Chapter 5

Fuzzy Observers: Disturbance Rejection Fault Accommodation

Chapter 5 *Fuzzy Observers: Disturbance Rejection Fault Accommodation*

Disturbances caused by unmeasured inputs, plant perturbations or faulty actuators degrade the robustness and performance of both control and diagnostic systems. When these disturbances are unknown, robust accommodation of disturbances requires elaborate architectures.

Current techniques for disturbance rejection require either a precise quantitative model of a disturbance or extensive supervised learning. This requirement limits the disturbance rejection capabilities of a single observer to a single narrow class of quantifiable disturbance. If a set of several disturbances need to be rejected a corresponding set of observers is required, with each individual observer designed to reject only one of the known possible disturbances. Optimal rejection of the entire set of disturbance then relies on an autonomous supervisor which selects the appropriate observer outputs to use in construction the state estimation.

Both the quantitative and learnmg approach suffer from some serious limitations, including:

- disturbance must be known a priori,
- multiple observers must perform in parallel to reject multiple classes of disturbances, and
- estimates of disturbances are not available.

New techniques for disturbances reject must be developed that allow the rejection of unknown set of disturbances with only one observer. Chapter describes two accommodation methodologies: quantitative and qualitative.

The quantitative methodology uses the integral action of the **Proportional Integral (PI) Observer and Adaptive Observer** to estimate and accommodate disturbances. Our second methodology is based on **qualitative models** and uses stability and tracking behaviors, implemented with fuzzy rules, to achieve robust rejection of disturbances. Validation studies show that integral action is effective in estimating disturbances caused by unmeasured inputs, plant perturbations or faulty actuators, and a PI Observer-based controller will even outperform a Linear Quadratic Regulator in the presence of nonlinear actuator faults. The qualitative approach was validated with the 1992 ACC Robust Control Benchmark. Fuzzy controller achieves stability robustness and tracking robustness comparable to the other compensators.

The PI observer was developed by Shafai to extend the robustness of observers by including an **integral action** in the observer equation. This section extends the integral action for the purpose of disturbance rejection.

- Integral action is extended to the adaptive case, where the parameters of the plant are unknown, with the new **PI Adaptive Observer (PIAO)**.

- The integral action of a PI observer is shown to effectively estimate and compensate for an arbitrary set of disturbances with n distinct injection points, if n independent state measurements are available. Increasing the integral gain allows for the rejection of faster perturbations, but with the negative side effect of decreasing the filter's stability margin.

The robustness of the PI observer may be adversely effected by transitory disturbances with unmodelled distribution matrices. This can in fact severely limit the applicability of the PI observer; simple integral action cannot alone provide a robust solution to the 1992 ACC Benchmark.

The chapter describes generalization of the PI observer, the **Proportional Fading-Integral (PFI) observer** which discounts the integral term over time. Fading enables the rejection of these transitory events with unmodelled distribution matrices and improve the stability margin of the observer, allowing an unstable PI observer to become a stable PFI observer, yet with still sufficient integral action to reject disturbances.

QRC is used for disturbance rejection, Qualitative Robust Control, is based on qualitative modeling. These qualitative abstractions are frequently created using fuzzy logic and are represented as **symbolic linguistic models**. These models provide a convenient mechanism for tuning quantitative observers and improving their disturbance rejection properties and may be used to build fully fuzzy observers.

Existing fuzzy observers rely on machine learning to model a system, rather than using symbolic linguistic models of the system. The symbolic linguistic model effectively improves the robustness of the derived observer. The QRC methodology for designing fuzzy controllers transcends the quantitative methods that are utilized by the majority of fuzzy controllers described in the literature and contributes a new qualitative methodology.

Instead of using the fuzzified versions of quantitative controllers, qualitative models of the plant behavior is used. Robust set-point control is achieved if a qualitative plant model that subsumes all specified plant perturbations is utilized in designing the controller. A rule-based fuzzy controller is then incrementally designed based on the "qualitatively robust" plant model. After characterizing stability behaviors, tracking behaviors are developed to augment the controller. The stability and tracking behaviors are robust over the extent of plant configurations that are subsumed by the qualitative plant model. The resulting fuzzy controller supports both stability and performance robustness and allows for simple compensator tuning by changing linguistically interpretable rule parameters. The following sections provide a brief overview of the background material.

5.1 Fault Detection

The two main applications of state observers are observer-based state feedback control and fault detection. Both applications rely on accurate state estimation and suffer from performance degradation when input disturbances corrupt the observer's state estimates.

Faults are often detected by monitoring the measurement residuals of state observers. Excessive measurement residuals are interpreted as being indicative of a fault. A tradeoff, however, must be made between detecting all faults and creating an excessive number of false alarms since measurement residuals can also be generated by unmodelled plant dynamics, parameter mismatch or plant input disturbances. Disturbance decoupling is required in order to distinguish between true faults and the effects of disturbances .

Accurate estimation and robust accommodation of actuator faults can greatly increase the reliability and flexibility of control systems. Estimation allows for the characterization and classification of the fault, while actuator fault accommodation increases the robustness of the control system and allows time for diagnostic evaluation of the fault mechanism. With sufficiently accurate fault estimates and sufficiently robust accommodation, the dynamics of a fault can be closely monitored and used for the preemptive scheduling of repairs, without interrupting normal plant operation.

Robust accommodation also allows for the utilization of less expensive actuators . High accuracy and performance can thus be achieved with components that previously were not precise enough and did not have sufficiently stable performance characteristics.

No simple scheme exists for the design of a controller that both estimates and accommodates for unknown actuator faults. If the type of fault is known, and has a priori been characterized by a piecewise linear model, adaptive techniques exist for estimating the parameters of the fault model. Other techniques can accommodate, but not estimate, a class of faults with a known H^{∞} bound and much work has been done in accommodating actuator faults in systems with redundant actuators .More complex methodologies have also been developed that use computer-automated reconfiguration of control laws to accommodate for a set of known actuator faults .

Qualitative approach to fault detection requires the creation of qualitative models of the underlying quantitative system. These qualitative models must be designed so they support reasoning that is consistent with the quantitative system. This coupling of consistent qualitative models with a continuous, quantitative plant is called a **hybrid system.**

This section explains how qualitative models are abstracted from quantitative systems and then introduces the implementation of fuzzy systems from qualitative models. It concludes with an overview of fuzzy tuning systems for linear observers, linear observer-based fault detectors and Proportional, Integral and Differential (PID) controllers, and a discussion of full fuzzy controllers.

5.1.1 Learning Fuzzy Models

Learning or tuning allows the initial linguistic fuzzy model developed from heuristic domain knowledge to be optimized. Learning is achieved by using a neuro-fuzzy structure and exploiting the supervised learning strategies originally developed for neural networks. These strategies include gradient descent back-propagation , least-mean-squares, and a hybrid methodology that combines least-squares to optimize linear parameters and back-propagation to optimize the nonlinear parameters .

These same supervised learning methodologies can automatically learn any arbitrary nonlinear mapping between input and output without an initial linguistic fuzzy model . The resulting self-organized fuzzy models do not necessarily have a linguistic interpretation that would be recognized by a human expert. Often systems developed through self-organization are never interpreted linguistically, but are utilized effectively for pattern matching and curve fitting.

Moore, Harris and Rogers have used least-mean-squares learning to train a set of fuzzy networks, where each individual fuzzy network is trained with noisy data to track a target when perturbed by a single unique acceleration disturbance/maneuver. These fuzzy networks are then used in a hybrid scheme to detect and identify maneuvers and estimate target position.

A similar approach to solving the least-mean-squares estimation problem has been developed by Chao and Teng . A fuzzy network is trained off-line to estimate the state of a non-linear process from a sequence of noisy measurements. An estimation correction term similar to that utilized by a Kalman filter is included in the observer to ensure stability and convergence. As in the work by Moore, Harris and Rogers no linguistic qualitative model is employed in the design of the observer.

Fuzzy tuning of quantitative systems is a large field, but by far the largest application of fuzzy systems is control. The next section describes fuzzy derivatives of quantitative controllers and controllers based on qualitative models.

The majority of fuzzy controllers described in the literature fall into three main categories:

- fuzzy PID controllers,
- fuzzy sliding mode controllers, and
- fuzzy gain scheduling.

All three compensators realize close-loop control action and are based on quantitative control techniques. The fuzzy PID controllers and fuzzy slide mode controllers are fuzzy implementations of the linear quantitative PID controller and a nonlinear quantitative sliding mode controller. Both controllers use the error term and its derivatives and integrals as input into a fuzzy rule base. Coleman and Godbode compare the robustness of a fuzzy PID controller with that of a conventional PID and sliding mode controller and conclude that the fuzzy controller has equivalent robustness characteristics.

The fuzzy gain scheduler uses Sugeno type fuzzy rules to interpolate between several control strategies . This methodology is useful for controlling nonlinear plants that are piece wise linear or for linear plants that have a time varying parameter. An example of this is a controller built for an inverted pendulum with a variable length pendulum. Measurements of the pendulum length are used as inputs into a fuzzy rule base that interpolates the output of a small number of controllers that are optimized for controlling short, medium length and long pendulums .

Additionally, fuzzy controllers have been developed using qualitative models of target behavior. These controllers are often developed using the following steps:

1. rules are developed to realize a localized qualitative behavior,
2. global behavior caused by the interpolated localized rules is tested, and
3. Behavior is refined by tuning localized behavior and superimposing additional localized and global behaviors.

Two solutions are presented. The first uses the PI Observer and its new variants to estimate and accommodate unknown disturbances. The second uses fuzzy control based on qualitative models to accommodate for disturbance. The solution steps involving the integral action are:

- characterize the quantitative PI techniques for disturbance rejection for PI observer and the PI Kalman filter,
- increase parameter robustness of the PI observer by introducing an adaptive PI observer, and
- increase the disturbance rejection robustness of the PI observer by developing the Proportional Fading-Integral (PFI) observer.

The solution steps for the approach using qualitative models and behaviors are:

- create design methodology for a robust state-feedback fuzzy controller that use qualitative behaviors that incorporates robust stability and tracking behaviors, and
- create a robust hybrid output-feedback controller that combines the fuzzy controller with a robust PFI observer.

5.2 Fault Estimation and Accommodation

Faults can take several forms, but the literature usually focuses on faulty actuators and sensors. Additional faults can be caused by excessive friction, component failure, or age degradation. This thesis will focus on actuator faults and will propose a benchmark for the accommodation of frictional faults to be used in future research.

Actuator faults can be linear or nonlinear in nature. Gain mismatch and gain offsets are common examples of idealized linear faults. Backlash and deadzone are often described as typical idealized "actuator nonlinearities" (Tao and Kokotovic 1996 [1]), and actuator saturation is often indicative of a nonlinear actuator power limit (e.g. all actuators have fixed output ranges). These faults can occur in combination, or vary in severity with changes in operating conditions, maintenance schedule and age. Fig 5.1-5.4 shows the effects of gain mismatch, deadzone, backlash and saturation on actuator outputs.

5.2.1 Gain Mismatch Faults

Faults caused by gain mismatch are often caused by component drift or mis-calibration. The output of an actuator with this simple linear fault is

$$u = f(w) = K_{amen} w \tag{5.1}$$

Fig 5.1shows a gain mismatch with $k_{atten} = 0.6$

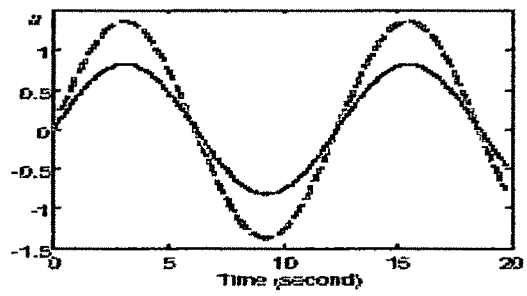


Fig 5.1: Gain Mismatch

5.2.2 Deadzone Faults

Deadzone faults suppress actuator output for a range of Input values. This fault is often caused by friction in the actuator. The output of the failed actuator is

$$u = f(w) = \begin{cases} w - k_{dz} & \text{if } w \geq \mathrm{k_{dz}} \\ 0 & \text{if } -\mathrm{k_{dz}} \leq w \leq \mathrm{k_{dz}} \\ w - k_{dz} & \text{if } w \leq -\mathrm{k_{dz}} \end{cases} \tag{5.2}$$

Fig 5.2 shows a deadzone fault with $k_{dz} = 0.3$.

