Data-based Optimization Assignment 2

Tobias Wejbora
ID: 1785486

March 2026

Classical Extremum-Seeking Control Approach

Question 1

The ESC parameters were tuned on the physical test set-up by first testing the results from Assignment 1 and then making some modifications. The filter cut-off frequencies $\omega_{\text{HP}} = \omega_{\text{LP}} = \omega_f$ were chosen so that the timescale separation condition $\omega_f \ll \omega$ still holds and the dither amplitude a was made so that there was sufficient gradient signal. Also when tuning a, the steady-state oscillation had to be kept within the 0.1 rad bound. The integrator gain k was tuned to find a balance between convergence speed and stability.

As mentioned earlier, the parameters from Assignment 1 Question 7 were used initially:

$$k = 3000, \quad a = 7.5\,\text{rad}, \quad \omega = 2\pi \cdot 1.0 \approx 6.28\,\text{rad/s}, \quad \omega_f = 0.287\,\text{Hz} \approx 1.8\,\text{rad/s}$$

Multiple experimental runs were made with different parameter settings. Table 1 shows the parameters and convergence times for each run.

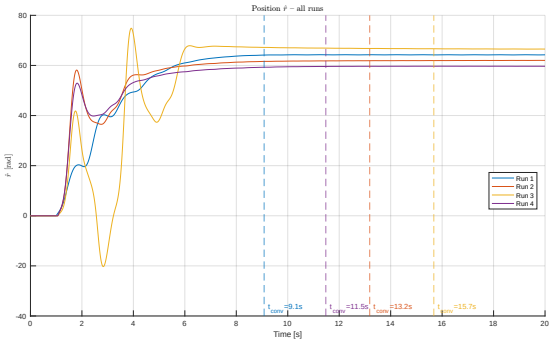
Table 1: Classical ESC experimental runs:

Run	k	a [rad]	ω [Hz]	ω_f [Hz]	ω_f [rad/s]	$\hat{r}_{\text{ss}}$ [rad]	J_{ss}	t_{conv} [s]
Run 1	3000	7.5	1.00	0.287	1.80	64.2	0.554	9.1
Run 2	4500	5.8	0.94	0.564	3.54	62.0	0.555	13.2
Run 3	6000	5.8	0.94	0.564	3.54	66.6	0.556	15.7
Run 4	4000	6.2	0.94	0.564	3.54	59.7	0.555	11.5

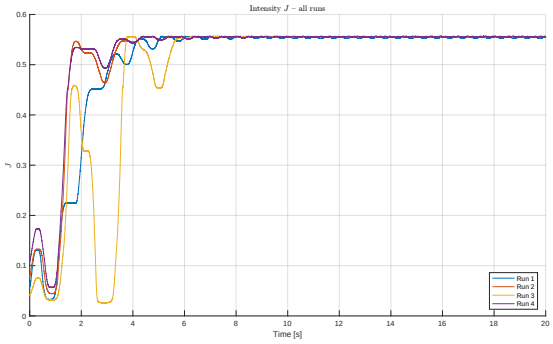
All experimental runs converged within 9 to 16 s ($|\hat{r}(t) - \hat{r}_{\text{ss}}| < 0.1$ rad) and run 1 ($k = 3000$, $a = 7.5$) had the fastest convergence at 9.1 s. This happened to be the same parameters as Assignment 1 Question 7 due to the fact that there was only time for four experimental runs. However, comparing run 1 with the rest, it can clearly be seen that run 1 benefits from a lower k value but a higher a value and half the ω_f.

Thus, the higher k in Runs 2–4 most likely caused the integrator to overshoot the optimum, and the lower dither amplitude of Run 2–4 created a weaker gradient estimate. Runs 2–4 also had a higher filter cut-off $\omega_f = 3.54$ rad/s (vs. 1.8 rad/s) which did not provide as much attenuation of the dither oscillation in ξ. A lower filter cut-off like what was used in Run 1 would have led to a smoother approach to the optimum also with less steady-state oscillations.

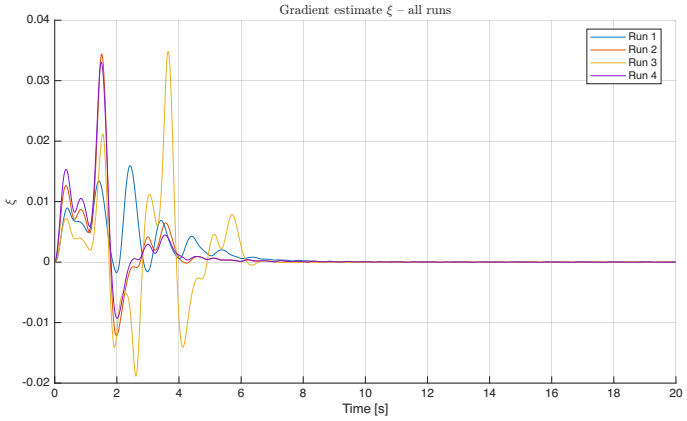
Figures 1a, 1b, and 1c show the convergence of $\hat{r}(t)$, the measured intensity $J(t)$, and the gradient estimate $\xi(t)$ for all runs. Figure 2 shows Run 1 in more detail.



(a) Convergence of $\hat{r}(t)$ for all runs.

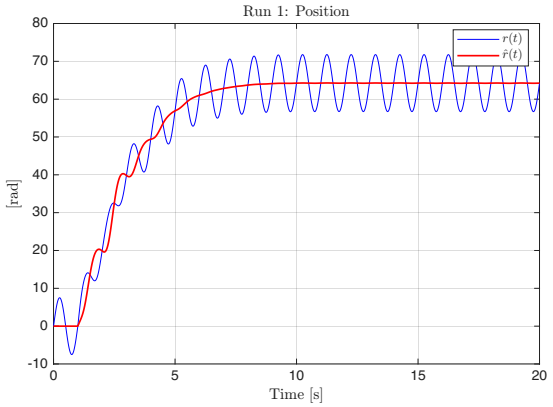


(b) Intensity $J(t)$ for all runs.

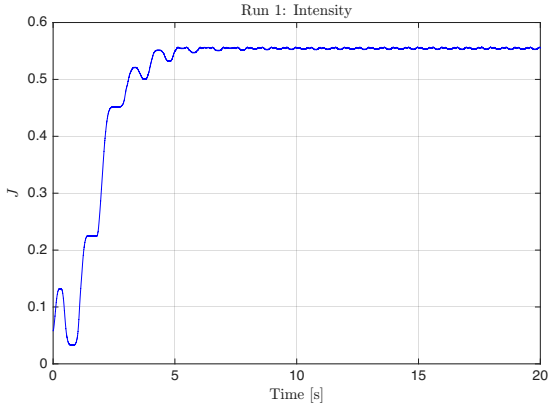


(c) Gradient estimate $\xi(t)$ for all runs.

Figure 1: Classical ESC experimental runs on the test setup.

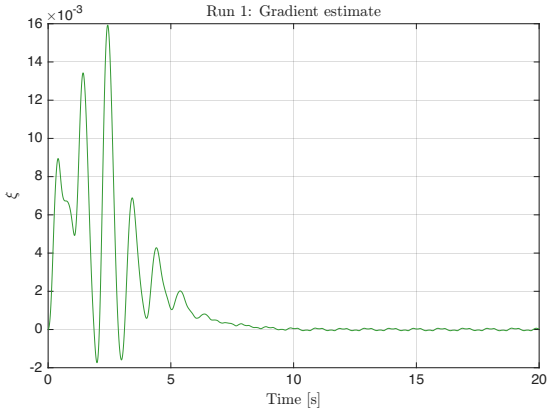


(a) Position $r(t)$ and $\hat{r}(t)$.

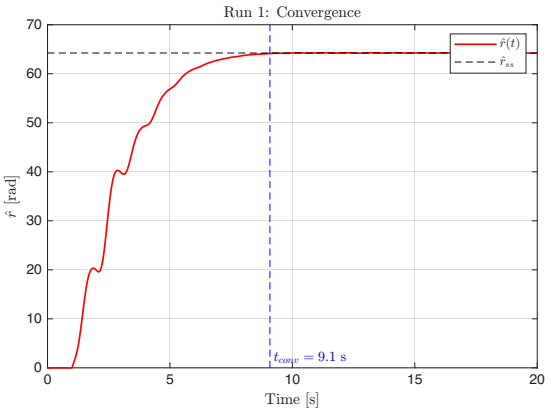


(b) Intensity $J(t)$.

Figure 2: Detailed view of Run 1 ($k = 3000$, $a = 7.5\,\text{rad}$).



(a) Gradient estimate $\xi(t)$.



(b) Convergence of $\hat{r}(t)$.

Figure 3: Detailed view of Run 1 (continued).

Encoder offset

On the physical setup, the system converges to $\hat{r}_{\text{ss}} \approx 60$–$67\,\text{rad}$, while on the analytical objective function, the optimum is at $x^* = 120\,\text{rad}$. Looking at the peak intensities, they are exactly the same (experiment ≈ 0.555 vs. model ≈ 0.554), which means that the issue is with the encoder (the motor encoder zero positionis not the same as the model's $x = 0\,\text{rad}$). This is fixed by aligning the experimental and model peaks which gives an offset of approximately 53–56 rad depending on the run, since each run settles differently due to steady-state error. Note that here the 53 rad offset is calculated from the position without the dither, thus the lower values of $\approx 45\,\text{rad}$ that can be seen in figure 4 are from the dither ($\hat{r} - a$).

Figure 4 shows the measured data shifted by this offset. The experimental data does seem to follow the model Gaussian near the peak, but the further away from the optimum, the steeper the function. This is most likely due to the lens being non-ideal.

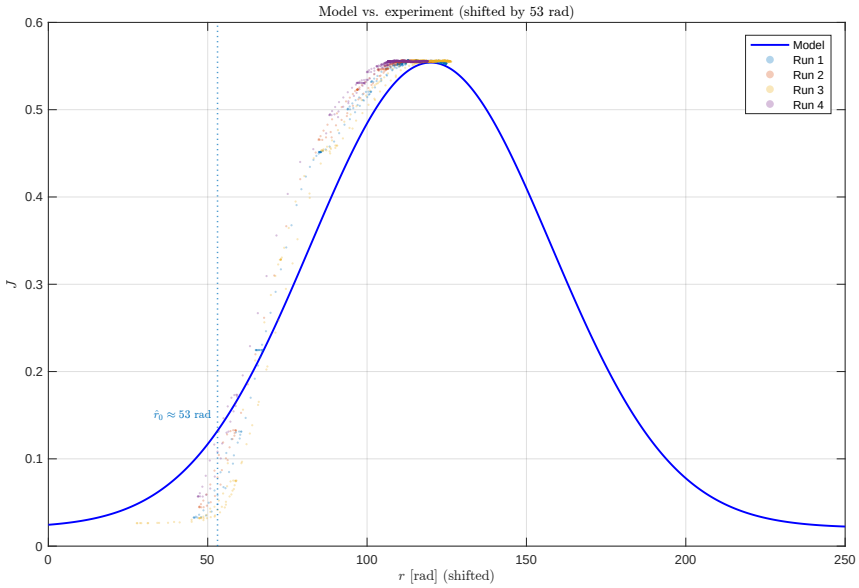


Figure 4: Model objective function $Q_J(x)$ vs. experimental measurements, with an offset of $\approx 53\,\text{rad}$.

