Introduction

Active Disturbance Rejection Control (ADRC) is a model-free control technique used for designing controllers in systems with unknown dynamics and external disturbances. it operates effectively even when the system’s behavior is not fully understood.

Disturbance Rejection

ADRC extends the system model by introducing an additional **fictitious state variable**. This virtual state represents everything that the user does not explicitly include in the mathematical description of the base system to be controlled.

The virtual state, denoted as the “total disturbance” accounts for unknown parts of the model dynamics and external disturbances. An extended-state observer estimates this **disturbance online** and uses it as the control signal.

ADRC scheme-control objective and adaptation mechanism

The control objective is to preserve the **stability** and **desired control performance** of the closed-loop system despite the presence of large model uncertainties (parametric and structural) and despite external unmeasurable disturbances acting on a plant.

The adaptation mechanism results from **on-line estimation** of the **total disturbance** (which includes all the model uncertainties and external disturbances) and from compensation of its influence on a plant by using an additional control signal in the inner compensation loop.

Description of the plant

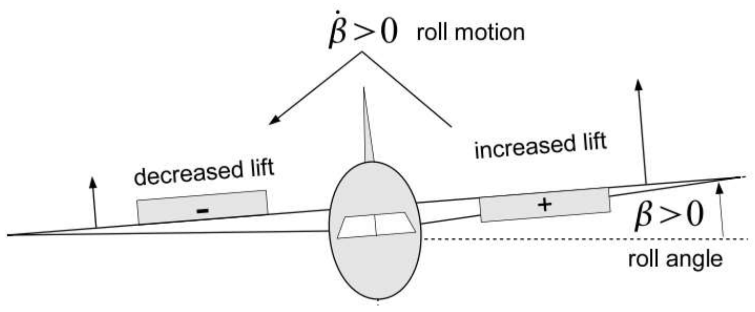


Fig1: A view of the delta-wing aircraft in the rolling motion with angular velocity

$$\ddot{\beta} = \theta_{10}\beta + \theta_{20}\dot{\beta} + \theta_{60}(\delta_a + d) + (\theta_{30}\,|\beta| + \theta_{40}\,|\dot{\beta}\,|)\dot{\beta} + \theta_{50}\beta^3$$

where θ10, θ20, . . ., θ60 are the true parameters of the plant, while the term d represents some external disturbance which affects the plant through the input channel.

Control performance requirements/objectives

- **R1**. signal yr(t) = βr(t) is a bounded time-varying reference trajectory for the aircraft roll angle such that dyr(t) and d2yr(t) exist and are bounded
- **R2**. tracking error e(t) ≜ yr(t) - y(t) converges with no overshoot to an arbitrary small vicinity of zero, that is |e(t)| ≤ε for t→∞, where ε ≥ 0 is a sufficiently small constant

- **R3**. The settling time Ts1% of the closed-loop system satisfies Ts1% ≈ α for α > 0, as expressed in sec.

Control System Design

Stages:

1. Determine a rough model of the plant:
2. Reformulate the rough model: refined to separate the nominal part and total disturbance f(·)
3. Design the Extended State Observer (ESO):
4. Design the inner compensation loop: this loop generates a new control input, **u,** which **cancels out** the effect of the disturbance on the system.
5. Design the outer-loop control input: designing the outer-loop control input, **u*,** which governs the **nominal part** of the plant model.

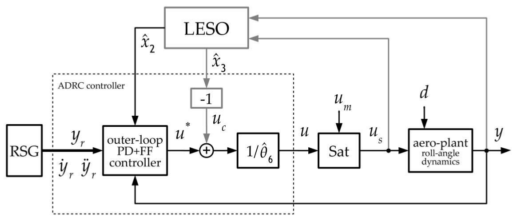
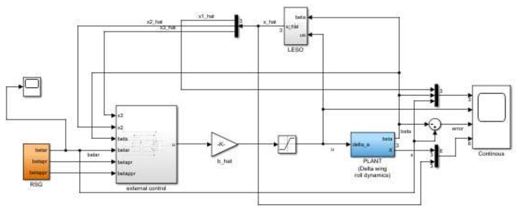


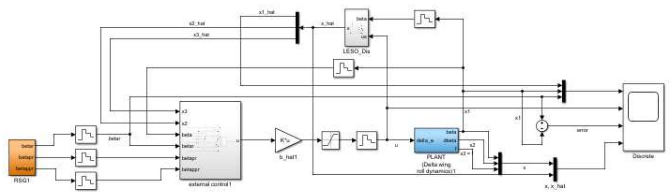
Fig2: Block scheme of the ADRC system for the aero plant

Source: lecture material

Simulation Result and Analysis



Continous Time Model



Discrete Time Model

Fig3: Complete Simulink model (for zoom in block, check the appendix)