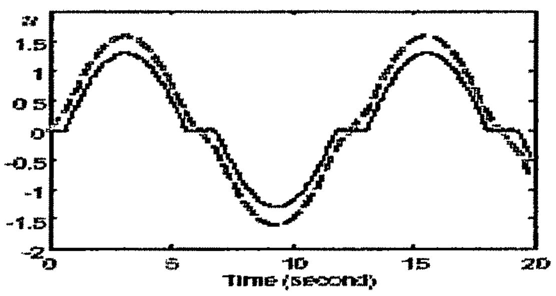


Fig 5.2 Dead-zone

5.2.3 Backlash Faults

Backlash is a simple form of hysteresis and differs from the other three faults described here in that it has memory; the current state of the fault mechanism is a function of it's previous state. A compact representation of the fault mechanism is

$$\dot{u}(t) = f(w) = \begin{cases} \dot{w} & \text{if } \dot{w} > 0 \text{ and } \mathrm{u(t) = w - k_{bl}} \\ 0 & \text{if } \dot{w} = 0 \end{cases} \tag{5 3}$$

Fig 5.3 shows a backlash fault with $k_{bl} = 1$.

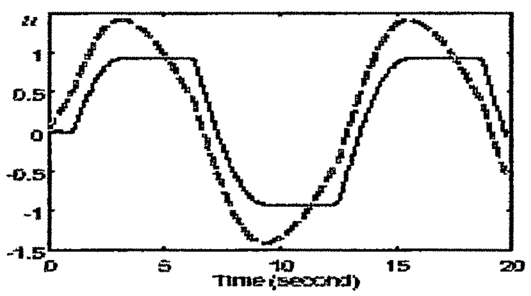


Fig 5.3: Backlash

5.2.4 Saturation Faults

Almost all actuators saturate at some extreme range of inputs, beyond the nominal operating range of the actuator. However, over time the performance of an actuator can deteriorate, and saturation begins to overlap the nominal operating range of the actuator. The output of the failed actuator is

$$u = f(w) = \begin{cases} k_{sat} & if\ w \geq k_{sat} \\ w & if\ -k_{sat} \leq w \leq k_{sat} \\ -k_{sat} & if\ w \leq -k_{sat} \end{cases} \tag{5.4}$$

Fig 5.4 shows a saturation fault with $k_{sat} = 1$.

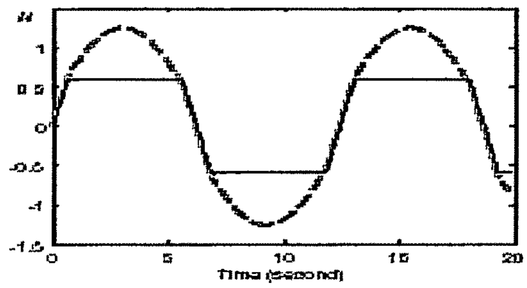


Fig 5.4: Saturation Fault

5.3 Validation Benchmarks

This section describes the validation benchmarks to verify the theoretical results. described in the previous chapters. The ACC Robust Control Benchmarks are used for evaluating the disturbance rejection robustness of controllers. The chapter continues with the additional benchmarks develop to validate the results of the dissertation and future research.

An extension to the Robust Control Benchmark is described to validate the frictional fault rejection properties of observers and observer based controllers. Modified benchmark is proposed to validate the disturbance rejection properties of adaptive observers.

5.3.1 Robust Control Benchmarks

The benchmark plant consists of a simple mechanical and the benchmark scenarios consist of rejecting input disturbances and tracking a unit step. The IFAC '93 (Graebe 1994[2]) was introduced at the 12th IFAC World Congress in Sydney Australia. The benchmark plant is seventh order with a nominal third order realization and has parameters with no physical interpretation. The exact plant is unknown to the control engineer and a tracking controller must be designed based only on noisy measurements from a *black box* simulation.

The ACC '92 benchmark, subsequently referred to as the *Benchmark*, was selected to validate the theoretical work in qualitative fuzzy control and observer design because the simple mechanical plant of the Benchmark lends itself to qualitative modeling The following sections describe the Benchmark plant and design scenarios.

ACC '92 Robust Control Benchmark

The Benchmark plant shown in Fig 5.5 is a simple flexible structure consisting of two masses connected with a single spring. This dynamic system has a **noncollocated** sensor and actuator; the sensor senses the position of **m₂** while the actuator accelerates **m₁**

input	u
output	$y = x_2$
disturbance	w

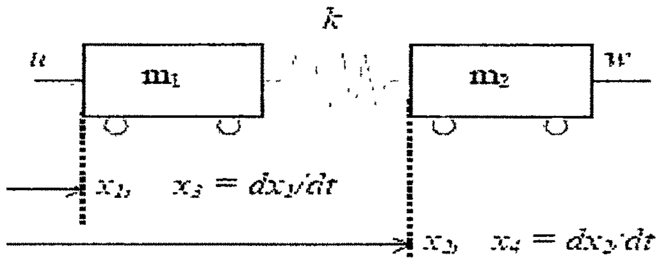


Fig 5.5: Plant used in the ACC benchmark.

The state space model for the plant is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -k/m_1 & k/m_1 & 0 & 0 \\ k/m_2 & -k/m_2 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1/m_1 \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/m_2 \end{bmatrix} w \qquad (5\ 5)$$

where:

x₁ is the position of **m₁**,
x₂ is the position of **m₂**,
x₃ is the velocity of **m₁**,

x_4 is the velocity of **m_2**,
y is the plant output **x_2**,
w is the acceleration disturbance on **m_2** and
u is the control acceleration on **m_1**

The following additional variables are needed to complete a description of the closed loop, series compensated system shown in Fig 5.6:

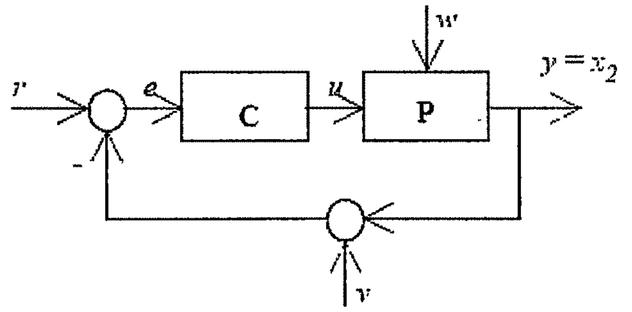


Fig 5.6 (a) Plant P with Series Compensator in a negative feedback loop.

v is the sensor noise,
e is the compensator input and
r is the reference input.

The corresponding transfer function between the plant input (actuator output) and plant output is

$$\tau_{uy} = \frac{(k/m_1 m_2)}{s^2\left[s^2 + k(m_1 + m_2)/m_1 m_2\right]} \tag{5.6}$$

The corresponding transfer function between the disturbance and plant output is

$$\tau_{wy} = \frac{(1/m_2)(s^2 + k/m_1)}{s^2\left[s^2 + k(m_1 + m_2)/m_1 m_2\right]} \tag{5.7}$$

5.3.2. Benchmark Extensions: Adaptive Observer

The ACC Benchmark presented in the Background Chapter is a robust control benchmark and as such is insufficient for benchmarking and validating failure detection and disturbance rejection properties of adaptive observers However, the ACC Benchmark can easily be augmented to include frictional failure scenarios and a simple disturbance rejection benchmark can be created from the plant used by Kudva and Narendra (Kudva and Narendra 1973 [3]) in their adaptive observer paper. The benchmark presented at the end of this section based on the Kudva and Narendra plant is called the **Adaptive Observer Benchmark**.

The ACC Benchmark can easily be augmented to test the robustness of the compensated system to system failure. The failures consist of Coulomb friction acting on the individual masses (Schneider and Frank 1996[4]). Three failure modes can exist, with friction on either **m**$_1$ or **m**$_2$, or friction on both masses.

The Adaptive Observer Benchmark uses the plant from Kudva and Narendra (Kudva and Narendra 1973[3]), described by the equations

$$\begin{aligned}\dot{x} &= \begin{bmatrix} -5 & 1 \\ -10 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 2 \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} x = x_1 \end{aligned} \qquad (5.8)$$

All four of the design scenarios utilize the input

$$u = 5\sin(t) + 5\sin(2.5t) \qquad (5.9)$$

a combination of two distinct sine waves, to excite the two states of the plant and ensure convergence of the parameter estimates. The initial state and state estimates are $\hat{x}(0)$ $=$ **x**(0) $=$ **0** , and initial parameters estimates, $\hat{a}(t)$ or $\hat{b}(t)$, are 102% of the actual plant parameters:

$$\hat{a}_0 = 1.02a = \begin{bmatrix} 5.1 \\ 10.2 \end{bmatrix} \text{ and } \hat{b}_0 = 1.02b = \begin{bmatrix} 1.02 \\ 2\ 04 \end{bmatrix} \qquad (5.10)$$

5.4 PI Adaptive Observer

The development of the PI adaptive observer parallels the development of the PI observer from the conventional P observer; a term proportional to the integral of the estimation error, *v*, is added to the conventional P observer equation. However, instead of the term **B***v(t)* being added to the observer equation, the term $\hat{b}(t)\,v(t\,)$must be added for the adaptive case. Because the magnitude of the integral offset is now dependent on the estimate$\hat{b}(t)$, and not **b**, the integral action is no longer unambiguously associated with the errors in estimating state With the adaptive version of the PI observer the integral action can now result from either state estimation error or parameter estimation error. Therefore, care must be taken in assuring that the parameter adaptation is not completely corrupted by the observer's integral action.

The PI adaptive observer is described by the equation

$$\begin{aligned}\dot{\hat{x}} &= K\hat{x} + [k - \hat{a}(t)]x_1(t) + \hat{b}(t)(u + v) + w_1(t) + w_2(t) \\ \dot{v} &= K_I(cx - y)\end{aligned} \qquad (5.11)$$

with the corresponding error equations

$$\dot{e} = Ke + \alpha(t)x_1(t) + \beta(t)u + \hat{b}(t)v + w_1(t) + w_2(t) \quad e_1 = ce$$
$$v = K_I e_1 \tag{5.12}$$

where $e = \hat{x} - x$, $\alpha = \hat{a}(t) - a$ and $\beta = \hat{b}(t) - b$. These error equations differ from the error equation for the P adaptive observer by the term $\hat{b}(t)\,v$.

Stability and convergence of this new error equation is shown by using an extension of the Gronwall-Bellman lemma (Narendra and Annaswamy 1989[5]) for almost time-invariant systems. First the error equations are converted to the following partition form:

$$\begin{bmatrix} \dot{e} \\ \dot{v} \end{bmatrix} = \left[\begin{bmatrix} K & \\ K_I c & 0 \end{bmatrix} + \begin{bmatrix} 0 & \hat{b}(t) \\ & 0 \end{bmatrix} \right] \begin{bmatrix} e \\ v \end{bmatrix} + \begin{bmatrix} \alpha(t) \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} \beta(t) \\ 0 \end{bmatrix} u + \begin{bmatrix} w_1(t) + w_2(t) \\ 0 \end{bmatrix} \tag{5.13}$$

The error equation has the structure $\dot{z} = (\tilde{A} + B(t))z + f(t)$, where

$$\tilde{A} = \begin{bmatrix} K & \\ K_I c & 0 \end{bmatrix}, \quad B(t) = \begin{bmatrix} 0 & \hat{b}(t) \\ & 0 \end{bmatrix} \text{ and}$$
$$f(t) = \begin{bmatrix} \alpha(t) \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} \beta(t) \\ 0 \end{bmatrix} u + \begin{bmatrix} w_1(t) + w_2(t) \\ 0 \end{bmatrix} \tag{5.14}$$

We can now use the following theorem to prove stability and convergence of the PIAO:

Theorem *The PIAO error dynamics* $\dot{z} = (\tilde{A} + B(t))z + f(t)$ *converges to zero for t > to if*

i) $\tilde{A}$ *is asymptotically stable,*

ii) there exists a b_0 *such that* $\| \mathbf{B}(t) \| \le b_0$ *for* $t > t_0$ *and*

iii) there exists a b_1 *such that* $\| \mathbf{f}(t) \| \le b_1$ *for* $t > t_0$.