Comparison with Assignment 1 simulation

The physical setup performance can be compared with the simulation model from Assignment 1 Q7, using the same ESC parameters. The initial condition was made the same so that the simulation starts at the same distance from the peak as Run 1. Run 1 started at 0 rad in terms of the encoder, which gave a peak at ≈ 64.2 rad, so the distance to the peak was 64.2 rad. The simulation is therefore started at $\hat{r}(0) = 120 - 64.2 = 55.8$ rad.

Figure 5 shows the ESC parameter comparison between the test setup and the simulation. There are some key take-aways from this figure.

- **Convergence speed:** The experiment converges just over 1 second faster (9.1 s) than the simulation (10.6 s), using the same ESC parameters. The fact that the convergence times are close together validates the simulation model. The small difference in convergence time is most likely due to the fact that the real objective function has a steeper gradient compared to the analytical model as seen in Figure 4, which should drive the integrator faster toward the optimum.
- **Steady-state oscillations:** The simulation has larger steady-state oscillations (± 0.09 rad) compared to the experiment (± 0.03 rad). Since these were both run with the same filter settings, this could mean that the real plant dynamics have some more high-frequency attenuation of the dither oscillation that the simulation doesn't model.
- **Gradient estimate $\xi(t)$:** The gradient estimation of the simulation is less volatile than the test-setup, which also hints at there being more plant dynamics that are not modelled in the simulation. The ESC controller when run on the test setup still manages to filter this and still converges to the optimum.

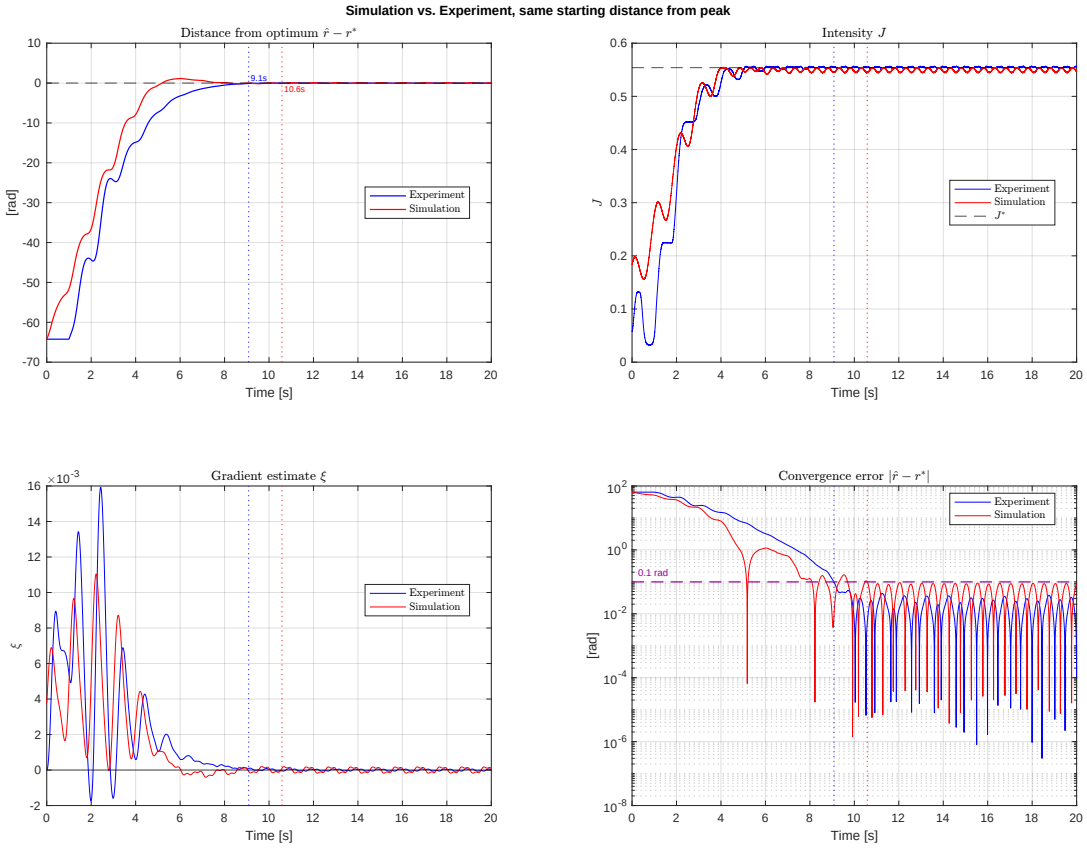


Figure 5: Simulation (A1 Q7 parameters) vs. experiment (Run 1), starting from the same distance to the optimum.

An additional comparison was also made that is not as relevant to the question but is still worthy of being mentioned. The simulation was also run from the analytical initial condition $\hat{r}(0) = 10\,\text{rad}$ which was the default in Assignment 1. Since this simulation starts further away, the ESC has to travel further, which resulted in the longer convergence time of 20.8 s. This is exactly the result of Assignment 1 Q7 but it is also valuable to see the comparison. This can be seen in Figures 19 and 20 in Appendix A.

Question 2

The accuracy of the position in extremum seeking control is found by the nominal position $\hat{r}$ plus the dither signal $a\cos(\omega t)$. This means that the actual position oscillates $r(t) = \hat{r}(t) + a\cos(\omega t)$, so when it has converged to x^*, the position still oscillates by $\pm a$ from the optimum. Therefore, with regards to the tuning in Question 1, if $a > 0.1\,\text{rad}$, then clearly the 0.1 rad requirement isn't met when the dither is also incorporated. The positioning accuracy can be improved by:

- **Reduce the dither amplitude** a: Smaller a directly reduces the oscillation around $\hat{r}$. However, smaller a weakens the gradient signal, making the estimate ξ noisier and convergence slower.
- **Reduce the integrator gain** k: A smaller k reduces the overshoot but slows convergence.
- **Reduce the filter cut-off** ω_f: Provides better noise rejection at the cost of slower gradient extraction.

The fundamental trade-off is between convergence speed and steady-state accuracy. For the dither-based accuracy requirement $|\hat{r} - x^*| + a < 0.1\,\text{rad}$, we need both $a < 0.1$ and $\hat{r}$ to converge close to x^*. A smaller a means weaker gradient information, which requires either longer convergence times or a more conservative (slower) feedback gain.

Question 3

Benefits and drawbacks of the moving average implementation

The moving average filter replaces the high-pass then demodulation then low-pass filter with just the demodulation then a moving average with $T = 2\pi/\omega$. One of the benefits of the moving average filter is that you get an exact gradient estimation since the high-pass and low-pass filters only approximate when trying to separate the frequency components. The moving average over one period exactly cancels out the DC component and all harmonics.

From Lecture 3, the demodulation product is:

$$d(t) = J(t) \cdot \frac{2}{a}\cos(\omega t) \tag{1}$$

$$J \approx Q_J(\hat{r}) + Q'_J(\hat{r})\, a\cos(\omega t) + \dots \tag{2}$$

Expanding J:

$$d(t) \approx \frac{2}{a} Q_J(\hat{r})\cos(\omega t) + Q'_J(\hat{r})\big[1 + \cos(2\omega t)\big] + \dots \tag{3}$$

Taking the moving average over one full period $T = 2\pi/\omega$ means taking the integral of cosine which is simply 0. This means the moving average calculates the gradient exactly. Furthermore, the high-pass and low-pass filters cause phase lag meaning at the dither frequency, the phase lag is $-\arctan(\omega/\omega_f)$ and for timescale separation: $\omega_f \ll \omega$ thus, ω/ω_f is a large number. The phase lag approaches -90 and this makes the demodulated signal delayed relative to the actual gradient. On the other hand the moving average only has a fixed time delay of $T/2$, which is more predictable for the system and also it is frequency independent. Lastly, the moving average is simply easier to tune since only the window length needs to be tuned.

The disadvantages of the moving average filter is that if the dither frequency is not known or changes over time then the window(T) no longer lasts an exact period. Furthermore, the moving average cannot be faster than $T = 2\pi/\omega$ seconds since the gradient estimate needs at least one full

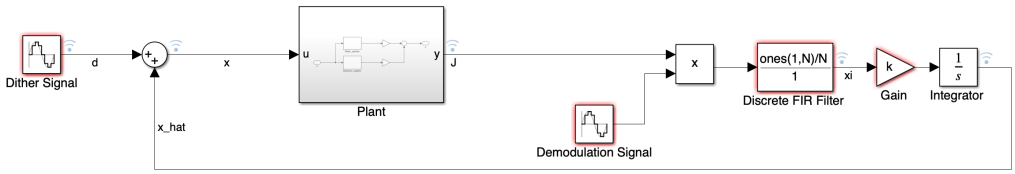


Figure 6: Moving average implementation in simulink

dither period of data to work. Here is where a low-pass filter can be faster but then more noise can be introduced due to worse filtering. Lastly, the moving average at the beginning (within the first T seconds) is not reliable at making a good gradient estimate since the moving average doesn't have a full period of valid data.

Implementation and tuning

The moving average ESC was implemented in Simulink as seen in Figure 6. Compared to the high-pass and low-pass implementation scheme, the high-pass filter was removed and the low-pass block was replaced by the discrete FIR filter and by making the coefficients all equal to $1/N$, a moving average filter over N samples is created. The simulation sampling rate is $F_s = 4000\,\text{Hz}$ and the FIR moving average window length is $N = F_s \cdot 2\pi/\omega$ samples so that the window spans exactly one dither period. This comes from the fact that in continuous time the period is $T = 2\pi/\omega$ seconds and if the simulation has a sampling rate F_s then there is $N = F_s \cdot T = F_s \cdot 2\pi/\omega$ discrete samples.

