KDC Alternative Project

Canbo Ye
canboy@cs.cmu.edu

Yiming Zuo
yzuo@andrew.cmu.edu

Zhipeng Bao
zbao@cs.cmu.edu

Abstract

The real robot control system inevitably faces noise interference problems, which greatly limit their performance in advanced manufacturing processes. In this project, we target at trajectory tracking problem with robot arms at noisy environment. We tried different denoise methods together with different controllers to improve the accuracy, robustness and response speed of our control system. We built a well-performing system robust with additional noises and evaluated its performance in different settings with simulation. Further analyses and detailed design of our system are also provided.

1. Introduction

Robotic system control plays an important role in many advanced manufacturing processes. Such systems have the potential to improve speed, robustness and precision for these processes. In real world, however, noise always exist due to the model design and sensor measurements [6]. If not addressed properly, the noise can greatly harm the control system performance. Hence in this project, we focus on robot arm trajectory tracking task and hope to explore possible approaches to address or mitigate the problem of noise.

The robotic arm trajectory tracking problem is modeled as follows: Given a set of desired time-position trajectory pairs, we want to control a robot manipulator to follow the desired path as close as possible. The task is usually called **path following** if we only pay attention to the closeness of two set of points. When we considered the timestamp, the problem is called **trajectory tracking**. There has been tons of study for both problems with different control strategies and robot types [7, 8, 2, 9]. In this paper we do not differentiate these two terms and we provide qualitative results for both in Section 3.

2. Model

Given a set of desired trajectory points for end-effector, we first use **Inverse Kinematics (IK)** to get the desired poses. We then use a control system based on the robot arm **Equations of Motions (EOM)** to find the torques for each joint. Finally, we use **Forward Kinematics (FK)** to get the actual trajectory. Besides, sensor noise would inevitably exist and we will model them accordingly. We will introduce each individual model and show the overall pipeline in the rest of this section.

2.1. Forward Kinematics

We use Screw Theory [3] to apply forward kinematics and get the position based on the equation: $g_{st}(\theta) = e^{\hat{\xi}\theta} g_{st}(0)$, where $\hat{\xi}$ is the joint twist, θ is the joint angle and $g_{st}(0)$ is the transformation matrix of the joint at zero configuration.

2.2. Inverse Kinematics

We implement inverse kinematics by using the following equation iteratively:

$$\dot{\theta} = \mathbf{J}^{\dagger}\mathbf{V} = \mathbf{J}^{\dagger}\begin{bmatrix} p_{error} \\ R_{error} \end{bmatrix}, \theta_{t+1} = \theta_t + \Delta t * \dot{\theta} \quad (1)$$

where $\mathbf{J}$ is the Geometric Jacobian matrix, and $\dagger$ means pseudo inverse. p_{error} and R_{error} measure the error of current position and orientation.

2.3. Equations of Motions

We calculate $M(\theta)$, $C(\theta, \dot{\theta})$ and $N(\theta, \dot{\theta})$ with:

$$M(\theta) = \sum_{i=1}^{N} J_i^T M_i J_i, N(\theta, \dot{\theta}) = \frac{\partial}{\partial \theta}(\sum_{i=1}^{N} m_i g h_i(\theta)) \tag{2}$$

$$C_{i,j} = \frac{1}{2} \sum_{k=1}^{N} (\frac{\partial M_{ij}}{\partial \theta_k} + \frac{\partial M_{ik}}{\partial \theta_j} - \frac{\partial M_{kj}}{\partial \theta_i}) \dot{\theta}_k \tag{3}$$

2.4. Decentralized Control

Decentralized control system fine tunes the coefficient of controller for each θ separately.

System Modeling and PID Controller Tuning The diagram for parameter tuning stage is shown in Appendix. We use PID control for this model and made some approximations. The robot EoMs are given by $\tau = M\ddot{\theta} + C\dot{\theta} + N$. If we ignore the term C and N and treat it as a linear system, we could use a PID controller at each individual joint and tune each controller parameters.

Decentralized Control with Gravity Compensation We then run control simulation with our tuned controller. Please refer to the appendix for the diagram of the whole system. We estimate the torque for countering gravity and compensate for it.

Adding Sensor Noise We add AWGN to θ and have tried several de-noising methods such as low-pass filter and observer. Detailed are discussed in Section 4.