5.4.1 Robust PI Kalman Filter

The PI version of the Kalman filter was first developed by Kim and Shafai (Kim, Shafai et al. 1989[6]; Kim and Shafai 1990[7]). The development was awkward and disturbance rejection was not explored. This section begins with a simplified development of integral action for the Kalman filter and then develops a necessary condition for the rejection of *n* simultaneous disturbances injected at *n* distinct injection points by integral action for either the PI observer or the PI Kalman filter.

The PI Kalman Filter contains the integral term **v**, which is added to the original estimator equation by using the pair of coupled differential equations

$$\hat{X}_{k|k} = A\hat{x}_{k|k-1} + K_k\left[y_k - C\hat{x}_{k|k-1}\right] + Bu(k) + B_I v(k)$$
$$v(k) = v(k-1) + K_I\left[y_k - C\hat{x}_{k|k-1}\right] \tag{5.15}$$

where the constant **K**ı determines the time constant of the integral action while **B**ı reflects the form of the plant perturbations or disturbance injection points. These quations can be manipulated to give one augmented equation that has the form of the estimator equation for the standard Kalman filter

$$\hat{z}_{k|k} = \overline{A}\hat{z}_{k|k-1} + \overline{K}\left[y_k - C\hat{z}_{k|k-1}\right] + \overline{B}u(k) \tag{5.16}$$

where

$$\overline{A} = \begin{bmatrix} A & B_I \\ 0 & I \end{bmatrix},\ \overline{\mathrm{B}} = \begin{bmatrix} \mathrm{B} \\ 0 \end{bmatrix}, \overline{\mathrm{C}} = [\mathrm{C} \quad 0],\ \overline{\mathrm{K}} = \begin{bmatrix} \mathrm{K_K} \\ \mathrm{K_I} \end{bmatrix} \text{and } \mathrm{z} = \begin{bmatrix} \mathrm{x} \\ \mathrm{v} \end{bmatrix} \tag{5.17}$$

The optimal gain $\overline{K}$ for a given process and measurement noise covariance can be derived using technique developed by Kim and Shafai (Kim, Shafai et al. 1989[7]). However, when optimizing $\overline{K}$ for perturbation and disturbance rejection, $\mathbf{K_K}$ becomes the Kalman gain and the gain associated with the integral action $\mathbf{K_I}$ should be selected such that

$$R = \begin{bmatrix} A - \mathrm{K_K}C & B_I \\ -\mathrm{K_I}C & I \end{bmatrix} \tag{5.18}$$

has eigenvalues within the unit disc.

The integral action of the PI Kalman filter is effective in estimating both plant perturbations and input disturbances that can be modeled by the term $\mathbf{B_{I}v}$. When rejecting plant perturbations, the perturbation must be in the form **DΔE**, and **D** becomes $\mathbf{B_I}$. As an example, the PI Kalman filter can be used to reject the perturbation to the Benchmark plant caused by changes in the spring constant provided that $\mathbf{B_I} = \mathbf{D} = [\ 0\ 0\ \text{-}1\ 1]'$. Correspondingly when rejecting input disturbances, the distnbution matrix of the disturbances must be used as $\mathbf{B_I}$. As an example, when rejecting a disturbance to m_I the injection point of the disturbance is x_3 and $\mathbf{B_I} = [\ 0\ 0\ 1\ 0]'$.

However for single output systems, a PI Kalman Filter can not simultaneously reject both a perturbation and slowly varying disturbance. With only one state measurement the PI Kalman Filter can reject either (1) a single rank one perturbation or (2) an input disturbance with a single known injection point. This can be generalized to the following:

Theorem *If the plant [A, B, C] of order n is completely observable, and ıf*

$$r \geq \rho\begin{bmatrix} A & B_I \\ C & I \end{bmatrix} - n$$

where r is the number of independent outputs. then all perturbations and disturbances modeled by $\mathbf{B}_I$ *can be rejected.*

The proof is a generalization of the development given by Saif (Saif 1997[8]). The theorem requires that for the simultaneous rejection of a rank *a* perturbation and *b* disturbances with unique injection points requires (*a* + *b*) independent state measurements.

The robustness of PI Kalman filter to plant perturbations and disturbances can be adversely effected by transitory disturbances. These disturbances which are not modeled by $\mathbf{B}_I$ create an offset in the integral term and adversely impact the performance of the PI Kalman filter. Rejection of these transitory features can be achieved by discounting the integral term over time.

5.4.2 Kalman Filter with a Fading Integral Term

The fading of the integral term allows integral action to estimate and accommodate disturbances with known injection points in the presence of transitory disturbances with unknown injection points. Without fading any small transitory disturbance, who's injection point is not incorporated in $\mathbf{B}_I$, will add a permanent offset to the integral action, precluding the convergence of the disturbance estimate and the disturbance. The fading-integral is developed in this section for the Kalman filter; an equivalent development is also valid for the Luenberger observer.

The Proportional Fading-Integral (PFI) Kalman filter decays the integral term of the PI Kalman filter, allowing the effects of transient disturbances to decrease with time. The PFI Kalman filter is described by

$$\begin{aligned}\hat{x}_{k-|k} &= A\hat{x}_{k|k-1} + K_k\left[y_k - C\hat{x}_{k|k-1}\right] + Bu(k) + B_I v(k)\\ v(k) &= (I - K_F)v(k-1) + K_I\left[y_k - C\hat{x}_{k|k-1}\right]\end{aligned} \tag{5.19}$$

where the constant $\mathbf{K}_F$ determines the amount of fading. As for the PI Kalman Filter these equations can be manipulated to give one augmented equation with the following parameters

$$\overline{A} = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix},\ \overline{\mathrm{B}} = \begin{bmatrix} \mathrm{B} \\ 0 \end{bmatrix}, \overline{\mathrm{C}} = \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix},\ \overline{\mathrm{K}} = \begin{bmatrix} \mathrm{K_K} & \mathrm{B_I} \\ \mathrm{K_I} & -\mathrm{K_F} \end{bmatrix} \text{and } \mathrm{z} = \begin{bmatrix} \hat{\mathrm{x}} \\ \mathrm{v} \end{bmatrix} \tag{5.20}$$

Using the same technique as for the PI Kalman filter, the optimal gain $\overline{K}$ for a given process and measurement covariance can also be derived for this augmented system.When designing the filter for perturbation and disturbance rejection it is necessary to insure stability for the PFI Kalman filter by having the eigenvalues of

$$R = \begin{bmatrix} A - K_K C & B_I \\ -K_I C & I - K_F \end{bmatrix} \tag{5.20}$$

inside the unit disk. As shown by **R,** the additional freedom afforded by the **K**$_F$ term allows the design of PFI Kalman Filter with a larger stability margins than the corresponding PI Kalman filter. Indeed, an unstable PI Kalman Filter can be made stable with the addition of fading.

Simple integral action alone can achieve time recovery, but can not reject transitory events that contribute to the integral action. However, with the PFI Kalman Filter the fading term **K**$_F$ can be tuned so that the effects of transients on the integral action decay over time. Faster suppression of transients is achieved by increasing **K**$_F$, but often it is also necessary to increase the integral action gain **K**$_I$ so that fading term does not completely suppress the integral action.

5.4.3 Tuning Integral Action

Proper rejection of perturbations or disturbances requires that $\overline{K}$ for the PFI Kalman filter, or equivalently the PFI observer, be designed by separation. The following four steps are required:

1. solve for the optimal Kalman gain for the system [**A**, **B**, **C**],
2. determine **B**$_I$ from the perturbation structure and/or the disturbance injection matrices,
3. tune **K**$_I$ so that the transient response of the integral action can follows the fastest perturbation and/or disturbance, and
4. tune **K**$_F$ so that effects of perturbations and/or disturbances with unknown distribution matrices decay satisfactorily and the PFI Kalman filter is stable.

- **Tuning the Integral Gain**

A large integral gain **K**$_I$ shortens the transient response of the integral action to disturbances, but also leads to overshoot and ringing. The critical integral gain can be determined by achieving a critically damped transient response to a unit step disturbance. The speed of the critically damped transient response limits the ability of the PI Kalman filter to reject fast repetitive disturbances. An optimal integral gain can often not be found for a given disturbance because the fastest possible transient response of the integral action is too slow to match a quickly varying input disturbance. It is however possible to decrease the transient response by:

1. increasing the modeled process noise or
2. reducing the measurement noise which allows for larger integral gains.

- **Tuning the Fading Rate**

The fading of the integral term discounts the integral term over time. Decaying the integral term allows the rejection of transients that would otherwise cause large offsets in the estimates of plant perturbation and disturbance. This is analogous to the discounting of old measurements in a P Kalman Filter by increasing the magnitude of the process noise covariance used by the filter. In fact, increasing the model process noise will also fade the integral term of a PFI Kalman Filter; however large levels of process noise will increase the variance of state estimates.

With small values of K_F the PFI Kalman filter performance approaches that of the PI Kalman filter, while for large values of K_F the integral action of the filter is suppressed and the PFI Kalman filter performance approaches that of the standard P Kalman Filter.

5.5 Robust Fuzzy Control and Benchmark

The fuzzy controller design begins with the qualitative plant model. First state information must be extracted from the plant output by using a combination of linear methods and fuzzy process models. The crisp outputs of linear operators and fuzzy process models are used as input to a Sugeno FIS controller (Fig 5.6).

The design of the rules for the FIS is suggested by the plant's qualitative state transition diagram. Given the connection between qualitative input events and changes in qualitative states, as shown by the plant's state transition diagram, the FIS controller is constructed to generate the qualitative events that will result in the qualitative state transitions required to realize the desired control actions.

The first control objective is the stabilization of the plant. Stability for the Benchmark entails the dampening of vibrations after an external disturbance is applied. After stabilization, the FIS is augmented with rules to achieve performance objectives.

Fuzzy logic lends itself to this methodology, because fuzzy logic deals with possibilities and reasoning remains consistent even when conflicting requirements generate conflicting rules. Further tuning of the composite compensator can be made adjusting the relative weighting of the outputs of the different behaviors and by gating behaviors, allowing them to be performed only during specified states.

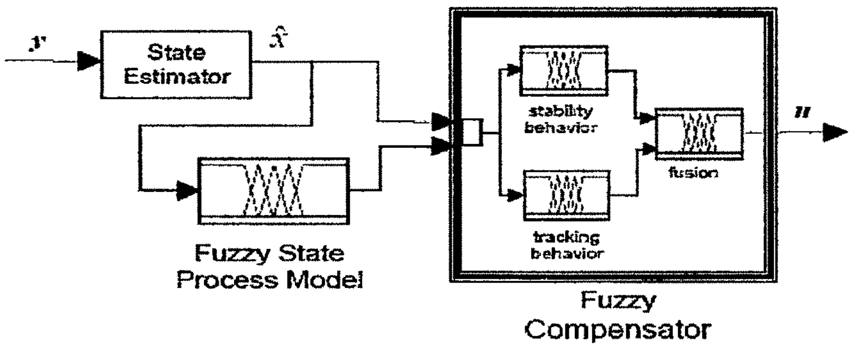


Fig 5.6(b): Fuzzy Controller with State Estimator

The Benchmark problem requires both the achievement of stability and tracking performance. The physical intuition is that the spring oscillations caused by disturbances must first be dampened to achieve stability, and then after stability is achieved the goal of tracking can become paramount. This nonlinear dichotomy is easily supported by fuzzy logic.

QRC allows the separate development of stability and tracking behaviors; the superimposition of these behaviors then achieves the final control objective. An analysis of the transfer function of the mass-spring-mass plant indicates that these behaviors should exploit the rigid-body-mode of the plant, where the plant behaves as if the masses are rigidly connected. If the stability behavior can be made to achieve this rigid-body-mode, then the tracking behavior can treats the mass-spring-mass system as a simple single mass. This allows a simpler tracking behavior to be effective.

The stability behavior is derived from the heuristic that a control action is most effective in suppressing plant vibration if it is applied when the spring is neutral, and the control action opposes the motion of the spring. As an extreme example of this effect, an impulse to a stationary plant can be rejected with only one complementary impulse of equal magnitude, if the impulse occurs exactly as the spring relaxes.

