NUS

National University of Singapore

Module Code: EE5111 Selected Topics in Industrial Control & Instrumentation

Name: Jin Chengzhe

Student ID: A0224599N

Part 1

Based on the lecture notes, and any additional reading materials of yours, answer the following questions:

1. It is observed that a sensor has fault. The input/output data is shown in the table below. Please fit the nonlinear relationship between the input and output in an equation below

Input(x)	0.9	2.2	3.5	4.5	5.7	6.7
Output(y)	1.1	1.6	2.6	3.2	4.0	5.1

$$y = b_0 + b_1 x \quad + b_2 x^2 + \cdots + b_p x^p$$

(1)

a) Please select p=1,2,3,4,5 and compute the corresponding fault model (1) and plot the data in the graph;

If p=1, the fault model is $y = b_0 + b_1 x$. By bringing the input/output data into the model, we get $y = 0.6832 + 0.2576x$.

If p=2, the fault model is $y = b_0 + b_1 x + b_2 x^2$. By bringing the input/output data into the model, we get $y = 0.0454 + 0.3378x + 0.7369x^2$.

If p=3, the fault model is $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3$. By bringing the input/output data into the model, we get $y = 0.0033 + 0.0083x + 0.4564x^2 + 0.6440x^3$

If p=4 the fault model is $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4$. By bringing the input/output data into the model, we get $y = 0.0128 - 0.1918x + 1.0093x^2 - 1.4968x^3 + 1.7595x^4$

If p=5 the fault model is $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5$. By bringing the input/output data into the model, we get $y = -0.0024 + 0.0596x - 5283x^2 + 2.1047x^3 - 3.0588x^4 + 2.4955x^5$

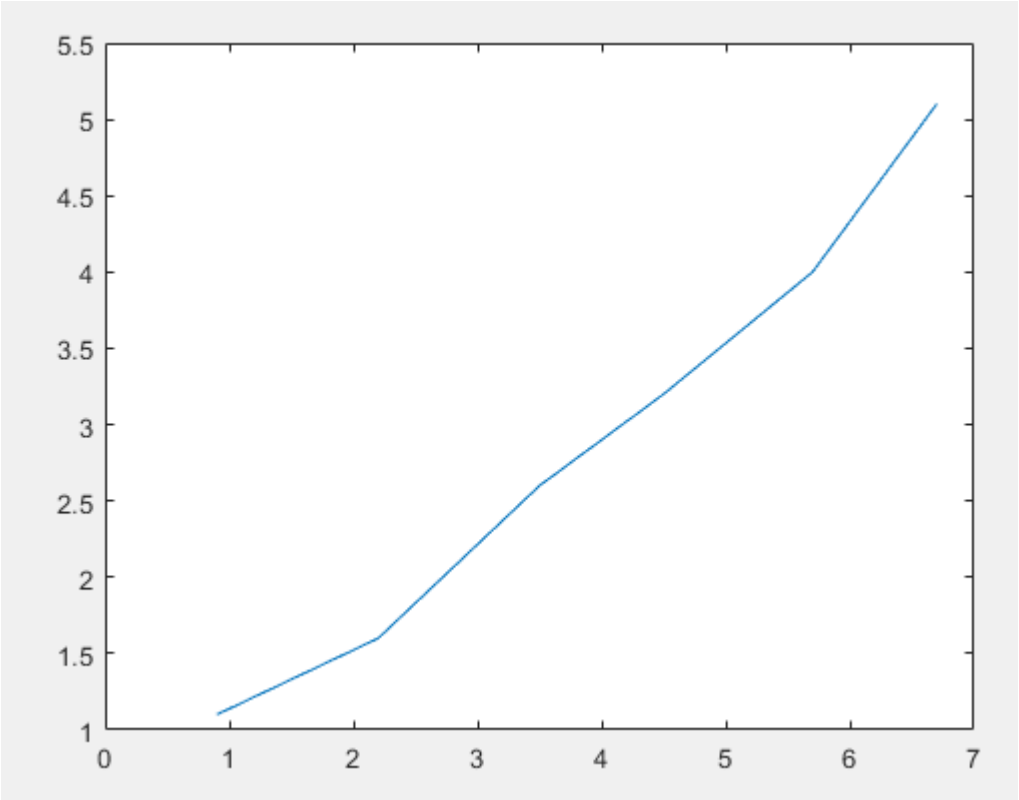


figure 1 data of I/O

b) According to the point (a) and working range of input [0.9, 6.7], please give the best fault model among them (p=1,2,3,4,5) and explain it.

