Intelligent Hierarchical Thrust Vector Control for a Space Shuttle
We present the design of a thrust vector controller for a space shuttle vehicle with multiple engines. This controller will maintain vehicle trajectory and thrust vector while minimizing risk and damage to each engine and to the propulsion system as a whole by independently controlling the thrust magnitude and exhaust cone gimbal angles of each engine. A statistical model from reliability theory is used to estimate overall engine damage accumulation and failure risk. An intelligent control system framework which functionally decomposes the control task into a formal hierarchical structure composed of nominally independent task coordinators is used to design and analyze the control structure for the entire propulsion system. One subtask that appears repeatedly within the hierarchy is that of distributing the required control actions among multiple engines or actuators dynamically in response to changing damage, risk, and status information. We identify this as the "control relegation problem" and solve it using linear quadratic optimal control techniques with variable weighting matrices. The individual controllers, when integrated into the formal structure, achieve the control objectives while minimizing mission risk and component stress and maximizing operating efficiency.

Space Shuttle Propulsion System Requirements

As described in the recent report of the Task Force on Intelligent Control, intelligent control techniques are used to address the control of large interconnected dynamic systems with complex overlapping goals and constraints. We claim that such systems can be dealt with in an organized methodology that involves:

1. identification of conceptual and structural hierarchies,
2. modeling and analyzing subsystem dynamics and interactions,
3. performing a formal goal-task decomposition,
4. establishing specific, standard control problems corresponding to the task decomposition,
5. completing the controller design by solving the standard control problems (4) and other required tasks (3).

In this article we consider a space shuttle propulsion system as an illustrative example of the above design sequence. As in all design studies dealing with large, complex dynamic systems, the process involves the understanding of many subsystems, many goals, and many techniques.

Current studies in reusable rocket engine control design are driven by the need to improve the utility and flexibility of the vehicle-engine system, to improve its reliability and performance, and to increase vehicle availability through reduced maintenance [1]. Furthermore, experience with the space shuttle main engine (SSME) has demonstrated the need to utilize more sophisticated planning and optimization to address engine durability issues and component problems induced by large vibrations, rapid thermal transients, acoustics, and other mechanisms associated with high performance engines.

Meeting these requirements will require an intelligent vehicle control system. An initial proposal for such a control system, the Reusable Rocket Engine Intelligent Control System Framework Design (RREICS), has been developed by Rockwell International/Rocketdyne Division under contract to NASA Lewis Research Center [2], [3]. Within the RREICS framework, vehicle control is viewed as a three-level hierarchy consisting of a vehicle mission level, a propulsion system level, and an engine control level. Each level is composed of nominally independent coordinators which perform the various vehicle control and evaluation tasks. The overall objective of this program is to develop an integrated control-diagnostic system for a rocket propulsion system which achieves fault tolerant operation and maximizes engine life within mission constraints. The objective of the work reported here is to provide preliminary design and simulation testing of propulsion level coordinators to be used on future launch vehicles employing reusable propulsion systems.

Several practical issues are encountered when implementing the coordinators. The first is relegation: the distribution of the control action required to achieve some closed-loop objective among multiple actuators or engines in a dynamic and adaptable way. It represents an added degree of freedom which can be exploited in order to introduce additional system constraints and control objectives, for example, risk and damage minimization, while still maintaining desired output behavior. In this article, we formulate the control relegation task as an optimal control prob-
The motivation for the ICs framework proposed by Rockwell and NASA Lewis and adopted here is to treat the vehicle and engine cluster as a single system rather than as a collection of individual engines and to allow for accommodation of engine degradation and failure at all levels of the ICs hierarchy. It will provide enhanced engine and vehicle performance through:

1. increased engine and vehicle reliability and useful life,
2. adaptive online mission planning and optimization, and
3. improved diagnostics and maintenance scheduling.

The proposal emphasizes the integration of all engine-vehicle control and evaluation tasks into a unified hierarchy of mission, propulsion, and engine level coordinators. Thus engine or vehicle variables which have historically been ignored, monitored off-line, or controlled by simple or isolated controllers are to be coordinated to improve overall vehicle performance.

The ICs framework, shown in Fig. 1, consists of three hierarchical levels of control and coordination: a mission level, a propulsion system level, and an engine level. Each level is implemented by one or more nominally independent subsystems called coordinators which perform the various vehicle control and evaluation tasks. Command flow is downward, from the most general level (overall mission goals and requirements) to the most specific levels (actuator commands), and the implementation of commands is deferred to the lowest possible level. Status information, in the form of interpreted expressions of success, availability, and risk, flows upward. The processing of sensor data is performed at the lowest level and the flow of raw (uninterpreted) data between coordinators and levels is minimized. Each individual coordinator, in the form of a mixed dynamical and discrete event/rule based system, is responsible for maximizing the performance of and minimizing the risk and degradation to its associated subsystem. Local decisions are constrained to meet system requirements from higher levels. The complete system is thus a decentralized, hierarchical, multi-timescale system.