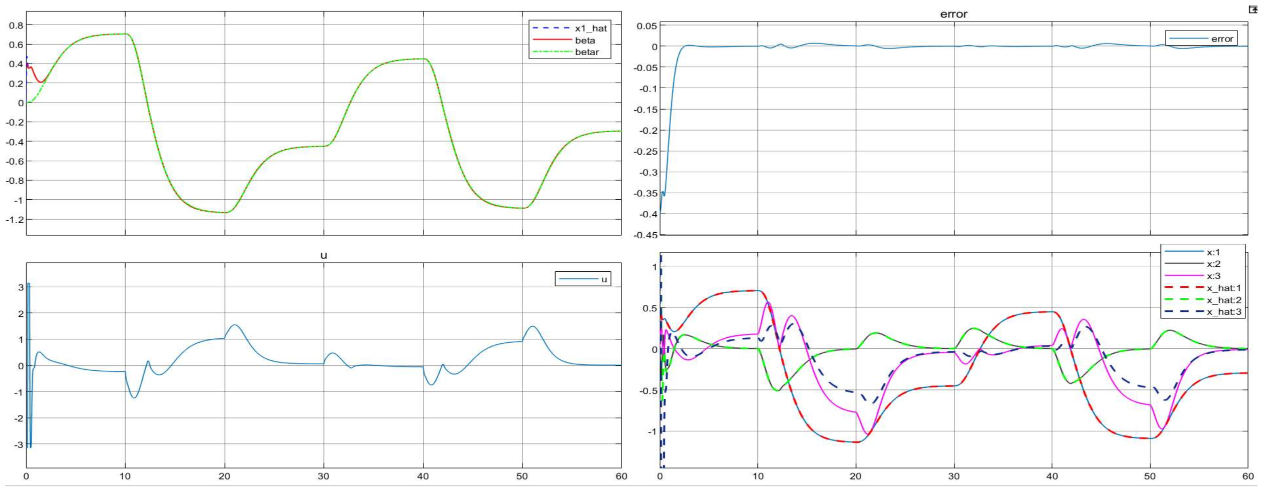


Fig4a: ADRC disturbance tracking performance at μ=0.2 with continuous random input

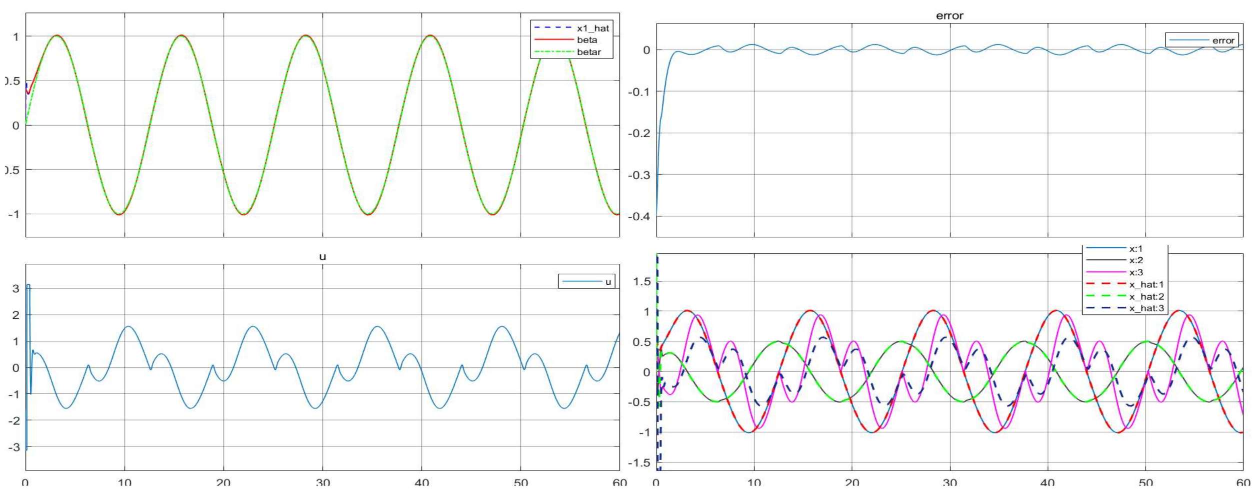


Fig4b: ADRC disturbance tracking performance at μ=0.2 with sinusoidal input

By comparing the above results (**Fig4a&b**), the plant used continuous ADRC and almost tracked the reference input and output from LESO also tracked the plant output as well for both type of reference signals.

when we consider the **total disturbance(x3 & x3_hat)**, LESO didn't perfectly track the output disturbance, however it is an acceptable error since the plant has many nonlinearities. The other estimates (**x1_hat & x2_hat**) of LESO were very good and matched to the plant output.

The other finding from the above result was, when sinusoidal reference signal was used, the **tracking efficiency was lower** compared to the uniform random signal which has a filter. The output showed some error compared to the reference input(**Fig4b: check the difference at pick values of beta & betar**).

In **both cases**, the error approached to zero as time goes infinity and the settling time is very short (around 2secs). So, the plant satisfied the control objectives **R2&R3** at μ=0.2.