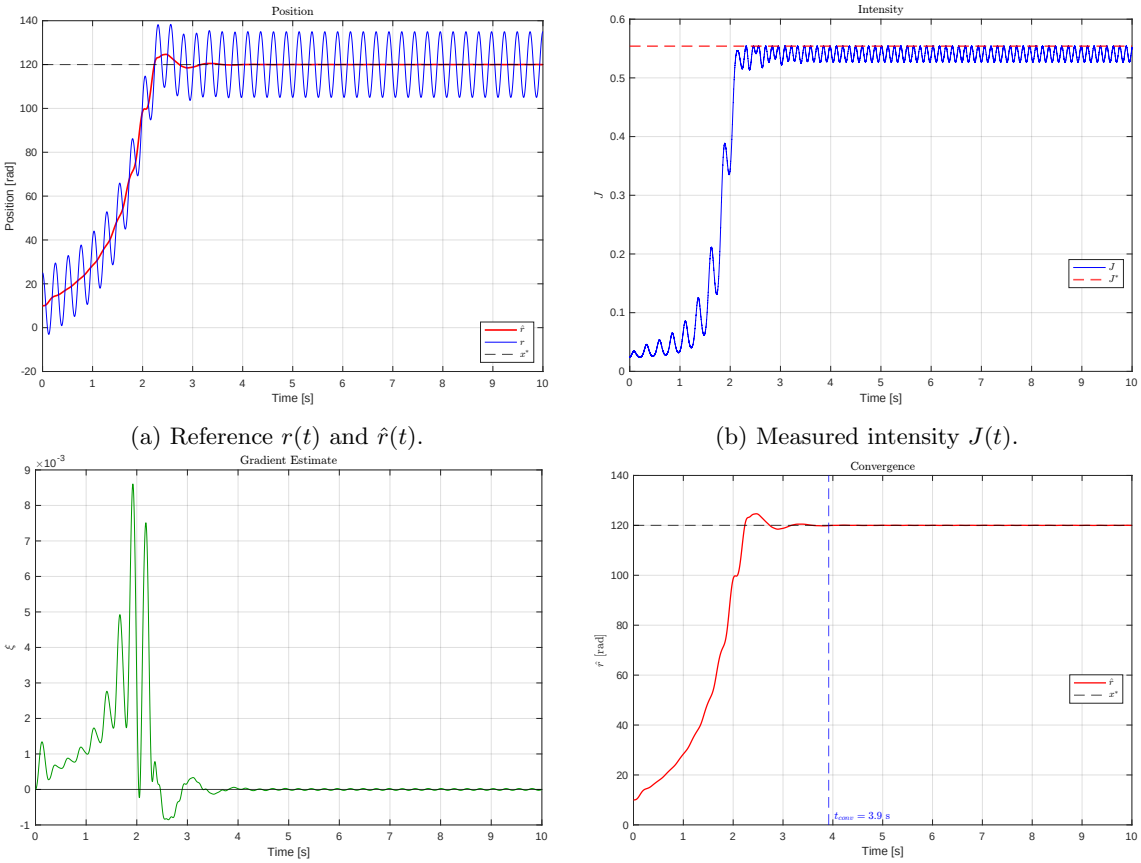
The parameters were tuned starting with the values found in Assignment 1 Q7 and then doing the following tuning steps:

1. Firstly, the dither frequency ω was tuned. Something that has to be taken into account is that the dither frequency has to be slow enough for the plant to react to it(static mapping), so that $|T(j\omega)| \approx 1$. Therefore, it is ideal to have $\omega \ll \omega_{\text{BW}} \approx 67\,\text{rad/s}$ but there is a trade-off. At the same time, a higher ω would minimize the moving average window T, since $T = 2\pi/\omega$. When we make the moving average window smaller the gradient estimate updates faster and the ESC can thus react faster. Tuning started with the Assignment 1 Q7 value of $\omega = 2\pi\,\text{rad/s}$ and increased to $3.9 \times 2\pi \approx 24.5\,\text{rad/s}$ which is where the convergence was still smooth and not full of noise. At $\omega = 24.5\,\text{rad/s}$ the dither frequency is still below $\omega_{\text{BW}} = 67\,\text{rad/s}$ so the static map assumption still holds. The reason a higher dither frequency is possible is, as mentioned earlier, there is no phase lag generated by the low-pass filter at higher frequencies.
2. Next, the dither amplitude was tuned to make the gradient signal stronger, but it's important to mention that the actual position oscillates with $r(t) = \hat{r}(t) + a\cos(\omega t)$, so the steady-state oscillation at the optimum is $\pm a$. However since the convergence criteria is based on the nominal position estimate, a was increased from $a = 7.5\,\text{rad}$ to $a = 15\,\text{rad}$. It can be argued that the larger a is still relatively small compared to the width of the objective function.
3. The integrator gain k was tuned taking into consideration not to be over-aggressive and overshoot since it is known that k decides how aggressively a step is made towards the optimum once the gradient is known. One thing to take into consideration is that an overly high k can also cause instability if $\hat{r}$ moves so fast that the gradient estimate can update since it is based on the previous dither period. Starting from $k = 3000$, it was increased to 24000. One of the reasons why k was able to be pushed to such higher values goes back to the dither frequency which was chosen to be increased so that the gradient is updated faster, keeping the gradient estimate more responsive even when $\hat{r}$ changes aggressively.

The final tuned parameters are:

$$k = 24000, \quad a = 15\,\text{rad}, \quad \omega = 3.9 \times 2\pi \approx 24.5\,\text{rad/s}, \quad N = 1026 \text{ samples}$$

The convergence time achieved was 3.9 s, starting from $\hat{r}_0 = 10$ rad compared to 20.8 seconds in Assignment 1 Q7. Looking more at what was done in Assignment 1 Q7, it's seen that the dither frequency was increased from 6.3 to 24.5 rad/s, the dither amplitude was increased to $a = 15$ rad and the integrator gain from 3000 to 24000. Since ω was increased it shortened the moving average window from $T = 1.0$ s to $T = 0.26$ s, meaning that the gradient estimate updates basically four times faster. The moving average achieves faster convergence since the shorter window gives more frequent gradient updates, which allows for an increase in k. At steady-state, $\hat{r}$ is only (± 0.03 rad) even though a large a was implemented, which also shows how the moving average filter cancels out the dither component from the gradient estimate.



(a) Reference $r(t)$ and $\hat{r}(t)$.

(b) Measured intensity $J(t)$.

(c) Gradient estimate $\xi(t)$.

(d) Convergence error on log scale.

Figure 7: ESC with moving average filter

Question 4

The moving average ESC scheme was implemented on the physical test setup. In total, seven experimental runs were performed with different parameters. As a starting point the parameters from Question 3 were used. Table 2 shows the parameters and results for each run on the test setup.

Table 2: Moving average ESC experimental runs on the test setup. MA1 and MA2 had an experiment runtime of 20 s compared to 10 s for the rest.

Run	k	a [rad]	ω [Hz]	T_{ma} [s]	$\hat{r}_{\text{ss}}$ [rad]	J_{ss}	t_{conv} [s]	Ripple [rad]
MA1	45 000	5.8	10	0.10	1.8	0.089	—	54.06
MA2	35 000	5.8	10	0.10	69.5	0.556	1.3	0.060
MA3	16 000	22.5	4	0.25	69.0	0.538	—	0.108
MA4	16 000	22.5	2	0.50	66.5	0.535	3.5	0.006
MA5	25 000	12.5	5	0.20	63.6	0.551	—	0.131
MA6	35 000	12.5	10	0.10	61.8	0.553	2.6	0.047
MA7	35 000	5.8	10	0.10	89.6	0.555	7.3	0.106

From table 2, in terms of convergence time MA2 was the fastest and MA3 and MA5 did not converge within the 0.1 rad bound because of residual oscillations at steady state, even though they reached high values of J. MA1 immediately went unstable, showing that the system has a maximum stable gain which is lower on the test setup compared to the simulation. MA7 used the same parameters as MA2, except with a shorter experiment time 10 s. It converged in 7.3 s but had large overshoot, thus there is some sensitivity to the initial conditions.

Figures 8 and 9 show the experimental results on the test setup for the five stable runs (MA2–MA6). All the runs performed on the test-setup including MA1 and MA7 are shown in Appendix B (Figures 21 and 22).

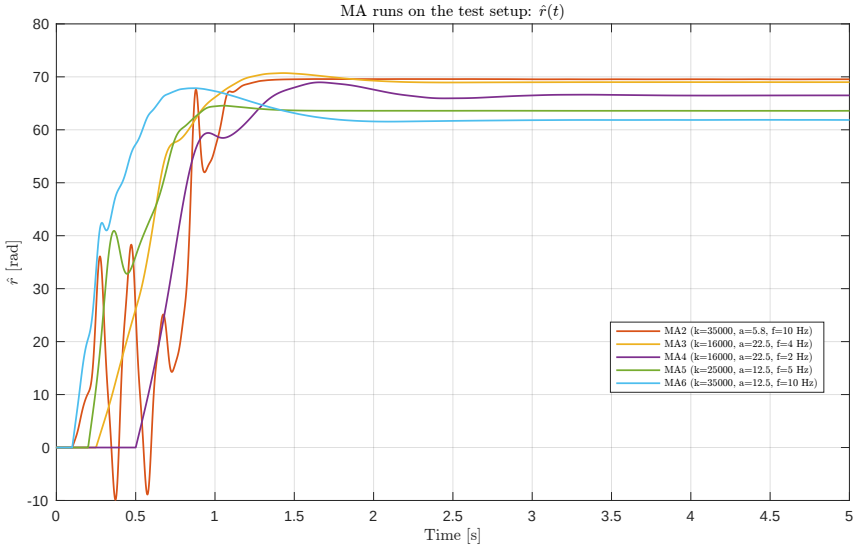


Figure 8: MA runs on the test setup: $\hat{r}(t)$ (MA2–MA6).

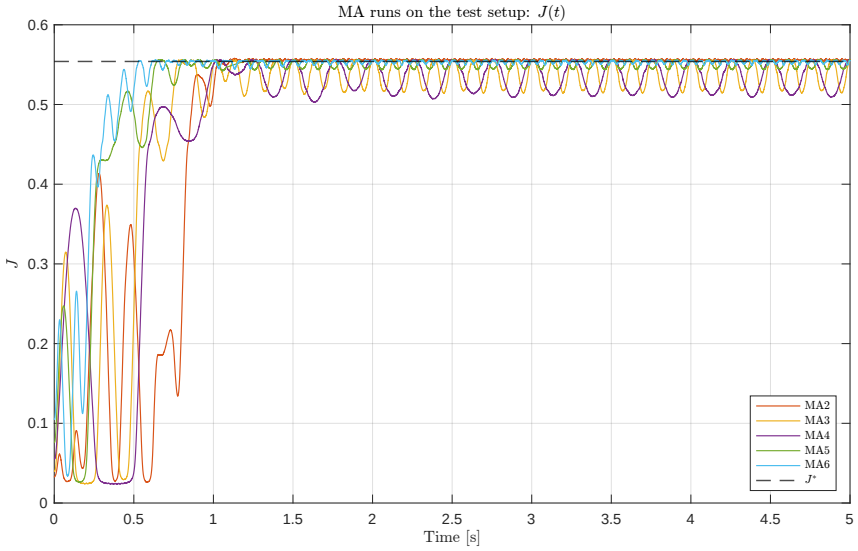


Figure 9: MA runs on the test setup: $J(t)$ (MA2–MA6).