The mission level coordinator is responsible for overall mission planning and success. It evaluates in real time the expected risk versus the expected benefit for various mission plans and provides adaptive responses to vehicle, payload, and crew status. It receives information concerning overall vehicle thrust, propellant utilization, engine status, and risk factor estimates from its subsystems and issues mission requirements for the currently selected mission to each subsystem. These requirements include the vehicle thrust vector, overall mixture ratio, and propellant usage.

The propulsion level coordinator is responsible for maintaining overall vehicle thrust, mixture ratio, propellant usage, and propellant tank pressure under varying engine states and degradations in a manner that minimizes risk to the propulsion system as a whole. It receives information concerning the thrust, mixture ratio, gimbal status, estimated risk, and degradation (in the form of downthrust and cutoff factors and sensor and actuator failure reports) from each engine and issues gimbal angle and engine thrust magnitude commands to each engine in order to meet mission level requirements. It provides status and risk assessments to the mission level but will not change mission requirements.

The engine level coordinator implements propulsion level commands via low level control of the engine in the manner which minimizes stress to the engine. It is also responsible for closed-loop control of the gimbal actuators. It performs low level status and diagnostics, and is reconfigurable to support continued engine operation under degraded conditions.

A simplified block diagram of a two engine propulsion level is shown in Fig. 2. Three major coordinators are shown, the tank pressure coordinator which provides propellant recirculant to maintain proper propellant tank pressure, the fuel utilization coordinator which regulates mixture ratio and propellant usage, and the thrust vector coordinator which determines individual engine thrusts and gimbal angles, based on current engine conditions, to achieve a desired vehicle thrust vector.

![Fig. 1. ICS propulsion system block diagram.](image-url)
Reusable Rocket Engine Model

The baseline engine for this work is the space shuttle main engine (SSME), a reusable engine which evolved from a long line of expendable rocket engines used in past vehicle applications. The SSME system is highly complex, with four propellant turbo-pumps, three combustors, a number of multi-phase heat exchangers, and many plumbing interconnections for cooling, propellant, and venting flow and hydraulic control. In spite of this complexity, the current SSME system uses only two control measurements and two valves to control the thrust and mixture ratio of the engine. All other engine control functions are provided by open-loop scheduling.

The analytic model of the SSME is based on a nonlinear mass-heat-power flow balance model provided by Rocketdyne [4], with the addition of several valves and sensors to study the impact of additional engine data and control authority. The model includes valve dynamics, turbine dynamics, and gas and combustion properties (in lookup tables), but does not accurately represent the engine during its open-loop startup state. As a consequence, all design work and simulations start after $t = 5$ s and assume that the engine has reached a known operating point. For simulation purposes the model has been implemented as a MATRIXx System Build model. The open-loop engine model has 63 states, including 39 engine states which are mostly mass flow rates, pressures, turbine shaft torques and speeds, and liquid and gas densities. The other 24 states are associated with the dynamics (second order dynamics plus backlash and stiction) of the six engine valves. The model has the 33 sensor outputs currently available from the SSME plus several additional sensors to provide greater engine observability.

Fig. 3 is a propellant flow schematic of the SSME with valve and sensor positions indicated. The six control valves and ten of the most important measurements are listed in Table I. Five of the valves are present on the SSME, while the OPFV valve has been added to study the impact of additional control authority on the temperature distribution between the high pressure turbines [5]. One other important quantity, the engine mixture ratio (MR), can not be directly measured but can be estimated from other measurements. This estimate is available as EMR. Other sensor outputs are also available, including turbine shaft speeds and a number of flow rates.

The low level engine controller is based on the multivariable linear quadratic servo controller described in [5]. This controller allows independent control of combustion chamber pressure ($P_c$), mixture ratio (MR), and two turbopump temperatures, does not require gain scheduling, and allows easy redesign for fault accommodation modes. The closed-loop engine is continuously throttleable from the 65% to the 109% power level. It should be noted that the slew rate of the $P_c$ command is limited to 10% of the 100% rated power level per second. An input shaping filter is included in the controller to enforce this constraint.

