MuJoCo: Jacobian/Inverse kinematics

Using template_writeData2.zip to get started

1. From tiny.cc/mujoco download template_writeData2.zip and unzip in myproject
2. Rename folder template to dbpendulum_ik
3. Make these three changes
    1. main.c — line 28, change template_writeData2/ to dbpendulum_ik/
    2. makefile — change ROOT = template_writeData to ROOT = dbpendulum_ik also UNCOMMENT (del #) appropriate to your OS
    3. run_unix / run_win.bat change <template_writeData2> to <dbpendulum_ik>
4. In the *shell, navigate to dbpendulum_ik and type ./run_unix (unix) or run_win (windows); *shell = terminal for mac/linux / x64 for win

MuJoCo: Jacobian, J (1)

$$\mathbf{f} = \left[f_1(\mathbf{q}), \quad f_2(\mathbf{q}), \quad f_3(\mathbf{q}), \quad \ldots \quad f_m(\mathbf{q}) \right]$$

size = m

$$\mathbf{q} = \left[x_1, \quad x_2, \quad \ldots \quad x_n \right]$$

size = n

$$\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{q}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \ldots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \ldots & \frac{\partial f_2}{\partial x_n} \\ \ldots & \ldots & .... & \ldots & \ldots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \frac{\partial f_m}{\partial x_3} & \ldots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

size = mxn

MuJoCo: Compute end-effector velocity, V (2)

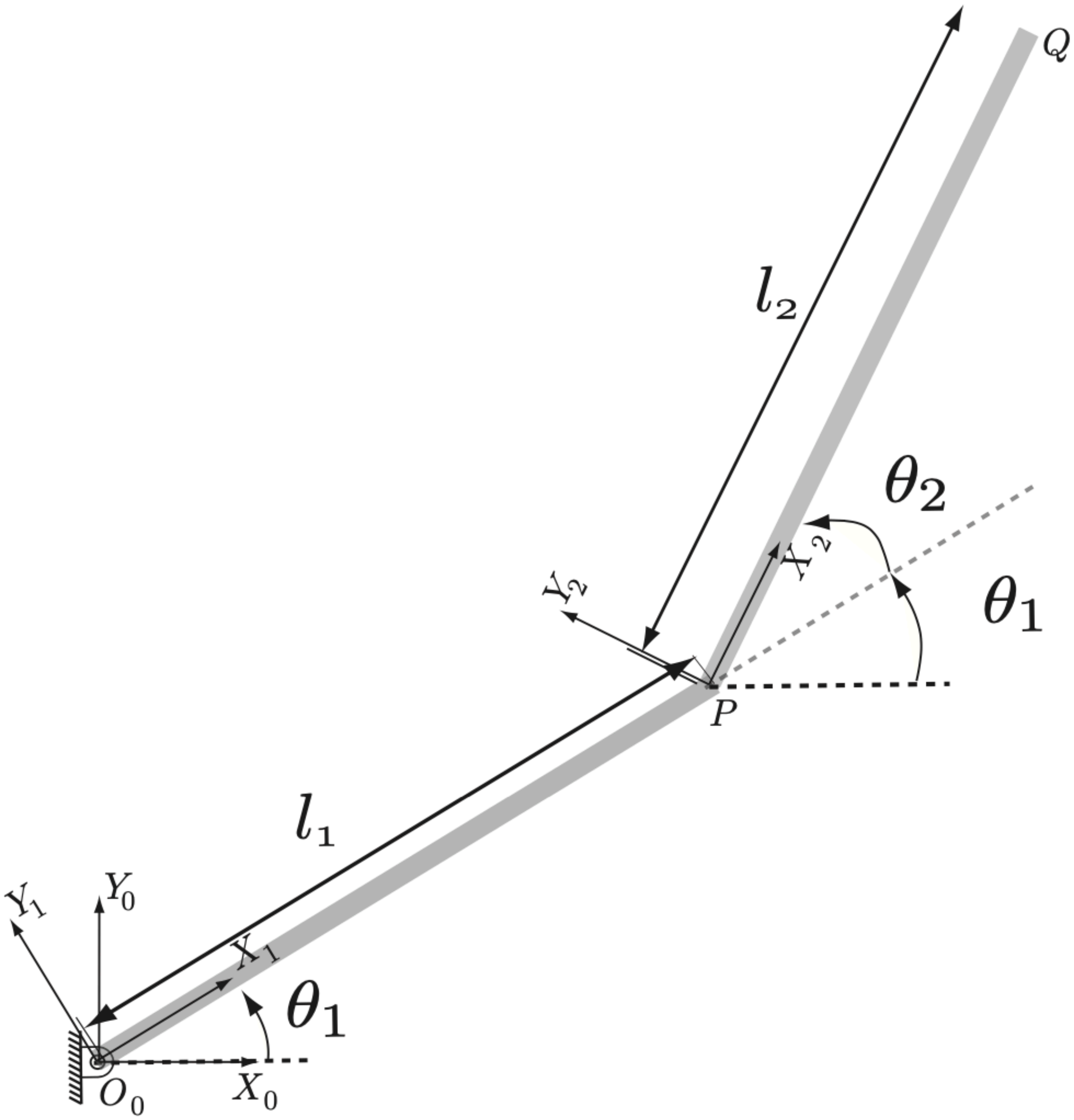
Position of Q

$$r_Q = f(q)$$

Velocity of Q

$$V_Q = \frac{\partial f}{\partial q}\dot{q} = J\dot{q}$$

Lets check this



MuJoCo: Inverse kinematics (3)