The best fault model is $y = -0.0024 + 0.0596x - 5283x^2 + 2.1047x^3 - 3.0588x^4 + 2.4955x^5$. Because we use most forms of the equation, we have six coefficients from 0 power to 6th power of x so we can simulate the variance of input and output data more precisely. Also, by calculating the error from the correct value, the difference between estimated number and output data is the smallest. So we choose the fault model $y = -0.0024 + 0.0596x - 5283x^2 + 2.1047x^3 - 3.0588x^4 + 2.4955x^5$.

2. Hardware sensor fault diagnosis method can be used for detecting various faults in machines. Now we use one sensor to detect the possible faults in one machine. From the experimental test, we collected the sensor readings from the healthy state as shown below and wanted to do fault diagnosis.

Table 2.1 Healthy state (time from 1 to 9)

Input (time)	1	2	3	4	5	6	7	8	9
Sensor readings	4.1	4.2	4.3	4.15	4.66	4.7	4.8	4.75	4.8

After time 100, it was observed that we had the following sensor readings.

Table 2.2 Sensor readings (time from 100 to 108)

Input (time)	100	101	102	103	104	105	106	107	108
Sensor readings	6.1	6.2	6.15	6.5	6.7	7.2	7.4	8.5	9.8

Please choose a **suitable technique** to check if the machine has a fault (if you select the threshold decision method, the tolerance number is 0.15) and explain why you select this method. You may use the graph to illustrate your conclusion. Also, please give the fault detection time if a fault is detected.

Answer:

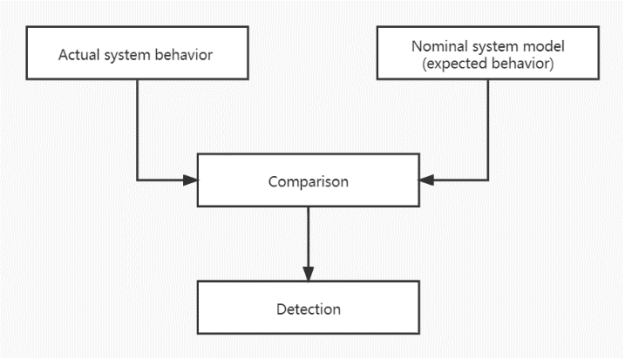
I want to use sensitivity analysis.

For the healthy state from 1 to 9, we can get the sensitivities are 0.1, 0.1, -0.15, 0.51, 0.04, 0.1, -0.05, 0.05. So the minimum value of sentivities is -0.15, the maximum value of sensitivities is 0.51.

Turning to the sensor readings from 100 to 108, the sensivities are 0.1, -0.05, 0.35, 0.2, 0.5, 0.2, 1.1, 1.3. Obviously we can find that the last two values are beyond the healthy state maximum sensitivities. So there is a fault in the system and the detection time is 107s.

I choose sensitivity ananlysis because the input is time and it increases 1 unit for each step, it's very convenient to calculate the sensitivities and sensitivity is quite direct to tell whether there is a fault.

3. For model-based fault detection method, please draw a block diagram to demonstrate it and explain it briefly.



For the steps in the fault detection method, first we want to compute estimated output. Then we generate a residual using the actual output by equation

$$r(k) = y_{true}(k) - y_{estimated}(k)$$

and then compare it with a given threshold. If the residual is larger than the given threshold, we declare fault detected, otherwise we say fault not detected. As for the threshold calculation, it is represented by