Table I

<table>
<thead>
<tr>
<th>Sensor and Actuator Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPPO: Fuel Preburner Oxidizer Valve</td>
</tr>
<tr>
<td>OPOV: Oxidizer Preburner Oxidizer Valve</td>
</tr>
<tr>
<td>OPFV: Oxidizer Preburner Fuel Valve</td>
</tr>
<tr>
<td>MIV: Main Fuel Valve</td>
</tr>
<tr>
<td>MOV: Main Oxidizer Valve</td>
</tr>
<tr>
<td>CCV: Chamber Coolant Valve.</td>
</tr>
<tr>
<td>Pc: Pressure — Main Combustion Chamber</td>
</tr>
<tr>
<td>THd: Discharge Temp — High Pressure Fuel Turbine</td>
</tr>
<tr>
<td>Ttd: Discharge Temp — High Pressure Lox Turbine</td>
</tr>
<tr>
<td>Pfd: Discharge Pressure — Low Pressure Fuel Turbopump</td>
</tr>
<tr>
<td>Pfd: Discharge Pressure — Low Pressure Fuel Turbopump</td>
</tr>
<tr>
<td>Pfd: Discharge Pressure — High Pressure Fuel Turbopump</td>
</tr>
<tr>
<td>Qfm: Vol Fuel Flow — High Pressure Fuel Turbopump</td>
</tr>
<tr>
<td>P4: Pressure — Nozzle Cooling Jacket</td>
</tr>
<tr>
<td>P5: Pressure — Main Combustion Chamber Cooling Jacket</td>
</tr>
<tr>
<td>P9: Pressure — Fuel Supply to the Preburners</td>
</tr>
</tbody>
</table>

Fig. 2. Propulsion level coordination block diagram.

Fig. 3. Propellant flow diagram of SSME.
and the motion of the vehicle is given by

\[ \begin{align*}
\dot{x} &= \frac{1}{m} F_x, \\
\dot{y} &= \frac{1}{m} F_y - g, \\
\dot{\theta}_y &= \frac{1}{J} M_y
\end{align*} \]

where the rotational inertia of the vehicle \( J \) is given by

\[ J = \frac{m}{12} (l^2 + w^2) = 1.89 \times 10^9 \text{ lb} \]

and \( g \) is the acceleration of the gravity field. The resulting thrust vector has magnitude

\[ F_{\text{thrust}} = \sqrt{F_x^2 + F_y^2} \]

and angle (in the inertial frame)

\[ \theta_{\text{thrust}} = \arctan \frac{F_x}{F_y} \]

Each engine gimbal and exhaust cone is modeled as the two pole transfer function

\[ \frac{C(s)}{F_i} = \frac{2.4}{2.2s^2 + 2.4s + 2.4} \]

which represents the dynamics from the gimbal angle command signal to the actual exhaust cone angle. A complete model of engine the gimbal motor and cone dynamics was not available so this model was chosen to represent known acceleration and velocity specifications. Each gimbal is nominally offset from the vertical axis of the vehicle by 3° (0.05 rad) left or right, and each gimbal has a range of ±9° (0.157 rad) from its nominal position.

The relationship between the thrust generated by an engine and the state variables of the combustion chamber was not available. To compute values of engine thrust for simulation, the complex relationship between combustion chamber gas pressure \( P_c \) and engine thrust is modeled by linear interpolation of actual engine data at the 65% and 100% rated power levels giving

\[ F = 156.2 \ P_c \ (\text{lbf}). \]

**Vehicle Model**

A simplified planar vehicle model was developed, in order to facilitate controller design and simulation. The vehicle is specified as a two dimensional rectangular rigid body of roughly the same dimensions and mass as the space shuttle orbiter module. The vehicle model includes rotational as well as translational inertia and provides for motion in a gravity field. It is powered by two reusable liquid propellant rocket engines whose dynamics are given by the SSME model described above. The engine exhaust cones are gimbaled independently to provide thrust vector angular control.

A schematic representation of the vehicle, showing the orientation of the vehicle both in its body frame and in an inertial frame, is shown in Fig. 4. In this diagram and in the following equations, \( F_i \) represents the thrust magnitude produced by each engine and ranges from 305 000 lbf at minimum engine throttle (65% of full power) to 512 000 lbf at maximum engine throttle (109% of full power), and \( \theta_i \) represents the exhaust cone angle for each engine. The vehicle has length \( l = 121.5 \text{ ft} \), width \( w = 18 \text{ ft} \), and each engine is offset from the center of the vehicle by a distance \( a = 4 \text{ ft} \). The mass of the vehicle is \( m = 150,000 \text{ lb} \), which is 2/3 the fully fueled mass of the space shuttle orbiter.

