Computer Games Exercises: 2024s s10 (non-physics)

https://www.umm.uni-heidelberg.de/miism/

Contents

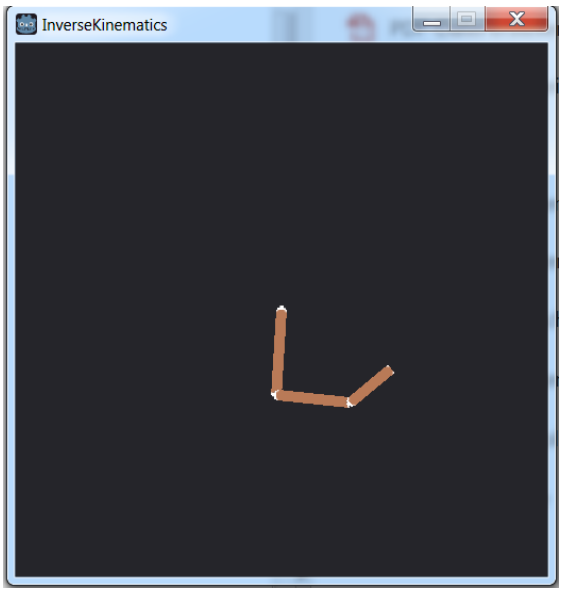
Answer header

Please put the author information in the header of all code files.

- `name (Name)`
- `coauthor list`

A04: Inverse Kinematics

Basics



For a three link planar robot arm, the forward kinematics is given by

$$\vec{X}_p = \begin{pmatrix} l_1 \cdot \cos(\Theta_1) + l_2 \cdot \cos(\Theta_1 + \Theta_2) + l_3 \cdot \cos(\Theta_1 + \Theta_2 + \Theta_3) \\ l_1 \cdot \sin(\Theta_1) + l_2 \cdot \sin(\Theta_1 + \Theta_2) + l_3 \cdot \sin(\Theta_1 + \Theta_2 + \Theta_3) \end{pmatrix}, \tag{1}$$

where $\vec{X}_p$ is the location of the moving end, l_1, l_2 and l_3 are the length of the arm segments, and Θ_1, Θ_2 and Θ_3 are the joint angles.

To implement inverse kinematics, the Jacobian inverse technique should be used. As the Jacobian matrix J is a 2×3 matrix in the current case, the right Jacobian pseudo-inverse

$$J^\dagger = J^T \cdot (J \cdot J^T)^{-1} \tag{2}$$

should be used to update the angles by

$$\Delta\vec{\Theta} = J^\dagger \cdot (\vec{X}_t - \vec{X}_p), \tag{3}$$

$$\vec{\Theta}_{k+1} = \vec{\Theta}_k + \kappa \cdot \Delta\vec{\Theta}, \tag{4}$$

where $\vec{X}_t$ is the target location of the moving end, and κ is the step size.

Task

Read the source code of the game "`inverse kinematics`" and extend the game. During the game, when there is one mouse click at one position in the scene, the robot arm should adjust its joints to move the moving end to the position.

- Calculate the Jacobian J.
- Calculate the pseudo-inverse $J^{\dagger}$.
- Calculate the angle update $\Delta\vec{\Theta}$.
- Update the angles and the location of the moving end for each frame with $\kappa = 0.01$.
- Rotate the arm segments to display the result. (HINT: In the current version, apply the negative angles to update the nodes as in the `ready()` function.)

Questions

Write the corresponding answers in the script file.

- If there are more degrees of freedom from the joints than necessary to reach out for a specified end position, the pseudo-inverse gives any solution. How could you influence this solution, for example to get another solution that evades a part of the 3D space due to potential collisions?