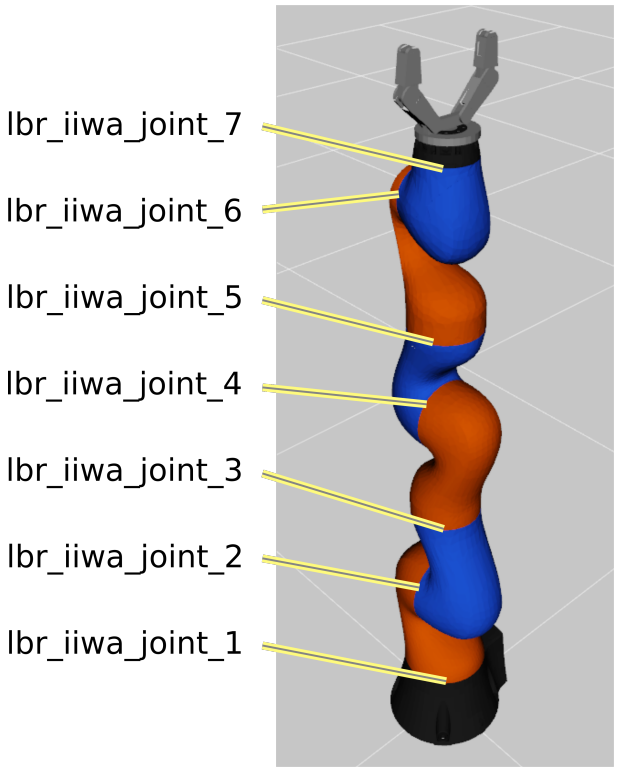
Comp 765 Assignment 1

Jonathan Pearce 260672004

April 17, 2020

Problem 1a For this question I modified the dynamics of the Kuka robot arm (pictured below) from the Kuka Gym Envirnoment in Pybullet [1]. This model is a representation of the Kuka LBR iiwa robot arm designed for applications with small robots. This arm has 7 controlled axes and the gym environment has added a gripper to the arm for interaction with objects. For this question I increased the weight of the linkages of the Kuka arm by a factor of 5 and examined how quickly the gripper dropped from an initial position due to gravity.



Problem 1b For every joint on the Kuka robot, there is a position and orientation. Therefore the state of a joint on the robot arm can be expressed as follows,

$$\mathbf{q} = [x, y, z, \theta, \phi, \psi]$$

Where (x, y, z) are the 3D position coordinates of the joint and (θ, ϕ, ψ) are the orientation coordinates of the joint (roll, pitch and yaw respectively). Every joint on the Kuka robot with a motor can receive a control input specifying the output torque. Therefore the action of a joint on the robot arm can be expressed as follows,

$$\mathbf{u} = [\tau]$$

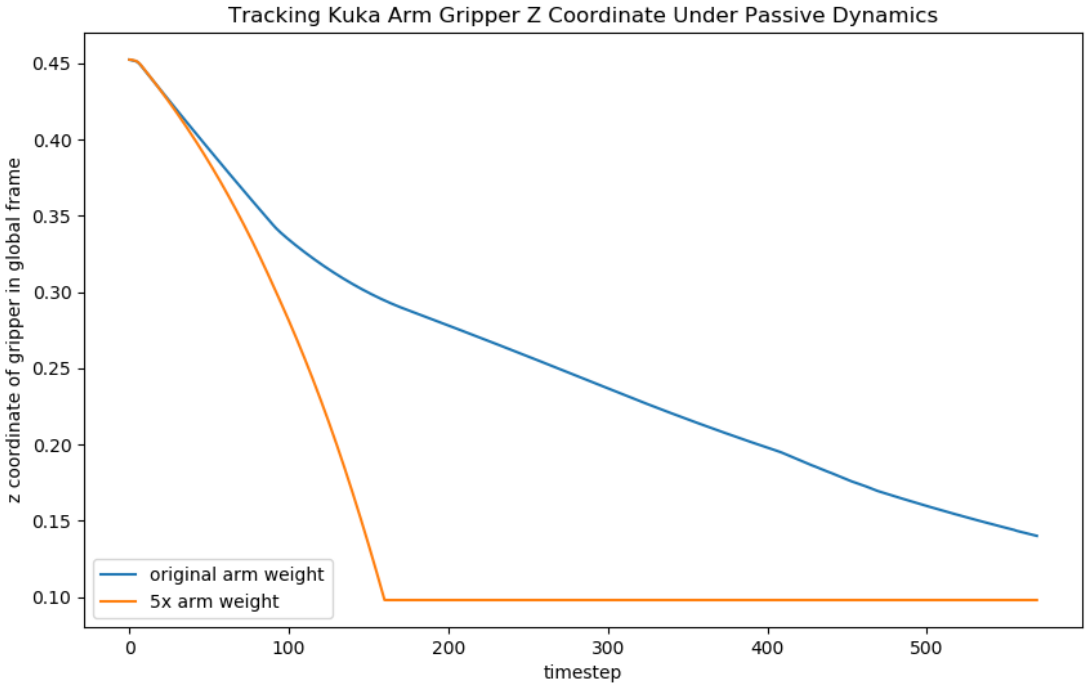
Where τ is the torque applied to the joint. As mentioned before there are 7 controllable axes on this arm therefore the state and action vectors above could be indexed by their joint index and combined to form one state and action vector for the entire robot arm.