$$Mean + deviation + small\ tolerance\ number$$

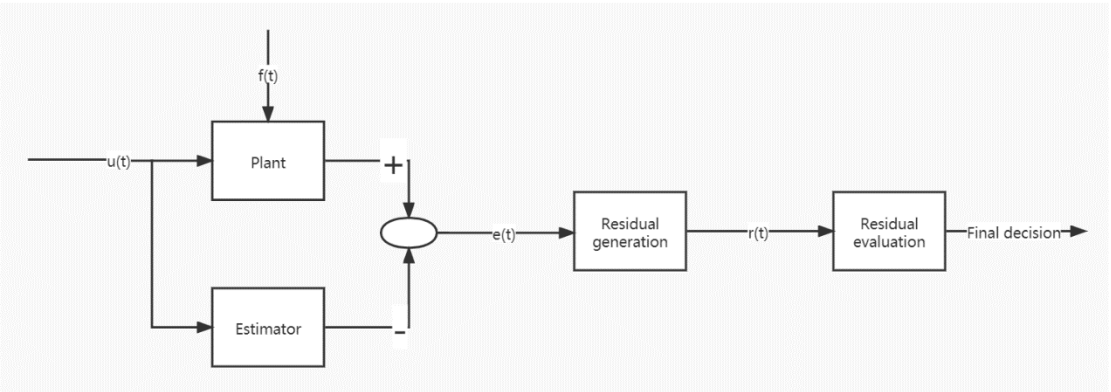
or more preciously $Mean + \alpha \times deviation\ (\alpha > 1)$.

We can do the above equation to get threshold. For example, linear system can be expressed by

$$\dot{x}(t) = Ax(t) + Bu(t)$$

We can compute estimated parameters (healthy state) and their threshold, then compare estimated parameters (current time) with fixed healthy parameter threshold. If parameter exceeds threshold, we assume there is fault, otherwise we say no fault occurs.

To build up a model, we need an estimator which can be a linear or nonlinear model observer; filters, which are usually linear or nonlinear Kalman filters and nonlinear neural network model. The model concept graph is like below:



Part 2

Based on the lecture notes, and any additional reading materials from literature, answer the following questions:

- **What is fault-tolerant control? Please give one or two application examples.**

Fault-tolerant control is concerned that a control system can accommodate faults and continues to work properly when faults occur. Its aim is to prevent those simple faults develop into serious failure. Fault-tolerant control (FTC) contains traditional FTC methods and modern FTC methods. Traditional FTC methods means if faults occur, we do manual service. Modern fault tolerant control methods will handle faults in industrial automation systems. Modern fault tolerant control methods include hardware methods, software methods and mixed methods. Hardware methods—use redundant hardware (spare) to accommodate faults. Software methods use software to implement the designed algorithms to accommodate faults (less sensors). Mixed methods---use both automatic handling or intervention of skilled personals.

Here I would like to use Pixhawk PX4 2.4.8 Flight Controller Kit Set as an example.

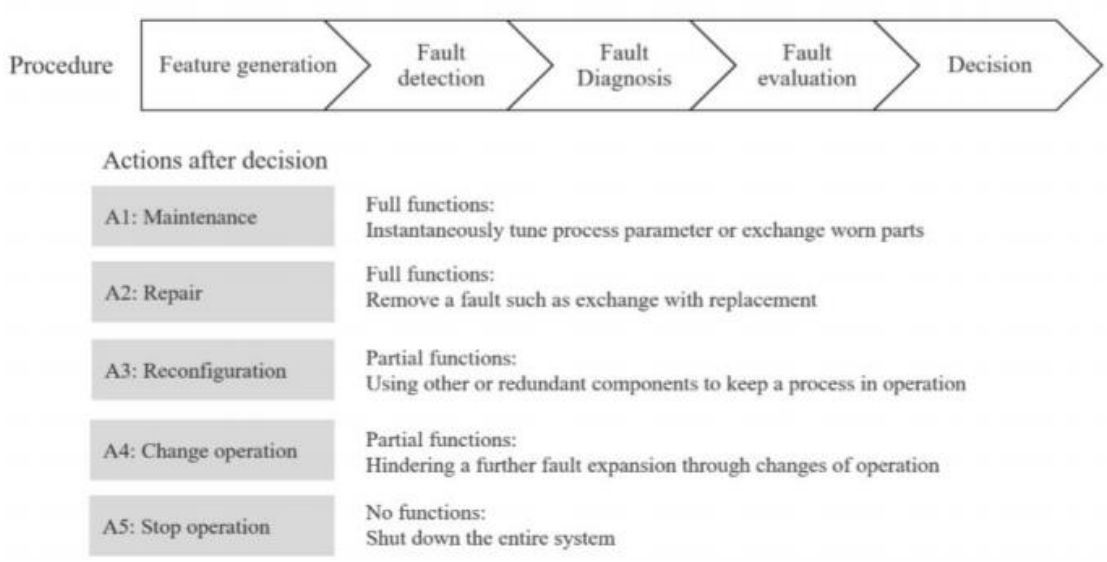
figure 2 Pixhawk PX4 2.4.8

The kit contains these sensors: L3GD20 3 axis digital 16-bit gyroscope, LSM303D 3 axis 14-bit accelerometer/magnetometer, MPU6000 6 axis accelerometer/ magnetometer (redundant component), MS5611 high precision barometer, GPS. Pixhawk uses two accelerometers for fault tolerant control: accelerometer ID1 and accelerometer ID2, usually ID1 is used as default. When velocity changes, accelerometer ID1 should change; if it changes, it works; otherwise, switch to ID2. Also, the fault detection checks the range of the accelerometer. If exceeding the range, switch to ID2.