2.5. Centralized Control

We implement Inverse Dynamic Control as centralized control method. From the robot arm EoM we know that

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta) \tag{4}$$

$$= M(\theta)\bar{z} + Q(\theta, \dot{\theta}) \tag{5}$$

We use $Q(\theta, \dot{\theta})$ as **feedback linearization** term and $\bar{z}$ is the feedback signal.

For the **feedback error compensation** part we use a PID controller. For the centralized control diagram and choice of the control parameters please refer to the Appendix.

2.5.1 Centralized Control with Sensor Noise

We assume we only have the observation of joint angles, $\bar{\theta}$. And this observation is not perfect. We model the sensor noise as an **Additive White Gaussian Noise Channel** (AWGN Channel). We implemented different denoising strategies and describe them here.

Butterworth Filter

Butterworth Filter is a kind of filter that aims to get as flat as possible frequency response in passband. Its frequency response is:

$$G^2(\omega) = |H(j\omega)|^2 = \frac{G_0^2}{1 + \left(\frac{j\omega}{j\omega_c}\right)^{2n}} \tag{6}$$

where G_0 is the DC gain (which is set to 1), ω_c is the cutoff (-3dB) and n is the filter order.

Luenberger Observer Since we only have access to the joint angle θ and we are interested in estimating joint velocity $\dot{\theta}$ which is required in feedback linearization, we need to find a way to estimate it.

Luenberger Observer is a practical way to estimate the system internal state given its input and output. It runs a forward dynamics simulation model in parallel to the physical robot. This estimation results can never be perfect because of sensor noise and model inconsistency. So we uses the error between observed output and simulation output to construct a feedback signal for the simulation model.

In its simplest form we have:

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) \\ \hat{y} &= C\hat{x} + Du \end{aligned} \tag{7}$$

2.6. Motor Noise

There are always discrepancy between desired torque and output torque for eclectic motor. Reasons can be mechanical (worn bearings, a bent shaft, etc.) and electrical (phase imbalance). We model this motor noise as an AWGN channel and study its effect on the stability of our control system.

3. Experiment Results

In this section, we evaluate the control models developed in section 2. We used MATLAB Simulink to build and test our control systems and Robotics Toolbox to build the physical arm robot. We build two

Metrics	Decentralized	Centralized
HD	48.61	4.86
FD	28.52	5.06
MSE	695.09	53.04

Table 1: Comparison of Control Strategies

Metrics	w/o noise	observer	filter
HD	4.86	12.05	14.06
FD	5.06	12.59	14.52
MSE	53.04	93.39	72.56

Table 2: Comparison of Noise Filtering Methods

physical model of the robot arm including one with estimated parameters for controller test and the other with real parameters for simulation.

We also evaluate the robustness of the control system by adding noises. We mainly consider two source of noise: sensor noise during measurement and motor noise, which models the discrepancy between desired torque and actual torque.

3.1. Qualitative Results

We compare the real trajectory of our best model with the ground-truth. Please find more results in the Appendix.

3.2. Quantitative Results

3.2.1 Evaluation Metrics

For the quantitative evaluation, we measure the closeness of the real trajectory to the desired one. We apply three commonly used metrics: the Hausdorff Distance, the Fréchet Distance and Mean Squared Error.

Hausdorff Distance (HD) is a method to measure how far apart two subsets of metric space are from each other[5]. The one-way modified Hausdorff Distance [4] is defined as:

$$d_H(X,Y) = \sup_{x\in X} \inf_{y\in Y} d(x,y) \tag{8}$$

Fréchet Distance (FD) measures the similarity between two curves [1] considering both location and timestamps along the curves:

$$d_F(X,Y) = \inf_{\alpha,\beta} \max_{t\in[0,1]} \{d(X(\alpha(t)), Y(\beta(t))\} \tag{9}$$

$\alpha(t)$ and $\beta(t)$ are the parameterization that connect two corresponding points from the two curves.

Mean Square Error (MSE) measures averaged L_2 distance between the two trajectories.

3.2.2 Results

Table 1 shows the performance of centralized model compared with decentralized model with zero noise. As we can see, the centralized model is significantly better. Table 2 shows the performance of different de-noising strategies.

3.2.3 Ablation Studies

We do ablation studies to find how the choice of individual parameters can affect the system performance. Please find the ablation studies in the Appendix.