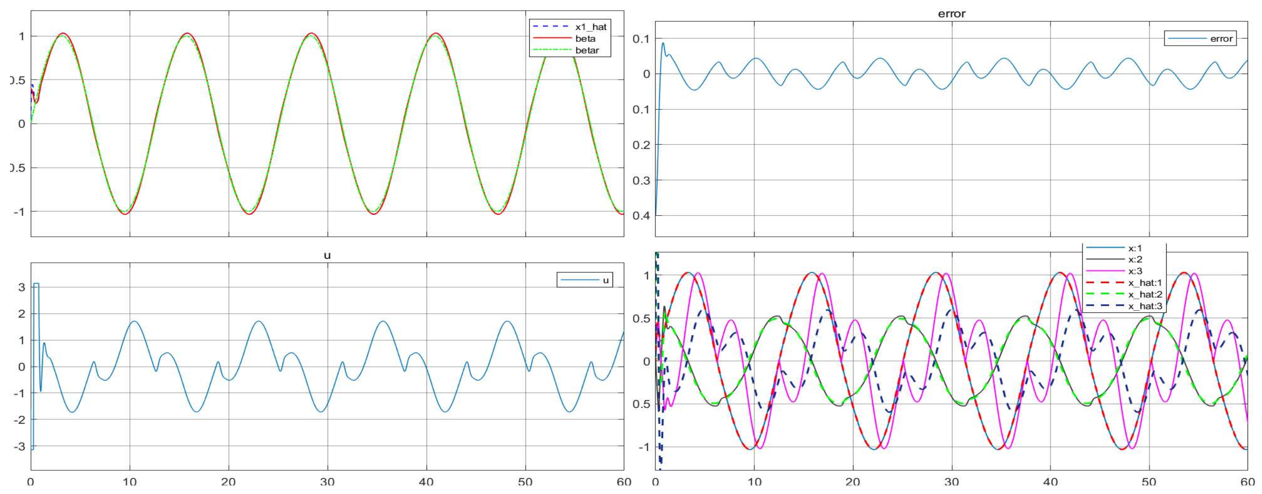


Fig5a: ADRC disturbance tracking performance at μ=0.5 with sinusoidal input

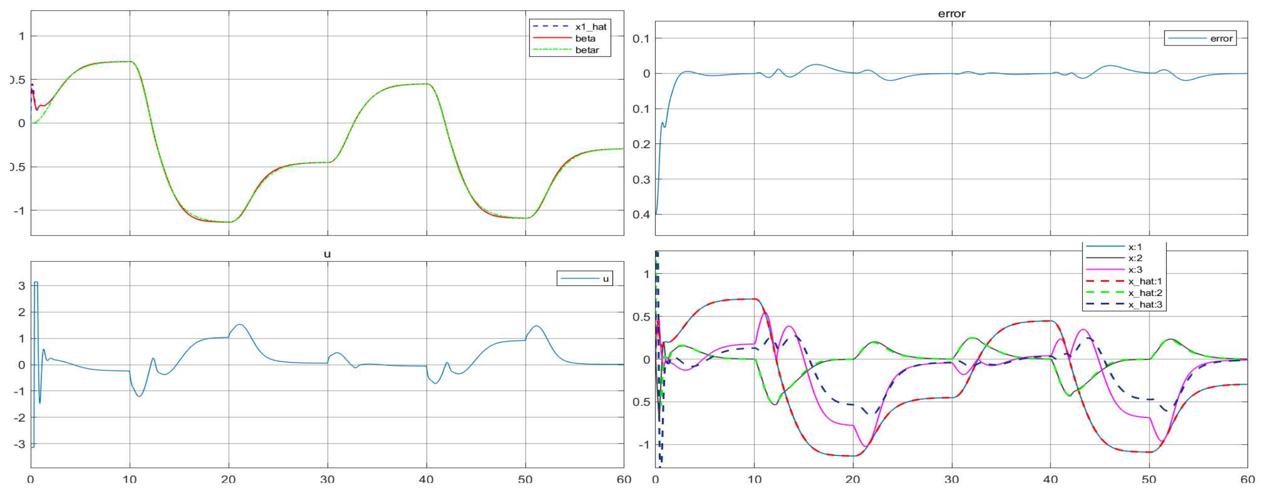


Fig5b: ADRC disturbance tracking performance at μ=0.5 with uniform random input

By comparing the above results (**Fig5a&b**), the controller didn't perfectly track the reference input, but output from LESO perfectly tracked the plant output for both type of reference signals.

when we consider the **total disturbance**, LESO didn't show much difference from the previous one on tracking the output disturbance.

The other finding from the above result was, the tracking efficiency was lower compared to the results in **Fig4a&b**. The output showed higher error, more value differences between the reference and the output signals.(**Fig5a&b: check the difference at pick values of beta & betar**).

In both cases, the error approached to zero as time goes infinity but not as perfect as the previous one at **μ=0.2** and the settling time was greater than 2secs. So, the plant satisfied the control objective **R2** only for both references.

