Inverse Kinematics

2020/05/19

Outline

- Project overview and demo
- Objective and explanation
- Score criteria
- Submission
- Hint and reminder

Project overview

- Understand the principle of IK and apply it to satisfy specified constraints
  - Mapping from Cartesian space to joint space
  - Compute joint angles to place the end effector to the desired positions

System Overview

- The same framework of assignment 2
- Include forward kinematics library

math Module

- math::MatrixN_t
  - a dynamic-size matrix
- math::VectorNd_t
  - a dynamic-size column vector
- dynamic size object need allocation first
  - math::MatrixN_t mat(10, 15)
    mat is a 10×15 matrix, with allocated but uninitiallized coefficients

Inverse Kinematics

- Mapping from cartesian space to joint space
- Review “Kinematics.ppt” from p.21-p.58

Solving Inverse Kinematics

- Analytic method
- Inverse-Jacobian method
- Optimization-based method
- Example-based method

Inverse-Jacobian method

- Jacobian maps velocities in joint space to velocities in cartesian space

$$f(\theta) = p$$

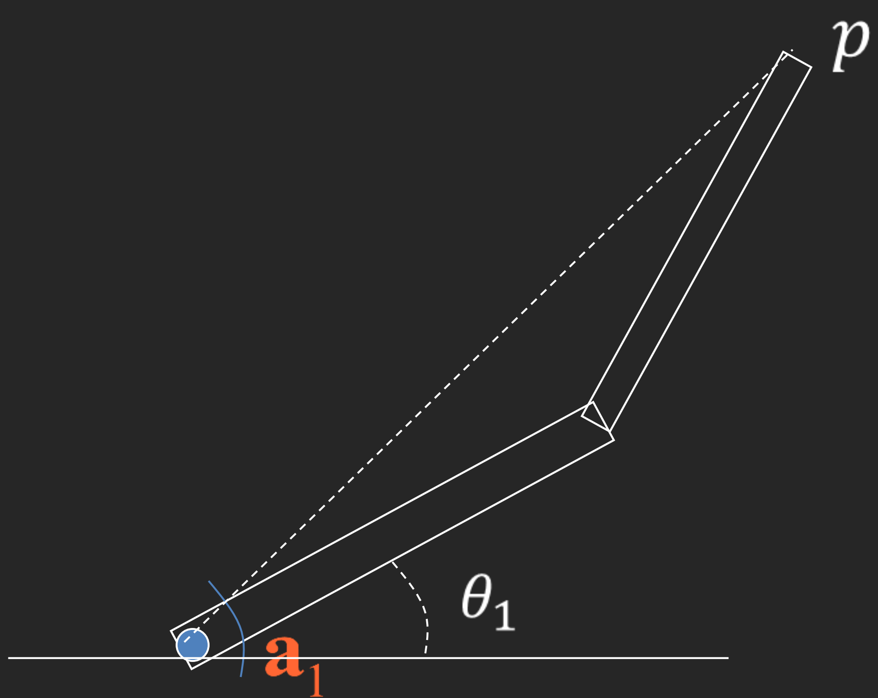
$$\frac{dp}{dt} = \frac{\partial f(\theta)}{\partial \theta}\frac{d\theta}{dt} = J(\theta)\frac{d\theta}{dt}$$

$$J(\theta)\dot{\theta} = V$$

$$J_{ij} = \frac{\partial f_i}{\partial \theta_j}$$

Computing Jacobian

- Take geometric approach to compute it
- You will need the result of FK
- Review “Kinematics.ppt” from p.40-p.47



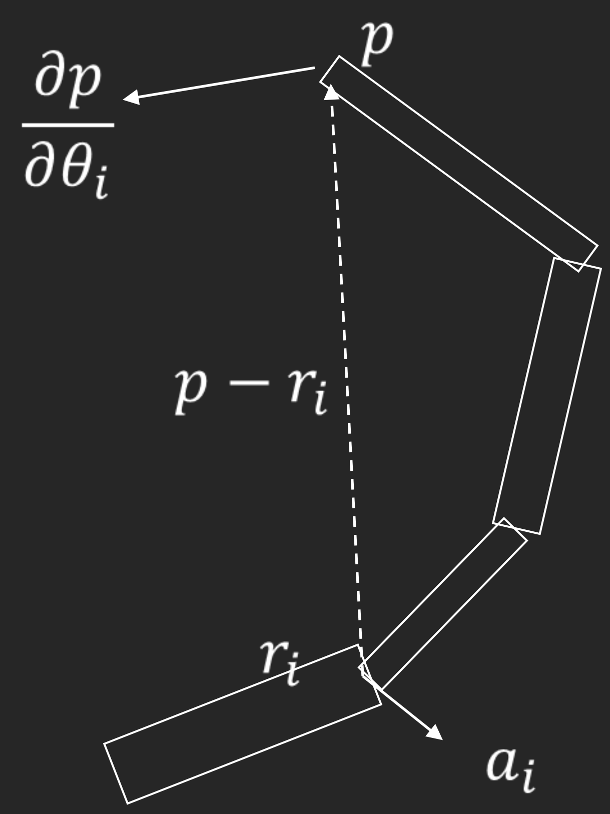
$$\frac{\partial p}{\theta_1} = \begin{bmatrix} \dfrac{\partial p_x}{\partial \theta_1} \\ \dfrac{\partial p_y}{\partial \theta_1} \end{bmatrix} = a_1 \times (p - r_1)$$

unit-length rotation axis vector

$$a_1 = \frac{\omega_1}{|\omega_1|}$$

Rotational DOFs

$$\frac{\partial p}{\partial \theta_i} = a_i \times (p - r_i)$$



- a_i: unit length rotation axis in world space
- r_i: position of joint pivot in world space
- p: end effector position in world space