• Fuzzy Spring Process Model

A fuzzy process model of the spring needs to provide the qualitative state information necessary to dampen the vibrations of the plant and achieve stability. This can be achieved by abstracting the quantitative state of the Benchmark plant to just one qualitative state that indicates whether the spring is at its neutral length and whether the spring is in the process of compressing or elongating. This fuzzy process model requires that any estimate provided by a linear filter of the plant's quantitative state effectively captures:

- the timing of the spring relaxation
- the direction of motion of the spring (e.g. compressing and stretching).

Because the fuzzy process model only requires accurate estimates of state with respect to these two metrics, the linear filter does not need be optimal (e.g. Kalman filter), but can be a sub-optimal filter derived with robust filter methodologies.

The fuzzy process model utilizes a qualitative spring state that is specified by a qualitative partition of the spring length, $L = x_2 - x_1$, and the spring length velocity, $\dot{L} = \dot{x}_2 - \dot{x}_1$.

A Mamdani Fuzzy Inference System (FIS) applies seventeen rules, shown in Table 5.1, to infer the qualitative spring state, Q_L from the inputs L and $\dot{L}$. The output Q_L is partitioned into the following qualitative states mapped with triangular and trapezoidal membership functions to the interval [-1, 1]:

- spring is compressing rapidly and is relaxed,
- spring is compressing and is relaxed,
- spring is not in State 1, 2, 4 or 5.
- spring is stretching and is relaxed,
- spring is stretching rapidly and is relaxed,

The input and output membership functions for the process model are shown graphically in Fig 5.7 where L, $\dot{L}$ and Q_L are partitioned by five membership functions.

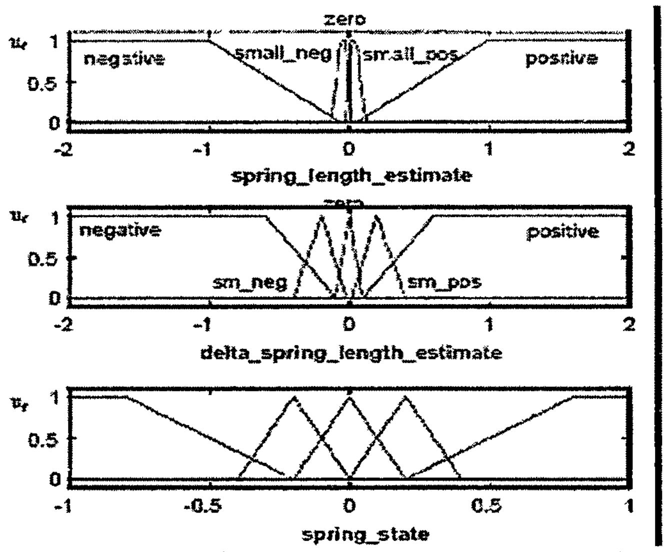


Fig 5.7: Input and output membership function of the Spring Observer.

1. **If** (spring_length_estimate **is** negative)
   then
   (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
2. **If** (spring_length_estimate **is** small_negative) **and** (delta_spring_length_estimate **is** negative)
   then
   (spring_state **is** compressing_fast_with_neutal_spring)
3. **If** (spring_length_estimate **is** small_negative) **and** (delta_spring_length_estimate **is** sm_neg)
   then
   (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
4. **If** (spring_length_estimate **is** small_negative) **and** (delta_spring_length_estimate **is** zero)
   then
   (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
5. **If** (spring_length_estimate **is** small_negative) **and** (delta_spring_length_estimate **is** sm_pos)
   then
   (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
6. **If** (spring_length_estimate **is** small_negative) **and** (delta_spring_length_estimate **is** positive)
   then
   (spring_state **is** stretching_fast_with_neutal_spring)
7. **If** (spring_length_estimate **is** zero) **and** (delta_spring_length_estimate **is** negative)
   then
   (spring_state **is** compressing_fast_with_neutal_spring)
8. **If** (spring_length_estimate **is** zero) **and** (delta_spring_length_estimate **is** sm_neg)
   then
   (spring_state **is** compressing_fast_with_neutal_spring)
9. **If** (spring_length_estimate **is** zero) **and** (delta_spring_length_estimate **is** zero)
   then
   (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
10. **If** (spring_length_estimate **is** zero) **and** (delta_spring_length_estimate **is** sm_pos)
    then
    (spring_state **is** stretching_fast_with_neutal_spring)
11. **If** (spring_length_estimate **is** zero) **and** (delta_spring_length_estimate **is** positive)
    then
    (spring_state **is** stretching_fast_with_neutal_spring)

12. **If** (spring_length_estimate **is** small_positive) **and** (delta_spring_length_estimate **is** negative)
    then
        (spring_state **is** compressing_fast_with_neutal_spring)
13. **If** (spring_length_estimate **is** small_positive) **and** (delta_spring_length_estimate **is** sm_neg)
    then
        (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
14. **If** (spring_length_estimate **is** small_positive) **and** (delta_spring_length_estimate **is** zero)
    then
        (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
15. **If** (spring_length_estimate **is** small_positive) **and** (delta_spring_length_estimate **is** sm_pos)
    then
        (spring_state **is** not_stretching_or_compressing_with_neutal_spring)
16. **If** (spring_length_estimate **is** small_positive) **and** (delta_spring_length_estimate **is** positive)
    then (spring_state **is** stretching_fast_with_neutal_spring)
17. **If** (spring_length_estimate **is** positive)
    then (spring_state **is** not_stretching_or_compressing_with_neutal_spring)

Table 5.1: Rules for determining spring state

The stability behavior is enabled during output states 1, 2, 4 and 5; these states indicate the spring is vibrating and transitioning through zero. Noise immunity was improved by requiring that when L is not zero, but rather small_positive or small_negative, that the level of $\dot{L}$ be large, either negative or positive, before Q_L is set to a state indicate a vibration. Only for zero values of L will small_positive or small_negative of $\dot{L}$ result in an output Q_L which indicates that a vibration needs to be suppressed.

- **Fuzzy Compensator**

The fuzzy compensator integrates both the stability and tracking behavior. First, a single Stability Behavior was developed and qualified, and then two tracking behaviors were developed. As when designing linear controllers, a tradeoff is required between maintaining the stability robustness of the Stability Behavior and increased tracking performance. Tracking Behavior A limits its control action to when the spring length is small, minimizing the interference with the Stability behavior, while Tracking Behavior B adds additional rules to improve the settling time performance, but at the expense of reducing stability robustness.

The Stability Behavior requires five rules while Tracking Behavior A required an additional three rules and Tracking Behavior B requires five rules. These rules used a combination of six (seven when Tracking Behavior B) Sugeno output functions where the output functions are a linear combination of the position and velocity inputs to the FIS.

b^{big}_{pos}	b^{big}_{vel}	b^{big}_{accel}	b^{small}_{accel}
-1	***-3***	***8***	***-0.6***

Table 5.2 Coefficients for output equations used by vibration suppression behavior.

Stability behavior is realized with 5 fuzzy rules that map the 5 fuzzy partitions of *QL* to five output membership functions. These rules are shown in Table 5.3. Vibrations are suppressed using the following five output functions:

big_stop_spring_stretching
$$y = b^{big}_{pos}x + b^{big}_{vel}\dot{x} + b^{big}_{accel} \tag{5.21}$$

small_stop_spring_stretching
$$y = b^{small}_{accel} \tag{5.22}$$

zero
$$y = 0 \tag{5.23}$$

small_stop_spring_compressing
$$y_{sssc} = b^{small}_{accel} \tag{5.24}$$

big_stop_spring_compressing
$$y_{bssc} = b^{big}_{pos}x + b^{big}_{vel}\dot{x} - b^{big}_{accel} \tag{5.25}$$

These output equations use four constants whose values are given in Table 5.2 actual vibration suppression is achieved only by the bias terms b^{big}_{accel} and b^{small}_{accel}. These terms generate a pulse which opposes the stretching or compression of the spring. The remaining terms bias the amplitude of the pulse, so that the pulse aids in zeroing both the position and velocity of the masses.

Rules to supress vibrations
1. **If** (spring state **is** compressing_fast_with_neutal_spring) **then** (control_output **is** big_stop_spring_stretching)
2. **If** (spring state **is** compressing_slowly_with_neutal_spring) **then** (control_output **is** small_stop_spring_stretching)
3. **If** (spring state **is** not_stretching_or_compressing_with_neutal_spring) **then** (control_output **is** zero)
4. **If** (spring state **is** stretching_slowly_at_zero_accel) **then** (control_output **is** small_stop_ spring_compressing)
5. **If** (spring state **is** stretching_fast_with_neutal_spring) **then** (control_output is big_stop_ spring_compressing)
Additional rules to achieve tracking of m_2
6. **If** (position_error **is** BigNegative) **then** (control_output **is** zero_large_position)
7. **If** (position_error **is** negative) **and** (spring_length **is** tinyBell) **then** (control_output **is** zero_small_position)
8. **If** (position_error is zero) **and** (velocity is zero) **then** (control_output **is** zero)
9. **If** (position_error **is** positive) **and** (spring_length **is** tinyBell) **then** (control_output **is** zero_small_position)
10. **If** (position_error **is** BigPositive) **then** (control_output **is** zero_large_position)

Table 5.3 Fuzzy Rules for Controlling Plant

- **Tracking Behavior:** Tracking behavior has the conflicting goal of settling the plant output so that $|y| < 0.1$ after 15 seconds and interfering as little as possible with the stability robustness provided by the stability behavior. Two versions of the tracking behavior were developed:
  1. Tracking Behavior A which attempts to minimizes any detrimental interaction with the Stability Behavior

  Tracking Behavior A is implemented with Rule 7 to 9, from Table 5.3. It uses these three rules to implement a Sugeno Fuzzy PD controller which becomes active only when the spring length becomes small. This gating of the tracking behavior minimizes any detrimental affect the tracking behavior may have on the stability behavior. A smooth gating is obtained by using a smooth bell curve, the `tinyBell` membership function, instead of a simpler triangle membership function. The width of `tinyBell` at half-height is smaller than the equivalent `zero` membership function used by the spring process model, but unlike a triangle membership function decays smoothly to zero.
  The single output membership function used is:

  zero_small_position $$y_{sssc} = P_{small}x + d_{small}\dot{x} \qquad (5.26)$$

  2. Tracking Behavior B which sacrifices stability, but attempts to minimize the peak overshoot and settling time for y for the nominal plant with $k = 1$.

  Tracking Behavior B are designed so that when there are large position errors the tracking behavior is superimposed directly on the stability behavior, with the assumption that stability and vibration suppression are less important for large errors. This modification of Behavior A requires an additional two rules that are activated when y is large. These additional two rules are used to increase the PD controller gains when the output error is large, irrespective of whether the spring length is small, but with the detrimental effect of reducing the efficacy of the stability behavior. Additional output membership function used is:

  zero_large_position $$y_{sssc} = P_{large}x + d_{large}\dot{x} \qquad (5.27)$$

A Sugeno PD controller is implemented with output membership function which consist of a linear combination of the position and velocity of m_2. Table 5.4 gives the values of the constants used in the output equation for both Tracking Behavior A and B.

	p_{small}	d_{small}	p_{large}	d_{large}
Behavior A	-0.4	-2.1		
Behavior B	-0.25	-2.5	-0.75	-1.2

Table 5.4 Coefficient used for tracking

5.6 MATLAB/SIMULINK: Simulation

Simulations validate the disturbance estimation and accommodation properties of the integral action for the PI observer and the PI adaptive observer, PFI observer and Robust Fuzzy observer Simulations have been performed using Matlab7 and Simulink 6

5.6.1 PI Observer

Fig 5.8 depicts the SIMULINK model of a Luenberger observer with integral action for non-adaptive observer. The *Integral Action* block implements the integral action using an integrator and constant block.

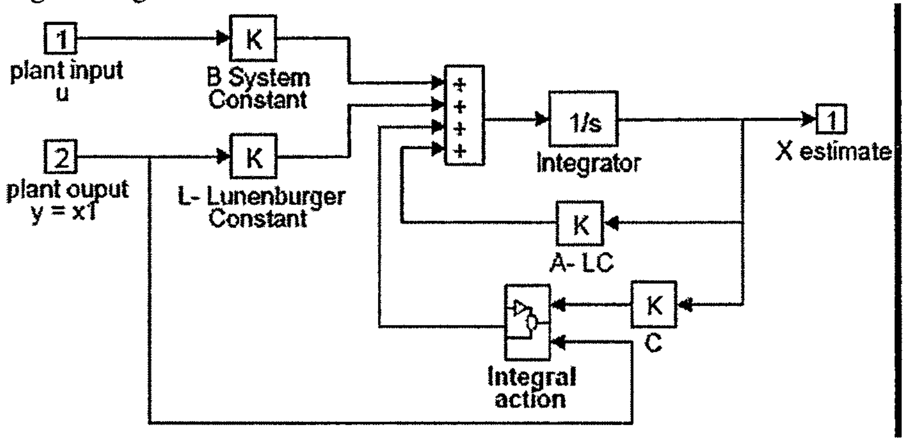


Fig 5.8: PI Observer block.