In the body frame of the vehicle, the forces and moments generated by the engines (ignoring the Coriolis damping moment, which in this case is relatively small) are

\[ F_y = F_1 \cos \theta_1 + F_2 \cos \theta_2 \]  \hspace{1cm} \text{(1)}

\[ F_x = F_1 \sin \theta_1 + F_2 \sin \theta_2 \]  \hspace{1cm} \text{(2)}

\[ M_l = \alpha F_1 \cos \theta_1 - 0.5 F_1 \sin \theta_1 - \alpha F_2 \cos \theta_2 - 0.5 F_2 \sin \theta_2 \]  \hspace{1cm} \text{(3)}

In the inertial reference frame, offset from the body frame by an angle \( \theta_i \), the forces become

\[ F_x = F_1 \sin (\theta_1 + \theta_i) + F_2 \sin (\theta_2 + \theta_i) \]  \hspace{1cm} \text{(4)}

\[ F_y = F_1 \cos (\theta_1 + \theta_i) + F_2 \cos (\theta_2 + \theta_i) \]  \hspace{1cm} \text{(5)}

Estimation of Engine Status and Accumulated Damage

Some method of estimating engine risk and damage is needed in order to design controllers which attempt to minimize risk and damage. There are several possible approaches. An analytic model of the engine and vehicle could be developed which includes the effects of engine wear and damage [6]. This approach might produce the best estimates of risk and damage, but the development costs and online computational requirements of its implementation for such a large system may be too high. Another approach is the use of sensor-actuator diagnostics and expert systems to diagnose faults as they occur. This approach has been considered in several works [7], [8]. Instead, we propose a simple statistical model from reliability theory [9], [10] in which a standard probability distribution from studies of mechanical reliability is used to predict engine life and damage.
Let \( f(t) \) be the probability distribution function for engine failure over time. Let \( R(t) \) be the survivor (or reliability) function defined by

\[
R(t) = 1 - \int_0^t f(x) \, dx.
\]  

(12)

The survivor function gives the probability that a system is still functional at time \( t \). \( R(t) \) is an inverted cumulative distribution function, that is \( R(0) = 1 \) and \( R(\infty) = 0 \). There are a number of well known choices for \( f(t) \). We use the Weibull distribution, a standard two parameter reliability function which is often appropriate for electromechanical components and systems which, having survived an initial burn-in period, fail primarily due to fatigue, corrosion, wear-out, and uniform chance failures [11]. The survivor function for the Weibull distribution is

\[
R(t) = \exp\left\{-\left(\frac{t}{\alpha}\right)^{\beta}\right\}
\]  

(13)

where \( \alpha \) is the scale parameter and \( \beta \) is the shape parameter.

Let \( \lambda(t) \) be the hazard rate defined by

\[
\lambda(t) = \frac{1}{R(t)} \frac{dR(t)}{dt}.
\]  

(14)

The hazard rate measures the rate at which failures are occurring in the remaining functional population at a given time. It is known, in actuarial science, as the force of mortality because it measures, in some sense, the force or acceleration towards failure that a currently functioning item experiences at time \( t \). The hazard rate for the Weibull distribution is given by

\[
\lambda(t) = \frac{\beta \alpha^{\beta-1}}{t^{\beta-1}}.
\]  

(15)

An example of \( R(t) \) and \( \lambda(t) \) for \( \alpha = 0.5587 \) and \( \beta = 3.3 \) is shown in Fig. 5.

Damage occurring during SSME operation is principally due to exceeding the rated operating conditions and wear damage and excess stress due to transients and oscillations around the operating point. With this in mind we define a function which will measure the cumulative effects of these two actions by

\[
x(t) = \int_0^t \sum_i a_i S(x_i(t) - x_i^{\text{max}}) + b_i \left| T_c(x_i(t) - x_i(t)) \right| \, dt
\]  

(16)

where the \( x_i(t) \) are important engine outputs and states, \( x_i^{\text{max}} \) are the safety limits for the \( x_i \), and \( x_i^{\text{set}} \) are the setpoint or nominal values for the \( x_i \). The constants \( a_i \) weigh the damage occurring when an \( x_i \) exceeds its operating limits and the constants \( b_i \) weigh the damage caused by oscillations of \( x_i \) around its setpoint.

Considering \( x(t) \) as the abscissa of the reliability functions defined above, \( R(x(t)) \) is a measure of how much damage has already been done to the engine and \( \lambda(x(t)) \) is a measure of likelihood of engine failure in the near future.

### Formal Hierarchical Decomposition

Large, complex systems, such as the multi-engine space vehicle described above, often possess a hierarchical structure, and recognizing or imposing such a structure has several practical benefits. System design is easier since the design process can proceed independently on smaller, simpler subsystems with predefined interfaces which are then integrated into a complete design. System analysis is simplified because the behavior of the complete system is often determined more by the gross qualitative behavior and interactions of components than by the possibly complex quantitative behavior of individual elements. Finally, system modifications can be made at local level, with limited and predictable effects on overall system performance.