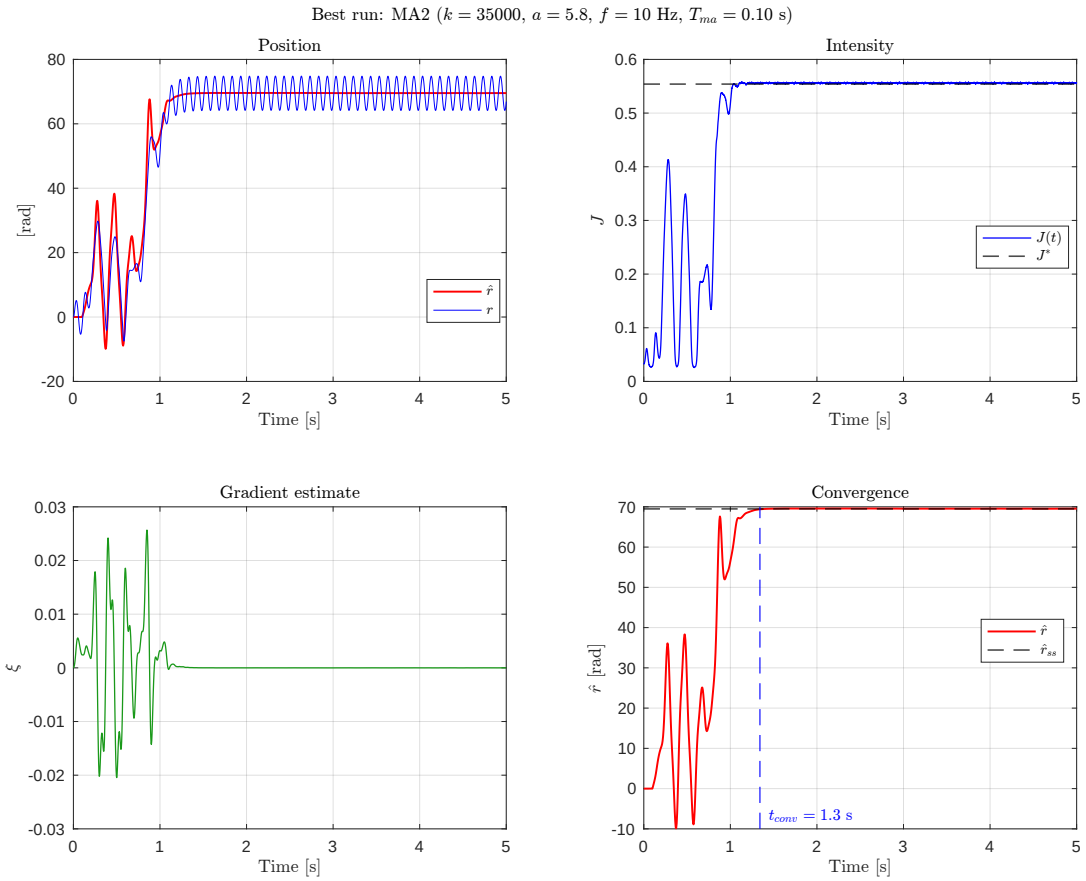


Figure 10: Detailed view of the best MA run (MA2).

Given the results of the moving average implementation, the advantages and disadvantages of the classical and MA approach can be discussed. Table 3 shows the classical approach vs. the MA approach in the simulation results (Assignment 1 Q7 vs. Q3) and experimental results (Q1 vs. Q4).

Table 3: Classical approach vs. moving average comparison.

	Simulation		**Experiment**	
Variable	Classical	MA	Classical	MA
t_{conv} [s]	10.6	3.9	9.1	1.3
J_{ss}	0.549	0.540	0.555	0.556
$\hat{r}$ oscillation [rad]	0.09	0.03	0.03	0.06

There are key differences between the classical and moving average approaches with regard to time-scale separation, gradient estimation errors, convergence rate, and the role of the window moving average filter. These are discussed below:

- Firstly, the time-scale separation: the classical approach strictly needs the following for the filters dither and plant $\omega_f \ll \omega \ll \omega_{\text{bw}}$. The moving average filter is less strict in a sense because it acts like a notch filter at the dither frequency and the higher order Taylor terms(harmonics); making sure to average over exactly one dither period($T = 1/f$) cancels out the dither component for any ω. This allows for higher dither frequencies which shortens the MA window and thus allows for faster gradient updates.
- Secondly, the gradient estimation errors: in the classic approach, the low-pass filter never completely gets rid of the dither sine wave but it can only attenuate it, leaving some residuals. In theory, the MA filter removes the dither completely through integration of the sine wave, thus having a more desired gradient estimate. Looking at table 3, the Q3 simulation has a smaller steady-state oscillation of ± 0.03 rad compared to the classical approach which had ± 0.09 rad. However, on the test setup the opposite was seen which could mean the tuning was not ideal or there was sensor noise.
- Regarding the convergence rate, in simulation, the MA manages to converge significantly faster(3.9 s vs. 10.6 s) because of the higher dither frequency and the higher gain. On the test setup, the moving average run MA2 converged in 1.3 s vs. the classic run of 9.1 s. However, this is a very fine line because when gains are too high like in MA1, the ESC became unstable. Furthermore, the lower frequency dither runs on the test setup like MA3 to MA5 converged slowly or had residual oscillations.
- The role of the MA window is that it has to exactly be one dither period $T = 1/f$ which is more trivial in the simulation. On the test-setup there are more factors that can affect things like delays or drift. It could be that the MA filter lets through some unwanted signals then this can affect the dither and be multiplied by high gains like 35000.

Taking all this into consideration the MA filter has more advantages compared to the classical approach. This can be seen even on the experiment runs where MA2 converged significantly faster and it still had the same steady-state accuracy. Like mentioned above the MA is still quite sensitive to the tuning when gain is too high and dither frequencies are too low, meaning that the classical approach can be more robust for a bigger range of operating conditions.

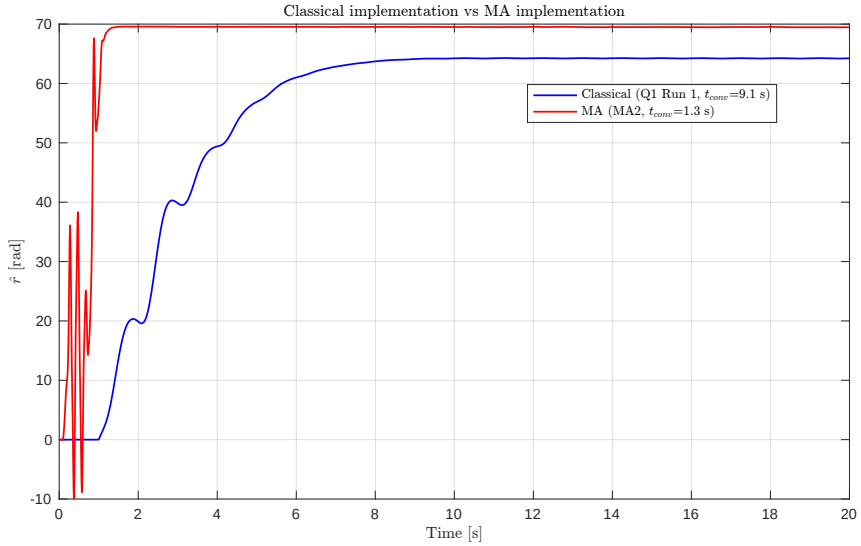


Figure 11: Classical implementation vs MA implementation on the physical setup.

Sampled-Data Extremum-Seeking Control Approach

Question 1

Instead of a dither in the form $a\cos(\omega t)$, a step-like dither signal c_v is implemented with the reference as $r = \hat{r} \pm c_v$ in alternating phases. In figure 12, the system responses for different c_v amplitudes are plotted where r is the reference position and x is the motor position which is the output of the plant. It is chosen to plot the different amplitudes near the optimum ($\hat{r} = 110\,\text{rad}$) because the gradient of the objective function is quite steep while also being close enough to the peak where the gradient is already getting weaker. Therefore this is a good location to test gradient estimation as the intensity changes ΔJ are smallest and the sensitivity to c_v is highest. Here the closed loop controller is used to find the performance tracking of the plant to the reference signal. Looking at the graphs there are some key points to be made:

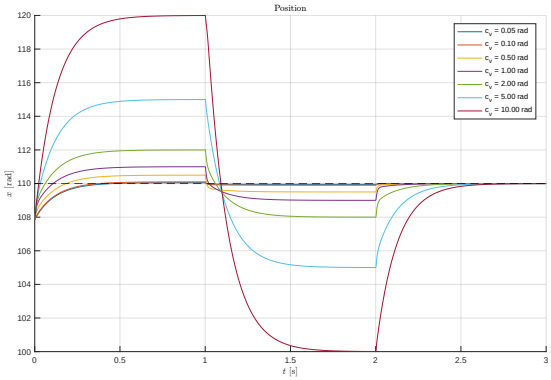
- When c_v is small (0.05 to 0.5 rad), the system has minimal overshoot and settles rather fast to the new reference and the intensity change $\Delta J = J^+ - J^-$ is thus also small but still measurable. Looking at ΔJ is useful when looking at the gradient estimate as it is in the numerator: $\xi = \Delta J/(2c_v)$ (which was discussed in Lecture 6, slide 14). If ΔJ is small, the signal to noise ratio must be kept in mind as sensor noise can become large relative to ΔJ, which means noisy gradient estimates and also more transient error. This is the case when $c_v = 0.05$ for example, as with noise, the gradient estimate is -0.001639 which is negative, while the analytical gradient is positive ($Q'_J(10) = 0.0036$).
- For middle values of c_v (1 to 2 rad), the system has good tracking and the gradient signal is still strong, and the finite-difference approximation is still approximately the same as the actual gradient. The only thing that has to be kept in mind is that the finite-difference approximation is accurate when c_v is small enough that higher-order terms in the Taylor expansion which is proportional to c_v^2 are negligible.
- For larger values of c_v (5 to 10 rad), the system clearly overshoots and the finite-difference gradient $\frac{J^+ - J^-}{2c_v}$ starts to deviate from the actual gradient because $Q_J(x)$ is nonlinear. This can be seen in the Taylor expansion of the central difference since there is $\frac{c_v^2}{6}Q'''_J(\hat{r})$ and higher order terms. Thus, for large c_v the higher-order terms become large and the estimate is not accurate anymore. Since Q_J is non-linear, the estimated gradient will be smaller in magnitude than the actual gradient. The objective function will then sort of flatten out over the interval at which we are measuring. Regarding the accuracy limit of 0.1 rad, if there are

oscillations of $\pm c_v$ around $\hat{r}$, the reference r will oscillate up to c_v from the nominal position. Thus for $c_v = 10\,\text{rad}$ and large values, this is clearly too high for the 0.1 rad accuracy bound.