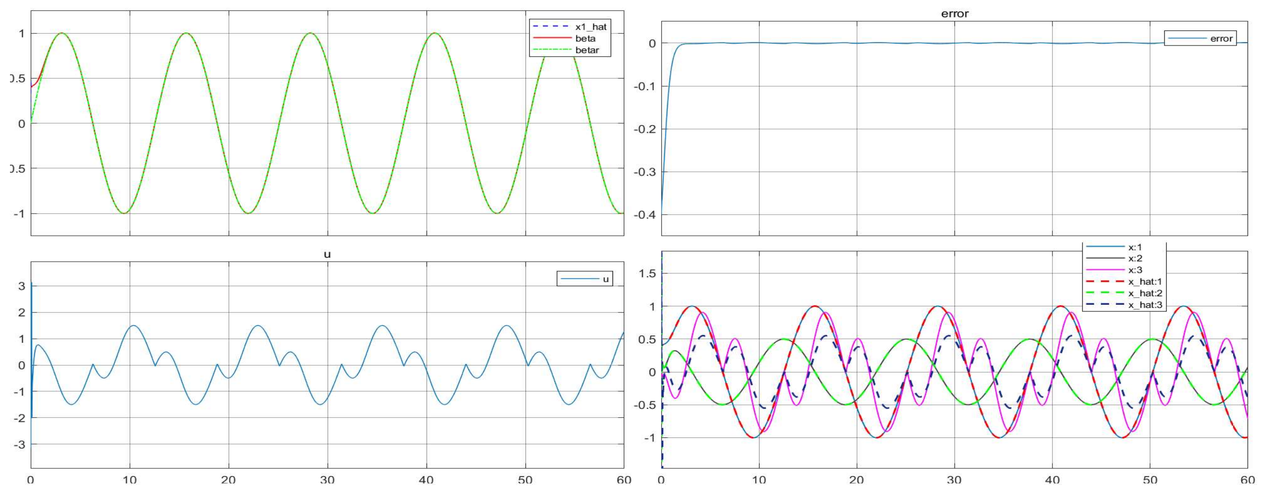


Fig6a: ADRC disturbance tracking performance at μ=0.03 with sinusoidal input

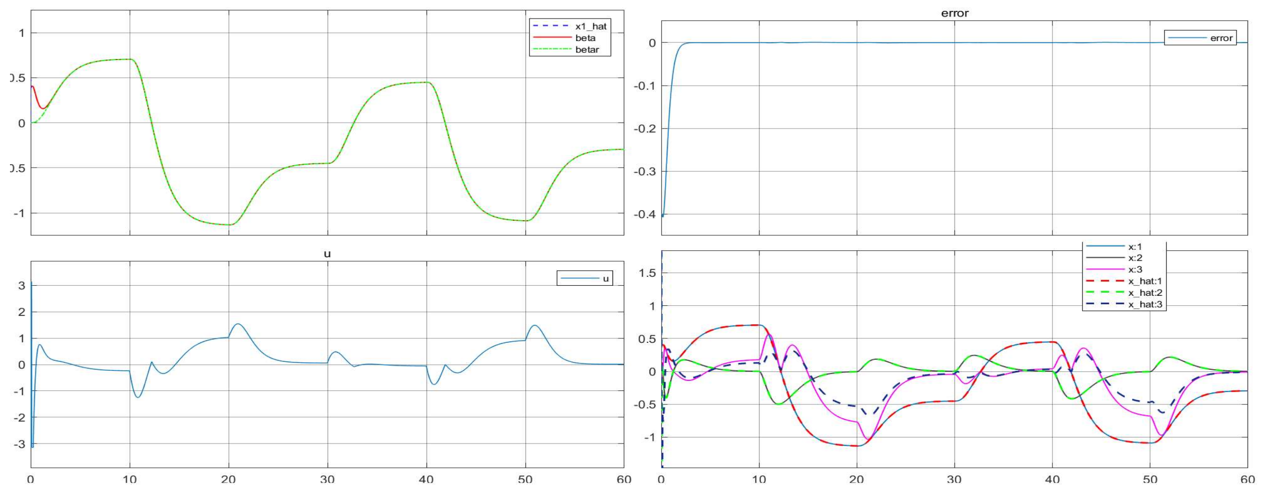


Fig6b: ADRC disturbance tracking performance at μ=0.03 with uniform random input

By comparing the above results (**Fig6a&b**), the controller perfectly tracked the reference input, and the output from LESO also perfectly tracked the plant output for both type of references.

when we consider the **total disturbance**, LESO still didn't show much difference from the previous two on tracking the output disturbance.

The other finding from the above result was, the tracking efficiency was very high compared to the results in **Fig6a&b**. The output showed no error after transient state.

In both cases, the error goes to zero as time passes its settling time and the settling time was very short(less than 2secs). So, the plant satisfied the control objectives **R2&R3** at μ=0.03 for both reference inputs.

Now by comparing all the above results(**Fig4a - 6b**), the bandwidth scaling factor μ has high influence on the controller efficiency. At lower μ value, the controller had higher bandwidth that improved the control performance and the reverse was also true.