There are several possible approaches for decomposing a large system into a hierarchical structure. One approach is to perform a physical, or syntactic, decomposition in which the elements of the hierarchy are chosen to match physical components and the interconnections between components define the hierarchical relationships. This is the natural approach when a mathematical model of the system is desired. Another approach is to perform a functional, or semantic, decomposition, in which the components of the hierarchy are chosen on the basis of functions performed or goals achieved and an ordering of goals and subgoals defines the hierarchy. This is the natural approach when concepts from artificial intelligence research are to be employed. In practice, a compromise between these two extremes is usually required.

A framework for the formal specification of hierarchical structures was developed by Acar and Özgüner [12], and we will use it to illustrate the hierarchical decomposition of one of the SSME coordinators. This structure is "knowledge-rich"; it is mainly a functional decomposition. It also produces a specification similar to object oriented or modular program design.

---

Fig. 5. Weibull distribution function.
First, we consider the structural aspects of the hierarchical decomposition. Denote the entire system as a set $\Sigma_0$, which is also identified as Node: 0. This represents the most general description, the coarsest decomposition, of the system. At each level of the hierarchy, the set $\Sigma_i$ is decomposed into possibly overlapping subnodes $\Sigma_j \in \Sigma_i$, which at least cover $\Sigma_j$. In this way more detail and a finer decomposition of the system is achieved at each level. A directed graph is formed connecting subnodes and supermodes according to the subset relationships and satisfying structural coordinability axioms which guarantee that the entire system $\Sigma_0$ is the top node and that the tip level contains the maximal number of disjoint nodes. This process is illustrated in Fig. 6.

There are several consequences of this procedure. The entire structure is defined at the tip level, and indeed at every level of the hierarchy. Also, the description of the system becomes increasingly detailed at deeper levels of the hierarchy, and in fact the tip level is guaranteed to have the finest decomposition.

Next, we consider the functional hierarchy generated in accordance with the above structure. At each node, a Goal is formulated which asserts the end result to be obtained at that node. This goal is satisfied by the successful completion of one of (possibly) several Tasks, which are elementary job descriptions for the node. A set of Procedures exists which accomplish tasks. A procedure may either be Current, in which case it is applicable at this node, or it may be a Subprocedure, in which case it is carried out by subnodes at a lower level of the hierarchy. The selection of procedures and tasks is controlled by Constraints in the form of exemptions or restrictions. These constraints may be system dependent (e.g., mechanical), environmental (e.g., measurement related), or goal enforced. In addition, each node has a list of the Measurements and Resources available locally or remotely which are required to satisfy goals. Note that the resulting structure is not restricted to a tree topology. The presence of subprocedures and nonlocal measurements and resources allows richer topologies to be defined.

The generic formal description of a node is shown in Table II. Two numeric quantities appear in this description. The stiffness constants specify the strictness of constraints and goals and the cost constants indicate the cost of obtaining measurements or using resources. Notice that goals and tasks are specified in a similar syntax which aids in the matching of goals with tasks.

Finally, we define how a node acts to fulfill its goals. The node receives a goal, coupled with constraints and resources, from one of its supernodes. The goal is decomposed into a set of tasks, and a list of the (possibly) many sets of procedures accomplishing the required tasks is compiled. From this list one set of procedures is selected which minimizes a cost criteria while satisfying constraint, measurement, and resource requirements. Current procedures in the selected set are executed when their constraints are satisfied. Subprocedures are used to form goals for subnodes.

The utility of this decomposition is that it forces one to identify and rigorously specify the hierarchical structure of the control and system requirements. It also provides a systematic framework for the implementation of the required controllers.

**Thrust Vector Coordinator**

Having specified a dynamic model of the rocket engine, a model for a simple planar craft, a measure of engine damage and failure risk, and a formal structure for the hierarchical decomposition of a system, we are now ready to apply these specifications within the RREICS intelligent vehicle control framework to design a thrust vector coordinator.

The thrust vector coordinator consists of three distinct subsystems: the first corrects the angle of the thrust vector utilizing the engine exhaust gimbals, the second corrects the magnitude of the thrust vector by distributing (or redistributing) thrust magnitude commands to each engine, and the third evaluates engine risk status and acts to minimize engine damage and protect the engines and the vehicle from catastrophic engine failure. These subsystems are designed to operate independently.

**Hierarchical Decomposition of the Thrust Vector Coordinator**

Using the formal hierarchical decomposition of Acar and Özyüner, the thrust vector coordinator is decomposed in the following manner. At the highest level of the hierarchy consid-

![Fig. 6. Partial decomposition into a structural hierarchy.](image-url)
ered in this paper, the propulsion system level (Level:2), the goal received from the mission level is to regulate the overall craft thrust vector around a commanded thrust magnitude and direction. A single task is generated to fulfill this goal, constrained by engine thrust limits and lockup status. This task is decomposed into two subgoals, thrust angle coordination and thrust magnitude coordination, and these goals are passed to the next lower level in the hierarchy (Level:2b).