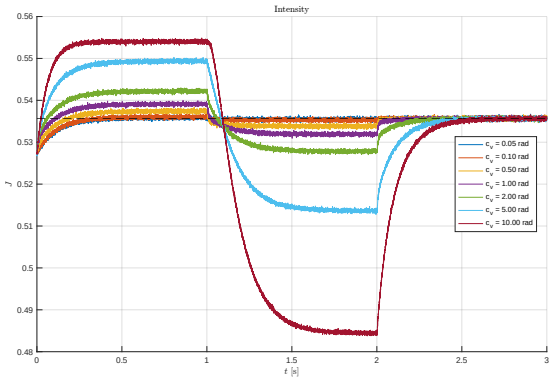
Therefore, an acceptable amplitude for c_v would be around 0.5–2 rad for when $\hat{r}$ is still far from the optimum as large position steps are needed to be made and a strong gradient signal is preferred over fine position movements. If the 0.1 rad bound needs to be met then it is obviously determined by $\hat{r} \pm c_v$. Thus c_v has to be chosen below 0.1 rad near the optimum, or simply $c_v < 0.1\,\text{rad}$ is made constant throughout the whole ESC movement.

In figure 12, the simulation started from a nominal position $\hat{r}_0 = 110\,\text{rad}$ and for each c_v value, firstly a step to $\hat{r}_0 + c_v$ for 1 s was made, then a step to $\hat{r}_0 - c_v$ for 1 s and then finally a step back to $\hat{r}_0$ for 1 s. The simulation was done without Simulink by modeling the full closed-loop dynamics making sure that the plant $P(s)$, PD+LP controller, transport delay, and actuator saturation were all modeled.

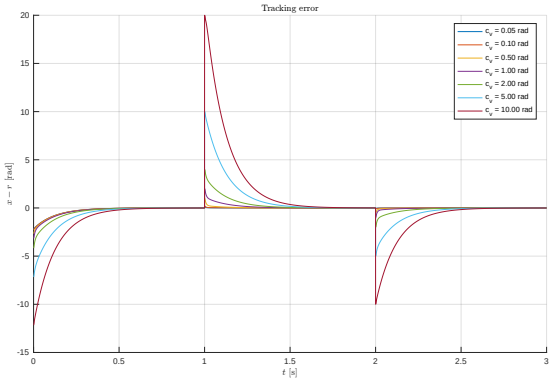
- Figure 12a shows the reference tracking of the closed-loop where small c_v tracks well but large c_v causes overshoot and longer settling time. The dashed horizontal line shows $\hat{r}_0 = 110\,\text{rad}$.
- Figure 12b shows $J(t)$ for each c_v, here ΔJ can be seen which relates to the numerator of the gradient estimate as mentioned earlier. The smaller c_v values obviously have smaller ΔJ which makes it harder to tell apart from noise and the large c_v gives large ΔJ but the finite-difference approximation becomes less accurate.
- Figure 12c shows the tracking error $x(t) - r(t)$ which shows the transient error which occurs especially when c_v is big. The larger c_v means larger steps and larger transient spikes that have to decay. This is an issue since the plant has to settle before J can be measured which is why the waiting time T is needed.
- Figure 12d shows the gradient estimate $\xi = (J^+ - J^-)/(2c_v)$ as a function of c_v on a log scale. Here $\hat{r} = 10\,\text{rad}$ is chosen which is different from the other figures which start at $\hat{r} = 110\,\text{rad}$. Thus in this figure, the reference is started far away from the optimum where the gradient is large. Note that the red dashed line is the analytical gradient $Q'_J(10)$. For small c_v the estimate is very close to the analytical gradient meaning that the finite-difference is accurate, however, as c_v increases, there is some divergence between the estimate and the analytical which shows the higher-order Taylor terms that are proportional to c_v^2 take over.



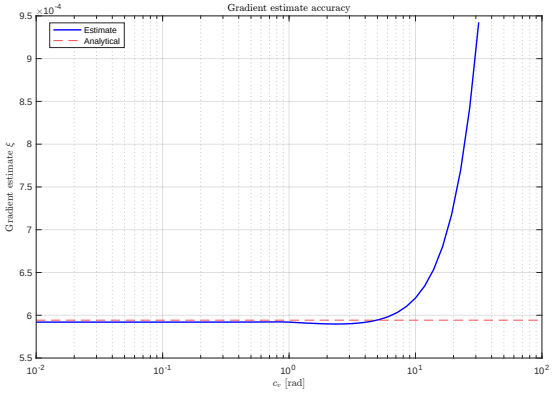
(a) Position $x(t)$ for different c_v.



(b) Intensity $J(t)$ for different c_v.



(c) Tracking error $x(t) - r(t)$.



(d) Gradient estimate quality vs c_v.

Figure 12: System responses for different c_v.

Question 2

The role that the waiting time T has is that after each step perturbation $\hat{r} \pm c_v$, the closed-loop system needs time to settle before you can have a reliable intensity measurement J. If T is made too short, then the position x has not reached steady state, and the measured J still captures transient errors that make the gradient estimate biased. Therefore, in order to find the minimum T, the settling time of the closed-loop transfer function has to be taken into consideration.

For 0.1 rad accuracy, c_v cannot be higher than 0.1 rad, but when finding the best value for T, the settling error has to stay small relative to the smallest c_v that are considered. Given the values chosen, the smallest c_v value is 0.05 meaning that 0.01 rad is a good option for the settling threshold. This way the measurement error in J is negligible relative to c_v perturbation, and the gradient estimate is not biased by left over transient error. With the threshold of 0.01 rad , the minimum waiting time is $T_{\min} = 0.41$ s as seen in fig. 13.

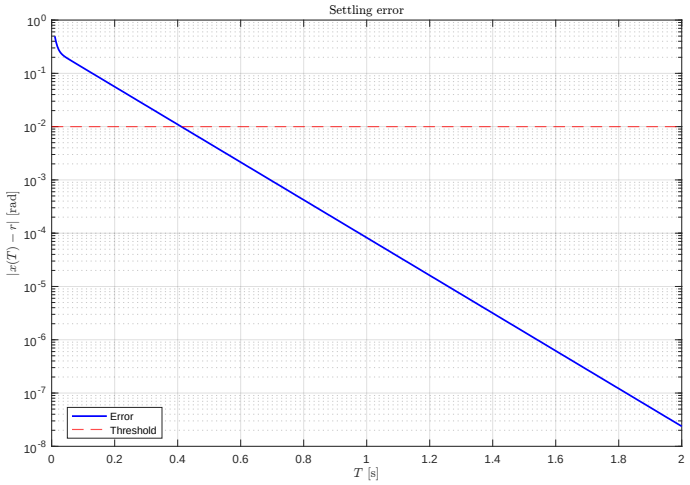


Figure 13: Settling error $|x(T) - r|$ vs. waiting time T. The red dashed line shows the 0.01 rad threshold.

Table 4 shows the acceptable waiting times for given c_v values. Based on the table, $c_v = 0.05$ rad and $c_v = 0.08$ rad are good options. If $c_v = 0.05$ rad with $T \geq 0.35$ s is chosen then there is more margin for steady-state error, while if $c_v = 0.08$ rad with $T \geq 0.31$ s is chosen then faster iterations with a stronger gradient signal can be achieved. However, this is at the cost of more strict positioning requirements ($|\hat{r} - x^*| < 0.02$ rad). Both are acceptable choices and a final choice can be made based on whether robustness or speed is more important. Obviously values of $c_v \geq 0.1$ rad are too high for the 0.1 rad accuracy bound however, as the c_v values get larger the gradient estimate does improve since ΔJ is larger relative to noise.

Table 4: Acceptable combinations of T and c_v for 0.1 rad accuracy. T_{min} is the minimum waiting time for the settling error to be less than 0.01 rad.

c_v [rad]	T_{min} [s]
0.01	0.52
0.02	0.44
0.05	0.35
0.08	0.31
0.10	0.29
0.50	0.21

Thus there is a clear trade-off as each ESC iteration takes $2T$ seconds, one for $+c_v$ and one for $-c_v$. Larger T means slower convergence times but better gradient measurements and smaller c_v means better accuracy but a worse gradient signal and more sensitivity to noise.

Question 3

The sampled-data approach was implemented in Simulink as seen in fig. 14.

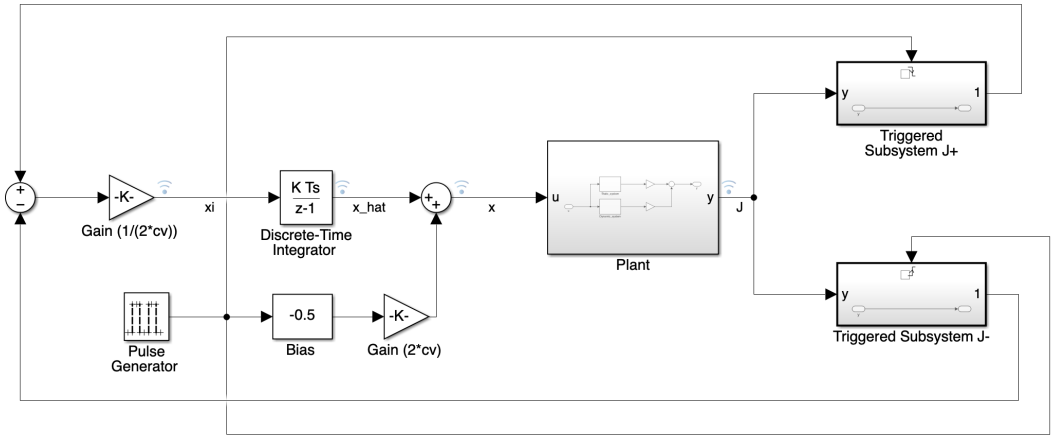


Figure 14: Sampled data implementation in Simulink

The following blocks were used to make the sampled data loop with a finite-difference gradient estimator:

- A Pulse Generator block was used set to sample-based mode with amplitude 1, period 2 samples, and pulse width 1 sample, with the sample time set as T. This creates a discrete clock with outputs 1 for T seconds then 0 for T seconds, which ends up being the total dither period $2T$. Sample-based mode was chosen to make sure that there is no continuous/discrete sample-time issues.
- Then the output of the pulse generator is put into a Bias block set to a value of -0.5 and then a Gain block of value $2c_v$. This changes the pulse from $[0, 1]$ to $[-0.5, 0.5]$ to $[\pm c_v]$ which is the dither signal.
- The two Triggered Subsystems are there to sample the intensity J at the correct moments. The J^+ subsystem triggers on the falling edge which is when the pulse goes from 1 to 0, and this is what signals the end of the $+c_v$ move after the plant has had time T to settle. The J^- subsystem triggers on the rising edge which is the opposite (end of the $-c_v$ phase). Each subsystem holds its sampled value just like a ZOH.
- A Sum block to create $(J^+ - J^-)$ then a Gain block with value $1/(2c_v)$ finally makes the finite-difference gradient: $\xi_k = (J^+ - J^-)/(2c_v)$.
- A Discrete-Time Integrator with: forward Euler method, sample time $= 2T$ integrates the gradient to update $\hat{r}_{k+1} = \hat{r}_k + \lambda \cdot \xi_k$. The initial condition is set to $\hat{r}_0 = 10\,\text{rad}$. An important detail to mention is that the discrete-time integrator in simulink multiplies its input by its sample time so the block's gain needs to be $\lambda/(2T)$ not just λ. Essentially this cancels out the block's multiplication by $2T$ and thus we actually achieve $\lambda \cdot \xi_k$. Finally, the integrator output being $\hat{r}$ is simply summed with the dither to get the reference $r = \hat{r} + (\pm c_v)$.