4. Discussion

4.1. Decentralized Control

Estimated v.s. Real Parameters Fig. 1 and Fig. 2 compare the result from the estimated parameters and real parameters. The overall trajectory in Fig. 2 works well but has some skews at the beginning. We also further monitor the individual θ error, which is shown in Fig. 6. θ_2 and θ_1 have the largest error which is maybe because these joints are more meticulous and more sensitive to the estimation errors.

Noise Control The de-noising process failed. We analyzed the reasons in the Appendix.

4.2. Centralized Control

We mitigate the additional noises for the centralized control system and find:

- PID controller could reduce the skew of the trajectory since the integral component restrains the progression of errors.
- first-order Butterworth filter could best filter out the noisy components. Lower cut-off frequency generally helps since the configuration of robot arm changes slowly.
- Luenberger Observer helps mitigate the noise by smoothing the signals. But the performance is not good under large-noise conditions.

References

[1] H. Alt and M. Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995. 3

[2] C. C. de Wit, N. Fixot, and K. Astrom. Trajectory tracking in robot manipulators via nonlinear estimated state feedback. *IEEE Transactions on Robotics and Automation*, 8(1):138–144, 1992. 1

[3] F. M. e. Dimentberg. *The screw calculus and its applications in mechanics*. Foreign Technology Division, 1969. 1

[4] M.-P. Dubuisson and A. K. Jain. A modified hausdorff distance for object matching. In *Proceedings of 12th international conference on pattern recognition*, volume 1, pages 566–568. IEEE, 1994. 3

[5] F. Hausdorff. *Felix Hausdorff-Gesammelte Werke Band III: Mengenlehre (1927, 1935) Deskripte Mengenlehre und Topologie*, volume 3. Springer-Verlag, 2008. 3

[6] N. Jakobi, P. Husbands, and I. Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *European Conference on Artificial Life*, pages 704–720. Springer, 1995. 1

[7] H. Moulin and E. Bayo. On the accuracy of end-point trajectory tracking for flexible arms by noncausal inverse dynamic solutions. 1991. 1

[8] P. Ouyang, W. Zhang, and M. M. Gupta. An adaptive switching learning control method for trajectory tracking of robot manipulators. *Mechatronics*, 16(1):51–61, 2006. 1

[9] A. Visioli and G. Legnani. On the trajectory tracking control of industrial scara robot manipulators. *IEEE transactions on industrial electronics*, 49(1):224–232, 2002. 1

A. Design Details

A.1. Decentralized Control

Fine-tune Stage The model design of fine-tune stage of decentralized system is shown in Fig. 1.
Overall system

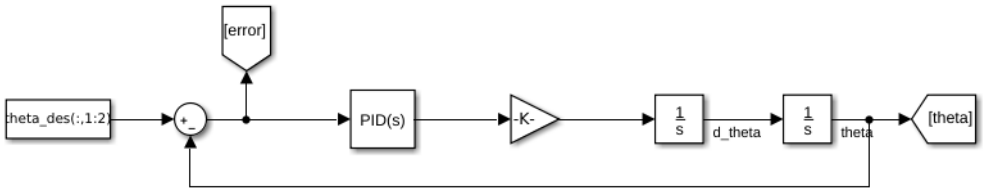


Figure 1: Fine tune stage of decentralized control model.

For the multiply unit in the figure, we first model the mass matrix $M(\theta)$ as $M(\theta) = \bar{M} + \Delta M(\theta)$. And we calculate $\bar{M}$ by averaging the mass matrix over the whole trajectory. Then the multiply unit is initialized with m_i, the i^{th} diagonal element of $\bar{M}$.

And the tuned parameters for PD controller are:

$$k_p = [15.747, 8.699, 5.498, 10.484, 5.422, 8.827, 0.419] \tag{10}$$
$$k_i = [0.400, 0.400, 0.400, 0.400, 0.400, 0.400, 0.400] \tag{11}$$
$$k_d = [10.478, 3.258, 2.317, 5.251, 1.967, 1.988, 1.952] \tag{12}$$

Simulation Stage We send the torque combined with a gravity torque estimated by the given parameters to the real manipulator. The model design of this stage is shown in Fig. 2.