Problem 1c The Kuka robot arm has 7 controlled axes, each controlled by a separate motor and requiring independent input to determine output joint torque at the particular joint. Therefore at each joint we can use the manipulator equation to understand the robot arm's motion from that joint to the gripper. The manipulator equation is as follows [2][3]

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{u}$$

Where $\mathbf{H}$ is the inertial matrix. $\mathbf{C}$ is the Coriolis force matrix. $\mathbf{G}$ is the potential matrix, which models things such as gravity.

Problem 1d For most manipulator systems the mass of the entire manipulator (or part of the manipulator) is a variable in the matrix entries in all three of the matrices mentioned above, $\mathbf{H}, \mathbf{C}$ and $\mathbf{G}$ (for example see [2]). Therefore a complete quantitative analysis checking whether my observed data matches the dynamics equations would be difficult. However consider $\mathbf{G}$, the potential matrix, which models things such as gravity. Under passive dynamics (which we used for the plot below) $\mathbf{G}$ can explain a lot. The force of gravity acting on the robot arm is a function of the mass of the arm (Newton's law of gravitation [4]). The greater the mass of the Kuka robot arm the greater the force of gravity applied to the arm will be, therefore pulling the arm towards the ground faster under passive dynamics. This result is confirmed in the plot below, the heavier robot arm drops to the ground faster under passive dynamics than the original unedited Kuka robot arm.



Problem 2a We are looking to find matrices A and B such that,

$$\dot{x} = Ax + Bu$$

Therefore we need to linearize our dynamics equations near the balance point $x = [0, 0, 0, \pi]^T$. In the provided dynamics equations the non-linear terms are $\sin\theta$, $\cos\theta$ and $\dot{\theta}^2$. We can use the Taylor expansions of sin and cos at $\theta = \pi$, and then take only the constant and linear terms.

$$\sin\theta = (\pi - \theta) + O((\pi - \theta)^3) \approx \pi - \theta$$

$$\cos\theta = -1 + O((\pi - \theta)^2) \approx -1$$

One problem arises with the sin θ approximation. Consider the dynamics equations for $\ddot{\theta}$, substituting in the sin approximation would result in the following term in the numerator,

$$2(M+m)g\sin\theta = 2(M+m)g\pi - 2(M+m)g\theta = \eta - 2(M+m)g\theta$$

Where η is a constant. To incorporate this constant into our equation we would need $x = [0, 0, 0, \pi, 1]^T$, where the 5th element is a dummy variable to allow for constants to be represented in the matrix vector calculations. However we can avoid doing this by simply translating the balance point by π. This leads to us solving for our control with a balance point of $x = [0, 0, 0, 0]^T$. This is done on line 86 of my solution code. With this new balance point we can again use the Taylor expansions of sin and cos at $\theta = 0$, and then take the constant and linear terms.

