5-2022

Nuclear Thermal Rocket Engine Control Autonomy Via Embedded Decision

David Sikorski
University of Tennessee, Knoxville, dsikorsk@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by David Sikorski entitled "Nuclear Thermal Rocket Engine Control Autonomy Via Embedded Decision." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Nuclear Engineering.

Richard T. Wood, Major Professor

We have read this dissertation and recommend its acceptance:

Richard T. Wood, Jamie Coble, Seddik M. Djouadi, Dianne Bull-Ezell

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Nuclear Thermal Rocket Engine Control Autonomy Via Embedded Decision

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

David Sikorski

May 2022

Abstract

This doctoral dissertation presents an investigation of embedded decision capabilities as a means for developing nuclear reactor autonomous control. Nuclear thermal propulsion (NTP) is identified as a high priority technology for development, and is the focus of this research. First, a background investigation is presented on the state of the art in nuclear thermal rocket (NTR) engine control and modeling practices, resulting in the development of a low order NTR engine dynamic model based on the literature. The engine model was used to perform the following investigation, and is intended to serve as a research platform for the future development of autonomous control in NTR engines. Next, since embedded decision techniques are argued to be the basis for autonomous control, additional background is provided on autonomy and embedded decision. This background served as the basis for this investigation, resulting in the first application of utility based decision in an NTR engine control system (ECS). The decision system was developed to accommodate faults detected in the primary control system, and is based on expected actuator availability. The behavior resulting from the actuator availability attribute was considered overly sensitive to perceived faults, and attributes based on engine performance are recommended for further development of the utility based decision approach. In addition to the research platform and the findings it produced, this work resulted in a collection of tools for future use in further research of engine modeling, control, and applications of embedded decision to NTR engine control.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

Nuclear reactor operation traditionally requires human operators to provide constant system supervision. However, there are many potential applications of nuclear power for which human availability is significantly restricted. Remote terrestrial power, planetary and lunar surface power, spacecraft power, and spacecraft propulsion are all examples of applications which will require reactor operation under some level of restricted human oversight.

NTP offers many benefits over conventional chemical propulsion. Perhaps most importantly, NTR engines have demonstrated significantly greater efficiency than is possible from chemical while still managing to produce thrust values desirable for space propulsion. These key advantages combined with the National Aeronautics and Space Administration's (NASA) interest in sending manned missions to Mars are what makes NTP a high value research target and a current focus of investigation under NASA and the Department of Energy (DOE).

Significant development efforts took place under project Rover and the Nuclear Engine for Rocket Vehicle Application (NERVA) program, and some rocket engine technologies have been matured since the end of those efforts. However, some of the necessary technologies are still limited or have not been addressed since the early 1970's. For example, engine control

is critical to realizing a new generation of NTP, but actual experience is limited to engine platforms developed and tested in the 1960's. This NTR engine control experience required significant human interaction to function under controlled testing conditions. Thus, the NTR engine control system (ECS) is a prime target for investigating and developing techniques for reducing the reliance of nuclear systems on human oversight.

Autonomous control capability for engine operation is a critical design issue for reviving NTR technology. Addressing NTR engine control is a step toward addressing the emerging need for reactor operational capability in the absence of human oversight. Developments from this work may also have future implications for various nuclear technologies, such as reducing staffing costs and increasing reliability of terrestrial power reactors.

1.2 Approach

The underlying goal of this research is to understand autonomous control in the context of nuclear systems. The intention is to begin developing the building blocks for autonomous capabilities in reactor operation. This will be done by investigating methodologies which have the potential to reduce reactor control system reliance on human interaction. Reduced human interaction will require a control system to take on some level of reasoning ability, and thus, embedded decision capability has been identified as a foundational element for autonomous control. This is the primary focus of the ongoing research in order to lay the groundwork for developing autonomy in nuclear applications.

Specifically, decision capabilities for fault accommodation systems to address urgent time sensitive scenarios such as failures in the control system will be investigated. For a terrestrial reactor, such scenarios would typically trigger a safety system shutdown, or require a human operator to evaluate the situation and act accordingly. Decision capabilities may be particularly useful for such off-normal conditions when human operators are not available, especially space propulsion systems, where an automated shutdown response could be detrimental to mission success.

This approach will serve as a proof of concept for reducing the reliance on human interaction for NTR engine control, and ultimately reactor operation, via embedded decision. This approach is intended to introduce features which will increase engine operational autonomy beginning at the lowest level of the engine control architecture. This "bottom up" approach differs from the more common "top down" approach used for designing autonomous control architectures for terrestrial reactors, typically referred to as supervisory control. This approach is intended to allow for incremental introduction of autonomous capabilities to a more conventional control system design, while still being applicable to more deliberately designed highly integrated control architectures proposed for advanced terrestrial reactor applications. However it is noted that while this research is intended to demonstrate that autonomy can be incrementally introduced to a control system, the system can be more thoughtfully designed and integrated with the physical hardware to a higher degree when autonomous behavior is addressed early in the control system design process.

1.3 Outcomes

This work required significant preliminary research to serve as a basis for the technological and theoretical understanding, and to build the development platform on which the research will be performed. So in preparation, three initial objectives were established, each resulting in a tangible outcome. First, a literature review was completed which identified the state of the technology for NTR engine control, engine modeling for control development purposes, and decision theory as the foundational element of autonomous control. Next, a dynamic model of a representative NTR engine was implemented to serve as the development platform. And finally, control systems were implemented which function in accordance with the traditional experience as defined by the literature.

With the groundwork in place, the research focus was shifted to the primary goal of understanding autonomous control using embedded decision techniques in NTR engines. The culmination of which is the proof of concept application of decision theory as an embedded fault accommodation feature in an NTR ECS. This was accomplished by applying

utility theory to the integrated engine and control system model as a fault accommodation feature for responding to the defined fault scenario. This approach was compared with a more direct logical flow control approach. The design, application, and analysis of this approach resulted in the development of recommendations for evaluating future decision approaches and additional recommendations for further research on the topic. The necessary framework which was built to perform this investigation is intended to serve as a tool to future researchers to continue pursuing the application of embedded decision algorithms in NTR engine control systems.

1.3.1 Original Contributions

There is currently a severe lack of digital tools available for performing research relevant to NTR engine dynamics and control systems. To satisfy this need, a research platform was developed to serve as a tool for conducting research on autonomous capabilities for NTR engines. This tool, delivered as a collection of Simulink blocks and Matlab functions, is also intended for use by future researchers to continue this work and to pursue future interests. This new research platform was then used to conduct research on the incremental development of autonomous capabilities by the application of decision theories to fault accommodation scenarios in NTR engine control systems. The original contributions generated to accomplish this work are listed below:

1. Research platform

    (a) A digital implementation of a low order historical NTR engine model was developed to serve as the basis for NTR engine dynamics. This includes an ECS which performs in accordance with historical applications to better isolate the effects of the autonomous features of interest.

    (b) A decision framework was designed and implemented for the ECS, which is unique for NTR engine control. This implementation includes modules for conducting research on embedded decision algorithms, and also for the future development of new autonomous features.

2. The demonstration of applied decision techniques to NTR ECS

   (a) In the interest of incremental development of autonomous features, a baseline was established using a logical decision approach. The outcomes from this implementation resulted in a better understanding of the fundamental behaviors and relevant desirable qualities of a decision system.

   (b) A utility based decision approach was designed and implemented, and constitutes the first application of utility theory to NTR engine control.

Chapter 2

Nuclear Thermal Propulsion

2.1 Introduction

NTR engines are a propulsion technology similar in concept to chemical rockets. Chemical rocket engines rely on combustion reactions to generate and heat a working fluid called the propellant. The heated propellant is accelerated through a converging-diverging nozzle, resulting in the desired reaction force referred to as thrust. In a typical bi-propellant liquid fueled rocket engine, separate fuel and oxidizer streams, such as hydrogen and oxygen, are mixed and combusted to produce the heated propellant. In an NTR engine a single propellant such as hydrogen is heated via fission reactions in a nuclear reactor. The simplified schematic in figure 2.1 illustrates how the heat is transferred from the reactor to the propellant. The primary engine components include a liquid hydrogen storage tank, hydrogen turbopump, nuclear reactor, and a nozzle.

NTR engines are a prime candidate for crewed deep space missions because they have the ability to generate high thrust with about twice the efficiency of modern advanced chemical rocket engines [1]. Thus, NTRs can significantly reduce transit time, reducing astronaut exposure to various health concerns such as space radiation and microgravity environments. Greater efficiency can also allow for larger payloads, and NTR engines have potential to be reusable.

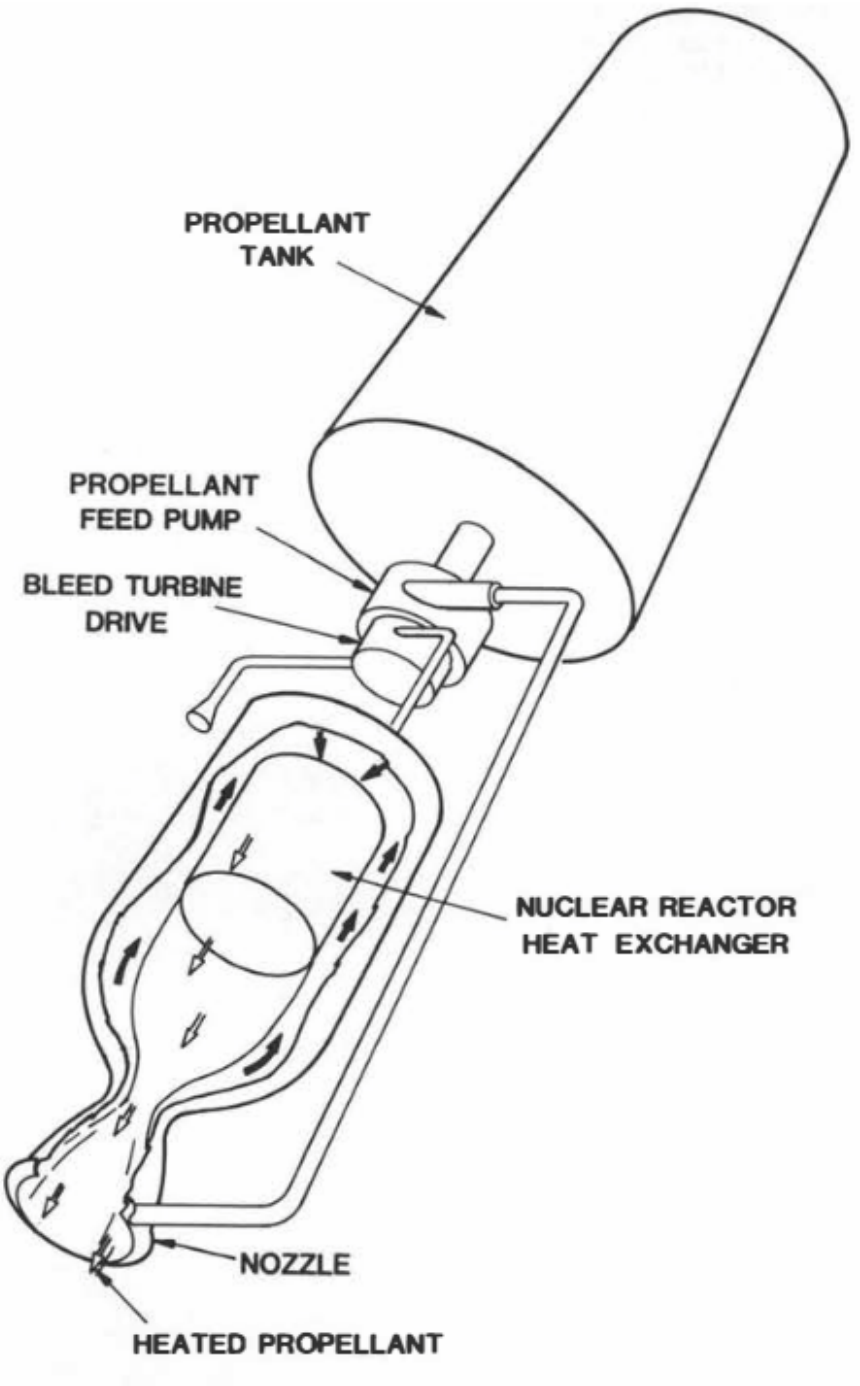

Figure 2.1: Generic NTR engine diagram displaying the primary components [2].