Tuning of the sampled data ESC was done starting from $\hat{r}_0 = 10\,\text{rad}$ and the three tuning parameters (λ, c_v, T) were tuned manually with this approach:

1. First, the waiting time T is tuned to be long enough for the plant to settle after each step of $\pm c_v$. Based on what was found in Q2, the minimum settling time for 0.01 rad accuracy is 0.3–0.5 s, but a shorter waiting time can be used if c_v is large enough that the residual settling error is small relative to c_v. $T = 0.05\,\text{s}$ gave a strong gradient estimate with $c_v = 9.75\,\text{rad}$, since the settling error at $T = 0.05\,\text{s}$ is a small fraction of the dither amplitude.
2. Next, the c_v was tuned taking note that very small c_v gives a bad estimate of ΔJ due to signal-to-noise ratio and very large c_v degrades the gradient estimation (Taylor approximation breaks down due to the curvature of Q_J). A value of $c_v = 9.75\,\text{rad}$ was found to give a strong gradient signal since the larger dither makes a large ΔJ. This is important for the flat region

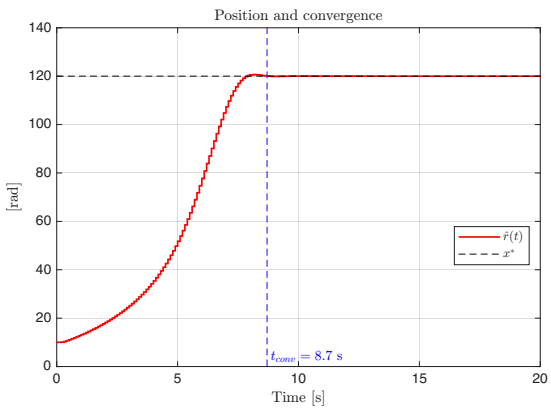
far from the optimum and since the objective function is generally quite wide, the higher-order Taylor error terms are still small, so the finite difference is still a good approximation of the tangent at this dither size.

3. Finally, λ was tuned by increasing it with T and c_v fixed. One important thing to take into consideration is that the objective function is flat far from the optimum, so λ has to be large enough that the step size can climb the flat slope in a good amount of time. At the same time, λ needs to be small enough so that there is no overshoot near the peak. The gradient does naturally help with this since it gets larger then shrinks near the optimum, so the step size adjusts itself. The best value found is $\lambda = 4\,600$ which gives convergence in 8.7 s with a clean gradient and no overshoot.

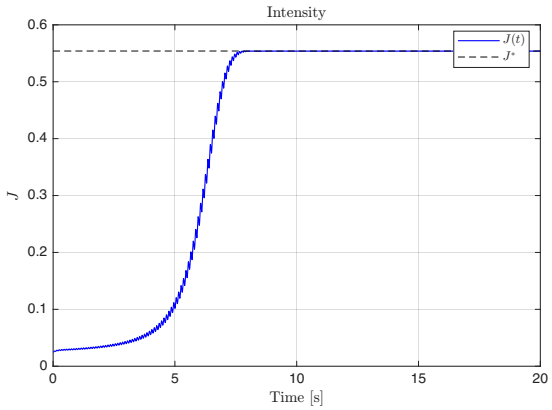
The final tuned parameters are:

$$\lambda = 4\,600, \quad c_v = 9.75\,\text{rad}, \quad T = 0.050\,\text{s}$$

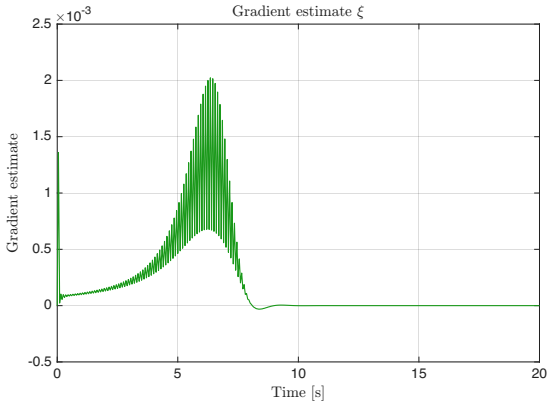
These results can be seen in figure 15 which shows the convergence time of 8.7 s, intensity and the gradient estimate.



(a) Position $r(t)$, nominal $\hat{r}(t)$, and convergence.



(b) Measured intensity $J(t)$.



(c) Gradient estimate ξ_k.

Figure 15: Sampled-data approach results ($\lambda = 4\,600$, $c_v = 9.75\,\text{rad}$, $T = 0.050\,\text{s}$): convergence in 8.7 s.

Question 4

During the testing session, a tuning for the sampled-data was done with starting from the following parameters: $\lambda = 30\,000$, $c_v = 5\,\text{rad}$, $T = 0.056\,\text{s}$. These parameters stem from an old parameter tuning in simulation from Q3, where it was found that the simulink integrator was implemented incorrectly. In the discrete-time integrator block, the gain was simply λ, however, it was supposed

to be $\lambda/(2 \cdot T_{wait})$. This blew up the required tuning for λ. As mentioned earlier, on the test setup, $\lambda = 30\,000$ caused the system to immediately go unstable in Run 11. After fixing the issue, new tuning, which is currently seen in Q3, is $\lambda = 4\,600, c_v = 9.75\,\text{rad}, T = 0.050\,\text{s}$. Sadly, the Simulink issue was only found after seeing the terrible results with $\lambda = 30\,000$, so the corrected parameters could not be validated experimentally. Nevertheless, twelve experiment runs were carried out, where one parameter was varied at a time. Table 5 shows the parameters and results. Runs 1 and 2 were run for 20 s; all others were run for 10 s.

Table 5: Sampled-data experimental runs on the test setup.

Run	λ	c_v [rad]	T [s]	t_{conv} [s]	$\hat{r}_{\text{ss}}$ [rad]	Steady-state oscillations [rad]
1	5 000	5	0.050	—	79.8	1.10
2	7 000	5	0.055	—	32.6	82.1
3	4 000	5	0.045	—	78.8	0.30
4	5 000	5	0.045	—	76.4	0.48
5	5 000	2	0.050	—	180.6	50.1
6	5 000	4	0.050	—	77.1	0.97
7	5 000	6	0.050	10.1	74.9	0.58
8	5 000	5	0.055	10.1	63.8	0.28
9	5 000	5	0.060	—	61.2	0.33
10	5 000	5	0.050	—	—	—
11	30 000	5	0.056	—	—	—
12	2 500	0.2	0.490	—	—	—

Looking at the results, none of the runs converged within the 0.1 rad bound. Nevertheless, Runs 7 and 8 were the most stable trajectories. Run 8 settled closest to the optimum as it was only 3.1 rad from $\hat{r}^* = 66.9\,\text{rad}$ and it also had the lowest steady-state oscillations of $\pm 0.28\,\text{rad}$. Run 7 which only changed c_v to 6 was also stable and showed a similar trajectory but it settled further from the optimum (7.9 rad away).

If further experiment sessions/runs could be made, then the corrected simulation parameters ($\lambda = 4\,600$, $c_v = 9.75\,\text{rad}$, $T = 0.050\,\text{s}$) would be the first parameters to test. Increasing c_v beyond 6 rad (to 9.75 rad) to match the simulation would also be worth testing, since Run 7 ($c_v = 6$) did show improvement over $c_v = 5$, however that was not clear during the testing session itself.

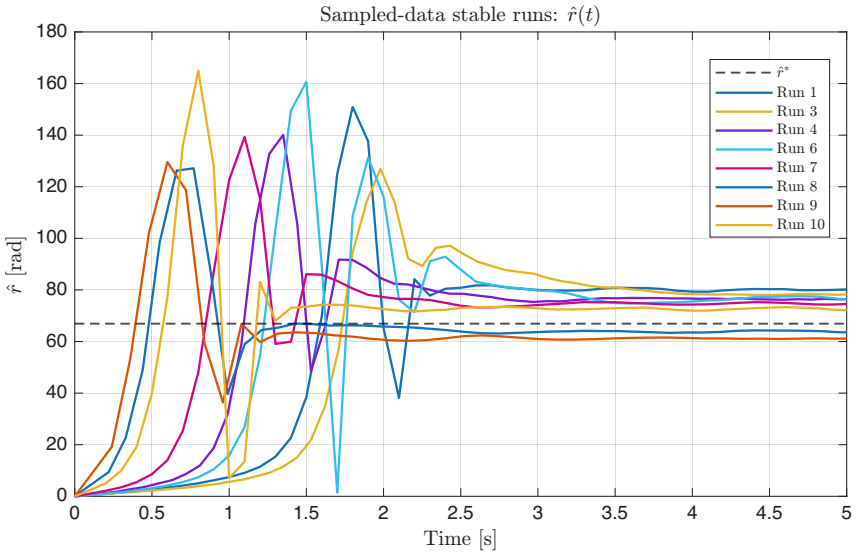


Figure 16: Sampled-data stable runs on the test setup: $\hat{r}(t)$.