$$\sin\theta = \theta + O((\pi - \theta)^3) \approx \theta$$

$$\cos\theta = 1 + O((\pi - \theta)^2) \approx 1$$

At the balance point we have $\dot{\theta}$=0, therefore we simply approximate $\dot{\theta}^2$ as follows,

$$\dot{\theta}^2 \approx 0$$

Using these linear approximations at the balance point we can simplify the dynamics equations as follows,

$$\ddot{x} = \frac{2ml\dot{\theta}^2\sin\theta + 3mg\sin\theta\cos\theta + 4(u - b\dot{x})}{4(M+m) - 3m\cos^2\theta}$$

$$= \frac{0 + 3mg\theta + 4(u - b\dot{x})}{4(M+m) - 3m}$$

$$= \frac{3mg\theta + 4u - 4b\dot{x}}{4M + m}$$

$$= \frac{3mg\theta}{4M+m} - \frac{4b\dot{x}}{4M+m} + \frac{4u}{4M+m}$$

And,

$$\ddot{\theta} = \frac{-3(ml\dot{\theta}^2\sin\theta\cos\theta + 2((M+m)g\sin\theta + (u - b\dot{x})\cos\theta))}{l(4(M+m) - 3m\cos^2\theta)}$$

$$\ddot{\theta} = \frac{-3(0 + 2((M+m)g\theta + (u - b\dot{x})))}{l(4(M+m) - 3m)}$$

$$\ddot{\theta} = \frac{-6(M+m)g\theta - 6(u - b\dot{x})}{l(4M+m)}$$

$$\ddot{\theta} = \frac{-6(M+m)g\theta - 6u + 6b\dot{x})}{l(4M+m)}$$

$$\ddot{\theta} = \frac{-6(M+m)g\theta}{l(4M+m)} + \frac{6b\dot{x}}{l(4M+m)} - \frac{6u}{l(4M+m)}$$

With the now linearized dynamics equations we can write out A and B

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \ddot{\theta}\\ \dot{\theta}\end{bmatrix} = A\begin{bmatrix}x\\ \dot{x}\\ \dot{\theta}\\ \theta\end{bmatrix} + Bu = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & \frac{-4b}{4M+m} & 0 & \frac{3mg}{4M+m}\\ 0 & \frac{6b}{l(4M+m)} & 0 & \frac{-6(M+m)g}{l(4M+m)}\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ \dot{\theta}\\ \theta\end{bmatrix} + \begin{bmatrix}0\\ \frac{4}{4M+m}\\ \frac{-6}{l(4M+m)}\\ 0\end{bmatrix}u$$

Q and R are derived experimentally, this procedure is outlined below in 2b.

Problem 2b During experimentation, A and B were kept fixed to the derived form from question 2a. I then proceeded to find appropriate Q and R.

Trail 1: Start with identity Q and R

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R = [1]$$

Performance: Pole fell over very quickly, robot drifted from position $x = 0$

Trail 2: Try to encourage little to no angular velocity

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R = [1]$$

Performance: Pole stays balanced! But cart drifts a little bit away from position $x = 0$

Trail 3: Try to encourage cart to stay centered at $x = 0$

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R = [1]$$

Performance: Cart barely moves from $x = 0$ and pole stays balanced

Problem 2c Substituting in the physical constants of the system, we can explicitly write out

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.16 & 0 & 5.892 \\ 0 & 0.480 & 0 & -47.136 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 1.6 \\ -4.8 \\ 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R = [1]$$

Important note is that when computing the state of our robot x, I subtract π from the angle θ, as discussed in 2a. This can be seen on line 86 of my code. On the next page there is a plot showing the balancing success. There is a brief period of oscillation (both in position and angle) which quickly stabilizes. After stabilizing the LQR method has great success at balancing and does not suffer from any position drift.