2.2 History

Preliminary investigations into NTP technology were first carried out by the Douglas Aircraft Company and North American Aviation in the 1940's. The findings were documented in classified white paper studies and provided to the United States government [3]. Higher efficiency, shorter trip times, larger payload, and potential for high reliability and reusability were among the many proposed benefits that NTR concepts offered over chemical propulsion options being pursued at the time. Benefits such as these gained NTP enough interest with the U.S. government that project Rover was established in 1953 as a propulsion option for various potential uses such as an upper stage for the intercontinental ballistic missile (ICBM) program, and lunar transit [2]. By 1955, two exploratory reactor research programs had begun. The Los Alamos Scientific Lab (LASL) developed the Kiwi program, and Lawrence Livermore Lab worked on the short lived Tory program, which was redirected to project Pluto. An advanced phase of the Rover project, called the Nuclear Engine for Rocket Vehicle Application (NERVA) program, was initiated in 1961 in an effort to develop flight style engines capable of withstanding the stresses and extremes of launch and spaceflight, while making use of advanced Kiwi reactors. Figure 2.2 illustrates the chronology of these efforts and their relationships to each other. Overall, 21 reactors were produced and tested under project Rover between 1959 and 1972 [2][4][5]. Figure 2.3 provides a more detailed look into the timeline of the individual reactors and their respective programs. The reactors tested were:

- 9 Kiwi reactors (including transient nuclear test, or "TNT")
- 3 Phoebus reactors
- 1 Peewee reactor
- 1 Nuclear Furnace
- 5 NERVA development reactors referred to as the nuclear reactor experiment (NRX)
- 1 NRX engine system test (EST)
- 1 flight style experimental engine (XE-Prime)

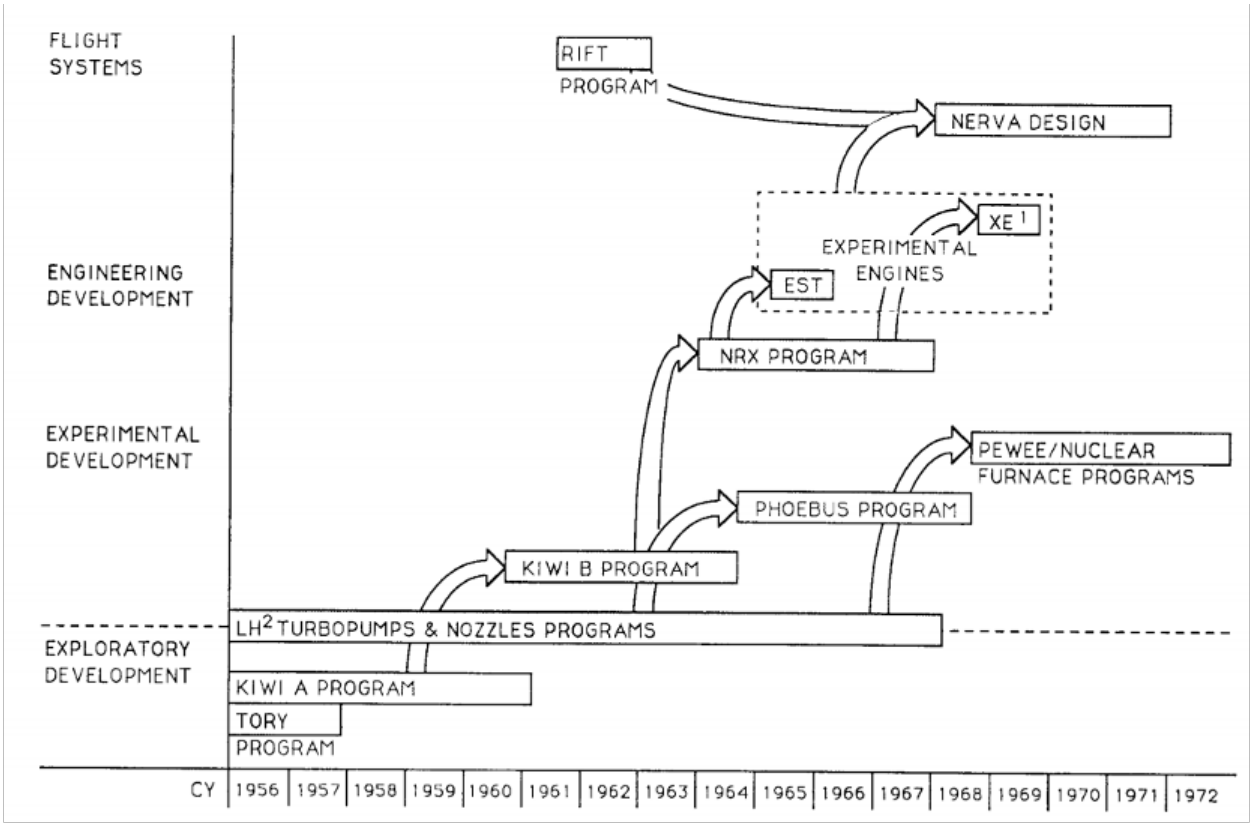

Figure 2.2: Chronology of NTR reactor and engine development efforts in the United States through the end of the Rover project [3].

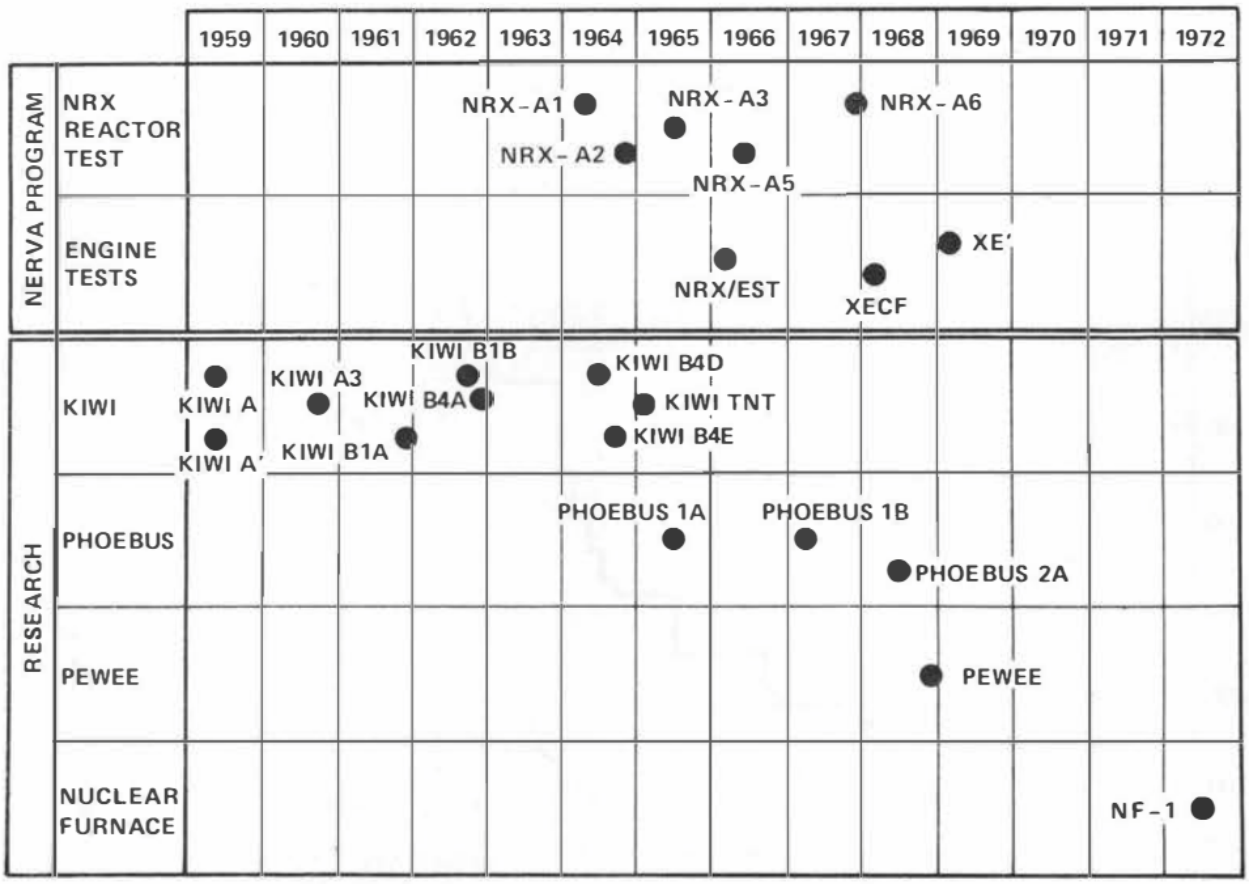

Figure 2.3: NTR reactor and engine test chronology [2].

The NRX-EST and XE-Prime were the only test articles of the entire effort to be considered engines. Prior reactor development tests used decoupled propellant feed systems, and were mounted in an upward firing position [3]. The NRX-EST was considered a "breadboard" engine because it used a self-powered experimental turbopump, and control systems mounted in a shielded housing adjacent to the reactor [3], though figure 2.4 shows that it was still mounted with an upward oriented nozzle. The culmination of testing efforts resulted in the XE-Prime engine test, which was the only nuclear rocket engine to be tested in a close-coupled, flight configuration [6]. As can be seen in figure 2.5, this meant that the engine was oriented in a downward firing position, exhausting directly into a low pressure altitude simulation system [3] [7]. Figure 2.6 illustrates how all of the relevant components such as the turbopump system and control valves were packaged between the reactor subsystem and the mounting structure as expected on a flight system.

Over the course of project Rover, long lead times and expensive full sized reactors led to the development of the Nuclear Furnace and Peewee reactors for more cost-effective fuels experimentation. However, economic constraints ultimately resulted in the cancellation of project Rover along with the NERVA program in January 1973 [2]. Though recently, NTRs are being reconsidered as an advanced propulsion option due to NASA's recently revived interest in crewed deep space missions [8].

2.3 Challenges

With renewed interest in development, it is important to understand the unique challenges NTR reactors face in comparison to terrestrial power reactors. There are many significant challenges presented by the extreme environment a nuclear thermal rocket engine is expected to experience. Simply existing in space for some period of time without operating, such as between burns, may result in cold soak temperatures as low as 20 K. The space environment also presents harsh conditions such as exposure to hard vacuum and space radiation sources such as galactic cosmic rays, solar particle events, and trapped belt radiation. During operation, the reactor and engine systems will be exposed to hydrogen ranging from liquid

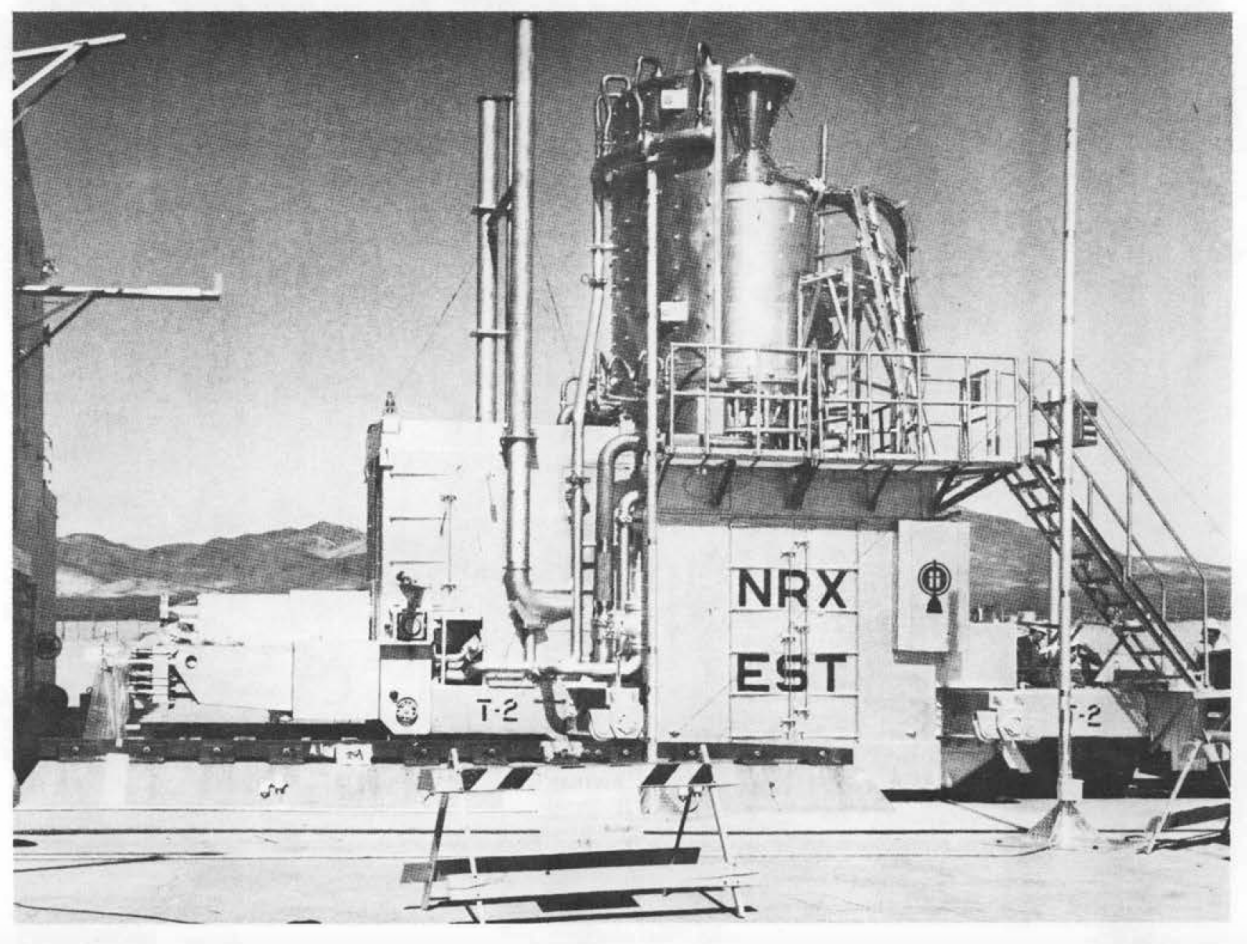

Figure 2.4: The NRX-EST breadboard engine mounted on a test stand in the upward firing position [2].