And we can do software-based fault-detection control which is also called the model-based FTC. The main goal of the model-based FTC system is to design the controller, which enables stability and satisfactory performance, not only when all control components are healthy, but also in cases when there are faults. Model-based FTC can be divided into these categories below: Passive FTC, which is designed to be robust against a set of predefined faults, active FTC, which reacts to the faults actively by reconfiguring control actions, and hybrid FTC which uses

a mixed techniques to do FTC or controls a mixed system.

Besides hardware-based and software-based FTC, we have mixed methods FTC. Mixed method FTC uses various algorithms and techniques, even intervention of skilled personals, to handle faults to adapt new requirements, extend functionality, eliminate faults (effect), and improve quality features. Usually, this method is used in industry. The graph below shows the usual steps of doing mixed methods FTC.



- **Please explain three soft-based fault-tolerant control methods, i.e., model-based fault-tolerant control methods, and draw the corresponding flowcharts, respectively.**

The three model-based fault-tolerant control methods are passive FTC, active FTC and hybrid FTC.

Passive FTC is designed to be robust against a set of predefined faults. The flowchart is like below:

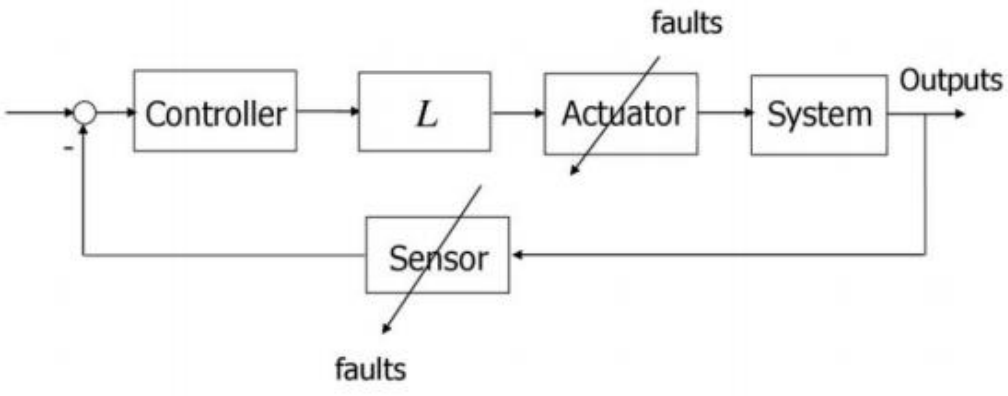


figure 3 Passive FTC flowchart

Passive fault-tolerant controller is similar to the robust approach when uncertain systems are considered. Further, such a controller works sub-optimally for the nominal plant because its

parameters are prearranged so as to get a trade-off between the performance and fault tolerance. The main feature of passive FTC is that is has robust fixed structure controller and faults have been considered at the controller design stage.

Suppose the pre-defined fault is $f = B_f u$. Then a linear system plus actuator fault is given by $\dot{x} = Ax + B(u + f)$. In this system, we use the feedback control $u = Kx$. The close-loop system is given by $\dot{x} = [A + (B + BB_f)K]x$. The goal of our control is to design the control gain K such that $[A + (B + BB_f)K]$ and $A + BK$ are stable.

The advantages of passive FTC are that we can design the robust control the achieve an acceptable performance and avoid the time delay. The main disadvantages are that only a limited number of faults can be tolerated and solution is often conservative.

Active FTC reacts to the faults actively by reconfiguring control actions.

An active FTC is sometimes referred to as self-repairing, reconfigurable, restructurable, or self-healing control systems. To achieve fault tolerance, the control system relies heavily on fault diagnosis.

There are two types of active FTC shown as follow.