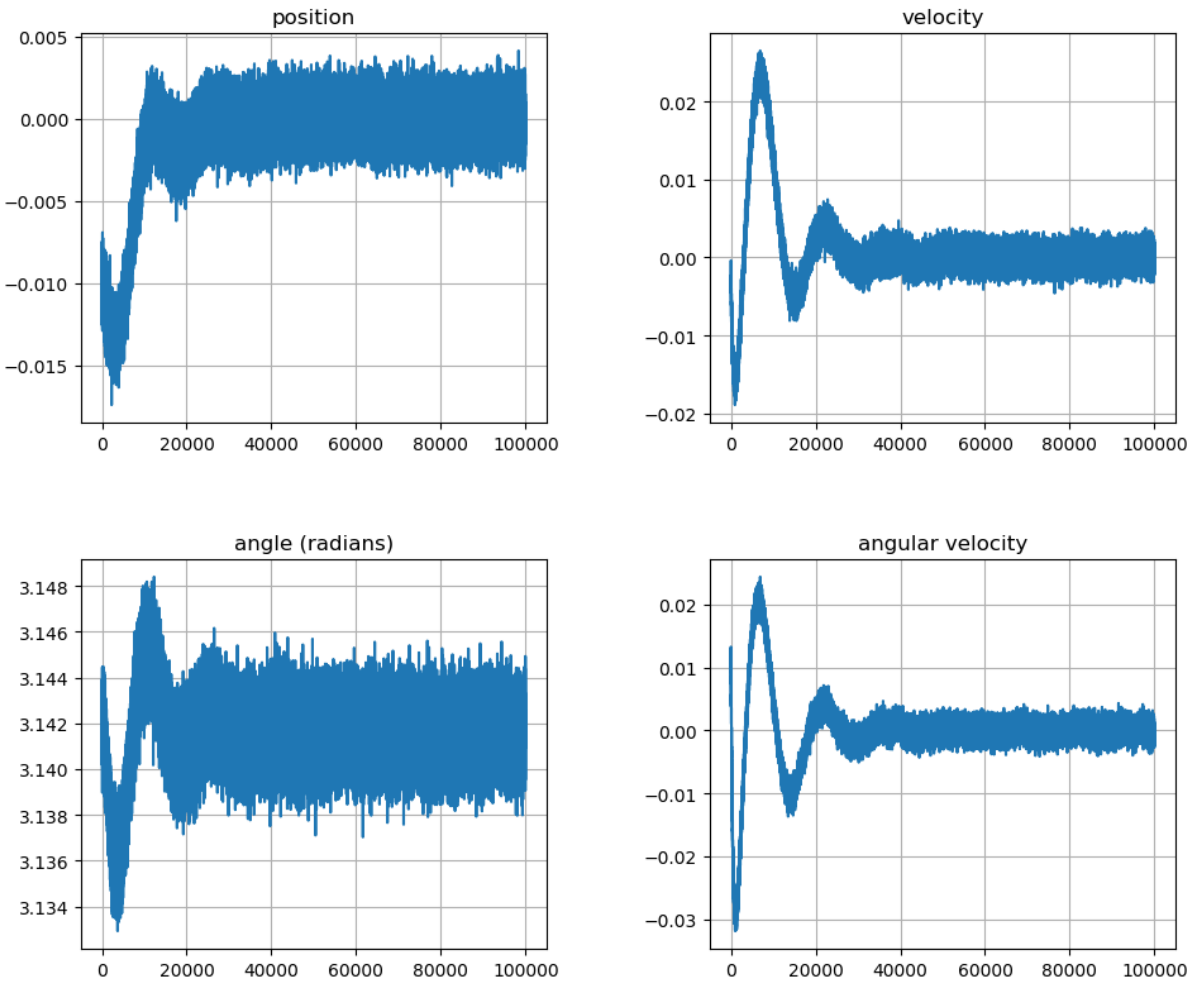


Figure 1: LQR Data

Problem 3 For question 3 I attempted solve the swing up task with Value Iteration as well as Prioritized Sweeping [6]. I was unable to achieve a successful swing up using either of these methods because the discretized state space was too large to work with in reasonable time. The results I present below are from the simple balancing task of question 2 where the pole starts upright. These results are presented to demonstrate successful implementations of both value iteration and prioritized sweeping.

The Value iteration method takes approximately 5 minutes to converge and the prioritized sweeping method takes 2 minutes to complete the specified number of value update iterations. The most significant observation of the results below is the oscillation that is prevalent in both methods. This oscillation is caused by the action space (and state space) being discretized, which makes it difficult or impossible for the cartpole to become almost completely stable like in question 2. It appears that prioritized sweeping was able to keep slightly better control of the cartpole. The prioritized sweeping method experiences less position drift, less angular drift and the magnitude of the deviation of the velocities is smaller when compared to value iteration. This makes a lot of sense since prioritized sweeping focuses more on learning about the values of states near the goal (and in this case the start position) where as value iteration computes updates uniformly over the discretized state space. If I had more time I would definitely like to play around with Prioritized Sweeping more, as that seems to be a promising method to learn the swing up procedure in reasonable time.