The formal specification of the thrust vector coordinator (TVC) appears below. We have not completely specified the lowest level (Level:3b) because the goals and tasks at this level will be directly implemented as mathematical control laws or algorithmic procedures. We have also only specified the thrust vector controller. The thrust magnitude controller is similar.

**Optimal Control Relegation**

In many complex multi-input multi-output systems the required closed-loop output behavior can be achieved in a multitude of ways. This multiplicity is an added degree of freedom which can be exploited in order to introduce additional system constraints and control objectives, for example risk and damage minimization, while still maintaining desired output behavior. The control relegation task appears repeatedly in control task decompositions for the space shuttle intelligent control system. In particular, we must use control relegation in the optimal thrust angle regulation task of Node: Thr_Ang_Coord at Level: 2b.

In this article we formulate the control relegation task as an optimal control problem. Linear quadratic optimal control techniques are employed to design controllers which can perform both output regulation and control action relegation in an optimal manner.

The first step is to develop a simplified plant model. The plant is in general nonlinear, so it is linearized about an operating point. Plant states and exogenous inputs which are not directly manipulated or sensed by the controller are assumed to be fixed or slowly time varying and are taken as constants. If this assumption is not correct, or if the operating range of the plant makes linearization around a single operating point inadequate, then this information may be incorporated into the simplified plant model as time varying entries in the plant matrices. The resulting low order linear model is written in state space form as

\[ \dot{z} = Az + Bu + Ev \tag{18} \]

with control inputs \( u \), exogenous inputs or noise \( v \), and where the \( A, B, \) and \( E \) matrices may have slowly time varying entries which, we assume, can be calculated or measured in real time.

Next, a gain scheduling LQR optimal controller is designed using a cost criterion of the form

\[ J = \int_{0}^{T} x^TQx + u^TRu \, dt \]. \tag{19} \]

The weighting matrices, which may also be time varying, are used both to prescribe the state and output behavior of the plant and to introduce external system objectives and constraints on the behavior of the plant states or control inputs. For example, the diagonal entries of the \( R \) matrix may be manipulated to cause relegation of the control action among multiple inputs.

We know that, for the simplified system, the cost criterion above is minimized when the plant inputs \( u \) are chosen as a function of the plant states specified in terms of the solution of an appropriate algebraic Riccati equation (ARE). If the \( Q \) and \( R \) matrices are diagonal and the simplified plant model is of low order it is possible to solve the Riccati equation explicitly. In this case an analytic nonlinear gain scheduling control can be implemented. If the Riccati equation cannot be solved explicitly, it must be solved numerically in real time as the plant operating point and the weight matrices change. The result is a gain scheduling LQR optimal controller.

Now that we have formally specified the control structure, we can implement the required controllers.
Regulating Vehicle Angular Orientation

Now we can apply the optimal control relegation methodology described in the previous section to the specific problem at hand — namely the task of regulating thrust vector direction. In order to change the thrust vector direction without inducing a rotational moment in the vehicle it is necessary keep the principal (Y) axis of the vehicle pointed in the direction of the desired thrust vector. Therefore, instead of directly controlling the thrust vector direction \( \theta_T \) we will control the vehicle angle \( \theta_i \) with respect to the inertial frame.

The engine gimbal angle commands \( \theta_i \), \( i = 1, 2 \) are defined as

\[
\theta_i = \theta_i = 0.05
\]  

in order to include the nominal exhaust cone angle offsets. Assuming constant engine thrust \( F_i \), (6) can be linearized around \( \theta_i = 0 \) to obtain

\[
\begin{bmatrix}
\dot{\theta}_i \\
\dot{\theta}_j
\end{bmatrix} =
A \begin{bmatrix} \theta_i \\ \theta_j \end{bmatrix} + B \begin{bmatrix} u \\ e \end{bmatrix} + F_V
\]

where

\[
b_i = -(0.5\epsilon + 0.05\alpha)F_i, \quad e = \alpha - 0.025J.
\]  

The coefficients \( b_i \) and \( e \) can be trivially calculated in real time, so we can design a gain scheduling LQR optimal controller which will generate gimbal angle commands \( \theta_i \).

As shown in [13], the resulting controller has the form

\[
\theta_i = \frac{b_i}{r_j} \left[ n_{12} (\theta_i - \hat{\theta}_j) + n_{22} \hat{\theta}_j + m_{21} (F_1 - F_2) \right]
\]  

where \( \hat{\theta}_j \) is the commanded vehicle angle. The coefficients can be determined analytically from the weights in the quadratic regulator problem.