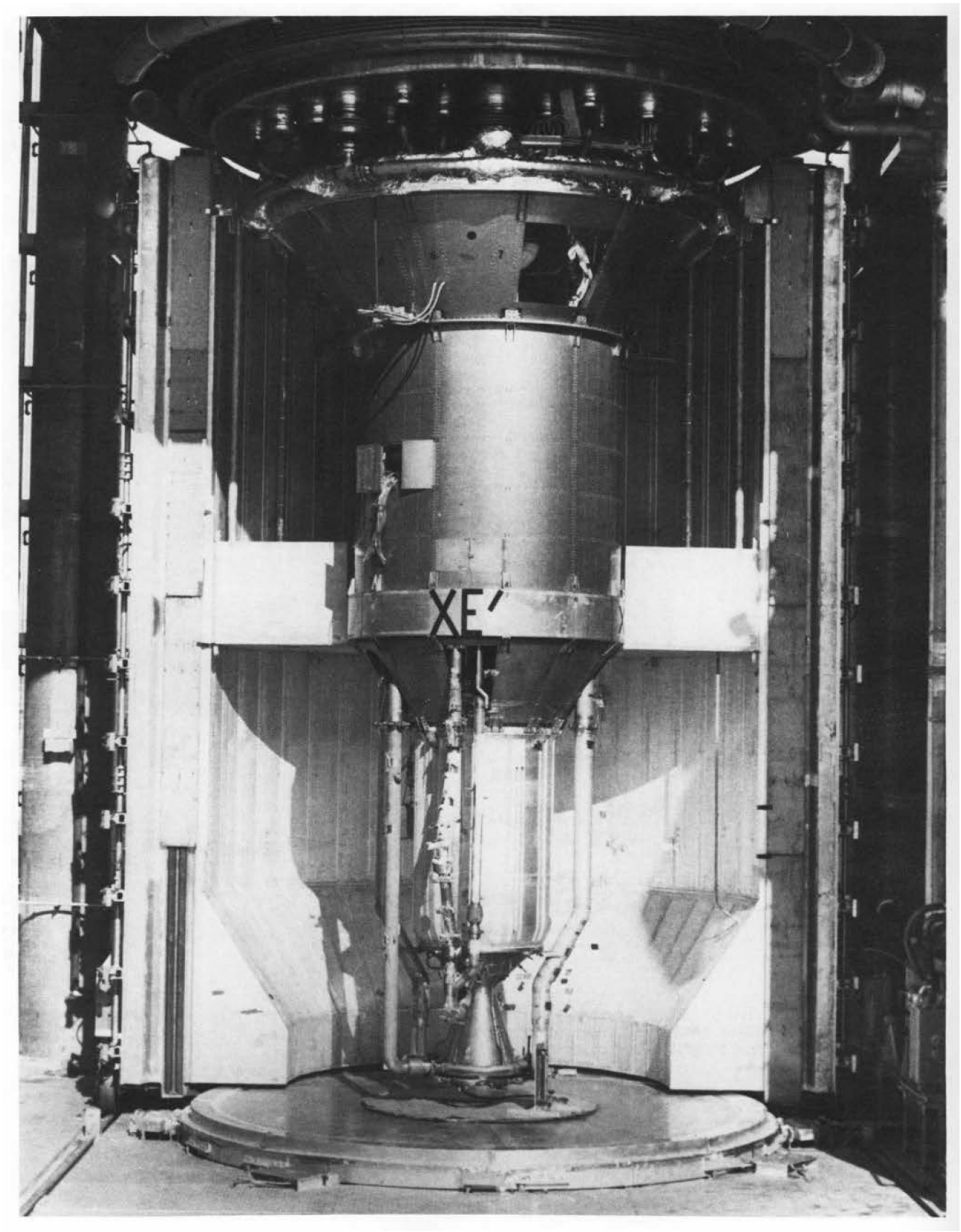

Figure 2.5: The XE-Prime engine mounted in a downward firing position [2].

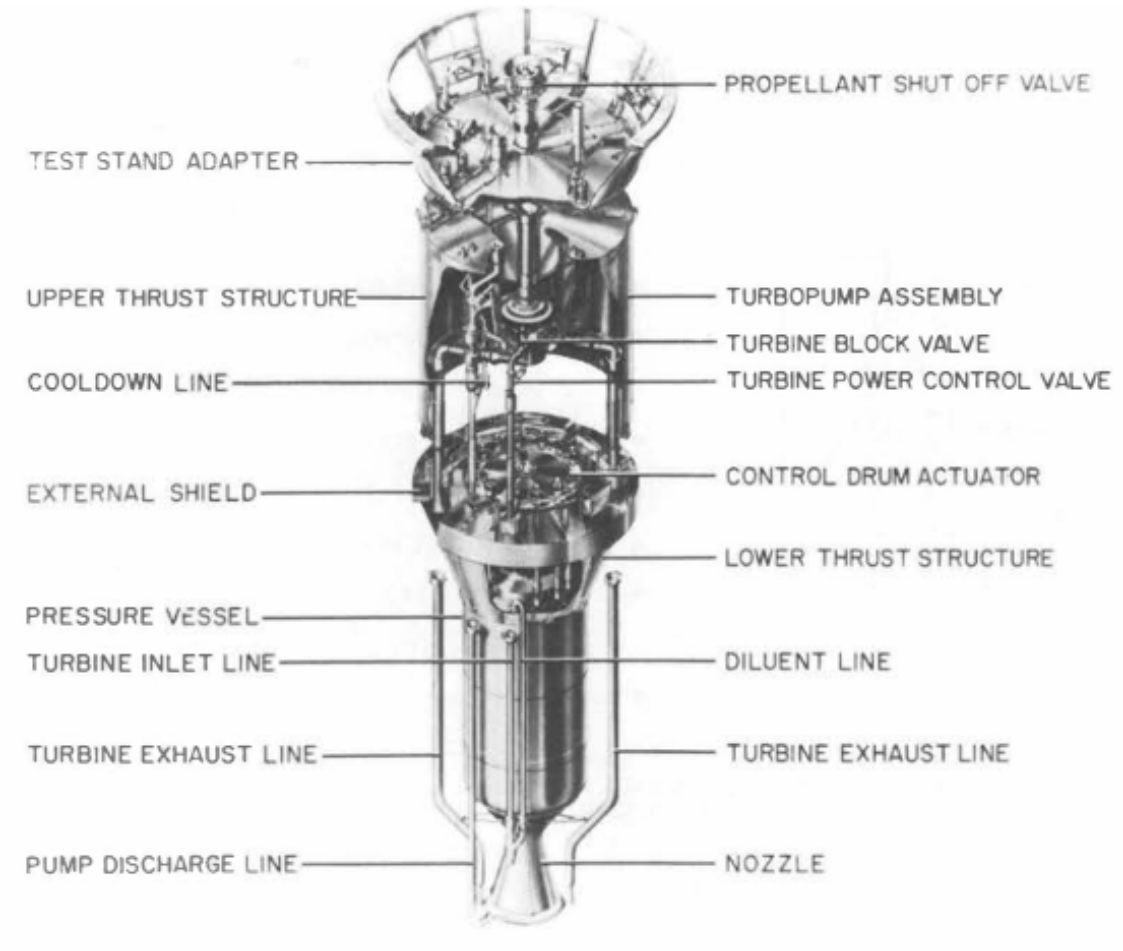

Figure 2.6: A diagram of XE-Prime highlighting the primary non-nuclear engine components [2].

temperatures to gaseous hydrogen greater than 2500 K. The engine will be expected to function reliably while enduring various combinations of these conditions over extended periods of at least several months, all while having endured the violent launch process to begin with. All of these conditions are significant departures from most reactor operation experience to date, and are expected to degrade the condition of the engine in ways not expected in terrestrial based reactors. Yet, the engine will need to function properly under these degraded conditions, because human maintenance and repair is unlikely to be feasible as it is on earth.

Terrestrial reactors, including XE-Prime, rely on an emergency shutdown procedure , called "scram", to stop the fission process in circumstances where a safety boundary is violated. However, this won't be a practical fallback because it is currently thought that a scram during engine operation will be undesirable for the mission outcome. For space reactors, and especially NTR reactors, operational paradigms must depart from traditional experience and employ methods which may allow the engine to continue running even in the event of failures. For example, cutting back power and extending the burn time may suffice for a given scenario where scram would be executed otherwise. Though, maintaining engine performance would be the most desirable goal. This kind of response to changes in system behavior is referred to as adaptation, and is a necessary capability in order to maintain performance under degraded conditions or during unanticipated events. However, current reactor operation and previous experience with experimental NTR engines required constant human oversight and input to manage these scenarios and make decisions about how to respond. For some scenarios, communications with earth may allow for ground based assessment and assistance in evaluating and responding to perceived issues between burns. However, communications delays and blackouts make immediate ground based assistance in assessment and response to anomalies or failures impossible during engine operation. Thus, to provide the necessary mission assurance, a nuclear rocket engine must be able to respond to rapid events and adapt to evolving or degraded conditions with minimal reliance on human oversight.

The need to reduce reliance on human interaction naturally leads us to pursue autonomous operation. There are many features which can improve operational robustness, helping to build autonomous behavior. Features such as signal validation, diagnostics, mode selection, and decision making capabilities are highly desirable for providing additional operational robustness. Many of these techniques improve robustness by preventing a system from requiring external intervention by being resilient to degradation and failures. Though, circumstances may still result in situations which may exceed the capabilities of such features, still requiring decisions to be made about the operation of the system. Thus, embedded decision capabilities are considered to be foundational to autonomous control. As such, this research focuses on the limitations of human interaction with space based reactors, and the challenges which arise from them.

2.4 Engine Control

2.4.1 Introduction

Addressing autonomous operation of NTR engines first requires a firm understanding of their control approaches, and knowledge of how they were traditionally operated. Since the XE-Prime engine was the first and only experimental nuclear rocket engine to be built and tested in a close-coupled, flight style configuration, it will serve as the basis for our understanding of engine control. The XE-Prime test program was designed to investigate various NTR engine functions and components to inform the further development of a flight engine. The focus of XE-Prime testing was to characterize engine dynamic performance, identify and demonstrate multiple control methods, and investigate transient operations. Specifically, the experimental plan (EP) general objectives as stated in volume I of the final report [6] were:

1. Investigate engine startup characteristics with a particular emphasis on different candidate control concepts and startup-restart from a range of initial conditions;

2. Investigate system characteristics under low power, low flow conditions, with and without the turbopump operating;

3. Obtain steady-state and dynamic performance data for selected engine operating conditions and demonstrate the reproducibility of the data;

4. Investigate engine shutdown and pulse cooling characteristics;

5. Investigate engine nuclear and non-nuclear component performance, including system controls;

6. Demonstrate the Engine Test Stand 1 (ETS-1) design concept and obtain facility performance data under nuclear firing conditions;

7. Obtain additional experience with remote handling of the engine.

The complete XE-Prime test program took place after cold flow experiments (XECF), and consisted of 40 total runs spread over 10 experimental plans. The tests took place between December 4th 1968, and September 11th 1969. During the testing program, the engine was successfully started 24 times, and ran for a total of 115 minutes [6]. Each of the objectives were met or exceeded during one or more of the tests. The XE-Prime test program managed to successfully demonstrate several control modes which amount to the most recent experience with nuclear thermal rocket engine control, and thus represent the state of the art.