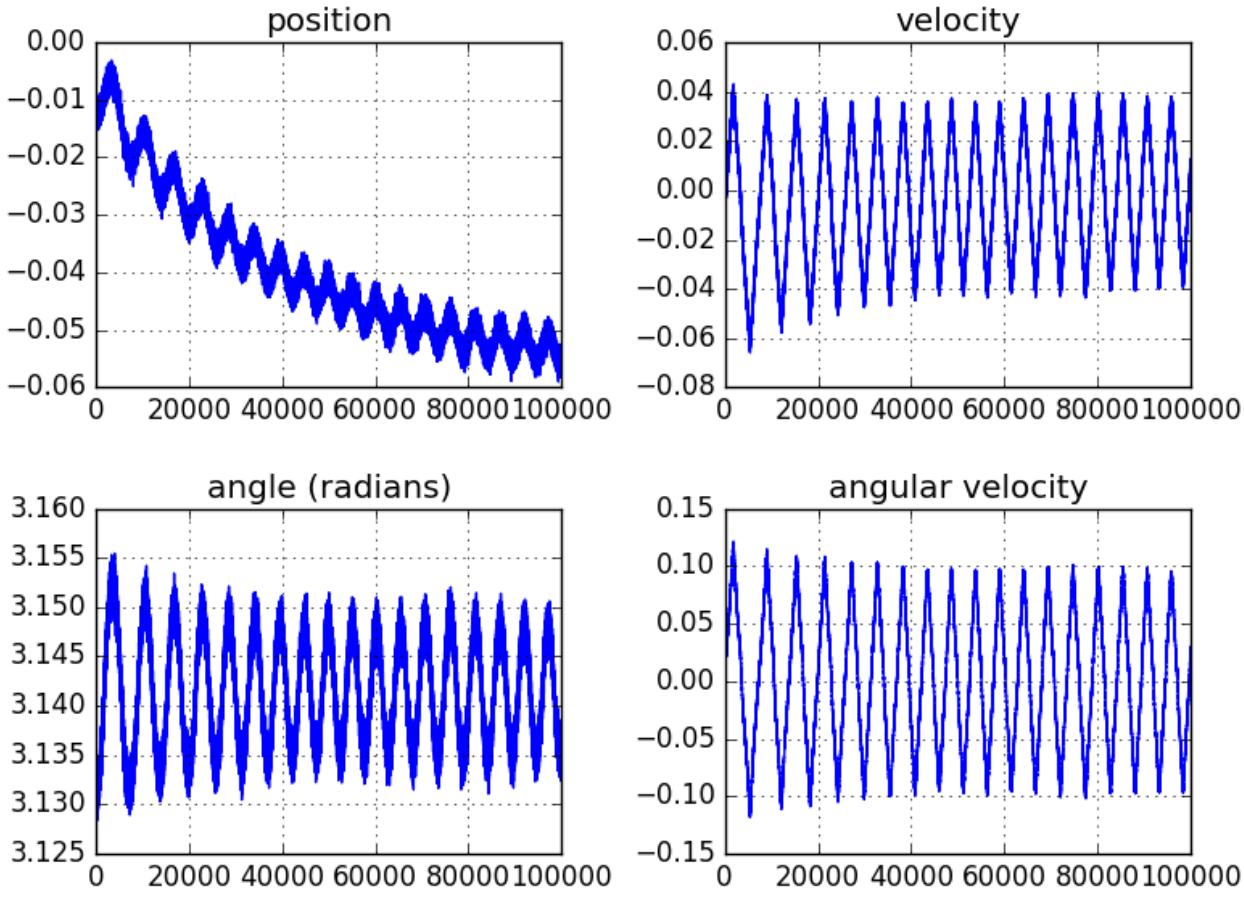


Figure 2: Value Iteration Data

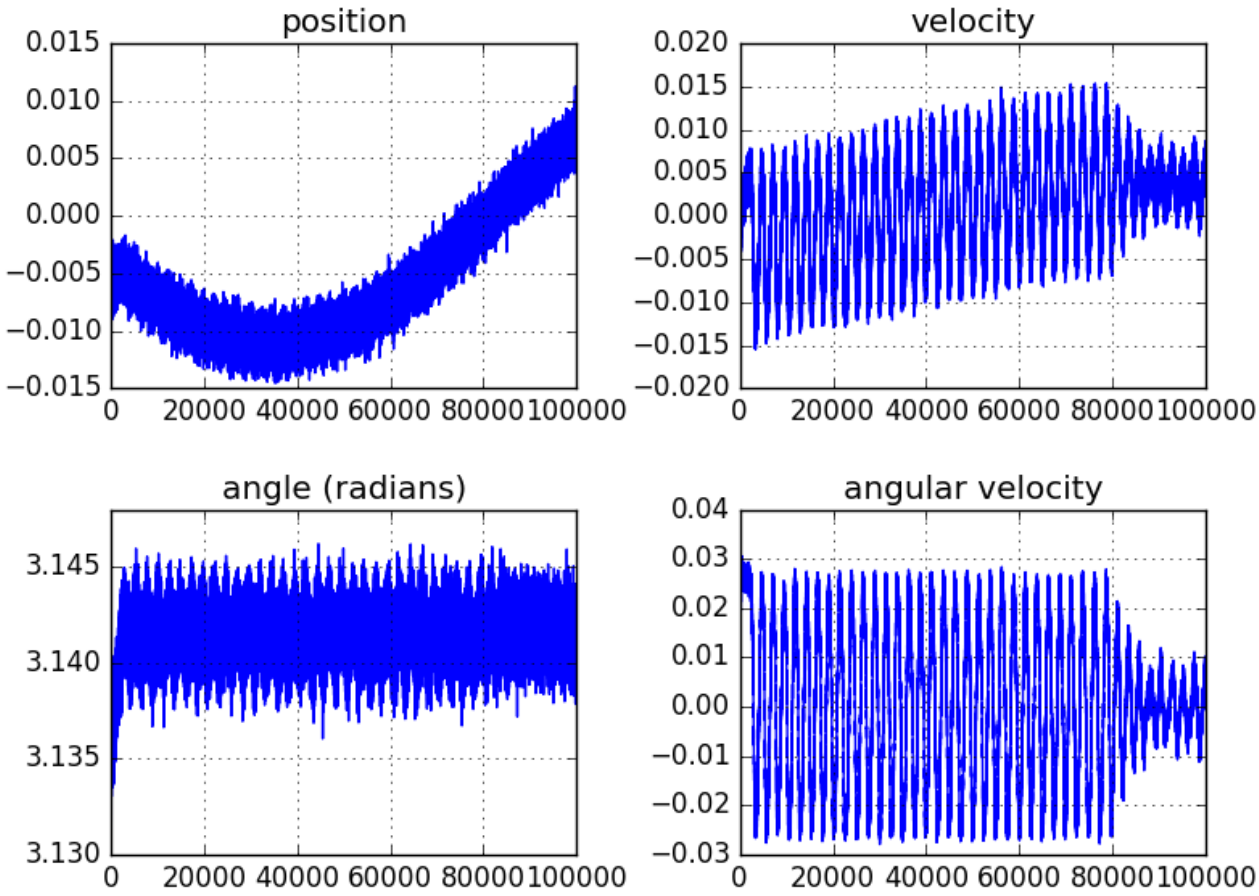


Figure 3: Prioritized Sweeping Data

References

1. https://github.com/bulletphysics/bullet3/blob/master/examples/pybullet/gym/pybullet_envs/bullet/kukaGymEnv.py
2. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-832-underactuated-robotics-s readings/MIT6_832s09_read_appA.pdf
3. https://ethz.ch/content/dam/ethz/special-interest/mavt/robotics-n-intelligent-systems/rsl-dam/documents/RobotDynamics2017/7-dynamics3_ex.pdf
4. https://en.wikipedia.org/wiki/Newton%27s_law_of_universal_gravitation
5. https://people.eecs.berkeley.edu/~pabbeel/cs287-fa19/slides/Lec3-discretization-of-continuous-state-sp pdf
6. https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume4/kaelbling96a-html/node30.html