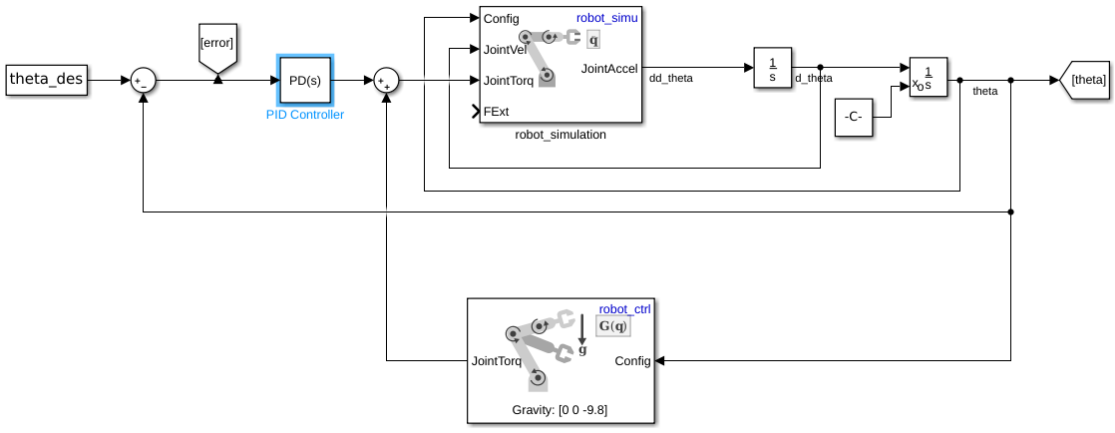


Figure 2: Simulation stage of decentralized control model.

Noise Control

All the de-noise modules working well for centralized control system failed in de-centralized control system. We analyzed and thought the reasons may be related to:

- The compensated gravity torque and coefficients of PID controller are calculated by the estimated parameters, which is not precise. The error could be further amplified when we add noises.
- We have several approximations for the total torques. These approximations may not hold with additional noises.
- The parameters of decentralized control is tuned individually so that they are sensitive to noises and the change of each parameter will have a great influence on the final result.
- The parameters of the de-noise filters may not reach the best.

A.2. Centralized Control

A.2.1 Control Diagram

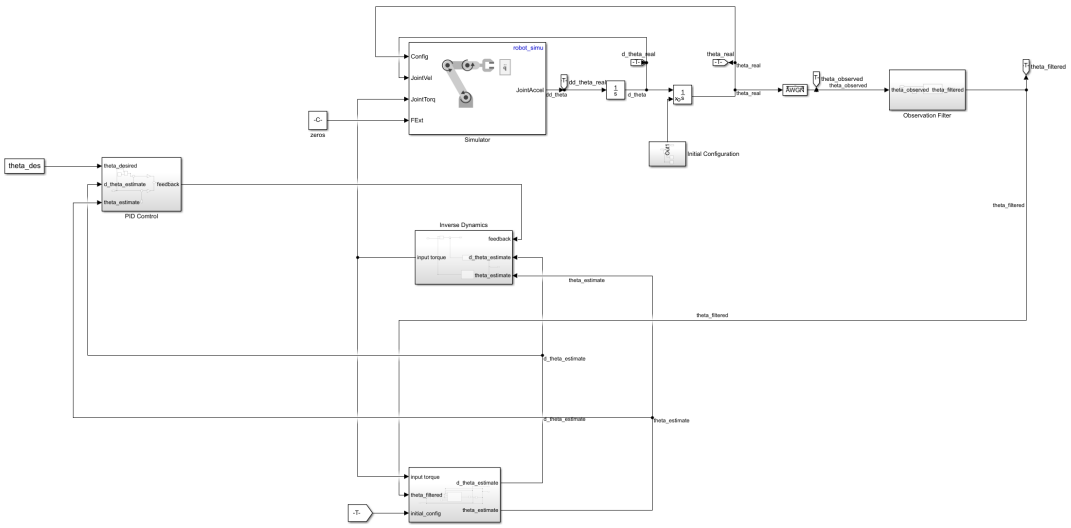


Figure 3: Centralized Control Diagram

A.2.2 Control Diagram

For the PID controller, we use $K_p = 10, K_D = 10, K_I = 1$. We use Butterworth Filter with cutoff frequency $20Hz$ and order $n = 1$ for the Observation Filter.