2.4.2 XE-Prime Control Experience

Engine control was accomplished by manipulating the two control variables: temperature and pressure of the Hydrogen propellant in the nozzle chamber. Control was performed manually or automatically via two primary engine components: The turbine power control valve (TPCV), and a set of 12 control drums located radially around the reactor core in the reflector region. The nozzle chamber pressure was controlled by actuating the TPCV, and the chamber temperature was controlled by regulating reactor power via the control drums. However, feedback mechanisms do not allow for the individual adjustment of a single control parameter without interactions with the other. The Engine Control System (ECS) compensated for these effects under automatic control modes, however in manual control

modes, the operator was responsible for these adjustments to maintain desired operating conditions. The ECS provided six operating modes:

1. manual drum control
2. reactor power level control
3. chamber temperature control with three control mode options
4. manual TPCV control
5. chamber pressure control
6. program control

In the manual drum control mode, the engine operator commanded the drums to a desired position with a console mounted potentiometer. The operator was provided with indications for all twelve individual control drums, as well as an averaged value. The drums were operated in a "ganged control mode" where all drums were actuated simultaneously using a single mechanism. Single drums could be removed and adjusted individually when not operating, but it does not appear this was possible during operation. An error signal was generated by comparison between the operator input demand and a measured position feedback signal. The control was accomplished by reducing the error signal via control drum servo action.

Reactor power level control is a simple mode where a power level demand was selected by the operator between the limits of 55 and 5500 MW with a selectable period of 2, 5, 10, or 20 s. An error signal was calculated between the demanded and measured power, and the error was reduced via drum actuation.

Chamber temperature control generated drum position demands based on a desired temperature produced by a temperature demand generator. The operator had the option to choose between three separate temperature control modes. The single range temperature controller used fixed compensation, independent of the engine operating point. The on-off

temperature controller provided drum position demands only if the temperature exceeded a preset deadband. The state-program temperature controller made use of one of three compensation networks, automatically selected based on preset chamber pressure values. This was the best chamber temperature control method because it provided more optimal compensation depending upon the engine operating point. The feedback signal used for temperature demand comparison was the average of 5 chamber temperatures, with the option to use the average of 10 in-core temperatures instead.

Manual control was the default TPCV mode unless selected otherwise. Just like manual drum control, TPCV position was manually commanded by the operator via potentiometer. The measured position of the TPCV was provided by an average of three potentiometers housed in the TPCV actuator. The difference between measured position and demand was reduced by actuation of the TPCV servo valve.

Chamber pressure control allowed the operator to manually input a chamber pressure demand, which was achieved at a rate of 1, 2, 5, or 10 psi per second. Chamber pressure was evaluated as the average of three chamber pressure transducers and compared to the operator demand. The difference provided a demand signal to TPCV position control loop.

Program control provided three predetermined operating behaviors defined as high thrust, high specific impulse, and normal operation trajectories, defined on the engine operating map shown in figure 2.7. The engine behavior was chosen using pre-programmed plug-in circuit boards. Program control provided a direct coupling between pressure and temperature demand generators via temperature-to-pressure demand relationship, where pressure was the primary demand, and temperature was secondary. Thus, a pressure demand was generated first, and the temperature-to-pressure relationship determined the temperature demand. The other major difference with program control is that it could only be selected once the engine had been started, whereas the other TPCV and drum control modes could be combined and used at the operator's discretion in order to achieve engine start conditions. The engine had to be started and idling before program control could be initiated.

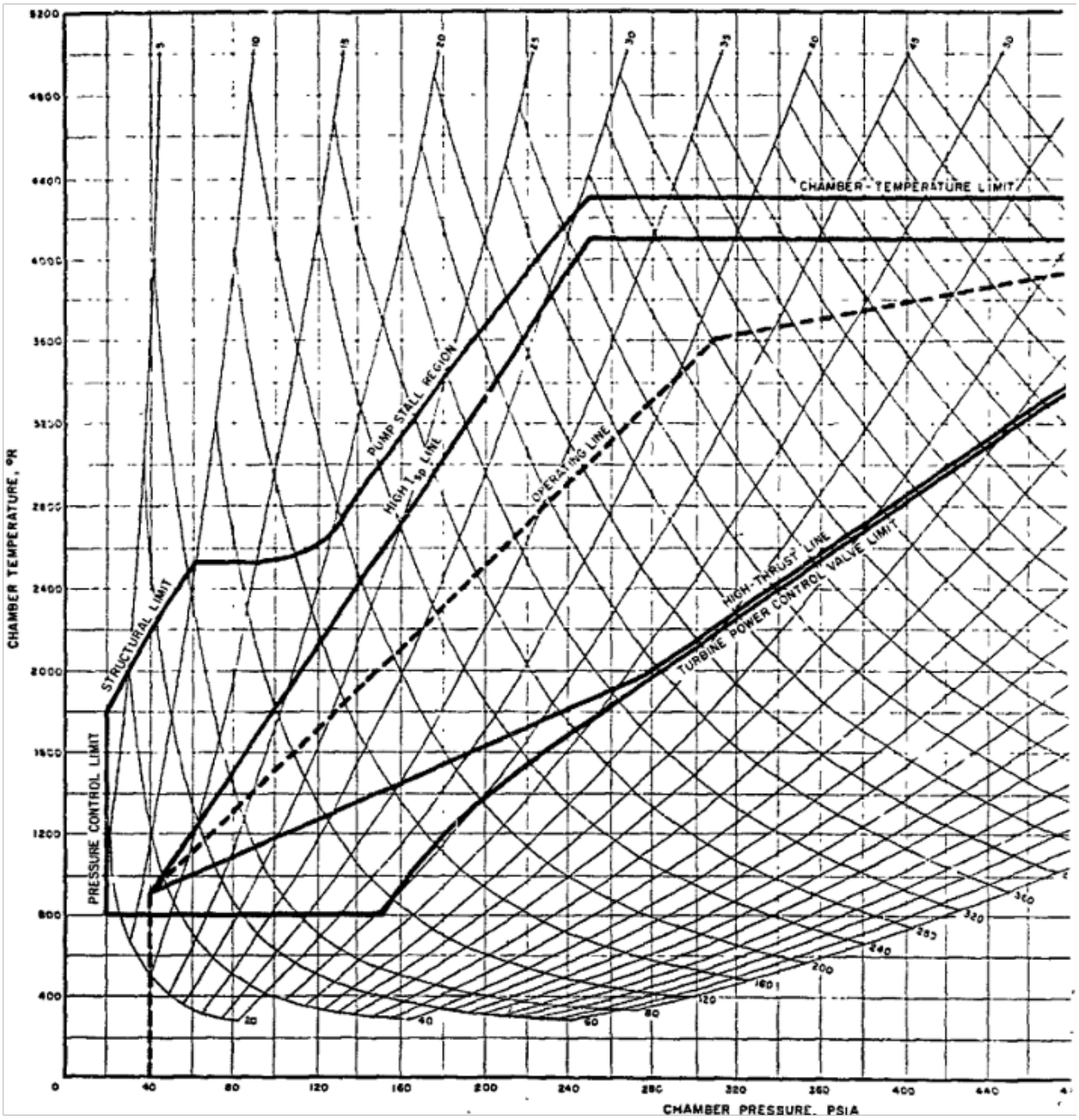

Figure 2.7: XE-Prime engine operating map defined by the temperature and pressure of the hydrogen propellant in the nozzle chamber [6].

2.4.3 Engine Start

However, as noted in the XE-Prime EP general objectives, engine startup was a primary point of investigation for the XE-Prime test series, including two primary automatic methods:

1. nuclear autostart
2. dry temperature autostart
3. damp temperature autostart
4. manual open loop start
5. closed temperature loop start

Nuclear autostart was performed using nuclear measurements for power determination. For the purpose of discussing XE-Prime documentation, "nuclear measurements" refer to any measurements made using instrumentation deemed "nuclear" in nature. This includes in-core calorimeters and various neutron measurements. Boron lined compensated ion chambers were specifically used for feedback in the nuclear control systems. Boron triflouride proportional counters were used for source, or startup range neutron signals, however it is unclear at this point whether they were used as feedback signals in the nuclear control systems. The nuclear autostart system would initiate startup by first rotating the control drums to a predetermined position past the critical angle, where the fission reaction is considered self sustaining for the given conditions. The reactor was considered started when some predetermined power level was achieved. Then, power control mode was initiated and the system was ready to continue engine startup with hydrogen flow. Bootstrap was initiated by opening the TPCV after completion of pump chill-down. Pump chill-down was performed because the cold flow tests revealed flow oscillations could be avoided when the reflector inlet was cooled below a certain temperature. Proper chill-down also helped to avoid pump cavitation. Fluid impedance and system energy balance are also primary reasons cited for chill-down [9]. Although possible, nuclear power control was not considered to be an optimal bootstrap control mode due to large variation in bootstrap timing.

Temperature autostart consisted of two differing start sequences, and was performed without the use of nuclear instrumentation by using temperature control instead of power control. Wet temperature autostart (WAS) initiated hydrogen flow at the same time as drum rotation. When the reflector inlet temperature dropped below a certain point, power was added by further turning the drums. Heating was sensed by nozzle chamber thermocouples. Once the desired nozzle chamber temperature was reached and pump chilldown achieved, the TPCV was opened and pressure control activated till bootstrap was complete. Dry temperature autostart (DAS) typically initiated hydrogen flow after the drums, when a preset core temperature was achieved. Notably, heating was sensed by in-core thermocouples as opposed to chamber thermocouples. Again, once chilldown was complete, pressure control was activated and primary propellant flow was initiated via TPCV opening, initiating bootstrap. The damp temperature autostart was a combination of WAS and DAS. Similar to WAS, damp temperature autostart initiated control drum motion and propellant flow at the same time. However, like DAS the in-core temperature measurements were used to detect heating.

The manual open loop start sequence began with pre-programmed drum movement, quickly followed by TPCV ramp to a preset value. The drums were stopped at a predetermined condition, and eventually turned back a few degrees to stop power increase at a predetermined value. Bootstrap was then initiated as soon as chilldown was complete. The engine was then manipulated within the operating map via fixed rate drum and TPCV ramps.

Closed temperature loop startup was performed by manually adjusting power till a temperature was reached which could be maintained by some specified flow condition. Once the temperature was maintained using a steady flow rate, the temperature loop was considered closed and bootstrap was initiated.

2.4.4 Further Developments

After XE-Prime, conceptual engine development continued under the NERVA E-1 engine series. Though none of these engines were physically realized, the efforts resulted in more complicated configurations and investigated more advanced control approaches [10]. The first iteration of the NERVA E-1 engine illustrated in figure 2.8 included an additional turbopump and added two support structure bypass valves (SSV) as additional control inputs. This design provided three control inputs for two control variables, though the specific control strategy for this configuration is not well defined in available literature.

Several variations of the E-1 engine were developed through the end of the program, with each iteration providing incremental design improvements. The E-1 revision 6.1, illustrated in figure 2.9, is believed to be the most recent NERVA engine design. Like all of the NTR engine designs, this one maintained control drums as a means of primary reactivity insertion. The additional turbopump and support structure valves introduced in revision 1 remained, but engineers removed the TPCV and added turbine bypass control valves (BCV), as well as pump discharge valves (PDV). This means revision 6.1 had 4 control inputs for the two primary control parameters. Recognizing the highly cross-coupled nature of the engines, engineers developed multivariable controllers as an improvement over previous single-input, single-output control approaches [11]. Linear quadratic optimal controllers with two region gain scheduling were designed for thrust buildup, and supervisory startup logic modules were under development as a prototype. Some as-yet undetermined level of supervisory adaptation logic for expected component degradation was prototyped, and diagnosis and adaption to single turbopump failure was implemented. The engine dynamics were further complicated by the new closed loop engine cycle. This cycle directs all of the propellant used to power the turbopump back into the engine to be further heated and used for thrust. The benefit of this engine cycle is increased efficiency, at the cost of greater complexity.