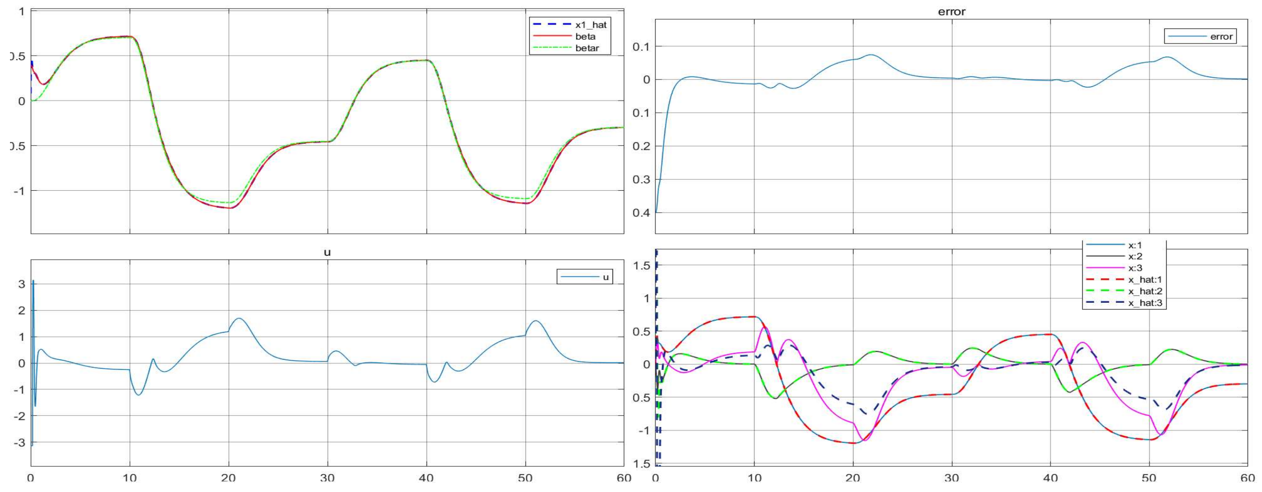


Fig7a. ADRC disturbance tracking performance at **μ=0.2** with square wave and without stochastic noise when **x3_hat = 0** (gain = 0)

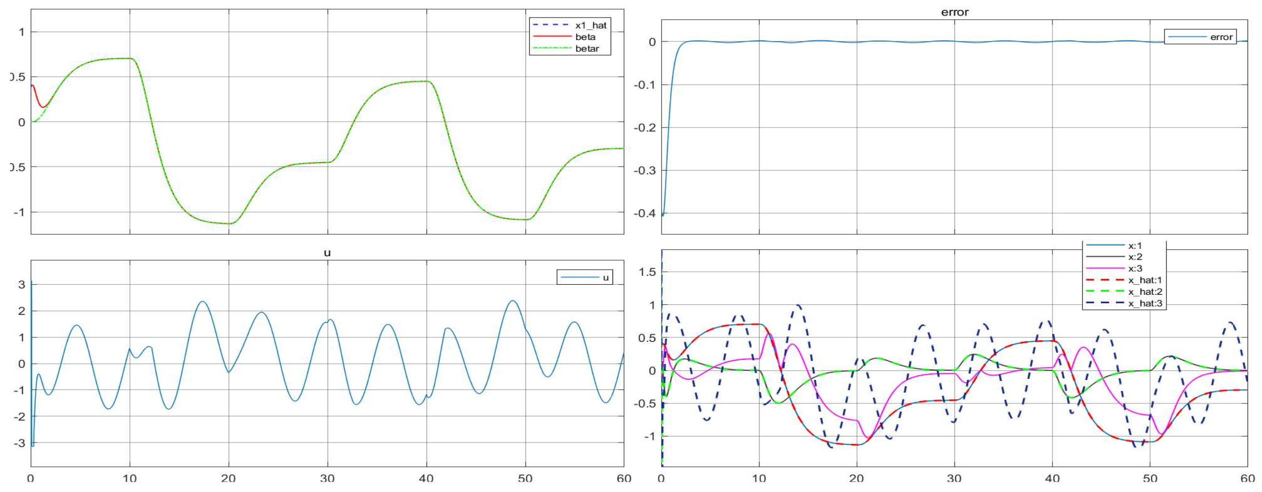


Fig7b: ADRC disturbance tracking performance at μ=0.2, d !=0 with square wave and without stochastic noise when **x3_hat != 0 (gain = 1)**

By comparing the above results (**Fig7a&b**), it was easy to say that the controller didn't satisfy any of the control objectives when disturbance rejection is turned off **(Fig7a)**, and the plant had high error, unable to tracked the reference. In addition, when external disturbance was added, the total disturbance wasn't tracked by LESO.

Whereas in the second case when the disturbance rejection was **on**, controller showed a great performance and the control objectives were satisfied even when the plant external disturbance is included. But still LESO didn't perfectly track the total disturbances.

So, we can conclude that ADRC has well done its job.

We have repeated the simulation by changing μ=0.03, 0.1, and 0.5 by turning the external distrurbance (**d(t)**) and the active disturbance rejection signal(**x3_hat**) on and off.