B. Qualitative results

B.1. Decentralized Control

Figure 4 and Figure 5 shows real trajectory for the decentralized model with : 1) manipulator with estimated robot model; 2) manipulator with ground-truth robot model.

Figure 6 shows the individual error for each θ. We could see the second and first joint have the largest error.

Figure 4: Control with estimated model parameters

Figure 5: Control with real model parameters

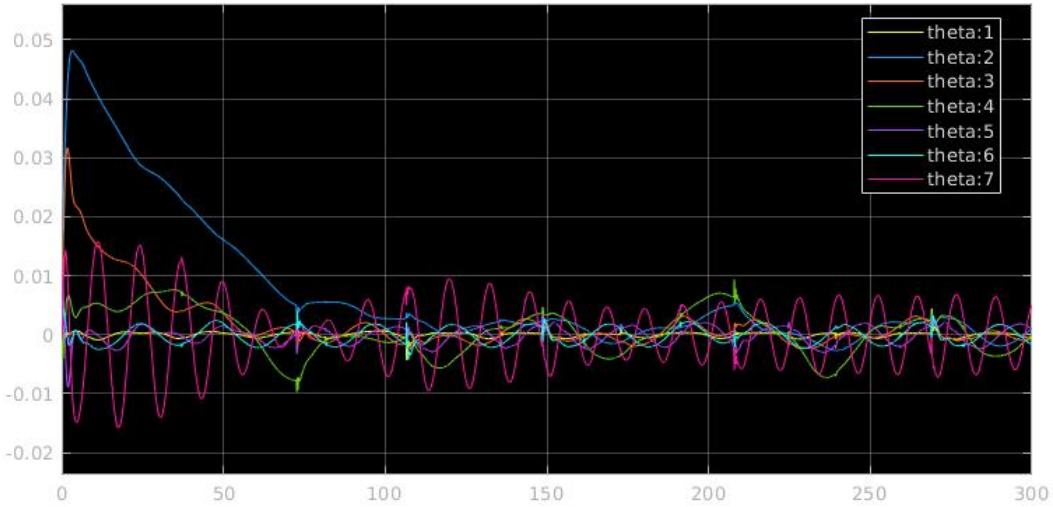


Figure 6: θ error for decentralized model.

B.2. Centralized Control

Figure 7 shows the visualization results of centralized control in different setting. Figure (a) is the real trajectory in noise-free situation, which is close to the desired trajectory. When we add noise to the system, we use either Luenberger Observer or Butterworth filter to mitigate its impact. Figure (b) shows the results using Luenberger Observer with sensor noise while Figure (c) using Butterworth filter. We could see that the trajectory of model using Luenberger Observer in (b) is more stable than using Butterworth filter in (c). We then add motor noise to the system and the result of model using Luenberger Observer in such situation is shown in figure (d).

C. Ablation Studies

C.1. PD controler v.s. PID controler

If we only use PD controller, we find that the system will have accumulated errors but is unable to eliminate it, so we decide to use PID controller instead to address this problem. The quantitative results are shown in Table 3, we could see that the performance of the system is greatly improved after we add the integral component to the controller. We also visualize the trajectories to see the difference in Figure 8.

(a) Noise free

(b) Sensor noise added and using Luenberger Observer

(c) Sensor noise added and using Butterworth filter

(d) Motor&sensor noise added and using Luenberger Observer

Figure 7: Visualization results of the centralized control in different settings

Metircs	PD	PID
HD	49.64	12.05
MSE	1824	93.39

Table 3: The performance difference when considering motor noise

Filter order	1	2	3
HD	12.05	27.58	doesn't
MSE	93.39	143.25	converge

Table 4: Comparison of Filter Order

C.2. Choice of Butterworth Filter Parameters

The frequency response of Butterworth filter is:

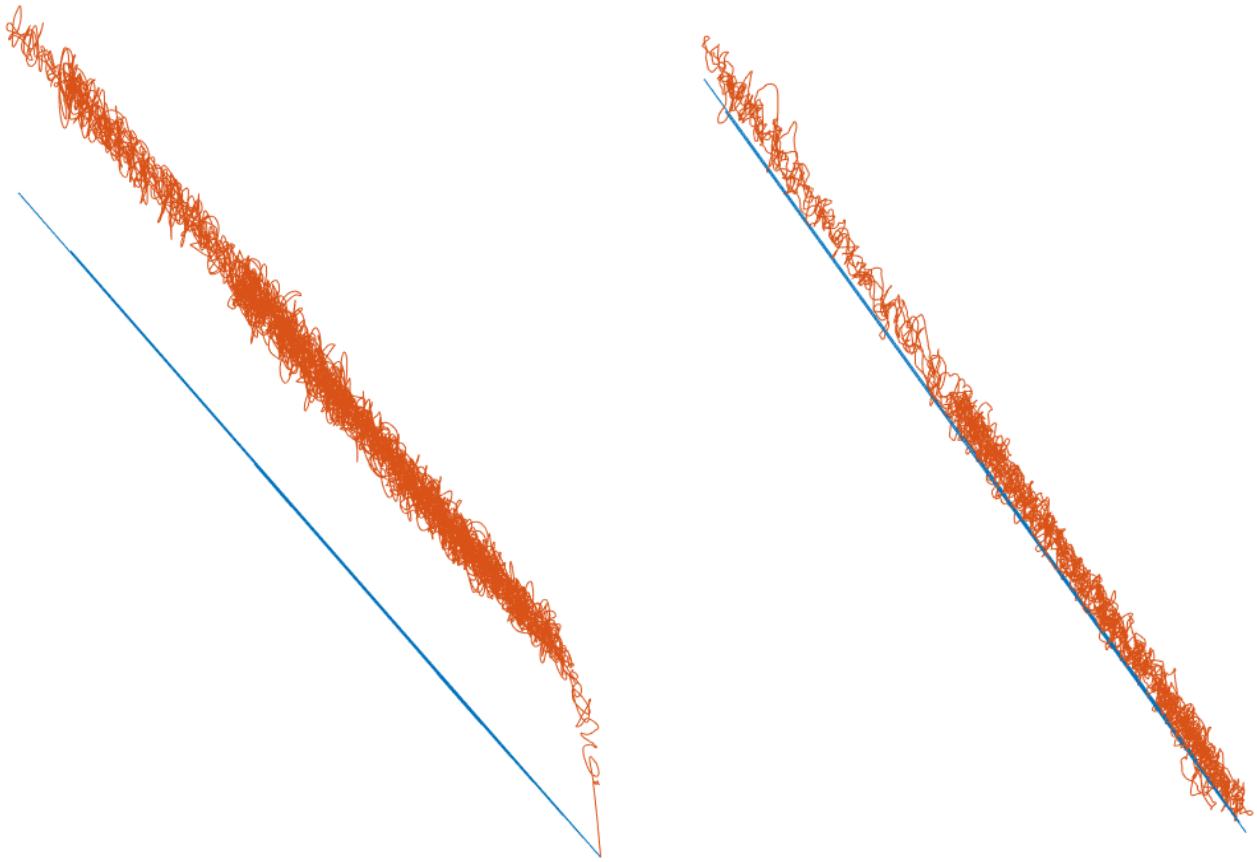
$$G^2(\omega) = |H(j\omega)|^2 = \frac{G_0^2}{1 + \left(\frac{j\omega}{j\omega_c}\right)^{2n}} \tag{13}$$

where ω_c is the cutoff frequency and n is the filter order. We experiment on the choice of ω_c and n. Results are shown in Table

As we can see from the results, The filter works best with cutoff frequency $\omega_c = 20Hz$ and order $n = 1$.

As for the cutoff frequency, a lower ω_c would filter out more noise power and increase SNR. But it also should not be set too low as we want all signal power to pass the filter.

As for the filter order n, typically a higher order Butterworth filter would have a flatter passband response and a larger cutoff slope. But it would also introduce larger phase shift and cause the system to be unstable.



(a) PD controller

(b) PID controller

Figure 8: Trajectory visualization using PD or PID controller

Filter ω_c (Hz)	5	20	200
HD	30.16	27.58	27.1274
MSE	107.35	143.25	85.0783

Table 5: Comparison of Filter Cutoff Frequency

Metircs	w/o motor noise	w motor noise
HD	12.05	13.94
MSE	93.39	101.95

Table 6: The performance difference when considering motor noise

C.3. The influence of motor noise

As we can see from table 6, the performance of the system remains almost the same even if we add motor noise, which indicates the robustness of our system. The visualization of the two trajectories is also shown in Figure 7b and Figure 8b.