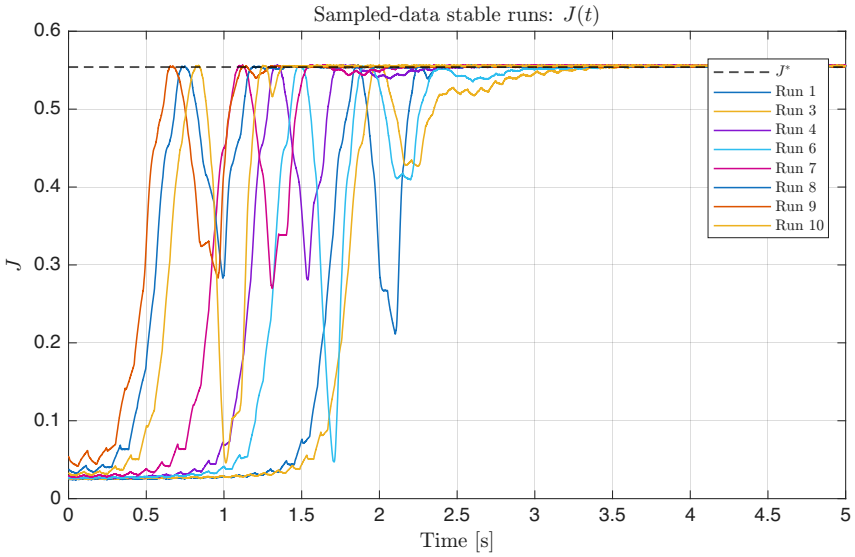


Figure 17: Sampled-data stable runs on the test setup: $J(t)$.

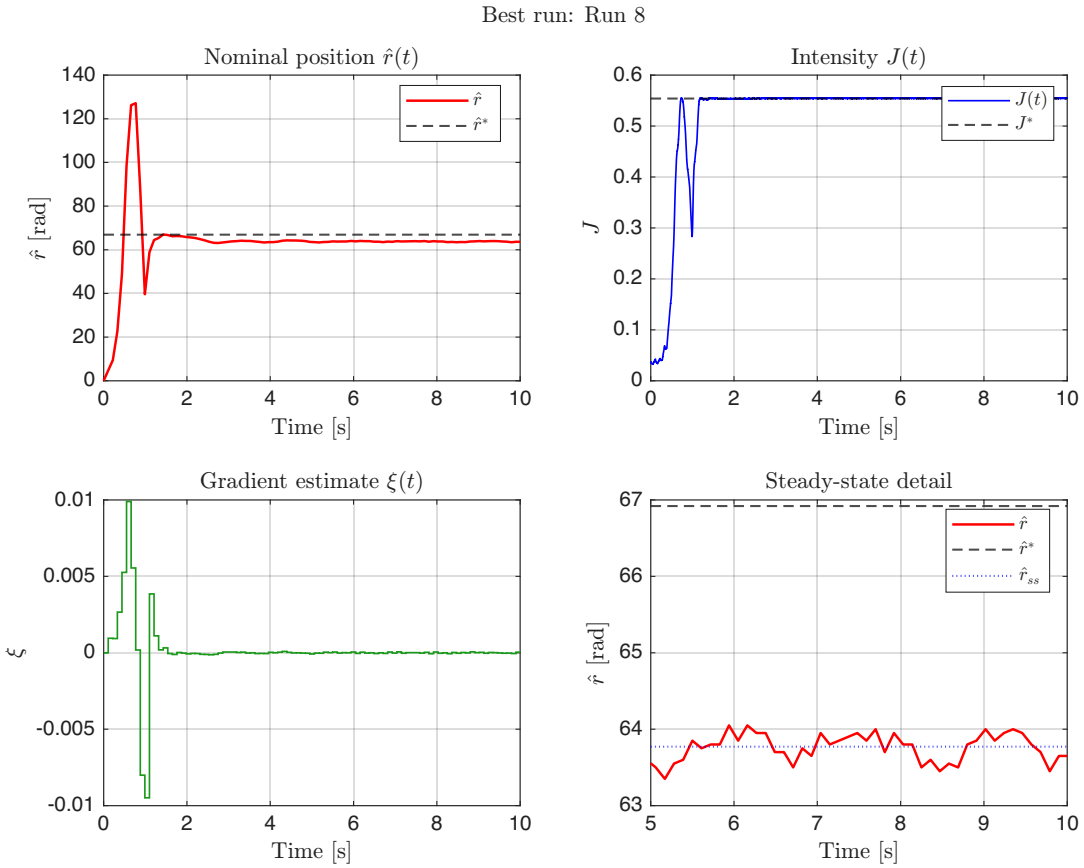


Figure 18: Detailed view of the best run: Run 8.

Figures of all twelve runs which include the unstable runs can be found in Appendix C (Figures 23 and 24).

Question 5

One of the most clear differences between the classical and sampled-data approach is the dither signal and how the gradient estimation framework is carried out. The classical approach uses a continuous sinusoidal dither $a\cos(\omega t)$ with demodulation and low-pass filtering to extract the gradient. The sinusoidal perturbation probes the objective function continuously and the demodulation extracts the gradient component from the measured output(mentioned in Lecture 3 regarding averaging). One problem is that even though the gradient is available continuously, the gradient estimate contains residual oscillations at harmonics of ω. The sampled-data approach on the other hand uses discrete step perturbations and the finite-difference formula to estimate the gradient. It is important to mention that each gradient estimate needs two measurements in the $\pm c_v$ direction with a waiting time of T for each measurement to settle. The gradient estimate is available only at discrete time steps, however, the estimate is exact/direct and doesn't have any residual oscillations like the classic implementation.

Regarding timescale separation, the classical approach needs $\omega_f \ll \omega \ll \omega_{\mathrm{BW}}$. This ultimately limits the convergence speed that one can get. The sampled-data approach has a discrete waiting time T instead of a continuous timescale and the only requirement is really that the waiting time is longer than the settling time of the plant. This is in a sense simpler and there is no need for continuous filtering.

The convergence speed of the classic approach is limited by the timescale separation requirements and k, ω_f, ω all have tradeoffs and have to be balanced. Thus the convergence speed can be slower but it is continuous and smooth. The sampled-data approach is limited by $2T$ since that is the time it takes to make an iteration but it is also limited by the step size λ. If there are large errors then since the discrete steps made are proportional to $\lambda \cdot \xi_k$ then large steps can be made, thus converging quite fast. However, each iteration takes $2T$ seconds no matter if the next update needs to be big or small making smaller steps take more time.

There are also some MIMO implications which were talked about in lecture 5 since the classical approach needs N dither signals at different frequencies for N optimisation variables. When using high and low-pass filters, these frequencies can't be normal multiples since they would otherwise create harmonic beats that the filters cannot handle and thus gradient estimates would be ruined. Also on top of that you need to make sure all these frequencies get put into a timescale window so that they are slow enough for the physical system to track them but also fast enough for the optimization loop to not take forever. The moving average filter helps with MIMO systems because the integral perfectly cancels out the ripples over exactly one window so the dither can be designed with integer multiples like $\omega_2 = 2\omega_1$ for example.

The sampled-data approach is even better in this case: firstly, it can use the central-difference which needs $2N$ function evaluations per iteration(measure up and then down). So this would be 2 measurements multiplied by 5 variables for example meaning 10 total function evaluations per update. The number of evaluations thus grows linearly with N meaning that if you double the number of variables then the controller has to do around twice as many calculations. Every single calculation needs the physical system to settle so if T_{wait} is 0.5 seconds and $N = 10$ variables then the controller can only update every 10 seconds. The benefit is that compared to the classical approach, if you had 10 variables, you need to find 10 unique frequencies that dont cause harmonic beating and that all fit into the tight bandwidth between the controller speed and the plants physical limits. Thus in the classical approach you quickly run out of frequency space when you scale up, but in the sampled data approach it just takes longer to run each iteration with no harmonic interference.

There are also some assumptions made on the objective function being in the classical approach, it assumes that it is a smooth/perfect parabola which is hardly the case in real life and that it can be approximated by a Taylor expansion around the current operating point. Since the system is continuously reading sensor data and passing it through high-pass and low-pass filters, if there is any sort of bump/step in the objective function then the continuous filters can read this as high-frequency noise and damage the Taylor approximation/gradient estimate. The objective function must be well-behaved, while the sampled-data approach doesn't need the function to be smooth

everywhere, but only for the finite-difference gradient to exist and be reasonably accurate. Looking at the central-difference gradient $\frac{J^+ - J^-}{2c_v}$, the ESC only needs those two specific data points to exist and it doesn't care about what happens between those points as long as the slope is still generally pointing towards the optimum. Therefore the sampled-data approach is more robust to irregular objective functions.

Furthermore, something important mentioned in lecture 6 is that the sampled-data approach is better at physical limitations/constraints. This is due to the fact that the position updates are discrete and predictable. Therefore a barrier function can be used. Instead of the system hitting a physical hard stop the objective function can be modified by adding a barrier term that goes to infinity as the system gets close to the constraint boundary. Comparing this to the classical ESC, enforcing constraints is more difficult because the position is continuously changing due to the sinusoidal dither and this risks the sine waves clipping against the physical constraints/boundaries.

Finally, regarding sensitivity to noise, the classical ESC is robust to high-frequency noise because the LP filter attenuates noise components above its cut-off frequency and the surrounding clean data absorbs any short spikes. The continuous averaging that the low-pass filter has over multiple dither periods is good at suppressing noise. The sampled-data approach is vulnerable because it relies on taking single discrete measurements as soon as the waiting time is over. Thus, if a random noise spike just so happens to occur right as the measurement is taken then the gradient estimate is completely off. This comes down to small values of c_v being dangerous for the signal to noise ratio. This can be seen in the following equations:

$$\xi_k = \frac{(J^+ + n^+) - (J^- + n^-)}{2c_v} = \text{Actual Gradient} + \frac{n^+ - n^-}{2c_v} \tag{4}$$

Some possible solutions to this is using averaging where the controller takes 100 samples and averages them for example. Otherwise a larger c_v value can be used which sacrifices the final positioning accuracy.

With all that being said, the classical approach should be used for continuous-time systems where the timescale separation can be kept and noise filtering is important. Otherwise, the sampled-data approach is better in general for discrete systems, MIMO problems, constrained problems, and when a longer waiting time is ok.