Generating Vehicle Angle Commands from Thrust Vector Direction Commands

As it happens, it is impossible to keep the vehicle thrust angle parallel to the principal axis of the vehicle without inducing a rotational moment in the vehicle except when both engines are producing the same thrust magnitude. However, it is possible to compute a minimal vehicle angle offset so that the desired vehicle thrust vector is obtained.

Consider again the equations for the vehicle moment

\[
M = \alpha F_1 \cos \theta_1 - 0.51 F_1 \sin \theta_1 - \alpha F_2 \cos \theta_2 - 0.51 F_2 \sin \theta_2
\]  

and the thrust vector directional offset (from the Y (vertical) axis in the body frame)

\[
\theta_T = \arctan \left( \frac{F_X}{F_Y} \right) = \arctan \left( \frac{F_1 \sin \theta_1 + F_2 \sin \theta_2}{F_1 \cos \theta_1 + F_2 \cos \theta_2} \right)
\]

We can minimize \( \theta_T \) constrained by \( M = 0 \) using the following procedure:

1. Solve \( M = 0 \) for \( \theta_T = f(F_1, F_2, \theta_i) \).
2. Substitute the expression for \( \theta_T \) into \( F_X \) and \( F_Y \).
3. Minimize \( \theta_T(t_i) \Rightarrow \text{Minimize } F_X(\theta_i) \) numerically (difficult).

Now \( F_X \) and \( F_Y \) are dominated by the \( O(0) \) terms of their Taylor's series around \( \theta_i = 0 \) when constrained by \( M = 0 \). Hence the \( O(0) \) terms

\[
\hat{F}_X = \frac{F_2 S}{T(1 + \frac{S^2}{4T^2})}, \quad \hat{F}_Y = \frac{F_1 + F_2 - \frac{S^2}{4T^2}}{1 + \frac{S^2}{4T^2}}
\]

are accurate estimates for \( F_X \) and \( F_Y \), and thus

\[
\hat{\theta}_T = \arctan \left( \frac{\hat{F}_X}{\hat{F}_Y} \right)
\]

is an accurate estimate of the offset between the vehicle orientation angle and the thrust vector angle required to avoid generating a rotational moment.

The vehicle angle command is generated by

\[
\hat{\theta}_i = -\theta_i + \hat{\theta}_T + \int_{\tau_i}^{\tau_f} (\theta_i(t) - \hat{\theta}_i) \, dt
\]

where \( \hat{\theta}_i \) is the commanded thrust angle. The integrator provides local error correction and is disabled and reset except during a state of small angular errors.

Regulating Vehicle Thrust Magnitude

The actions taken by the thrust vector coordinator to correct thrust magnitude errors should be based on the downthrust factors and operating limits of each engine.

The downthrust factor is based on the expected lifetime benefit of reducing the thrust of a particular engine. It may be specified as some function of engine damage or degradation estimates in combination with information about key engine states and their specified limits. For our thrust magnitude controller, the downthrust factor \( D_i, 0.0 \leq D_i \leq 1.0 \), for each engine is an externally specified quantity such that \( D_i = 0.0 \) indicates that there is no need to downthrust the engine and \( D_i = 1.0 \) indicates that an engine downthrust is imperative. We take \( D_i \) as

\[
D_i = \max \left\{ \frac{\lambda_i(\theta_i)}{\lambda(1.0)}, 1.0 \right\}
\]  

June 1994
the limited hazard rate from the engine damage estimator defined in (15) and shown in Fig. 5.

Two engine utilization factors are also defined as

\[
R_i^U = \frac{3275 - PC_i}{1325}
\]

(31)

\[
R_i^D = \frac{PC_i - 1950}{1325}
\]

(32)

\[ R_i^U \]

measures the remaining upthrust available from an engine such that \( R_i^U = 1.0 \) at the minimum (65%) power level and \( R_i^U = 0.0 \) at the maximum (109%) power level. In a similar manner \( R_i^D \) measures the remaining downthrust available from an engine.

Since we are concerned only with thrust magnitude errors, we disregard any angular errors and take

\[
\Delta F = F_{\text{com}} - F_{\text{mag}}.
\]

(33)

Then, assuming the difference between commanded and actual engine power output is small, engine magnitude commands are generated by a controller of the form

\[
\Delta F_i = \frac{R_i^U \Delta F - \beta (2D_i - \sum_{k=1}^{2} D_i)}{\sum_{k=1}^{2} R_i}
\]

(34)

\[
F_{\text{com}}(t) = F_{\text{com}}(t - \tau) + \Delta F_i.
\]

(35)

If there is a substantial difference between the actual engine thrust output and the command from the above controller, we set \( \Delta F_i = 0 \), effectively locking the thrust magnitude controller until engine outputs reach the current commanded values.