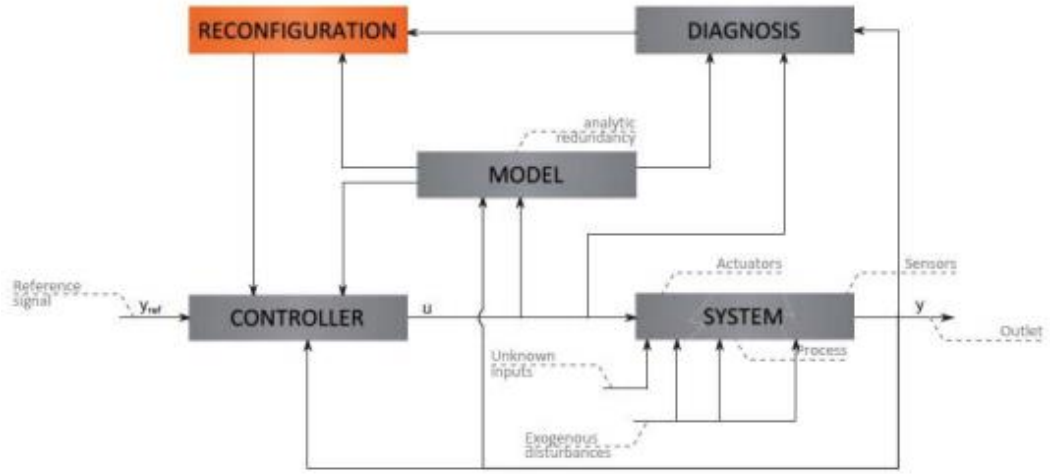


figure 4 active FTC type 1

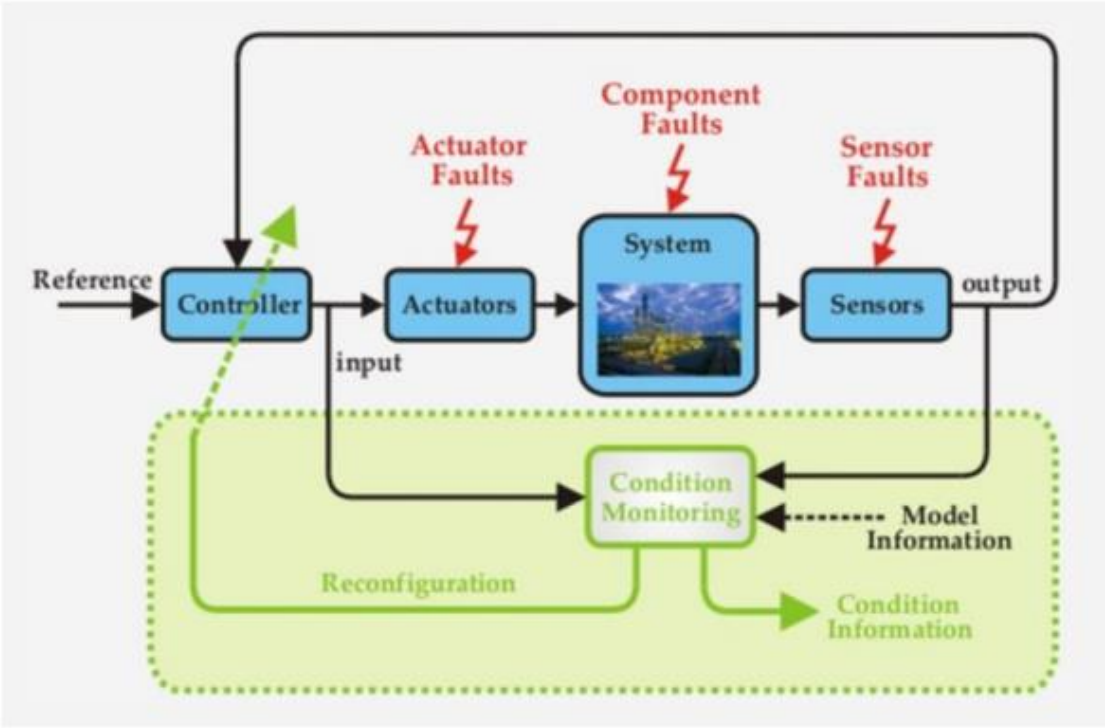


figure 5 active FTC type 2

We have three different methods to do with different conditions of facing fault function.

