(q)}{\partial q_n} \\ \vdots & & & \vdots \\ \frac{\partial \phi_d(q)}{\partial q_1} & \frac{\partial \phi_d(q)}{\partial q_2} & \dots & \frac{\partial \phi_d(q)}{\partial q_n} \end{pmatrix}$$

IK with null space resolution

- Cost to optimize:

$$f(q_{t+1}) = \|q_{t+1} - q_t\|_W^2 + \|\phi(q_{t+1}) - y^*\|_C^2$$

- Augmentation for null-space resolution:

$$\|q_{t+1} - q_t - h\|_W^2$$

- Resulting optimal step:

$$\delta q = J^\sharp \, \delta y + (I - J^\sharp J) \, h$$

- Where:

$$J^\sharp = (J^\top C J + W)^{-1} J^\top C = W^{-1} J^\top (J W^{-1} J^\top + C^{-1})^{-1}$$

- h is the null space motion component

IK algorithm

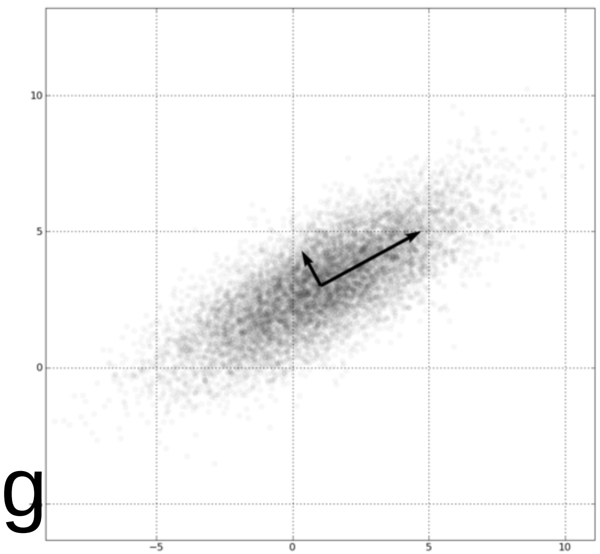
- **Input:** starting state q_0, desired y^*, forward map $\phi(q)$, Jacobian $J(q)$, weighting W, regularization C, comfortable pose q_{comf}
- **Output:** final pose q^*
- $q = q_0$
- $q_{\text{old}} = q + \epsilon$
- while $q - q_{\text{old}} > \epsilon$
  - $y = \phi(q)$
  - $J = J(q)$
  - $q = q + J^{\#}(y^* - y) + (I - J^{\#}J)(q_{\text{comf}} - q)$
  - $q_{\text{old}} = q$

Principal Component Analysis

- The kimatic map is highly non-linear

$$\phi_{\text{pos}}(q) = T_{W \to \text{eff}}(q).\text{translation} \quad \in \mathbb{R}^3$$

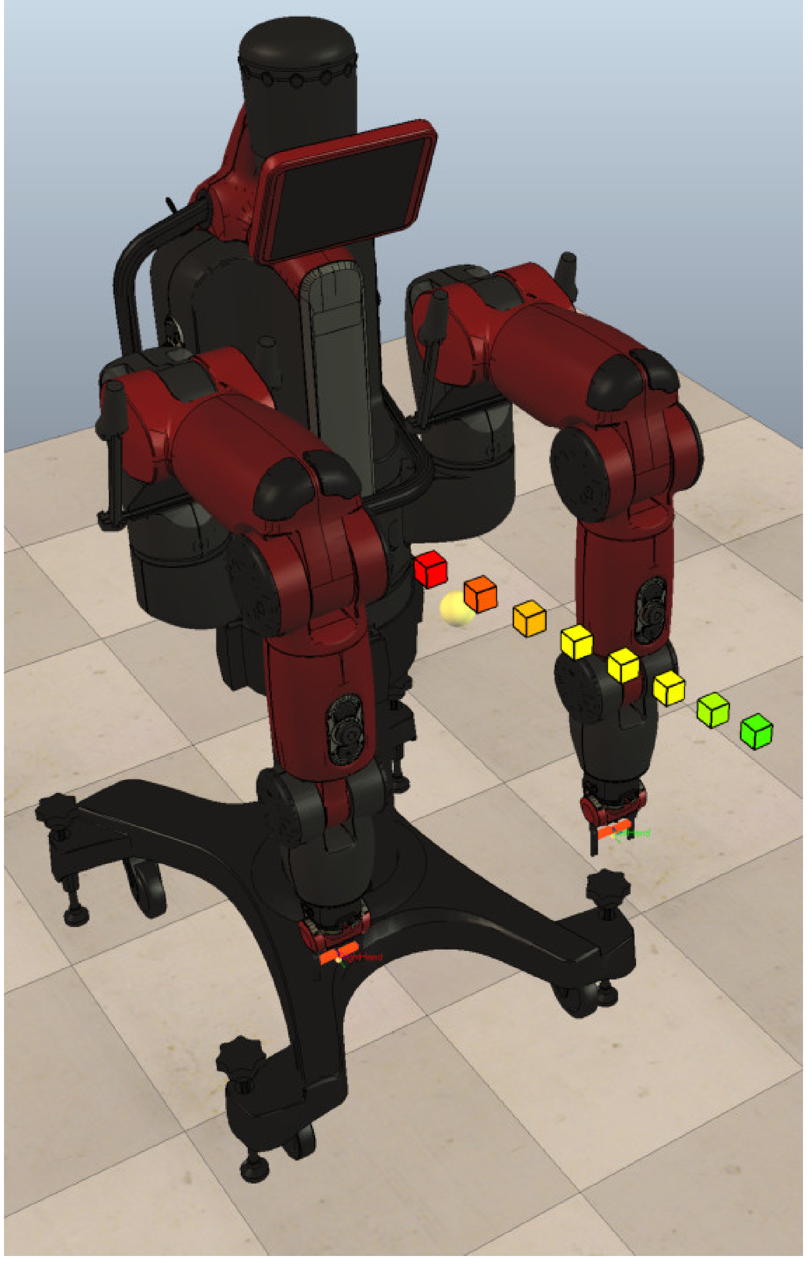
- If the task is linear, it can still be well represented using the principal components



- Example: The joint space pose samples of an end-effector moving on a surface have 2 principal components
  (the x,y axis of the plane)

Baxter robot

- 2x 7DOF arm
- Interchangeable grippers
- Fixed base
- Position control of all joints
- RGB camera
- Sonar array



Baxter Tools API

- Provides:
  - Simulator control (start, stop, advance)
  - Robot control (set joint angles)
  - Kinematics (FK and Jacobian)
  - Retrieve target positions
- Eigen library
  - Linear algebra tools (matrix and vector operations, transposes, inverse, ...)

Baxter Tools API

- Example code walk-through
- Q&A