Velocity of Q

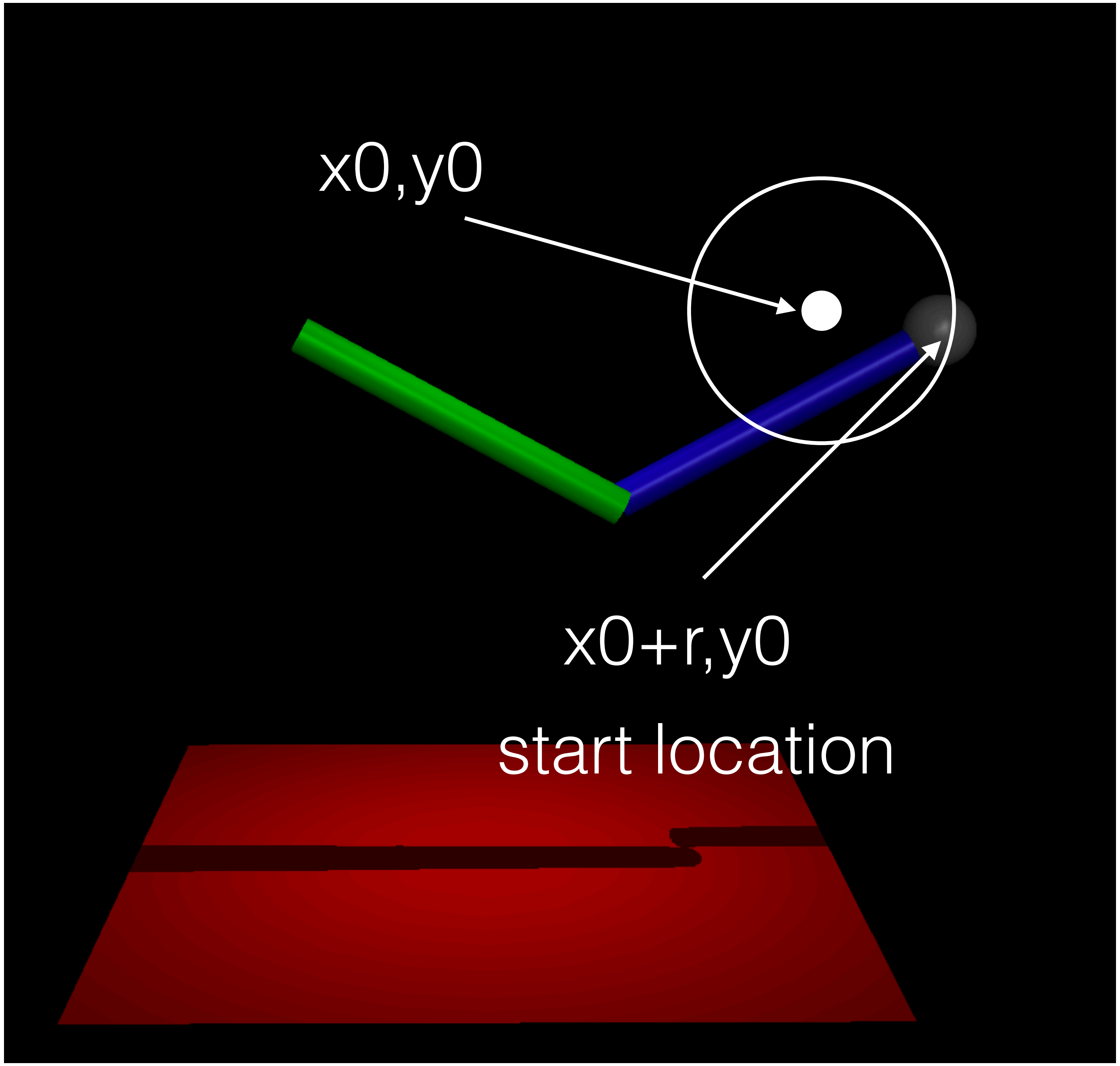
$$\left.\begin{aligned} &V_Q = J\dot{q} \\ &\frac{dr_Q}{dt} = J\frac{dq}{dt} \\ &\Delta r_Q = J\Delta dq \end{aligned}\right) \quad \text{Different ways of writing the same thing}$$

Key equation

$$\Delta dq = J^{-1}\Delta r_Q$$

Given: Delta r_Q (end-effector change),
compute Delta q (joint angle change)

MuJoCo: Draw a circle(4)



MuJoCo: Jacobian/Inverse kinematics (5)

- Summary of functions learnt
  - Locate point of interest; site, access position/velocity via sensors
  - Jacobian: mj_jac
  - Compute kinematics/dynamics: mj_forward
  - Compute kinematics/dynamics/integrate: mj_step