The result showed that the active disturbance rejection signal played a great role to minimize the error and increased the control performance depending on the bandwidth(μ value). Whereas the simulation result with no active disturbance rejection has showed poor control performance even at higher bandwidth and no disturbance. The figure below (**Fig7c**) showed the simulation result at μ = 0.03, d = 0). As we can observe from the result, it has high error at the best condtion. Whereas

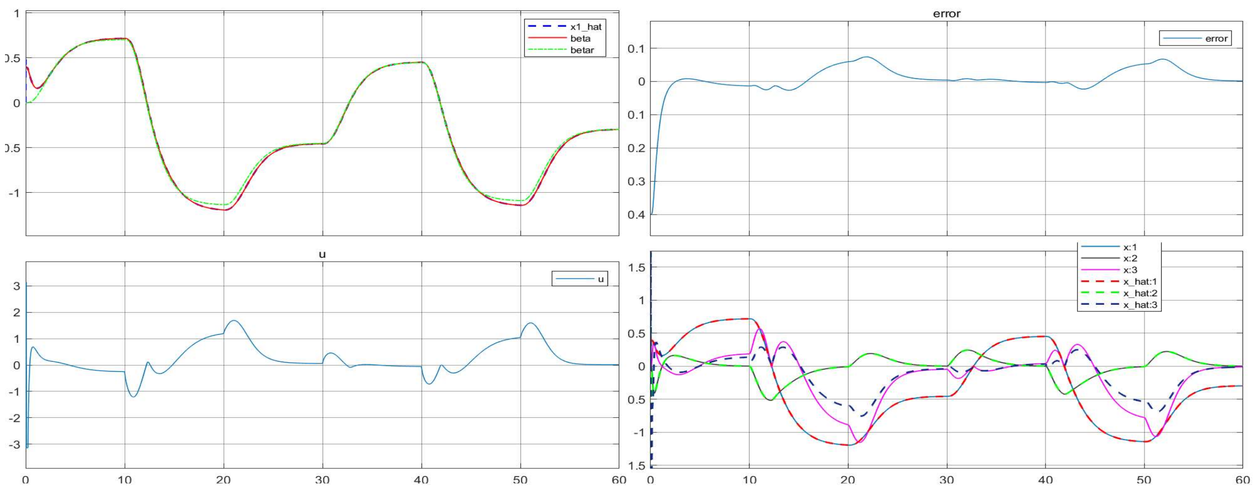


Fig7c. ADRC disturbance tracking performance at **μ=0.03, d=0** with square wave and without stochastic noise when **x3_hat = 0** (gain = 0)

Discrete ADRC Controller

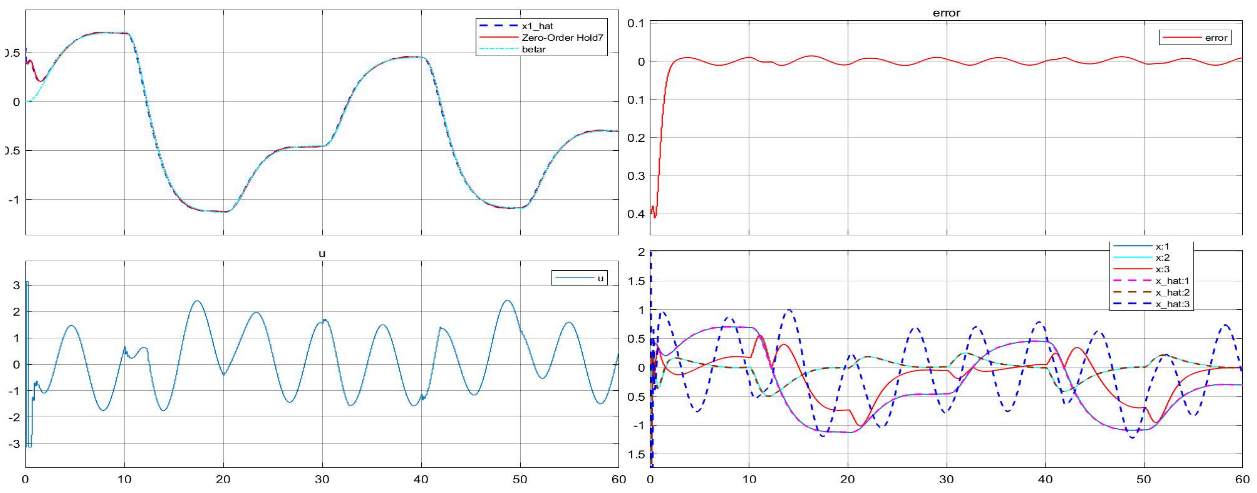


Fig8a. ADRC disturbance tracking performance at μ=0.1, d != 0 with uniform random input and with Ta=0.01