A Additional C_Q1 Comparisons

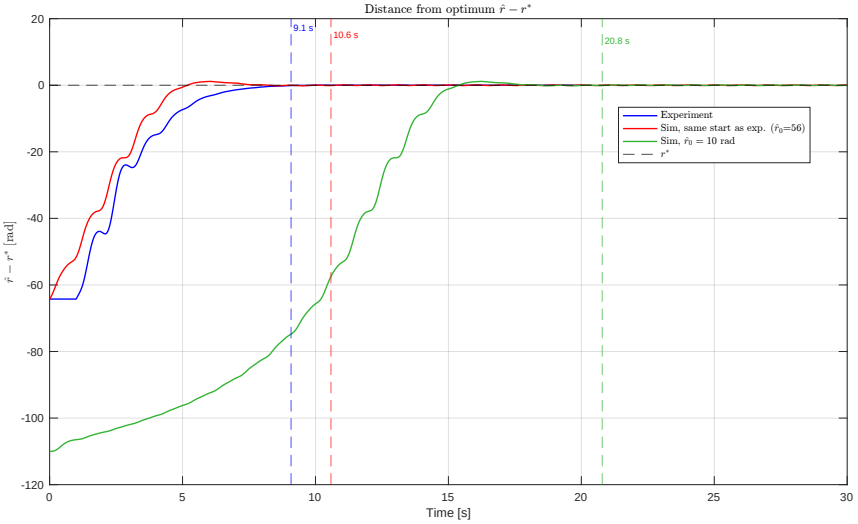


Figure 19: All three curves plotted showing the distance from the optimum over time. The three curves show the experiment on test setup, simulation starting at same position as experiment ($\hat{r}_0 \approx 56\,\text{rad}$), and simulation starting at $\hat{r}_0 = 10\,\text{rad}$.

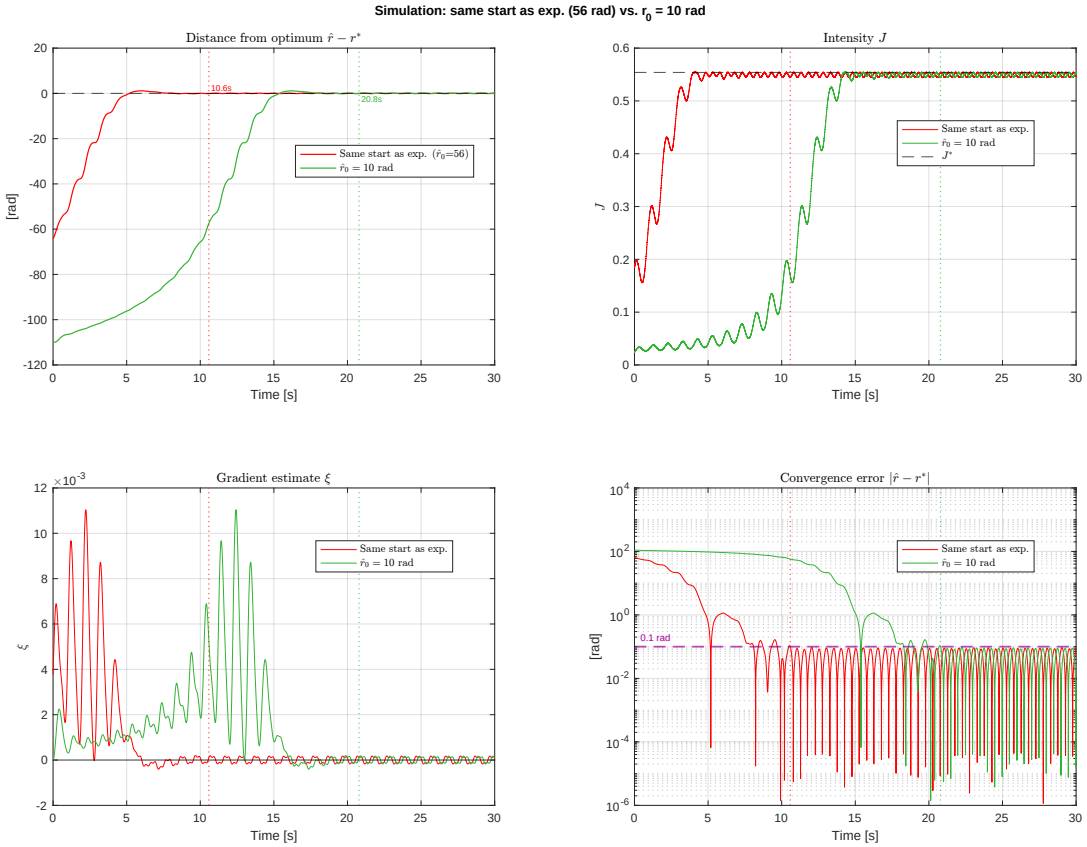


Figure 20: Simulink results applying the same tuning parameters for different starting positions: experiment starting point ($\hat{r}_0 = 56\,\text{rad}$) vs. starting at $\hat{r}_0 = 10\,\text{rad}$. Starting further away increases convergence time from 10.6 s to 20.8 s.

B Additional Classic Approach Q4 Figures

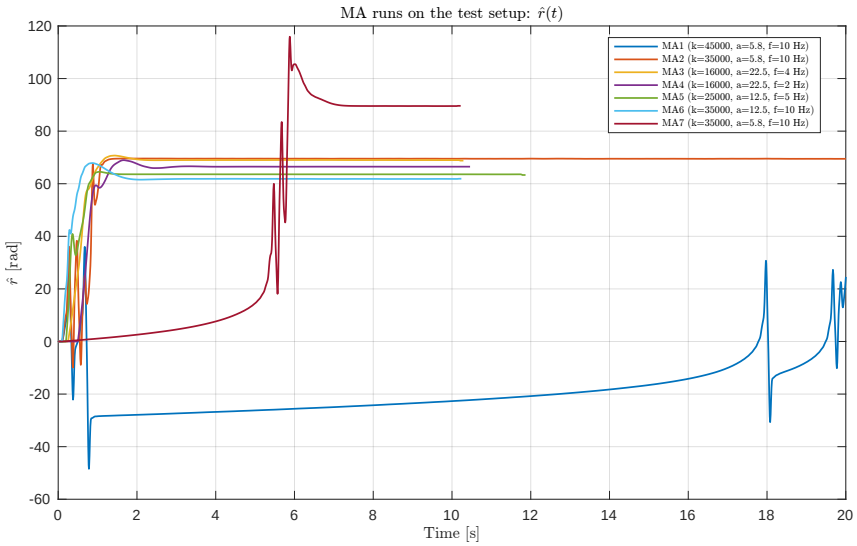


Figure 21: All MA runs on the test setup: $\hat{r}(t)$.

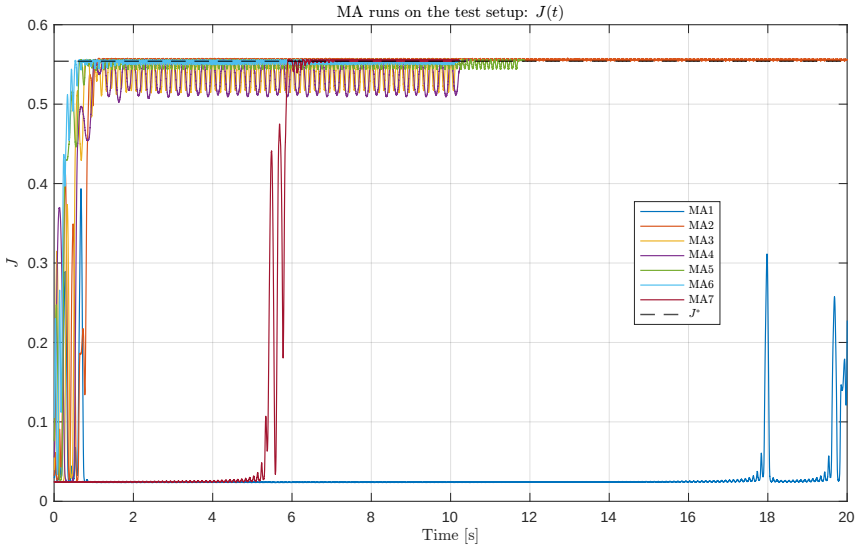


Figure 22: All MA runs on the test setup: $J(t)$.

C Additional Sampled-data Q4 Figures

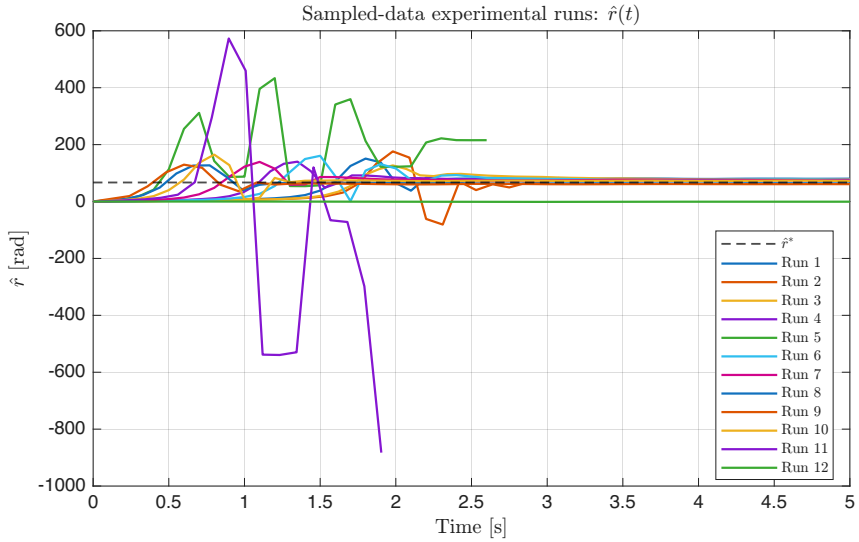


Figure 23: All sampled-data runs on the test setup: $\hat{r}(t)$.

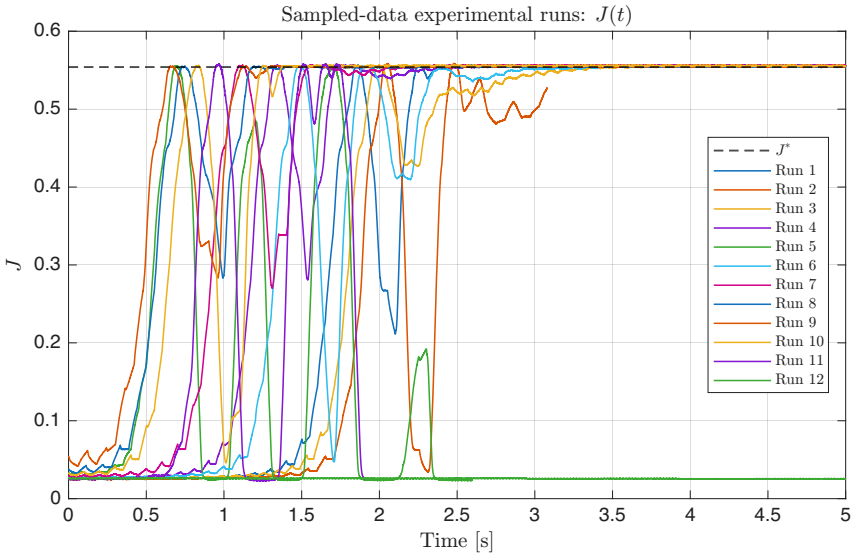


Figure 24: All sampled-data runs on the test setup: $J(t)$.