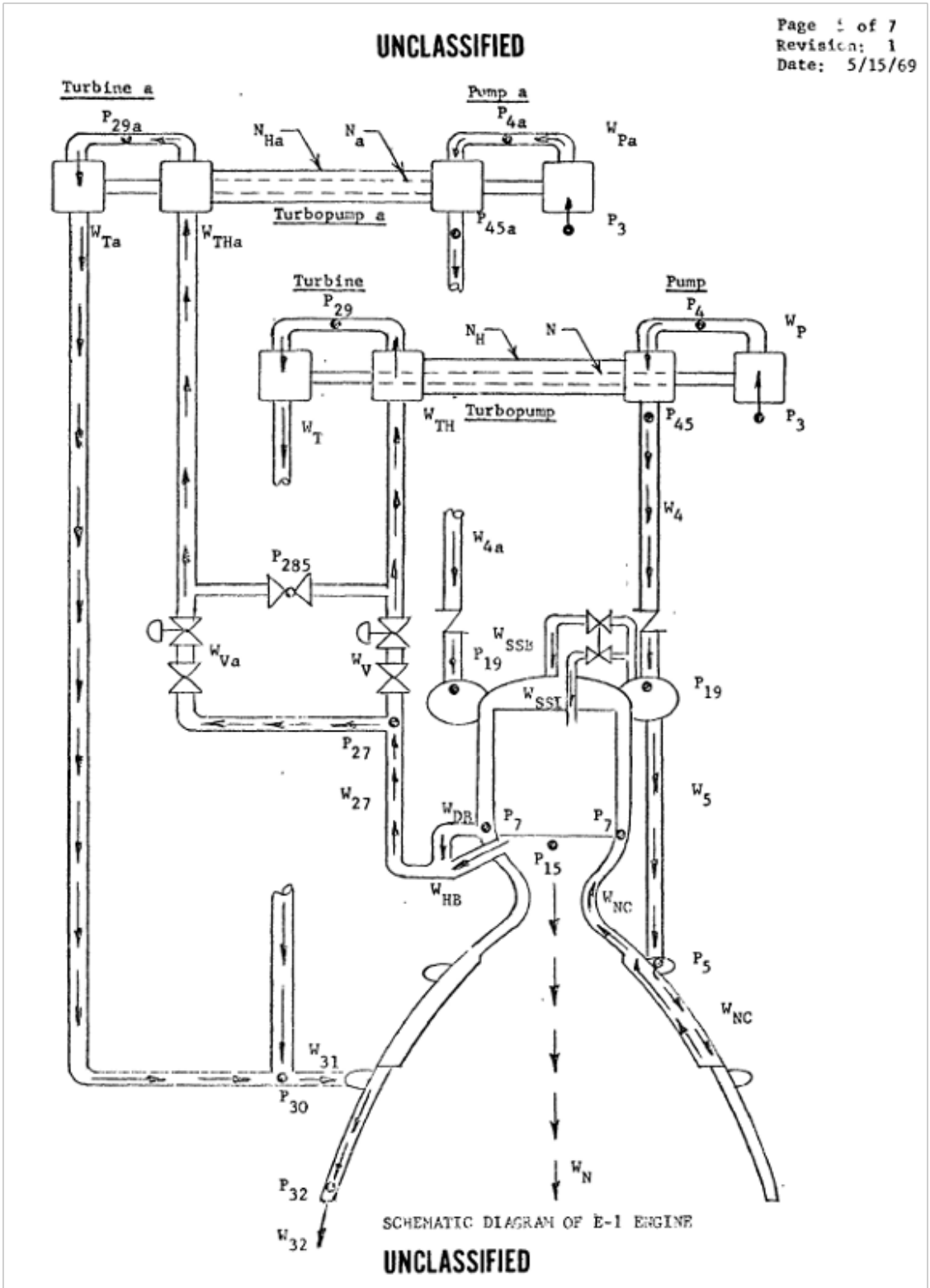

Figure 2.8: NERVA E-1 revision 1 engine diagram [12].

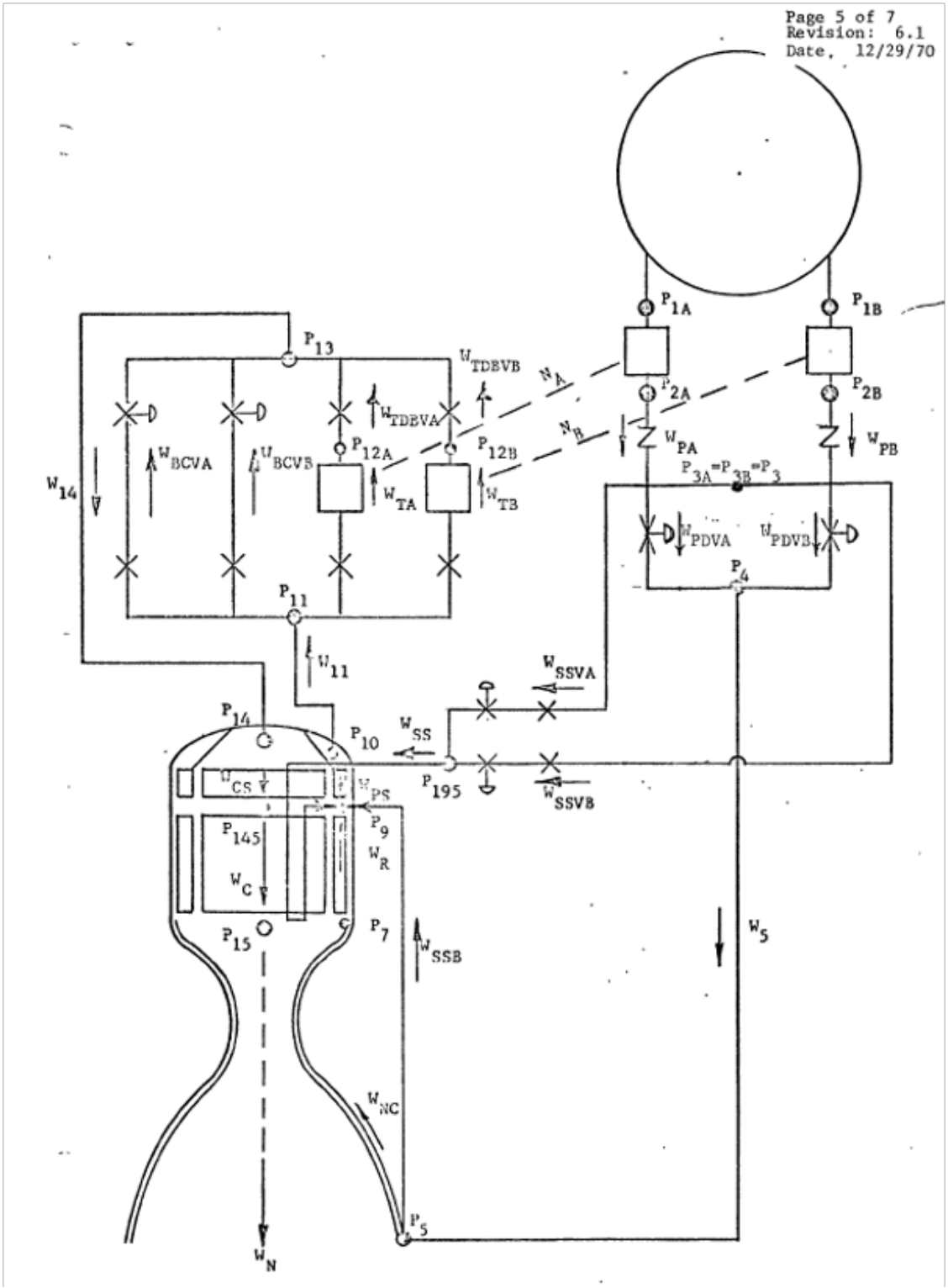

Figure 2.9: NERVA E-1 revision 6.1 engine diagram [13].

2.4.5 Summary

During the XE-Prime EP series, researchers were able to investigate concepts and operational capabilities of NTR engines important to understanding engine transient behavior and developing control capabilities. Some crucial engine operation capabilities demonstrated were engine bootstrap, multiple restart capability, and various startup and control methods. The NERVA E-1 series served to continue conceptual engine development and control approaches based on mathematical models. However, XE-Prime is the most recent experience with NTR engine control in the United States. These accomplishments are limited to a ground based testing facility with human oversight and emergency protocols in place. Each of the automatic control and startup methods required a person to monitor various systems and interact and make decisions along the way. Maintenance was available between runs, enabling repairs or replacement of faulty or damaged instrumentation and equipment. Additionally, the engine was subject to frequent scrams during startup due to reactor period limit violations [6][7][14][9]. In a mission oriented scenario, these human dependent activities are unlikely to be feasible, and scram is probably not desirable. Autonomous capabilities can reduce the need for human oversight and provide the ability to function under evolving conditions and unexpected events by building intelligence and adaptability into the system itself.

2.5 Engine Modeling

2.5.1 Introduction

In section 2.4, XE-Prime was introduced as the state of the art of NTR engine control experience. For this reason, the engine control philosophy and methodology for XE-Prime can serve as a basis for further development of NTR engine control. In section 2.4.4 it was noted that engine and control development did continue after XE-Prime, making use of mathematically modeled engines until the end of the NERVA program [10]. These models are crucial to control development and dynamic study of highly nonlinear cross-coupled systems such as NTR engines. Various engine model iterations are in development today for the same purposes [15]. While the majority of these models are relatively complex, consisting

of at least 50 differential equations, a relatively simple model with minimal computational requirements is desirable for conceptual controls development purposes. In the early 70's, a group of researchers recognized this need and developed a simplified low order dynamic model from the much more complex and computationally demanding models from the Rover and NERVA efforts. The following sections present these previous modeling efforts and my adaptation to a modern digital implementation for use as a research development platform.

2.5.2 Previous Efforts

Throughout the Rover and NERVA programs, engineers built dynamic models to perform transient analysis of the various engine and reactor configurations. The common analog model (CAM) was developed as an evolution of the analog models used for previous engine studies. This model basis meant that although the NERVA E-1 engine series was not physically realized, it remained "anchored" in test data from previous experiments due to the iterative process of model development and correction with test data [12].

The CAM was a time dependent mathematical description of the components of a given NTR engine configuration, implemented on state of the art analog and hybrid computers that were available at the time [13]. The various CAM iterations were relatively complex models based on conservation of mass, momentum, energy, and neutron density as described by reactor point kinetics. In addition, the ideal gas law was employed as the equation of state for hydrogen. The latest model consisted of 52 differential equations, at least as many algebraic equations, and empirical functions from experimental data. While the CAM was capable of providing detailed dynamic behavior studies of existing and proposed engine concepts, it was considered too complex for rapid controller development, resulting in the development of the simplified nonlinear model (SNM) [16] [17].

The SNM was derived from the NERVA E-1 revision 6.1 CAM by selecting dynamic variables which were considered essential for representing the predominant dynamic behavior of the system. This simplification was achieved through a three-part process. First, an initial variable selection based on engineering judgement was performed. Then, those equations

were linearized, and time constants were obtained for comparison with the CAM behavior. The differential equations with time constants less than one tenth of the dominant system time constant were reduced to algebraic equations. Subsequently, some of the algebraic equations were replaced with curve fits of E-1 revision 6.1 CAM data. This resulted in a reduced order model that primarily reflected the low frequency response of the system. Two of the most significant reductions to this model were the removal of the pump discharge valves, and one of the turbopumps. For the purpose of the SNM, the additional turbopump was considered redundant. This major simplification is not likely to be representative of a modern NTR engine design because current designs incorporate a boost pump that is required for normal engine operation. However, the SNM is a reflection of the engines it is based on and can serve as a generalization of NTR engine dynamics for the purposes of conceptual engine control development and demonstration. An unchanging fixed design like this is ideal for developing and testing the methodologies proposed in this work because modern engine designs are not complete and are continually changing.