If fault function is known, which means f is known in $\dot{x} = Ax + B(u + f)$. The configured controller should be capable of canceling the fault function f. Assume that the normal control is u_n. Then, when the fault is detected, the configured control is $u_n + \hat{f}$. The fault tolerant control is given by $u = \begin{cases} u_n, & if\ no\ fault \\ u_n + \hat{f}, & if\ fault\ is\ detected \end{cases}$.

If the structure of fault function is known, but the coefficients are unknown, the fault functin is like $f = c_1\varphi_1(t) + c_2\varphi_2(t) + \ldots + c_n\varphi_n(t)$. Then adaptive control is used.

If the fault function is unknown, we should turn to artificial intelligence for help to learn the unknown fault and achieve FTC. Neural network, fuzzy logic system and adaptive learning algorithms can be used to develop an on-line estimator of the fault function. The flowchart of AI-based fault-tolerant control is as follows:

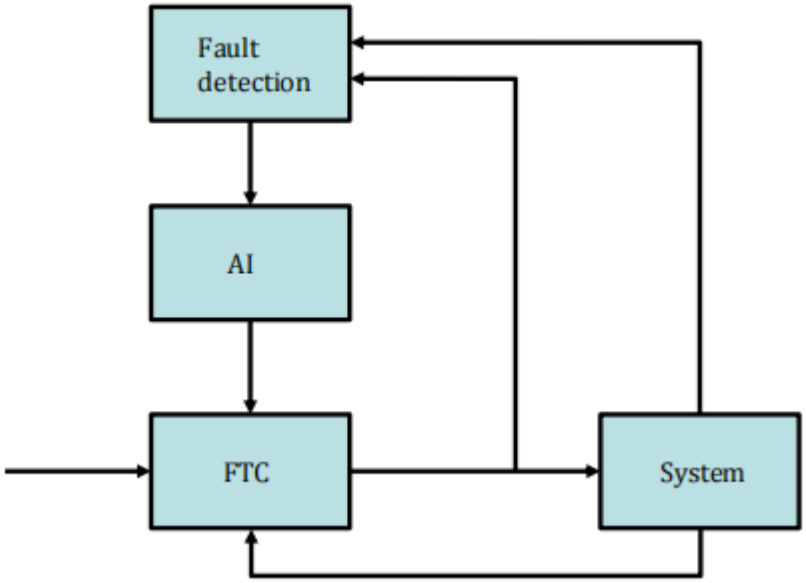


figure 6 AI based fault-tolerant control

Passive FTC and active FTC both have the same function which can accommodate faults. What is the difference between them two is that a passive FTC is a fixed controller for all situations while an active FTC reacts to the diagnosed faults by exercising the controls accordingly so that the stability can be maintained and the performance still be acceptable.

Hybrid FTC uses a mixed techniques to do FTC or controls a mixed system.

Hybrid FTC is mixed passive/active FTC (combining both) strategy. Other hybrid FTC---Hybrid systems are dynamical systems that involve the interaction of continuous and discrete dynamics.

The flowchart of hybrid FTC is:

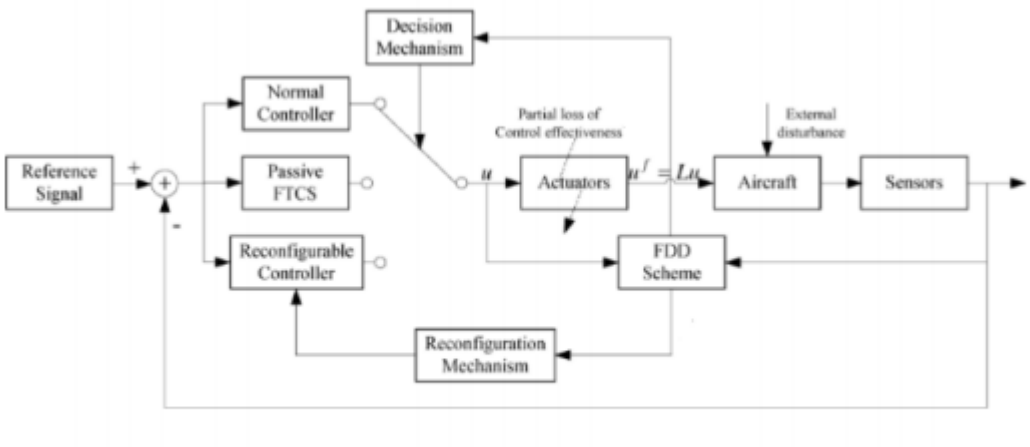


figure 7 Hybrid FTC