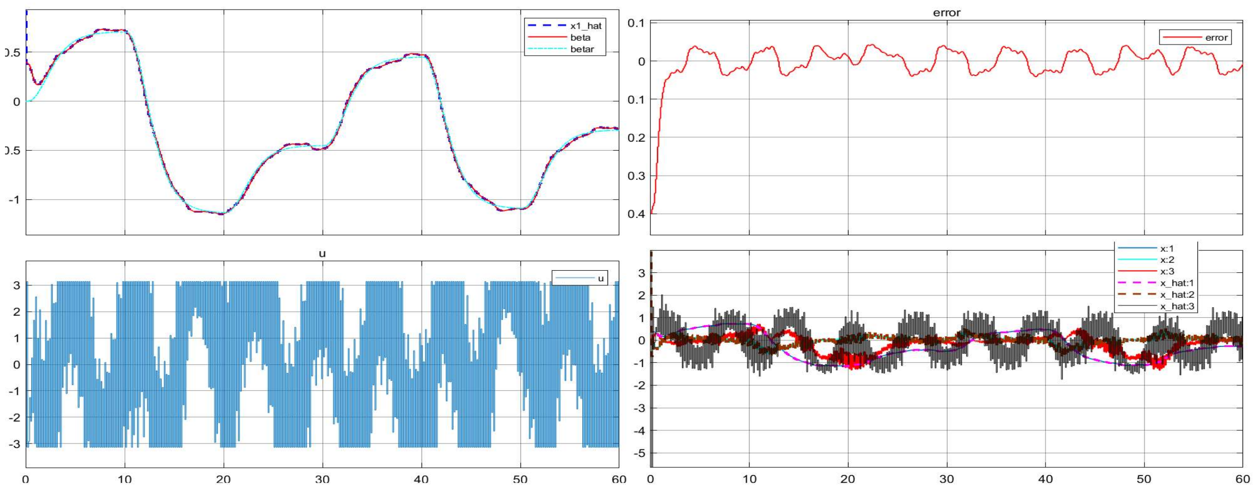


Fig8b. ADRC disturbance tracking performance at μ=0.1, d != 0 with uniform random input and with Ta=0.03

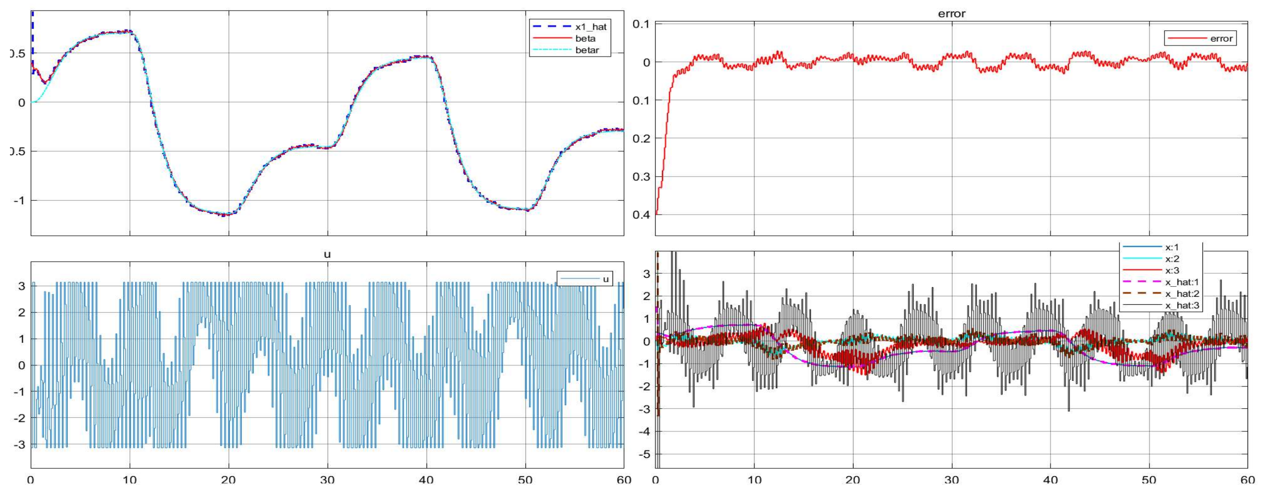


Fig8c. ADRC disturbance tracking performance at μ=0.1, d != 0 with uniform random input and with Ta=0.04

By comparing the above three figures (**Fig8a,b&c**), it was easy to say that the discrete LESO controller highly dependent on the sampling time **Ta** of the observer. As the the value of **Ta** increased, the controller performance goes worse and it didn't satisfy the control objectives when disturbance rejection is turned on **(Fig8c).** In addition, when external disturbance was added, the total disturbance wasn't tracked by LESO. The effect of external disturbance also higher when the value of Ta was larger.

So, proper selection of Ta is required to get a better performance of ADRC.

Conclusion

Active Disturbance Rejection Control (ADRC), also known as automatic disturbance rejection control, is a **model-free control technique** used for designing controllers in systems with unknown dynamics and external disturbances. It is the best control technique that has both robust and adaptive performances.

In this lab exercise, we have designed an ADRC method for the aero-plant rolling dynamics. The simulation results demonstrated the following qualities:

- **Smaller Overshoot**: The ADRC algorithm exhibited reduced overshoot during transient responses, leading to smoother control actions.
- **Lower Steady-State Error:** ADRC achieved better steady-state performance, minimizing deviations from the desired setpoint.
- **Faster Convergence Rate:** The ADRC controller converged more rapidly, ensuring quicker stabilization of the aero-plant.
- But **one limitation** of ADRC was its dependece on the observer sampling time **Ta.**

The ADRC algorithm outperformed the PD controller in terms of stability, precision, and speed. Its ability to handle disturbances effectively makes it a promising choice for self-balancing systems.