The SNM, illustrated in figure 2.10, models the effects of a reactor and the two control mechanisms on the properties of the hydrogen propellant travelling through the simplified NTR engine. First, the propellant is pumped into the system. When the support structure valve (SSV) is open, hydrogen is allowed to travel through the support structure for cooling. It is then mixed with the rest of the propellant, which is directed through the nozzle wall also for cooling. Next, the propellant which has been heated by the support structure and nozzle cooling pathways is directed through the turbine to power the propellant pump. When the turbine bypass valve, also referred to as the bypass control valve (BCV) is closed, all of the propellant flows through the turbine for maximum power. To control the pump power, the BCV can be opened to reduce the total flow of heated propellant to the turbine. The bypass flow is then re-mixed with the turbine outlet flow and is directed through the reactor to be heated. To control the reactor power, the control drums can be rotated to absorb or reflect neutrons back into the core. Finally, the heated propellant is exhausted through the nozzle to generate thrust.

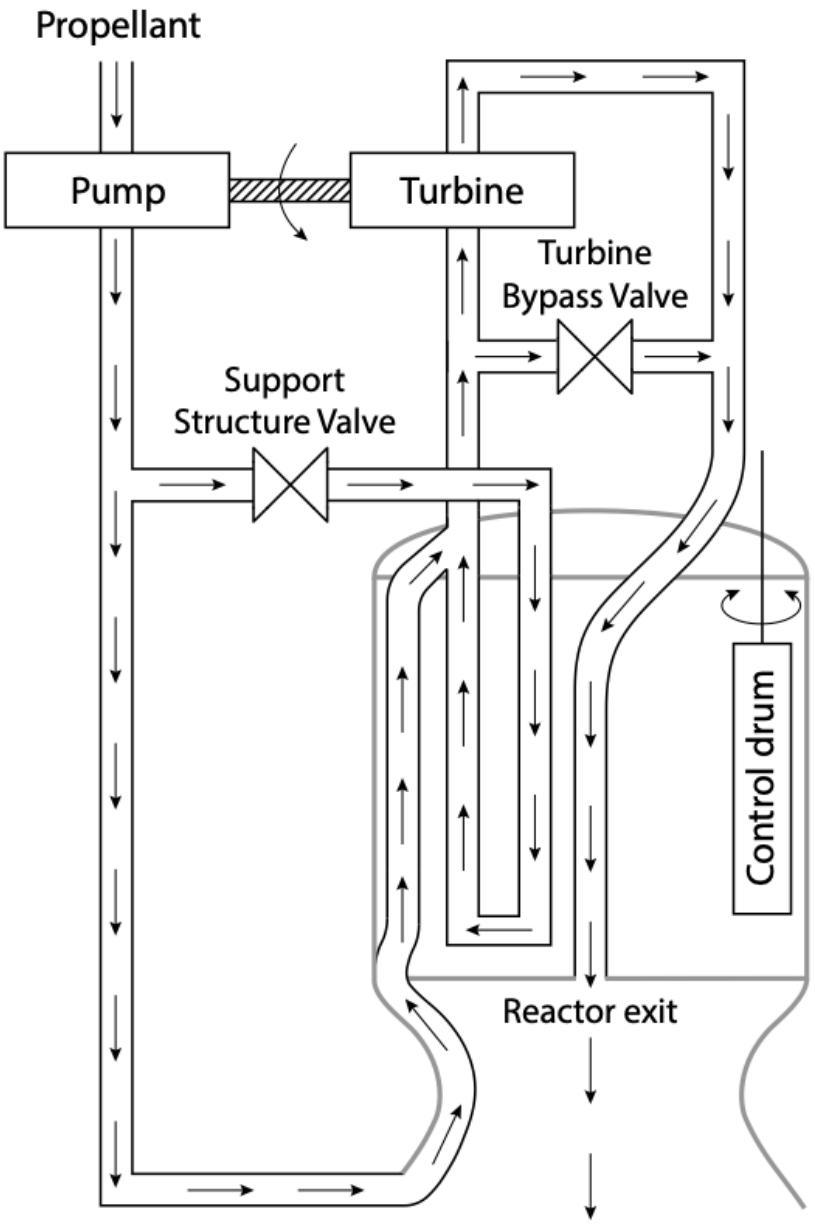

Figure 2.10: Simplified nonlinear model of NERVA E-1 revision 6.1.

To model the simplified system illustrated in figure 2.10, the SNM consisted of 11 differential equations and 13 algebraic equations. It was considered to be valid over the same range as the NERVA E-1 revision 6.1 model based on the results of validation studies. The region of validity includes the thrust buildup region and engine design point, with chamber pressures from 50 to 500 psi (344.7 to 3447 kPa) and chamber temperatures from 1000 to 5000 °R (555.6 to 2778 K). It is important to note that the CAM and SNM are not valid for modeling engine start conditions due to the assumptions on which their derivations are based. To our knowledge, one such evolution of CAM existed which could capture startup behavior, but the documentation is unavailable to date. Transient behavior and fault accommodation studies in the thrust build up region may be simpler to perform, and serve as the basis for this research. However, engine start is a key area of interest and developing automated startup procedures requires a model capable of representing engine start transients.

The SNM was originally implemented on an analog computer, while an experimental control system was implemented on a digital computer. This model was an important step for developing more accessible design and experimentation tools for NTR engine controls research. While this kind of model is perhaps even more relevant today due to greater accessibility to modern computers, the SNM was never ported to a digital system.

2.5.3 Modern Implementation For Use as a Research Platform

Simplified models are desirable because they can enable rapid dynamic studies and quick turnaround conceptual development. Model simplicity provides a low maintenance code base which can enable small teams or even individual researchers to focus on research priorities. A low order model also enables reduced computational requirements to allow studies to be performed in reasonable time frames on more widely available platforms. To support the effort to develop autonomous capabilities, the SNM has been recaptured and implemented using MATLAB and Simulink to serve as a testbed and research platform. The system of equations derived by Kendrick [16] was implemented in a Matlab function provided in Appendix A. This function was embedded in a Simulink block for simple time integration as a Simulink model. Figure 2.11 illustrates the model's structure in Simulink. The model's top

layer is shown in the top left with the three controllable inputs, the BCV, SSV, and drum angle, and two control parameters or outputs, nozzle chamber temperature and pressure. The center image shows the intermediate layer where input and output signals are routed to and from the embedded system of equations in the user defined central block, and the differential equations are integrated using time integration blocks. Finally, the bottom right image shows the system of equations which are embedded within the central block.

Next, to ensure the model was properly recaptured, the model behavior was verified by reproducing Kendrick's engine map studies that compared the nozzle chamber temperature and pressures of the original SNM to the CAM. The first study was conducted by performing BCV sweeps for support structure valve positions of 5°, 12.5°, and 30°. The BCV sweep range and drum angle of the original study are unknown, however they were reported by Kendrick to be performed to cover the operating range of the engine. This suggests the studies were performed with inconsistent input ranges in order to achieve results over the engine operating range. In order to keep the parameter sweeps consistent, the study was reproduced by sweeping the BCV angle (θ_{BCV}) from 35° to 90° for each SSV angle (θ_{SSV}), and the drum angle (θ_D) was chosen to be 20°. The results of the reproduced study are provided in figure 2.12 for comparison with the results reported from the original SNM and the CAM simulations. Upon initial comparison, it can be seen that the recaptured model behaves similarly to the original SNM and CAM, as expected. Although the values do not match exactly, the trends and orders of magnitude are well preserved. It is then noted that the pressure and temperature values generated by the new SNM do not span the same range as the historical models, particularly for smaller SSV angles. This is expected because the hydrogen density in the support structure region is known to have a strong moderating effect, and thus has the potential to significantly influence the reactor power. When the SSV angle is reduced the moderating effect is diminished, and the reactor power is reduced, thus requiring greater input from the other available control mechanisms to achieve the same range of outputs.

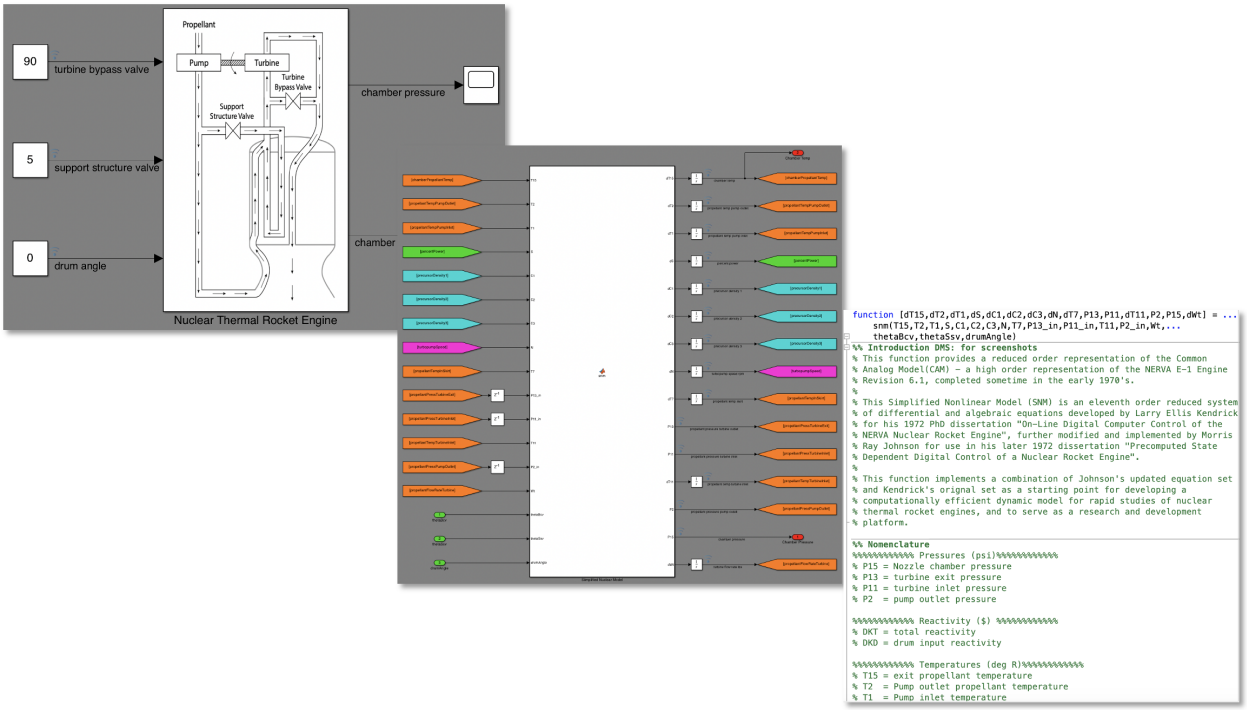

Figure 2.11: Simplified nonlinear model of NERVA E-1 revision 6.1 implemented in Matlab and Simulink.

This is in agreement with the expectation that the original studies were able to achieve similar temperature and pressure ranges for each SSV angle by compensating with different drum and BCV inputs.

The second study was conducted by performing BCV sweeps for control drum positions of 25°, 45°, and 65°. For this study, the original BCV values were again unknown, however one SSV position was stated to be 11.7°. As such, the engine map was reproduced using the same BCV sweeps as previously described for each of the three control drum angles, and the SSV was fixed at 11.7°. The results for this reproduced study are provided in figure 2.13 for comparison with the original SNM and the CAM simulation results. Again, the order of magnitude is correct, and the trends agree with the historical studies. Similar to the SSV study, pressure and temperature ranges output by the recaptured SNM are smaller than the historical models. This is again attributed to the likelihood that BCV input values were adjusted between runs in an attempt to cover the engine operating range. While the SSV angle was specified for the drum angle studies, it is unclear whether the SSV was fixed at 11.7° for each of the historical runs, which is expected to have a significant impact on the system behavior. More information could certainly help improve the reproduction of these studies, and provide a firmer basis on which to make comparisons, but it is unclear whether these details are available in the literature. However, this isn't considered necessary to complete the current research.

Recreating the historical engine mapping studies demonstrates that the modern digital implementation of the SNM produces similar results and behaves in accordance with the original model. These results have provided enough confidence that the original SNM has been properly implemented. The recaptured model is thus considered to be representative of a generalized NTR engine, which perfectly suits the need for a research development platform to support control studies. Finally, In order to finalize the SNM as a research platform for investigating autonomous control features, a baseline control system was implemented.