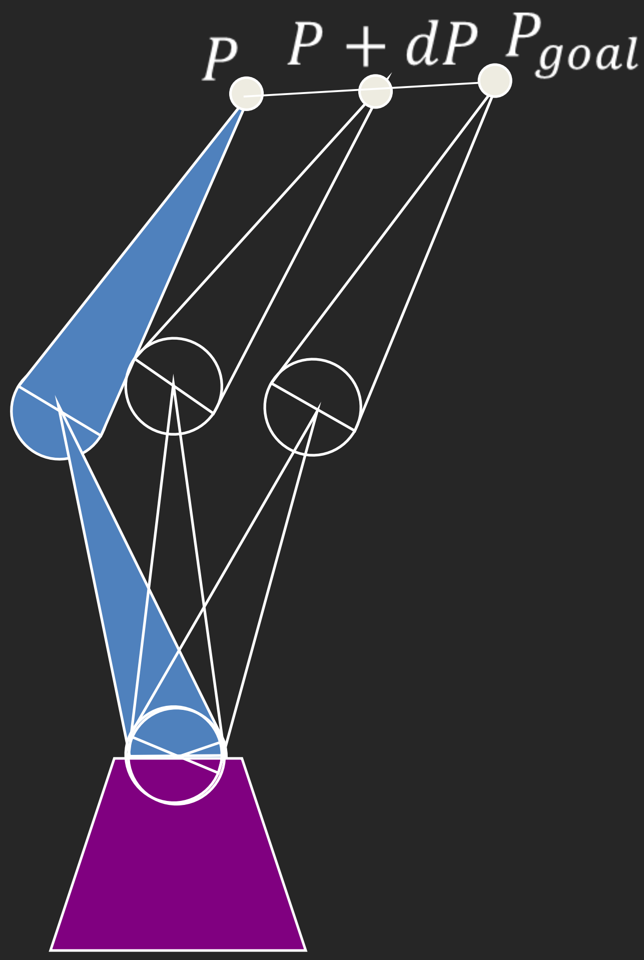
Iterative IK using Inverse Jacobian

$$\theta = f^{-1}(P)$$

$$V = J(\theta)\dot{\theta}$$

$$\dot{\theta} = J^{-1}(\theta)V$$

$$\theta_{k+1} = \theta_k + \Delta t J^{-1}(\theta_k)V$$



What you need to do

- Finish the methods in ‘kinematics_inverse_jacobian_ik_solver.cpp’
- Search for “TO DO”
- Test <start_bone_idx> with 25(arm) and 11(upperback)
- Test specified target
    - (x = -1.0, y = 1.49, z = -1.5)

InverseJacobianIkSolver::
Solve Parameters

- target_pos
  - The target position of end-effector
- start_bone_idx/end_bone_idx
  - Specify the bones used to reach target position
  - Default value is rhumerus/rhand
- original_whole_body_joint_pos6d
  - Motion data of the reference pose

InverseJacobianIkSolver Configuration

- skeleton
  - the loaded skeleton, applied to forward kinematics
- linear_system_solver
  - solver of $\dot{\theta} = J^{-1}(\theta)V$, applied to solve inverse Jacobian
- step
  - linearization step t
  - $\theta_{k+1} = \theta + \Delta t \, J^{-1}(\theta_k)V$ (p.43)
- distance_epsilon
  - the desired tolerance of the distance b/t the target position &end-effector
- max_iteration_num
  - the maximum allowable iterations to solve IK

Pseudoinverse Method

- Given a linear system $J\ \Delta\theta = e$
- The pseudoinverse J^+ set $\Delta\theta$ equal to

$$\Delta\theta = J^+ e$$

What you need to Do

- Implement $\dot{\theta} = J^{-1}(\theta)V$
  - math::PseudoinverseSolver::Solve
- Input
  - coef_mat $J(\theta)$ (p.43)
  - desired_vector V (p.43)
- Output
  - $\dot{\theta} = J^{-1}(\theta)V$ (p.43)

Configuration File

- See inverse_kinematics_config.xml for options

Score criteria

- Inverse kinematics 80%
  - Pseudoinverse method 20%
  - related implementations 60%
- Report 20%

Suggested Outline of Report

- Introduction/Motivation
- Fundamentals
- Implementation
- Result & Discussion
- Conclusion

Report Requirements

- Discussion of iterative IK
  - effects of step, distance_epsilon, etc.

- Discussion of jacobian matrix at specified target (x = -1.0, y = 1.49, z = -1.5)
  - Difference between starting from arm(id = 25) and lowerback (id = 11)

Submission

- Compress all materials into a zip file
  - Ensure your solution could be built successfully
  - Naming rule: CA3_ID_Version
    e.g., CA3_1234567_v01.zip
- Your zip file shall contain
  - Source code (without boost & Eigen)
  - Report in pdf or MS word format, no more than 10 pages
- Upload to E3
  - No limit to the no. of times you can upload
  - The latest version is your final submission

Due Date

- 06/01, 23:55
- The earlier your start, the more chances you have…

Late and Cheating

- Late policies
  - Penalty of 10 points of the value of the assignment/day
- Cheating policies
  - 0 points for any cheating on assignments
  - Allowing another student to examine your code is also considered as cheating

Hints and Reminder

- DO NOT include any Chinese / Mandarin characters in the path to your project
  - The program will crash when loading files
- Please run the source code under release mode

You can find TA in...

- Email:
  - Through new e3, and please CC to other TAs
- Lab EC229B
  - Need appointment
  - Please briefly describe your question(s) in the email