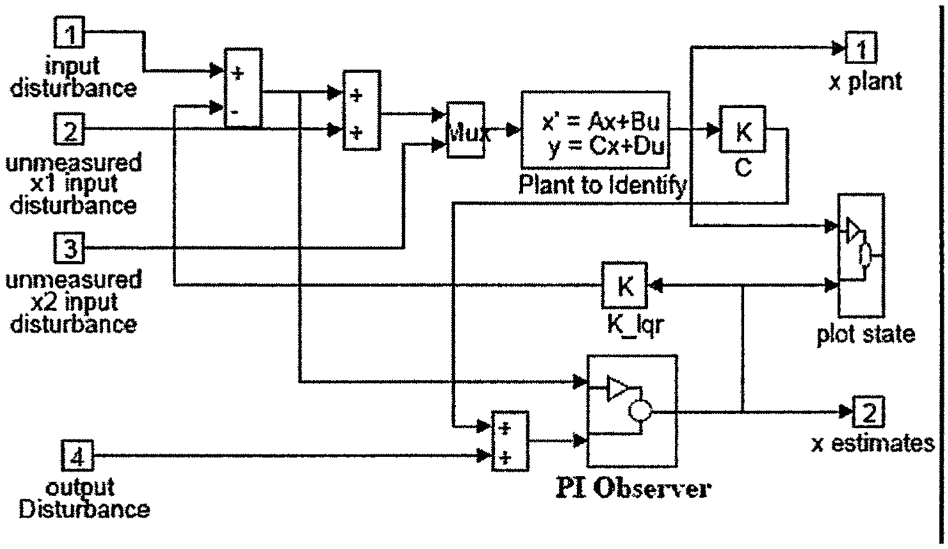


Fig 5.9: Observer-based regulator block

Fig 5.8 shows a detailed Simulink model of the observer based regulator. The observer output is multiplied by the constant block *K_lqr* to obtain the control output *u*.

PI observer is able to reject the step plant input disturbance by utilizing the integral offset *v*. Fig 5.9(a) reveals that after 3 seconds the values for *v* begin to settle to their final values. At this point the estimates of by for the PI observer converge, As shown in Fig 5.9(b) the cumulative estimation error for the hidden state x_2 reach a constant value for the PI observer while for the P adaptive observer the cumulative error is unbounded.

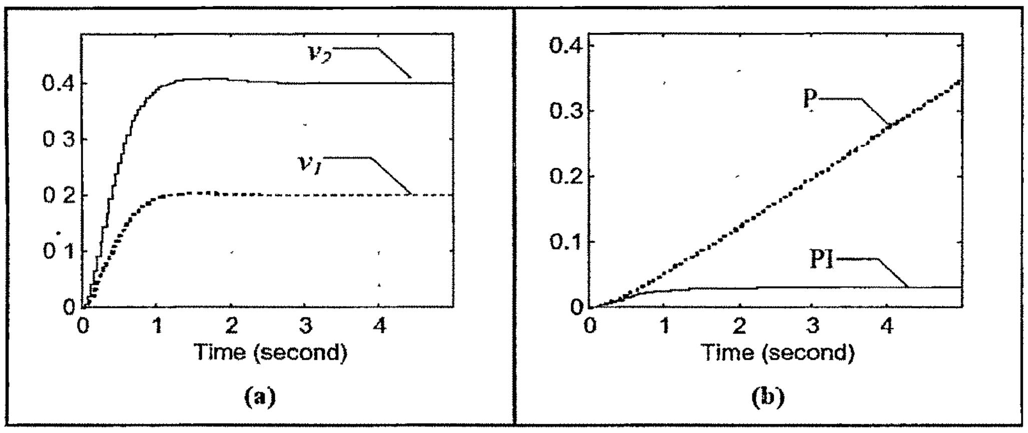


Fig 5.10: Response to a step input disturbance, din= 0.2,
Adaptive Observer Benchmark plant:
(a). Integral offset, *v*, from the PI
(b) the cumulative error estimating the hidden state *x2* for P and PI observer.

In the closed loop regulator configuration the PI observer was able to reject the step plant input disturbance faster than in the open loop case; after only several seconds the integral offset *v* shown in Fig 5.10(a) converges. During this initial settling time for *v* the PI regulator fails to track the performance of the full state feedback regulator. After settling the graph of cumulative error in Fig 5.10(b) shows that the PI regulator fully 33recovers the performance of the optimal full state feedback regulator, while the P regulator maintains a proportionally larger error and never achieves full loop recovery.

5.6.2 PI Adaptive Observer

The simulation for non-adaptive observers used the Simulink model of an adaptive observer shown in Fig 5.11

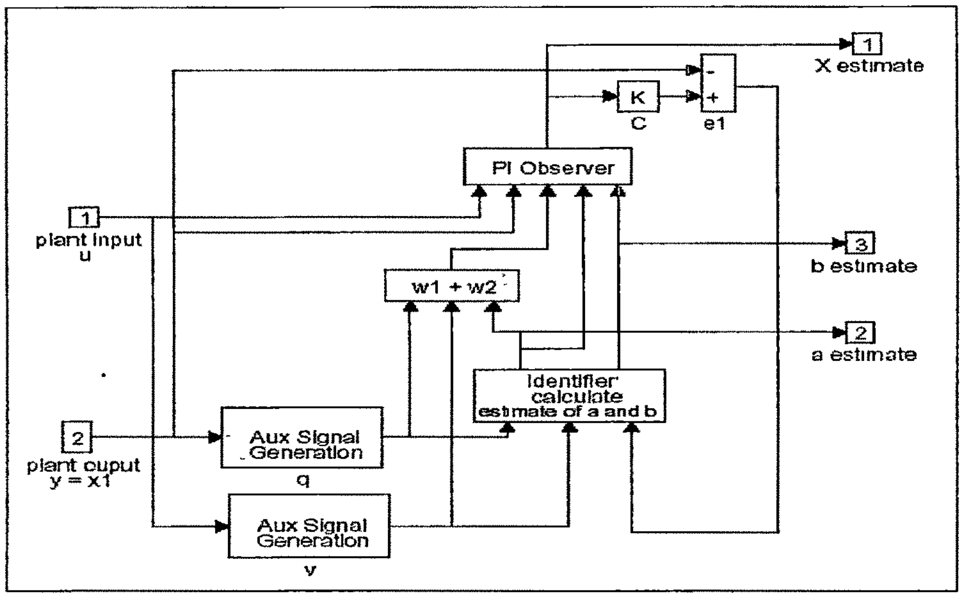


Fig 5.11 PI Adaptive Observer.

Integral action is added to the adaptive observer by adding integral and constant blocks to the *PI Observer* block of the model The disturbance rejection properties of the PI adaptive observer is evaluated with a step measurement input disturbance, d_{in} of magnitude 0.2, an integral proportionality onstant, K_I, of -4. The initial simulations used these adaptive gains from Kudva and Narendra (Kudva and Narendra 1973[3]) for both the P and PI adaptive observers,

$$\Gamma_1 = \begin{bmatrix} 140 & 0 \\ 0 & 75 \end{bmatrix}, \text{and} \quad \Gamma_2 = \begin{bmatrix} 5 & 0 \\ 0 & 7.8 \end{bmatrix} \tag{5.28}$$

but the estimates of a_2 and b_2 failed to converge for the PI adaptive observers. Convergence however was achieved for both benchmark scenarios after increasing the gains for the PI adaptive observer to

$$\Gamma_1^{PI} = \begin{bmatrix} 140 & 0 \\ 0 & 150 \end{bmatrix} \text{and} \quad \Gamma_2^{PI} = \begin{bmatrix} 5 & 0 \\ 0 & 15 \end{bmatrix}, \tag{5.29}$$

In the open loop configuration the PI adaptive observer was able to reject the step plant input disturbance by utilizing the integral offset v. Fig 5.12 (a) reveals that after 20 seconds the values for v begin to settle to their final values. At this point the estimates of both state and plant parameters for the PI adaptive observer begin to converge, as shown in Fig 5.12 (b).

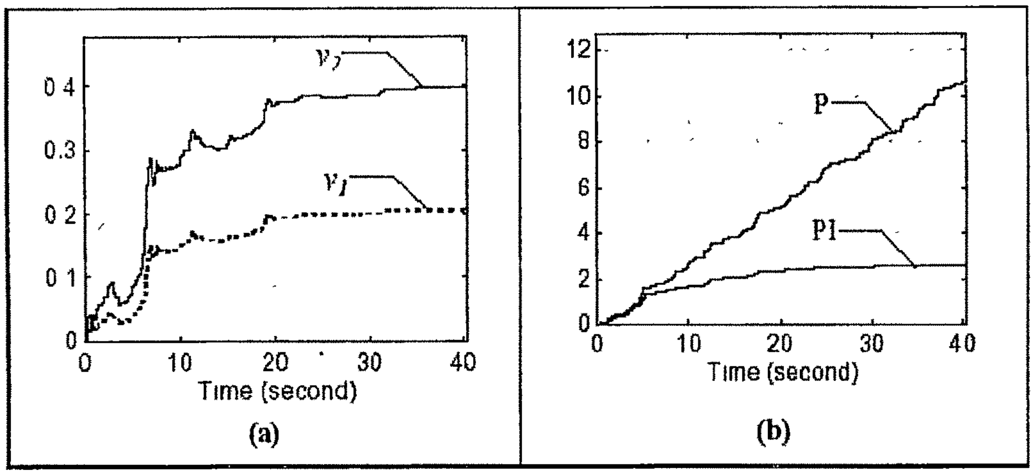


Fig 5.12: Response of *open-loop* P and PI adaptive observer to a step input disturbance, d_{in} = 0.2,

In the closed loop regulator configuration the PI adaptive observer was able to reject the step plant input disturbance faster than in the open loop case; after only several seconds the integral offset *v* shown in Fig 5.13(a) converges. During this initial settling time for *v* the PI regulator fails to track the performance of the full state feedback regulator. After settling the graph of cumulative error in Fig 5.13(b) shows that the PI regulator fully recovers the performance of the optimal full state feedback regulator, while the P regulator maintains a proportionally larger error and never achieves full loop recovery.