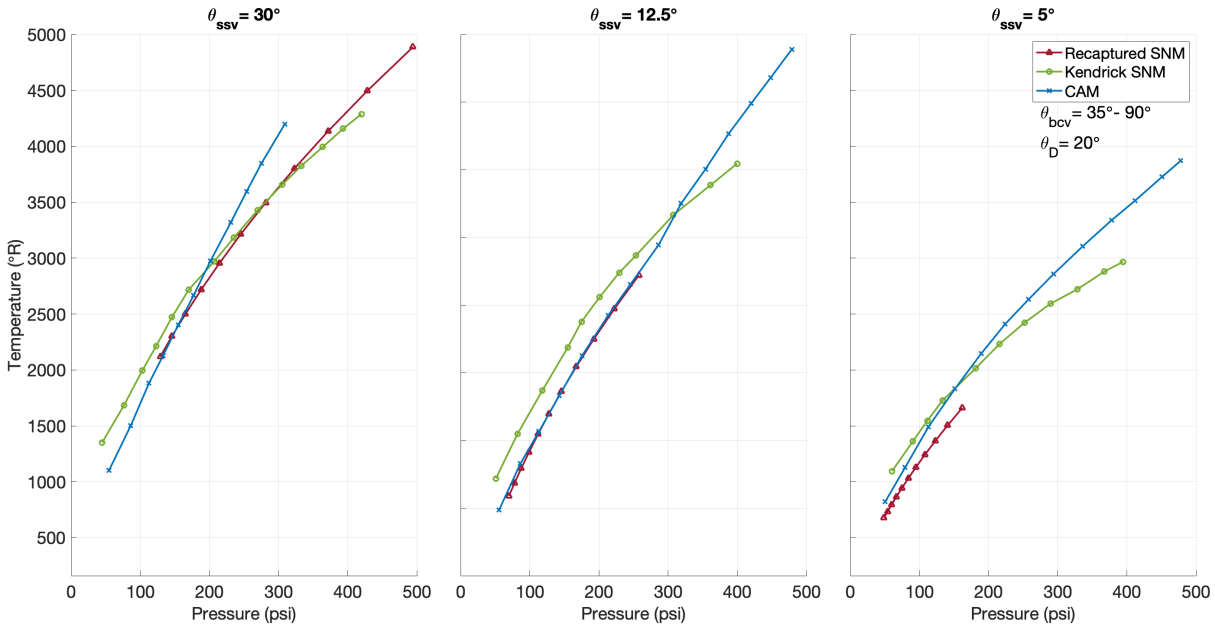

Figure 2.12: Results of the reproduced engine map studies with ramped turbine bypass control valve for three support structure valve angles.

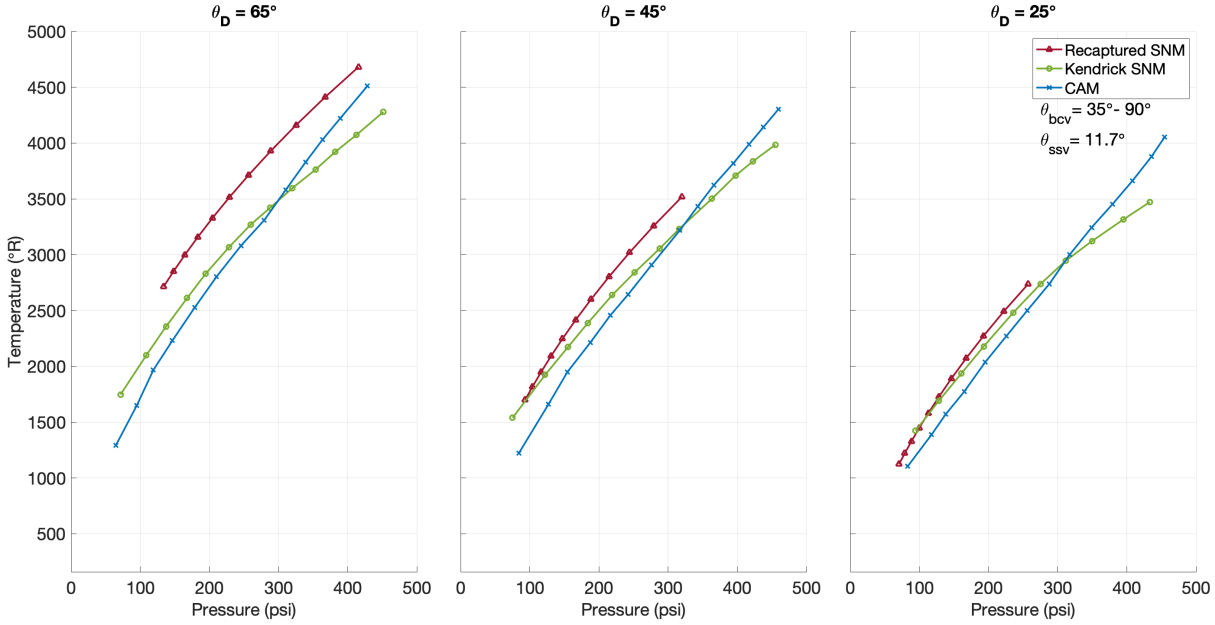

Figure 2.13: Results of the reproduced engine map studies with ramped turbine bypass control valve over three control drum angles.

2.5.4 Control System

The new SNM nominal control system is intended to implement a straightforward approach as seen on previous engine designs. While the engine model itself can serve as a versatile platform for more advanced control algorithm studies, this work is intended to investigate autonomous behavior in engine control systems. With a simple control system that behaves in a similar way to the historical experience, the research can be focused on developing autonomous capabilities.

The nominal control approach implemented on the new digital SNM is based on the XE-Prime pressure and temperature control. Individual PI control loops were built for nozzle chamber pressure and temperature control using Simulink. The diagram in figure 2.14 shows how the turbine bypass control valve is used as the pressure control element, and the drums are used to control chamber temperature. The target for SNM controller performance was chosen from the literature to be capable of bringing the engine from idle conditions to the operating point in about 30 s [17]. This transient operation is referred to as the thrust buildup stage and is demonstrated using the SNM with PI control in figure 2.15. In this figure, the nozzle chamber temperature and pressure are plotted against the commanded thrust buildup trajectory. For the majority of the transient, the SNM output is nearly identical to the commanded trajectory. However, the model deviates slightly from the demands in the lower power range of operation. This is hypothesized to be due to the nonlinearity of the system with regards to the exponential relationship between reactor power and neutron population. While it appears to be minimal, this behavior can be tuned using various available methods. This sort of effect was accounted for in the XE-Prime control system by making use of scheduled compensation as opposed to constant gains such as were employed for this particular implementation. Since these efforts are not intended to investigate or develop advanced control algorithms for NTR engines, this implementation is perfectly acceptable and provides a simple, nominal control system which will follow the commanded trajectory as needed.

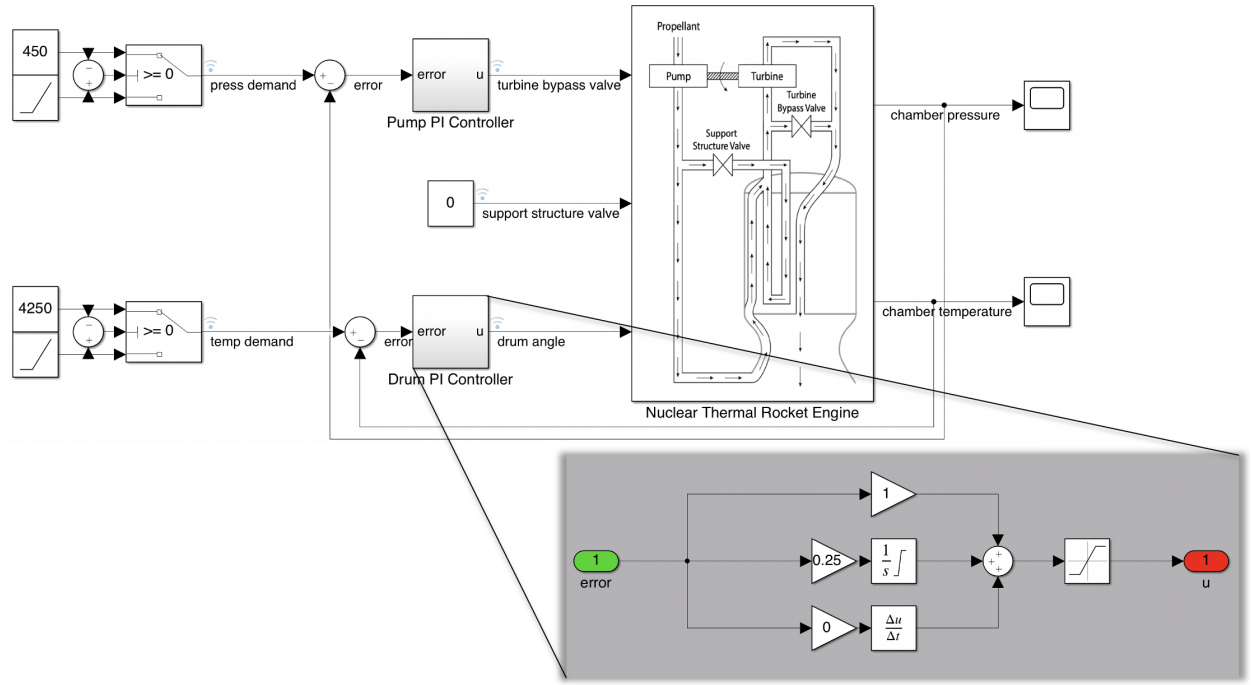

Figure 2.14: Simplified nonlinear model with PI loops implemented on chamber pressure and temperature controlling to a commanded thrust buildup trajectory.

Finally, as shown in figure 2.14, the SSV is normally operated by maintaining a constant position throughout the entire thrust buildup process. This simplifies normal operation, though the position value needs to be determined prior to a run. This leaves the SSV available as an additional control input to further develop the nominal control system. If deemed necessary, various approaches could be implemented to address the SSV operation such as a support structure temperature control loop, or perhaps something like a more simple desired position calculation or schedule. While for the purposes of this research the need for additional functionality under normal operation is not desired, it is known that the SSV position has a significant impact on reactivity for the SNM. This has the potential to serve as a backup or supplementary reactivity input as an option for fault accommodation logic. For this reason, another PI control loop has been developed to function as a backup control. Though this will be considered a part of the fault accommodation system, not the nominal control system, it is briefly described here.

The backup SSV temperature control functions similar to the nominal drum control shown in figure 2.14. Like the drum control, the SSV receives input commands from a PI controller which uses the nozzle chamber temperature error as the input. Figure 2.16 demonstrates the ability to perform a full thrust buildup run using the SSV backup temperature control with no input from the control drums. Though it is not intended to be used as a primary control method, figure 2.16 shows that it can provide similar performance to the nominal control approach, and will be useful in a backup capacity. Similar to the nominal control approach, the low power region also shows some difficulty in closely achieving the commanded trajectory. This isn't expected to be an issue, but it could be remedied using a variety of approaches such as gain scheduling or more advanced algorithms such as intelligent PI control if future researchers desire.

Now that the engine model has been completed with a nominal control system, and a backup which can be reserved as an option for emergency scenarios, the research platform is complete. This model can be used to perform all kinds of research relevant to NTR engines, and for this work, it is used to investigate autonomous control of NTR engines.

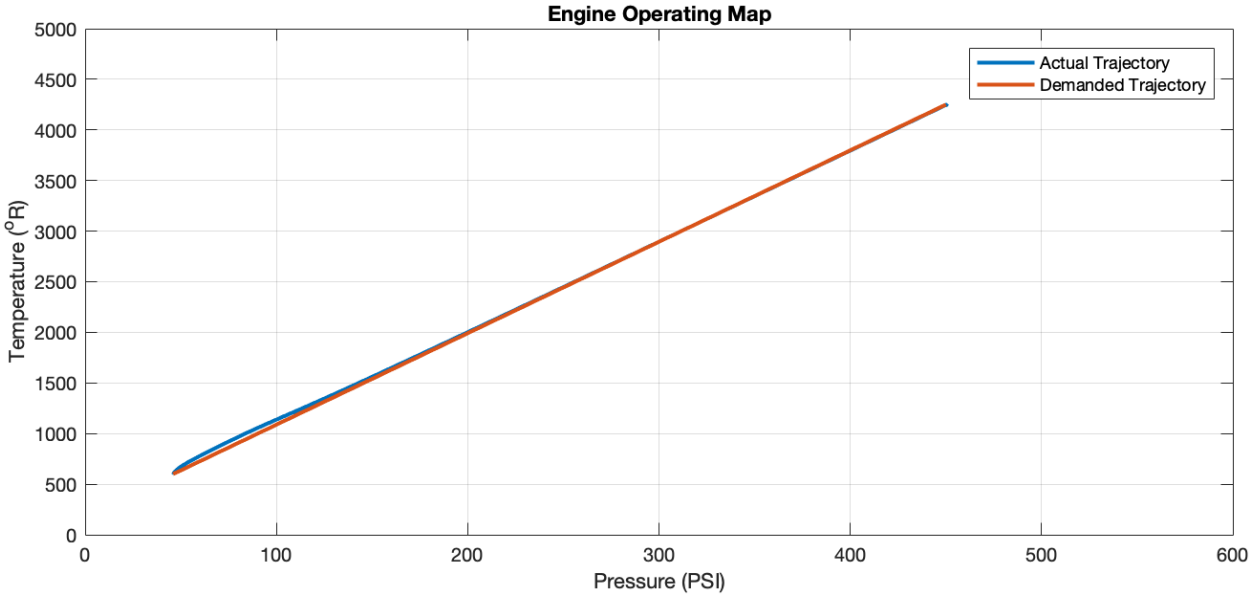

Figure 2.15: SNM thrust buildup achieved using PI control.