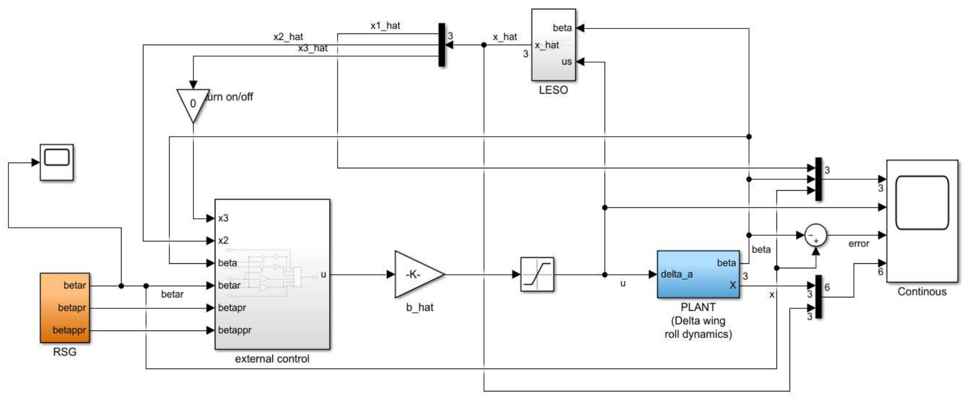
In general, the model-free design option, its robust and adaptive nature and its high performance made ADRC the best controller.

.

.

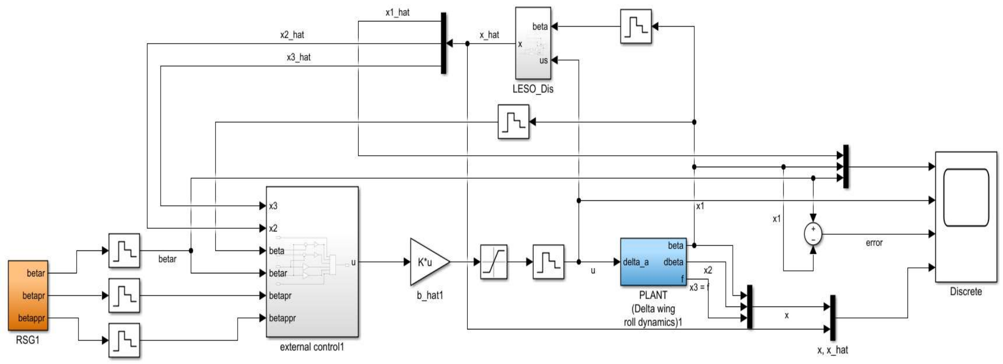
Appendix

Continuous ADRC Simulink model



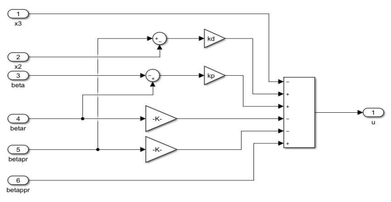
Continous Time Model

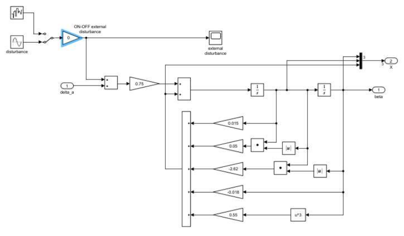
Discrete ADRC Simulink model

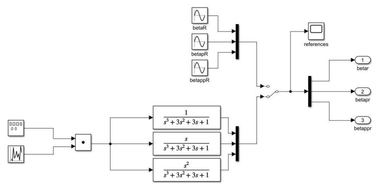


Discrete Time Model

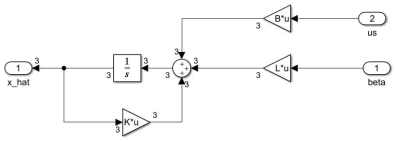
Supporting blocks

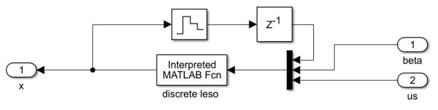






(a) external Control , (b) plant model , (c) reference input model ,





(d) continuous LESO , (e) discrete LESO

matlab function for discrete

```
function out = discrete_adrc(in)
global gamma_a La Ba
x = in(1:3)
y = in(4);
u = in(5);

x = gamma_a*x + La*y + Ba*u;

out = x
end
```

initials

```
% initial value declaration
global Ba La gamma_a
beta0 = 0.4;
betap0 = 0;
theta10 = -0.018;
theta20 = 0.015;
mu = 0.1;
alpha = 2;
um = pi;
wc = 2*pi/alpha;
w0 = wc/mu;
theta6 = 0.5;

A = [0 1 0; theta10 theta20 1 ; 0 0 0];
B = [0; theta6; 0];
C = [1 0 0];

l1 = 3*w0 + theta20;
l2 = 3*w0^2 + theta10 + l1*theta20;
l3 = w0^3;
L = [l1; l2; l3];
Ta = 0.01;
Tc = 0.1;
I = eye(3);
gamma = A - L*C;
gamma_a = I + Ta*gamma;
Ba = Ta*B;
La = Ta*L;
kp = wc^2+ theta10;
kd = 2*wc + theta20;
```