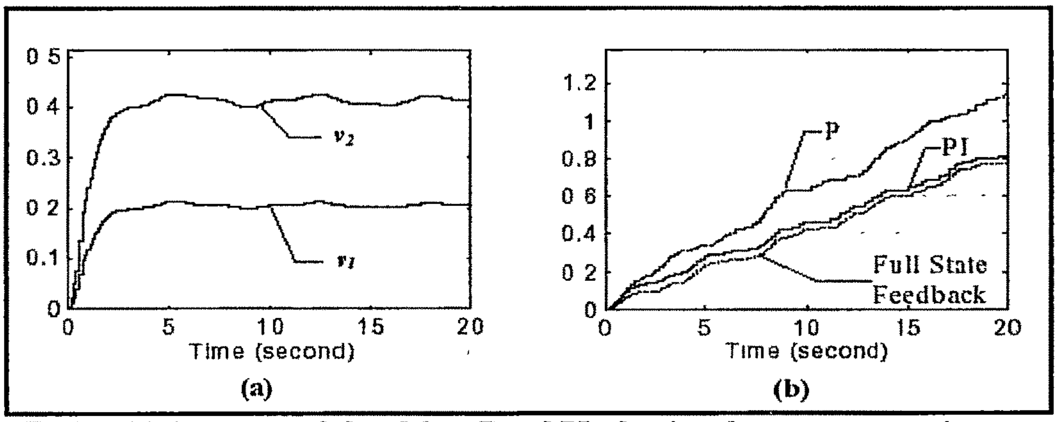


Fig 5.13(a): Response of closed-*loop* P and PI adaptive observer to a step input disturbance, d_{in} = 0.2

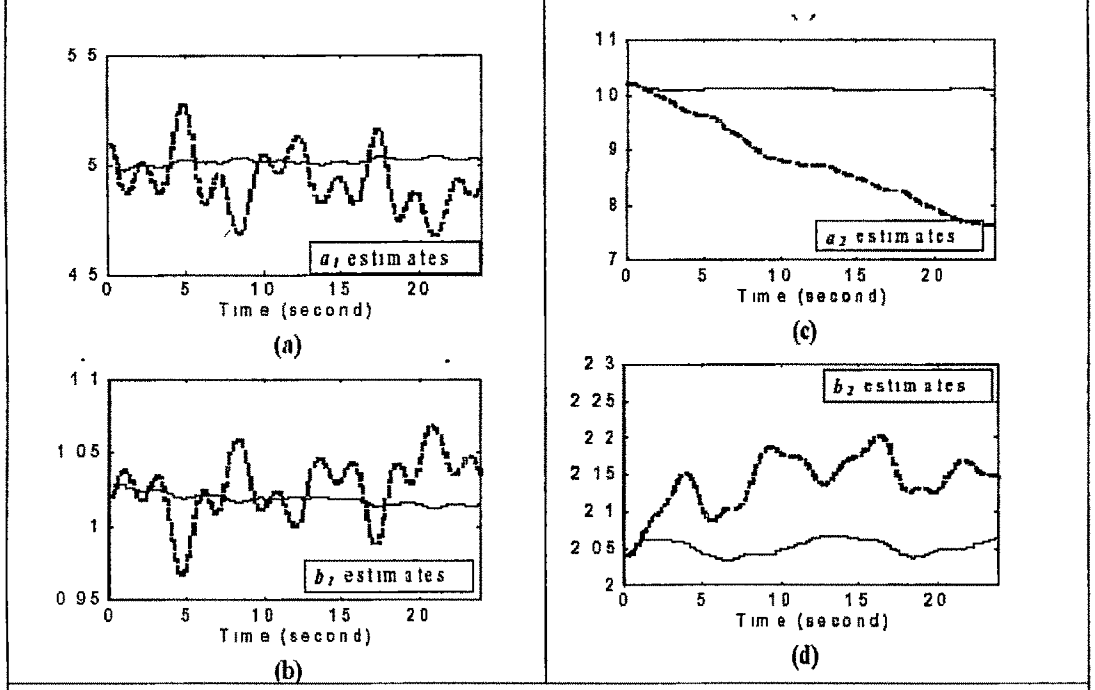


Fig 5.13(b): Estimates of a and b parameters from the closed-loop PI (solid line) and the P (dashed line) Adaptive Observer (triangular Disturbance)

- **Triangle Disturbance:** The input to the compensated plant is

  $u = 5sin(t) + 5sin(2.5t)$, the disturbance input is a triangle wave and the PI adaptive observer uses an integral gain of $K_I = -8$ for these simulations.

  Fig 5.13 shows that the estimates of the hidden state x_2 by the PI observer adaptive is significantly more accurate than for the P adaptive observer.

5.6.3 Fault Estimation and Accommodation

Simulations were performed for the plant using the four actuator faults described in the Theoretical Results chapter: of gain mismatch, deadzone, backlash and saturation. The parameters for each fault are:

Fault Type	Parameter Value
gain mismatch	$k_{atten} = 0.6$
deadzone	$k_{dz} = 0.3$
backlash	$k_{bl} = 1$
saturation	$k_{sat} = 1$

The same observer parameters, except where noted, are used in this section as in the Previous sections in this chapter.

- **Fault Estimation and Accommodation with the PI Observer**

An integral gain $K_I = -50$ for the PI observer is selected and an input $u = sin(0.5t)$

Fig 5.14(c) shows that the integral action of the PIO effectively estimates the disturbances caused by the actuator fault, leading to superior regulator performance, shown in Fig 5.14(a).

The graph of the cumulative output error, Fig 5.14(b), shows that the PIO-based regulator out performs both the PO-based regulator and LQR for all four faults. Only for the backlash fault does the performance of the LQR approach that of the PIO-based regulator.

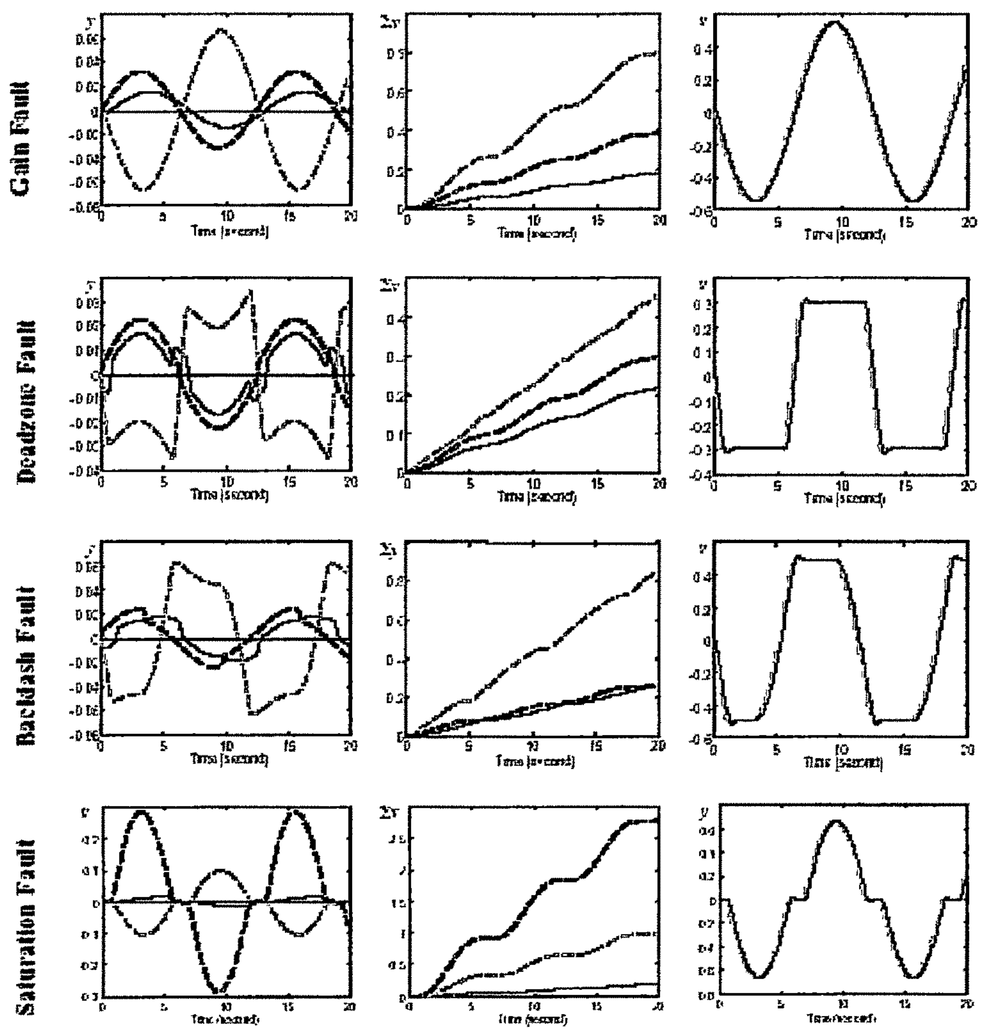


Fig 5.14. The performance of the LQR (dashed line), the P observer-based regulator (gray dashed line) and the PI observer-based regulator in the presence of four common actuator faults is compared in the first two columns:
(a) plant output (b) Cumulative output error. Column (c) Actuator disturbance

- **Estimation and Accommodation with the PI Adaptive Observer**

These simulations used an integral action gain for the PI adaptive observer of $K_I = -20$, an increase from the gain used by the previous simulations ($K_I = -4$). Fig 5.15 shows that the integral action of the PIAO precisely estimates the disturbances caused by the actuator fault, but with a slight degradation in estimation accuracy over the non-adaptive case. This degradation in performance can be attributed to inaccuracies in estimating plant parameters.

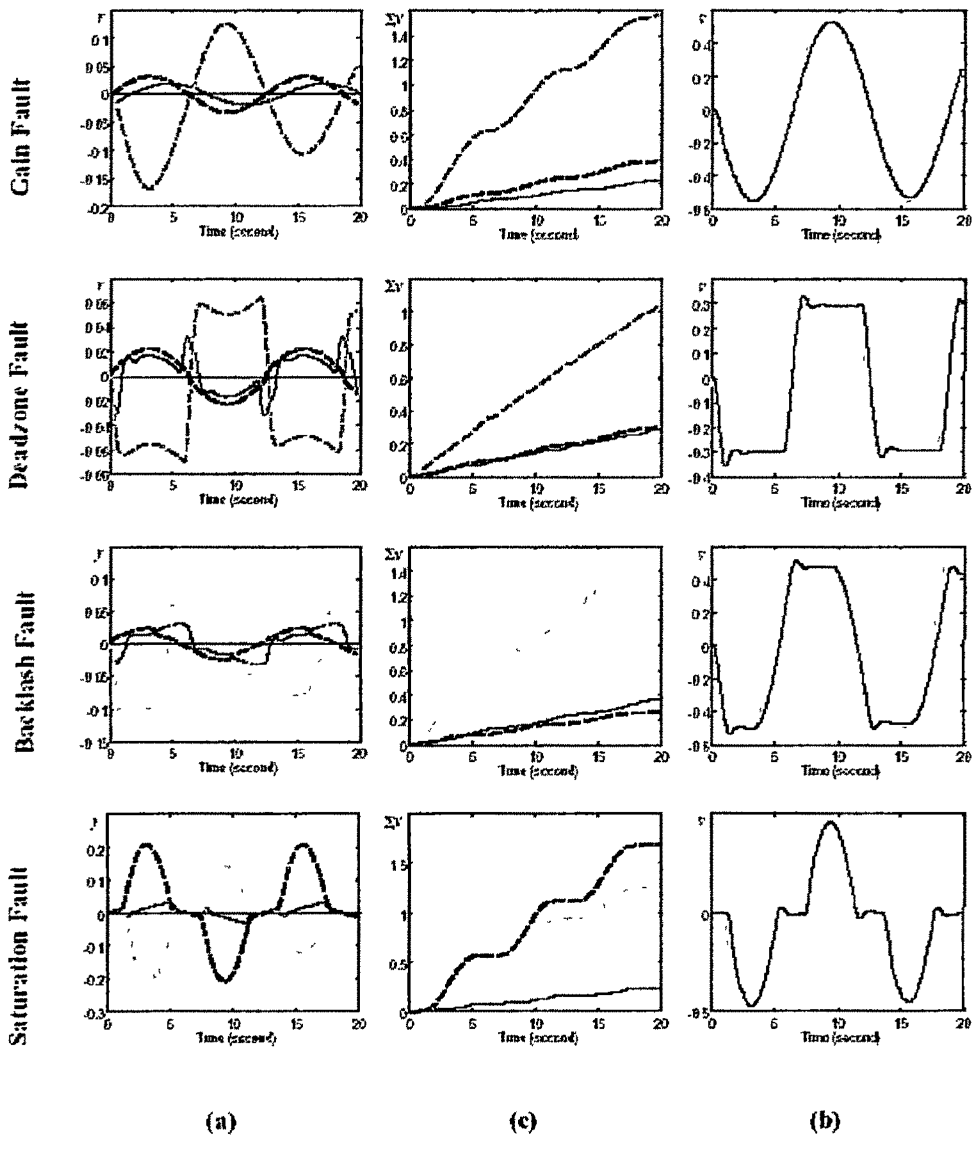


Fig 5.15: The performance of the LQR (dashed line), the P *adaptive* observer-based regulator (gray dashed line) with/without common actuator faults is compared in the first two columns;

(a) Plant output (b) Cumulative output error. (c) Actuator disturbance (gray line) and its estimate by the integral action (black line).

Fig 5.15(a) and Fig 5.15(c) shows that even with this reduced estimation accuracy, the PIAO-based regulator outperforms the PAO-based regulator. The graph of the cumulative output error, Fig 5.15(b) shows, however, that the PIAO-based regulator no longer out performs, in every case, the LQR using full state feedback.