This structure provides engine thrust magnitude commands which produce the specified vehicle thrust magnitude while distributing the thrust command among the engines based on downthrust factors and thus minimizing risk.

Thrust Vector Coordinator Simulation Tests

The vehicle, engine, and gimbal models were simulated under the control of the thrust vector coordinator specified above. Controller parameter values for thrust vector direction correction were chosen as shown in [13] to provide a good tradeoff between response time and angular overshoot. Parameters for the thrust magnitude correction were chosen [13] to provide reasonable engine response time without exceeding engine specifications. The simulation result for a step upthrust from 65% to 90% of full power and a step angle correction of 10° (0.175 rad) is shown in Fig. 7. In order to simulate engine degradation, a sinusoidal noise signal was introduced into the second engine during the period 10 < t < 30.

Benefits of Intelligent Propulsion System Control

One of the major attributes of so-called intelligent control systems is their utility in addressing the control of large coupled dynamical systems with multiple, complex, interrelated operating requirements. The understanding of goals, the creation of hierarchies, the identification of tasks, all with due consideration of fault tolerance, resource allocation, and sensor data fusion, are at the core of intelligent control. However, another important portion of intelligent control is the identification of control subproblems which can be solved using standard control techniques and their integration into the overall intelligent control scheme.

This article has presented the problem of intelligent control for a space shuttle vehicle powered by reusable rocket engines. An intelligent control framework which decomposes the vehicle and its mission into a hierarchical system was discussed. Possible solutions to several implementation issues, including control action relegation, damage estimation, and risk management, were discussed. Finally, all of these ideas were illustrated by the design and simulation of a space shuttle vehicle thrust vector coordinator.

Although the work reported here is ongoing, several conclusions can be drawn. The main conclusion is that the methodologies collected here under the title of "intelligent control" can be applied to problems that involve large coupled dynamics and interrelated goals. We have seen that hierarchical structures in systems and dynamics can be matched to task hierarchies. We have also demonstrated that a formal goal-task decompositions provides both an additional insight and structure to the control
problem and a suitable framework for controller implementation. Finally, we have illustrated these ideas on a nontrivial problem.

References


Keith Redmill received the B.S.E. degree in electrical engineering and the B.A. degree in mathematics from Duke University in 1989, and the M.S. degree in electrical engineering from The Ohio State University in 1991, where he continues his doctoral studies with the support of an Ohio Aerospace Institute Doctoral Fellowship. He is interested in intelligent control, large scale systems, dynamical systems theory, computational mathematics, and advanced computing technologies. He is a member of the IEEE and SIAM.

Umit Özgüner received the Ph.D. degree from the University of Illinois in 1975. He has held research and teaching positions at L.B.M. T.J. Watson Research Center, University of Toronto, and Istanbul Technical University. He has been with the Ohio State University since 1981 where he is presently Professor of Electrical Engineering and Director of the IVHS-OSU Center. His areas of research interest are intelligent control of large scale decentralized systems, flexible structures, automotive control, and intelligent vehicle highway systems. He is the Chair of the IEEE Control Systems Society Technical Committee on Intelligent Control and Chair of the Ohio Aerospace Institute Focus Area on Dynamic Systems and Control. He is General Chair of the 1994 IEEE International Symposium on Intelligent Control.

Jeffrey Musgrave received the B.S. and M.S. degrees in systems engineering from Case Western Reserve University in 1987 and 1989, respectively. He joined the Advanced Control Technology Branch at the NASA Lewis Research Center as a Research Engineer in 1989. He has recently completed research work as a Project Engineer for intelligent control of rocket engines. He is currently a Project Engineer for the development of electronic controls for single-engine general aviation aircraft and conducts research in the areas of fuzzy control, robust control, and nonlinear control. He is a member of the IEEE and Tau Beta Pi.

Walter Merrill specializes in aeronautic and aerospace propulsion control technology. He has been employed at the NASA Lewis Research Center since 1975 and is currently the Chief of the Advanced Controls Branch. His research interests include multivariable controls methodology, dynamic systems identification, fault tolerant control through analytical redundancy, diagnostics, and intelligent control systems. His current activities include the development of strategies to deploy 1) intelligent control systems for improved durability of reusable rocket engines and 2) integrated flight propulsion controls for aircraft. He received the B.S.E.E. degree in 1972 from General Motors Institute, the M.S.E.E. degree in 1971 from Auburn University, and the Ph.D. degree in engineering science in 1975 from the University of Toledo. He has authored over 70 technical articles, papers, and reports and is a senior member of the IEEE and an Associate Fellow of the AIAA.