The PIAO is superior for gain mismatch and saturation faults, slightly better for dead zone and inferior to LQR for the backlash fault. Noted that nonetheless even with this performance degradation with respect to LQR, only the PIAO-based regulator has the capability of estimating and characterizing the actuator fault.

5.6.4 Fading Integral Action

The PI observer-based regulator is not able to solve the impulse rejection Benchmark problem. Whereas the PI observer can compensate for the disturbance caused by a spring parameter mismatch, it can not also accommodate the unmeasured impulse input. The PFI variant of the PI observer can discount the effects of the impulses, and provides a solution for the Benchmark problem that is superior to that given by the P observer-based controller.

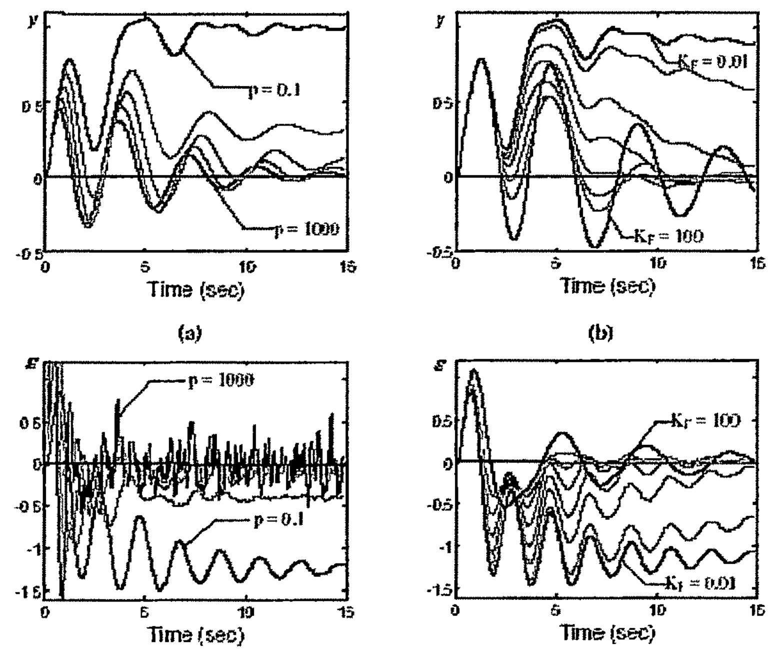


Fig 5.16: The effects of process noise and integral fading on the rejection of a unit impulse to m_2 for a perturbed plant with spring constant k =0.8.
(a) output for a PI Kalman Filter-based controller over a range of p
(b) output of the correspond PFI Kalman Filter-based controller (K_I = -20) for a range of K_F.
(c) and (d) give the respective error in estimating the state x_1.

A PI Kalman Filter-based controller was designed for the nominal plant and tested on a perturbed plant with a spring constant, $k = 0.8$. The efficacy of the fading term and high modeled process noise in rejecting transients was compared for a range of K_F and p. Fig 5.16(a) and (b) show that a P Kalman Filter-based regulator with high process noise and a PFI Kalman Filter-based regulator can achieve comparable output performance. However Fig 5.16(c) and (d) clearly shows that the estimation error for the PFI Kalman filter is much smaller than for the P Kalman Filter. Increasing the process noise enhances robustness of the filter, but has the detrimental effect of creating large estimation errors.

- **Estimating Plant Perturbations with PFI Kalman Filter**
  - ✓ The Benchmark problem requires that the integral term of the PFI Kalman filter estimate the perturbation Δx caused by changes in spring constants
  - ✓ simultaneously reject the effects of an impulse disturbance to m_2.

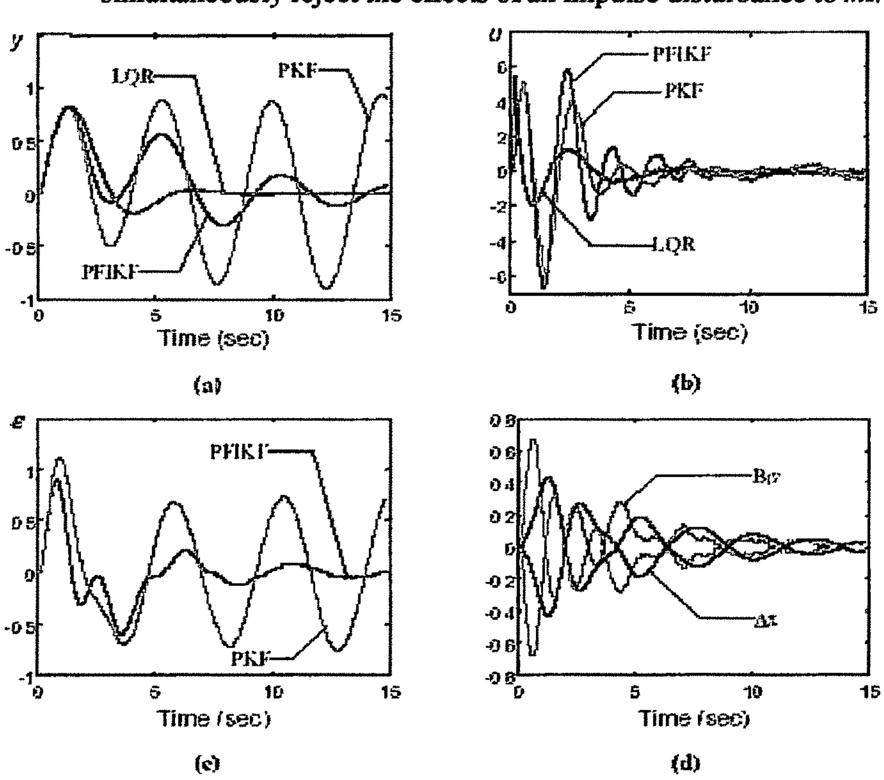


Fig 5.17: A comparison of the P , PFI Kalman Filter-based regulators and a LQR with full state feedback for a perturbed plant with spring constant k =0.8.

(a) Output *y*

(b) Regulator output *u*,

(c) Respective estimating error, ε, for state x_1 and

(d) the estimate B_{IV} of the perturbation term Δx from the PFI Kalman Filter.

Fig 5.17(d) shows that for a perturbed plant, with k = 0.8, the integral action $\mathbf{B}_{I}$, after seven seconds discounts the effects of the impulse disturbance and begins to effectively estimates the perturbation term Δx. The fading integral action allows the PFI Kalman filter-based regulator to achieve superior regulator output over the standard P Kalman filter-based regulator, shown in Fig 5.17(a), and improves the estimation accuracy of the filter, shown in Fig 5.17(c), while requiring only a nominal increase in compensator effort, Fig 5.17(b).

The robustness to plant perturbation of a Kalman filter-based regulator can be increased by using large modeled process noise. The tradeoff between robustness to plant perturbation and the level of actuator effort required to achieve this robustness is shown in Fig 5.18

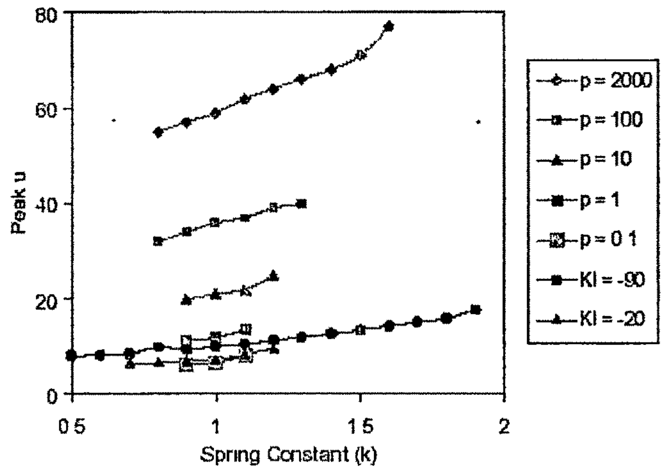


Fig 5.18: Peak regulator output versus plant spring constant for a P Kalman Filter -based controller and for a PFI Kalman Filter-based regulator.

5.6.5 Robust Fuzzy Control

The Simulink model of the of the robust control benchmark plant is shown in Fig 5.19

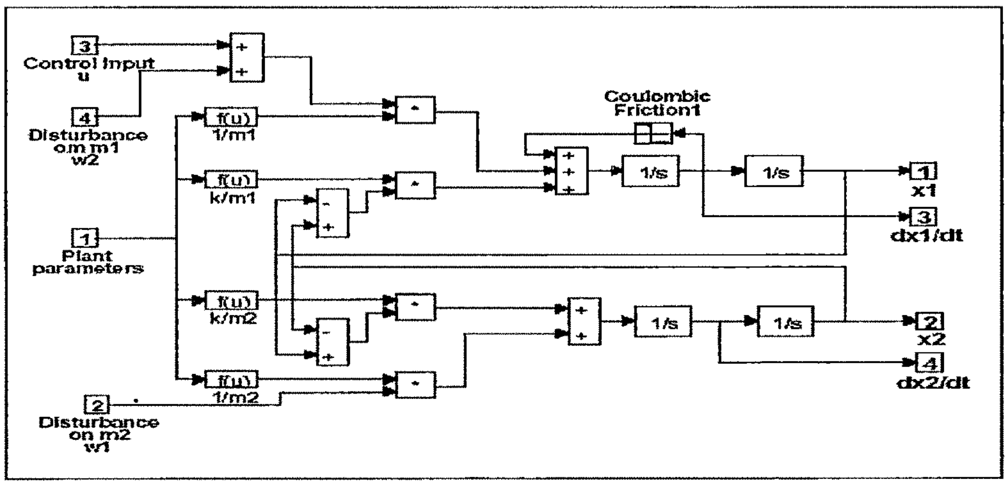


Fig 5.19 Plant used for Robust Control Benchmark.

The model was designed so that the three plant parameters, m_1, m_2 and **k**, can be adjusted dynamically. This model is used the Simulink Simulation shown in Fig 5.20.

A unit impulse was simulated by a pulse of amplitude of 4 units and duration of 0.25 seconds.

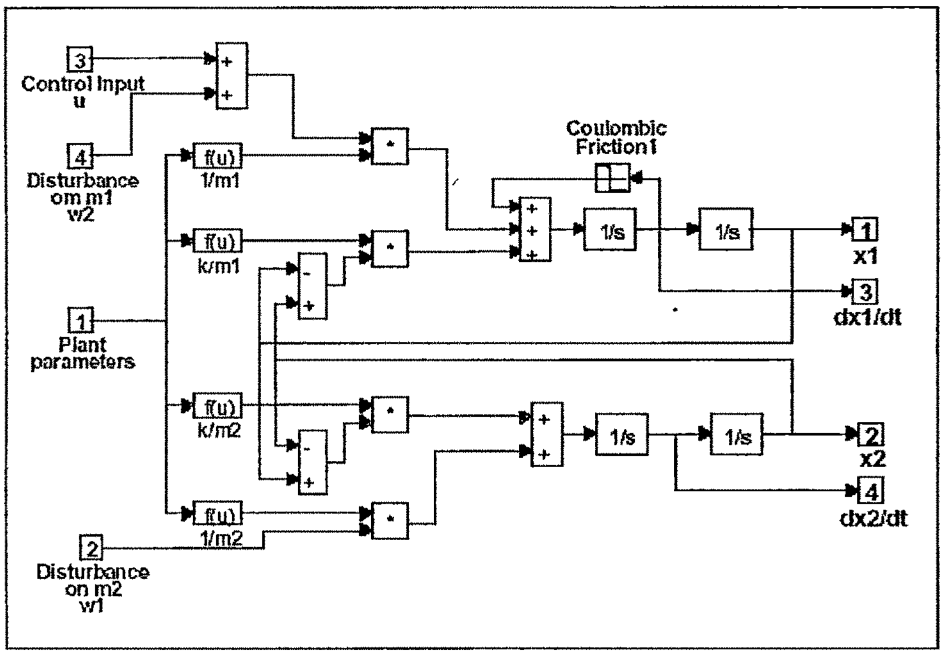


Fig 5.20: Robust Control Benchmark system with series compensation.

The performance of our QRC fuzzy controller was investigated using computer simulations for two scenarios: when complete state information was available and when a state observer was required to estimate the plant state. The QRC controller using Full State Feedback (FSFB) was benchmarked against the Linear Quadratic Regulator (LQR).

The LQR was selected because it shows the optimal robustness to plant perturbations of any linear controller (Anderson and Moore 1989). Equivalently, the QRC controller using output feedback and a robust Kalman Filter to estimate state was compared to the H^2 compensators: **based on stability Robustness and Performance Robustness.**

The compensators were evaluated for their ability to reject an impulse disturbance to m_2 using several metrics. $t_{|L|\leq 0.05}$, the time it takes L to settle within ±0.05 units of the final value, is used to measure the stability of a compensated plant. The effect of vibration suppression on stability robustness was evaluated by comparing the range of spring constants k for which L settles within ±0.05 units of the final value in less than 15 seconds to the stability radius of the compensated plant.

Tracking performance was measured by comparing the metric y_{max}, the maximum value of the plant output, and $t_{|y|\leq 0.1}$, the settling time of plant output y to within ±0.1 units of the final value. Since stability robustness and performance is enhanced by increased compensator output, $\Sigma_{u15\,0}$, the total actuator output for the first 15 seconds of the simulation was measured to insure that comparable levels of effort were used by all compensators.

All measurements, of both plant outputs and states, were corrupted with zero mean Gaussian noise with a period of 0.01 seconds and a standard deviation of 0.02. The LQR and LQG controllers use the same LQR gain matrix $\mathbf{K}_{LQR}$ = [0.8997 3.6077 0.5634 6.5076]'. The PFI Kalman filters used a model based on the nominal plant, with $m_1 = m_2 = k = 1$ and process and measurement noise covariance matrices

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \text{ and v=0.01}$$

which gives a Kalman gain **K** = [41.3051 6.4508 29.1938 20.8062]. Integral action, which compensates for spring constant perturbations, used a distribution matrix $\mathbf{B}_I$ = [0 0 1 1] ' and an integral gain K_I = -20.

Both the Kalman gain and the LQR gain were calculated with the standard MATLAB functions. All the fuzzy compensator configurations modeled the actuator as first order system with T = 40/(s+40). These simulation were performed with SIMULINK.

The robustness of the stability behavior used in the full fuzzy controller was characterized for both FSFB and output feedback. Figure 41 shows the response to a unit impulse disturbance to m_2 for a range of spring constants: k = 0.5 to 3.0 in steps of 0.5. Both compensator configurations show excellent vibration suppression properties.

The range of spring constants for which $|L| < 0.05$ after 15 seconds is $0.1 \leq k \leq 1000$ for FSFB and $0.4 \leq k \leq 2.3$ for the output feedback configuration. As expected, FSFB suppresses vibrations faster, over a wider range of plant perturbations and with less compensator effort, than output feedback.

The fuzzy stability behavior is so effective in suppressing vibrations, that even the output feedback configuration suppresses vibrations over a wider range of spring constants than a LQR using FSFB with equivalent actuator effort; the LQR regulator settles L only in the range $0.5 \leq k \leq 1.6$.

The addition of tracking behavior reduces the stability robustness of the compensator. However, if designed correctly a compensator with the less evasive Tracking Behavior A should be more robust than the compensator incorporating Tracking Behavior B. The effect of these two tracking behaviors is compared for the output feedback case in Fig 5.21

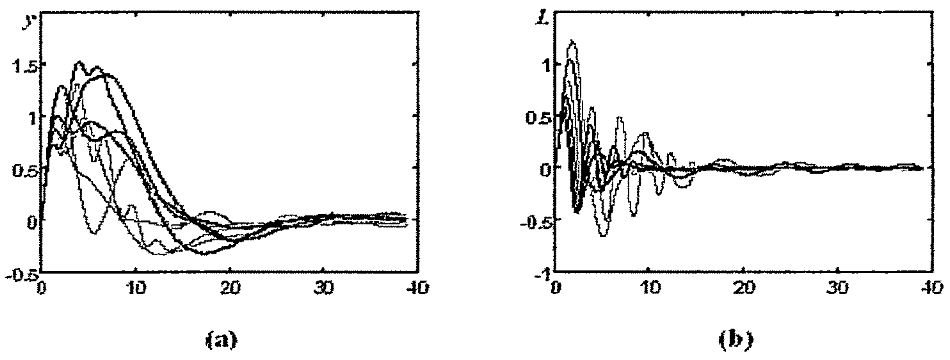


Fig 5.21Comparison of the two tracking behaviors using only output feedback,

While Behavior A shows higher peak responses and longer settling times, it is faster and more robust in settling L. This corresponds to a 50% larger stability radius for Behavior A than Behavior B. The Stability Behavior settles L within 15 seconds to $|L| \leq 0.05$ for $0.4 \leq k \leq 2.3$. The addition of Tracking Behavior A only marginally reduces the range of k to $0.7 \leq k \leq 2.3$, where as the performance oriented Tracking Behavior B reduces the range even further to $0.8 \leq k \leq 2.0$.

The stability robustness and tracking performance of the full QRC controller with Tracking Behavior B (**QRC B**) using FSFB was compared to LQR.

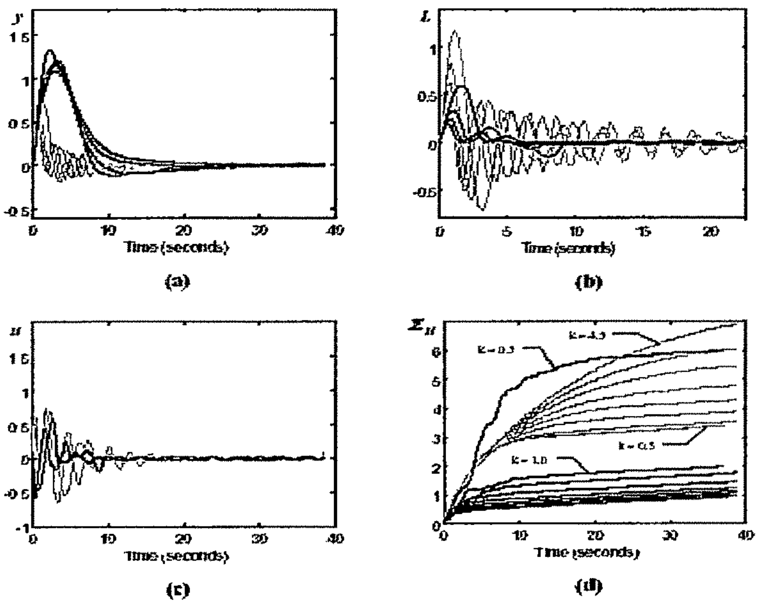


Fig 5.22: Performance comparison of two full state feedback controllers: the fuzzy controller (black and the LQR (gray) and Fuzzy Controller (black) after unit impulse disturbance to m_2 for k = 1, 2, 3, 4.

(a) Plant output y (b) Spring length L (c) Compensator output u

(d) Cumulative compensator output Σu (for k = 0.5 to 4.5 in steps of 0.5).

Fig 5.22 superimposes the output of these two controllers for a range of spring constants after a unit impulse disturbance to m_2. In order to insure a fair comparison, the LQR gain $\mathbf{K}_{LQR}$ was selected so that the peak LQR output was about the same as for the fuzzy controller; in fact, Fig 5.22(c) shows that in the range $1 \leq k \leq 4$ the LQR produces larger peak compensator output.

While the LQR over this range of k suppresses the effect of disturbance on the plant output y faster, shown in Fig 5.22(a), Fig 5.22(b) shows that the LQR compensated Plant has significantly larger oscillations in L. The superior vibration suppression properties of the QRC compensator contributes to the significantly larger stability margin of the QRC compensator: $0.2 \leq k \leq 1000$ for the **QRC A**, $0.4 \leq k \leq 1000$ for the **QRC B**, versus $1 \leq k \leq 4$ for the LQR. Fig 5.22(d) shows that the superior performance of the QRC compensators is achieved while having a total compensator effort that is significantly smaller for the QRC controller, except when $k = 0.5$.

5.6.6 Fuzzy Control with Output Feedback

The stability robustness and tracking performance of the full QRC controller with output feedback was evaluated using a PFI Kalman filter to estimate plant states. The QRC controller performance was compared to LQG using an identical PFI Kalman filter and to the two Marrison and Stengel Compensators: **Compl** given by

$$T = \frac{-79.3(s-0.8)(s+5.7)(s+0.11)}{(s^2+3.84s+10.24)(s^2+6.882s+13.69)(s+0.46)} \tag{5.30}$$

and **Comp3**, given by

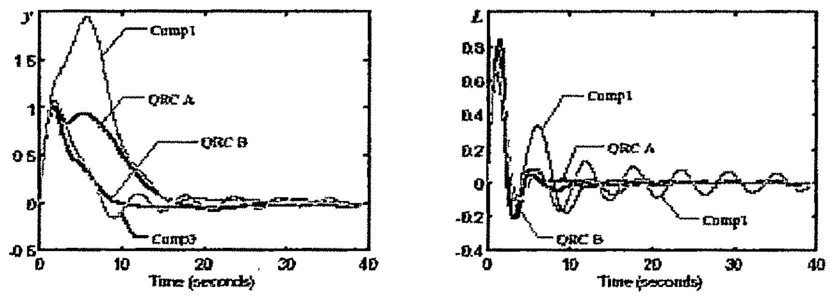
$$T = \frac{-8.2(s-4.7)(s+3.9)(s+0.24)}{(s^2+4.662s+13.69)(s^2+3.132s+7.29)(s+1.6)} \tag{5.31}$$

The stability robustness of the two QRC controllers proves to be far superior to that of the PFI Kalman filter-based LQG, but less robust than **Compl** and **Comp3**. The QRC A is stable for $0.1 \leq k \leq 3.0$ and QRC B is stable for $0.5 \leq k \leq 2.1$, while the LQG is for $0.8 \leq k \leq 1.4$ and **Compl** is stable for $0.5 \leq k \leq 5.5$. However **QRC B** has superior tracking performance than **Compl** in the range of spring constants specified by the Benchmark problem, $0.5 \leq k \leq 2.0$. As shown in Fig 5.23(a) **QRC B** has smaller peak output when perturbed by an impulse to m_2, and as shown Fig 5.23(d) has smaller overshoot when tracking a unit step. Additional, Fig 5.23(b) and (c) show that **QRC B** has superior tracking performance when m_1 and m_2 are perturbed.

When comparing metrics for the **QRC A**, **QRC B**, **Compl** and **Comp3** compensators at k = 0.5, 1.0 and 2.0, the QRC compensators show better vibration suppression behavior and comparable compensator output.

k	0.5	1.0	2.0	0.5	1.0	2.0
$t_{\|y\|\leq 0.1}$	∞^4	10.3	10.2	>40	8.0	19.4
$t_{\|L\|\leq 0.05}$		33.4	9.5	>40	8.8	14.9
y_{max}		0.87	1.17	1.38	1.02	1.33
$\Sigma_{u15.0}$		2.3	2.8	5.71	2.2	4.96

The table cells with the best results for the nominal plant, $k = 1$, are highlighted. QRC B has the best settling time for x_2, but takes longer to dampen vibrations in the flexible structure than QRC A. Comp3 has the smallest peak, but takes longer to settle than QRC B and is not stable in the range $0.5 \leq k \leq 2.0$ specified in the Benchmark problem.



(a) (b)

Fig 5.23: Responses to an impulse at m_2 for the nominal plant ($k = 1.0$) for Fuzzy controllers QRC A and QRC B and the linear controllers Comp1 and Comp3. (a) The fuzzy compensators have a lower overshoot in o/p (b) The fuzzy controllers dampen vibrations faster.

Fig 5.23 compares the nominal plant($m_1 = m_2 = k = 1$) output y and the spring length L response to a unit impulse disturbance to m_2.

The techniques developed give powerful tools to the field of disturbances rejection. Using these techniques disturbances caused either by unknown inputs, plant perturbations or actuator faults can be estimated and accommodated. The integral action approach, using variants of the PI observer, provide accurate disturbance estimate that allow for the accommodation and identification of disturbances given: (I) the injection points of the disturbances, (II) at least one independent state measurement for each disturbance and (III) disturbances with time constants longer than the time constant of the plant. Qualitative Robust Control (QRC), shows that controllers built using qualitative models and behaviors are as effective at accommodating disturbances as controllers built using H^2 and H^∞ techniques. The QRC controller requires only a small set of simple linguistic rule, with all parameters having simple interpretations, to achieve the same level performance as the H^2 and H^∞controllers built with complex mathematical constructs, and using obscure parameters.