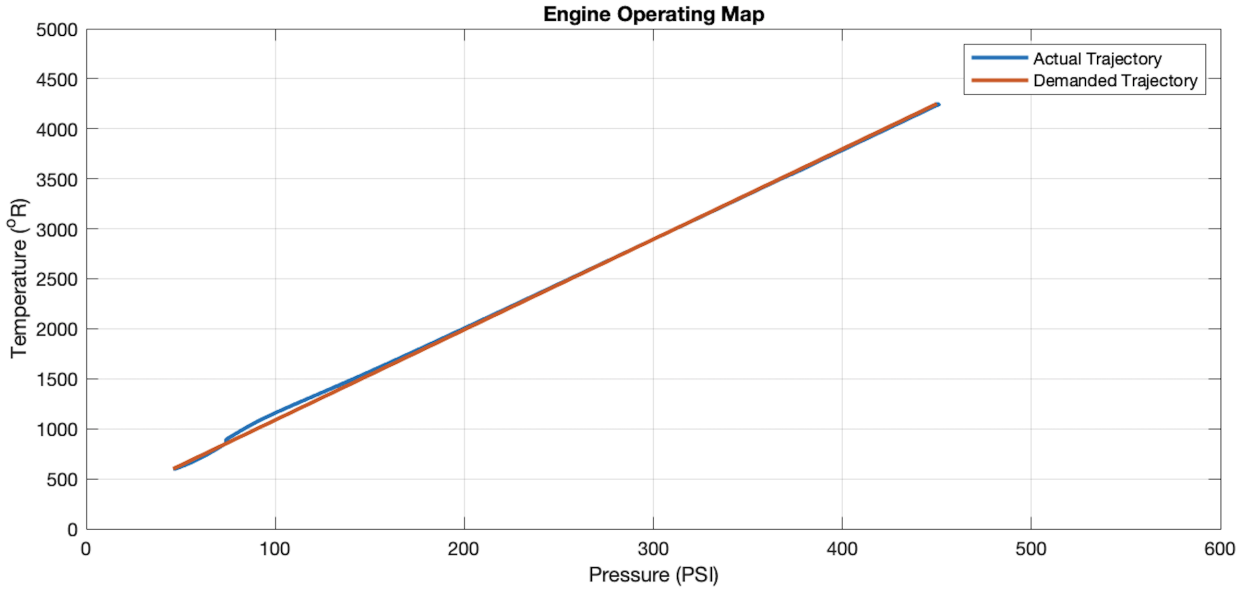

Figure 2.16: SNM thrust buildup achieved using backup SSV temperature control.

Chapter 3

Autonomous Control

3.1 Autonomy

The concept of autonomy can be understood from its Greek roots to mean the property of self-governance. As such, for the purpose of this work, autonomy is defined as the ability of a system to make decisions about itself. It is important to note that this is distinct from the concept of automation, which can be understood to mean self-action. An automated system is one which can act to execute predefined tasks, but has no independent authority over itself. Automated systems can only function under direct supervision of human operators, while autonomous systems are capable of functioning with some level of independence from humans.

These distinctions are also applied to control systems. The self-action of automated control does not suggest independent action or autonomous behavior. It may not govern its own actions. In other words, automated control executes straightforward, predefined actions. These control actions are provided by predetermined, fixed algorithms (measure, compare, act), with application only to the localized system. Although real-time interaction is unnecessary for normal operations under expected conditions, any decisions are left to human operators. An autonomous system may make decisions and determine its own action. This behavior requires some level of embedded intelligence to be present. With decision making being central to the definition of autonomy for this research, embedded decision capability

is considered to be a foundational element of autonomous control. However, there are many features which will be required to realize fully autonomous control. Though the focus of this work will be on embedded decision capability, these additional features will be briefly addressed here to provide a more complete discussion of autonomy. Features such as diagnostics, prognostics, performance assessment, signal validation, and condition monitoring can all contribute to reduce the necessity for human oversight of a given system. Some key characteristics of autonomous systems which arise from these features include intelligence, robustness, optimization, flexibility, and adaptability [18].

Robustness is achieved by an environmentally rugged implementation, accounting for design uncertainties and unmodeled dynamics, fault management, and self-maintenance or self-healing. Fault management can involve many deeper and distinct capabilities such as fault tolerance and forecasting, themselves built upon features such as redundancy, design diversity, error detection and recovery, reliability modeling, data collection and data driven modeling, and rare event prediction. Finally, self-maintenance and self-healing may be enabled by making use of captured design knowledge and prognostics for predicting failures, fault detection and isolation, and other self-correcting features.

Optimization is characterized by rapid response to demands, minimal deviation from target conditions, and efficient actuator actions. These capabilities are made possible by enabling flexibility and adaptability of the control system. Otherwise known as functional reconfigurability, flexibility and adaptability features make use of diverse measurements, communications, and alternate control solutions to adapt to a changing or degrading environment. This functionality is ultimately enabled by the inherent ability of the autonomous control system to make the decisions required to reconfigure itself.

These characteristics represent the possibilities of autonomy but they do not constitute a necessary set. Therefore, autonomous control can be viewed as providing a spectrum of capabilities with automated control representing the lowest extreme or baseline of the continuum. This also implies that autonomous control itself can be defined as a spectrum.

The incorporation of increasing intelligence and fault tolerance moves the control capabilities further along the spectrum of autonomy. In their work on remote operated undersea vehicles, Sheridan and Verplank provide a convenient detailed description of the levels of autonomy based on the amount of human interaction required by a given system [19]. For easy reference, this spectrum is summarized as a brief chart in figure 3.1.

First, to arrive at a decision point, a scenario requiring a choice has to be initiated. This way, the human is considered to be the starting point because they initiate the request to perform some general action. Even if that initial action was simply to flip an on switch before the system takes over, there is always a human involved. It is after this point where decisions must be made that systems are evaluated on their level of control autonomy. Level 1 on this spectrum is manual control. Just as with the XE-Prime manual control modes discussed in chapter 2.4, manual control relies entirely on human operators to make decisions about the system, such as choosing a desired operating pressure for example. Though the control actions themselves may be automated to achieve the objectives determined by the operator. Level 2 is defined by the introduction of computer based intelligence, where the operators are provided with a list of available choices determined by the computer. The operator has the option to choose any or none of these suggestions. Level 3 provides slightly increased intelligence by informing the operator which options may be more desirable for the given conditions. Level 4 further refines the level 3 choices by providing the human operator with just one presumably optimal suggested action. The operator still has complete discretion over whether to choose the computer defined course of action. Level 5 is where the paradigm shifts toward control autonomy as is more generally understood. In level 5 autonomy, the computer not only determines what it considers to be the best option available, it also makes the choice to request that this action be taken. However, the decision has to made by the operator whether the computer is allowed to perform the action. This is almost as if the computer is asking permission to take what it perceives as the best course of action, and it is up to the operator to decide. In level 6, the computer chooses the action and then informs the operator that it will be performed after some time limit. The operator is allowed to deny the action within this time limit. In level 7, the operator interaction is now reduced

to a passive role. The computer chooses and performs the action itself, and simply informs the operator of what was done. Level 8 autonomy also performs the chosen action, but only informs the operator of what was done if asked to do so. Level 9 again allows the computer to choose and perform the action, but the operator is only informed if deemed necessary by the computer. At this point, the computer is also making decisions about whether the operator needs to know about its actions. The final level functions the same as level 9 by choosing the proper course of action and informing the operator if deemed necessary. However, being the definition of full autonomy, it should have full autonomy in its actions. Therefore it is also able to determine whether the original action requested is to be performed in the first place. In other words, in level 10 autonomy, the system will first decide whether or not to accept the initial request to perform some general action.

In short, the more decisions which can be automated, the less reliance on human input is required, and greater autonomy is exhibited by the system to choose its own actions. In practice, the ability to perform decisions is enabled by additional information or calculations on which to base the decisions. This would inherently require the system to be more complex than a simple controller. This difference is illustrated by comparing the simple automated control diagram in figure 3.2 with the autonomous control diagram in figure 3.3. The traditional controller in figure 3.2 performs its task by comparing input measurements to some predefined target, executing some fixed control algorithm, and then outputting the results of that algorithm as actuator commands. The diagram in figure 3.3 shows that the same information which is used to generate the control action, is also used to generate diagnostic data, which is then fed to a decision module for making decisions about the system. The choices performed by the decision logic can then affect the controller as well as the diagnostic module. Sometimes this kind of control system is referred to as supervisory control, and can be extended to include many more complex features such as those previously discussed in this section [18]. The realization of full autonomy may involve many different capabilities and would likely look different for various applications, but a fully autonomous system will be completely independent of human interaction or supervision.

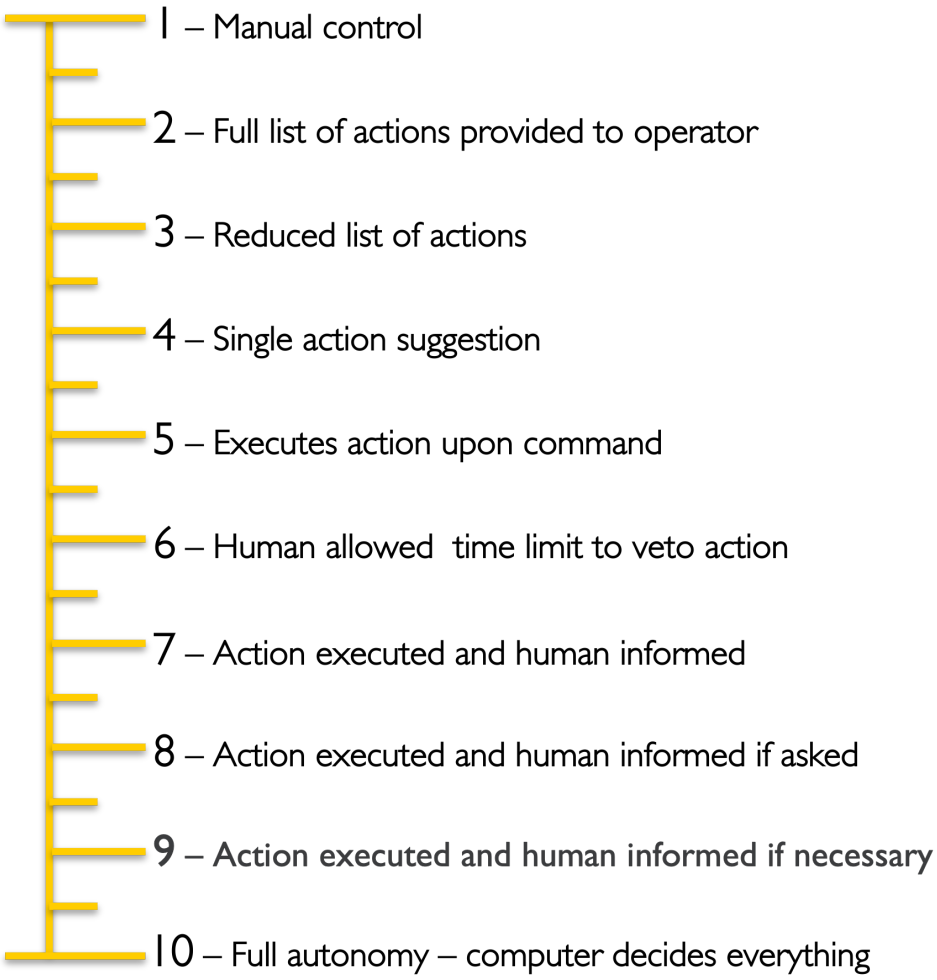

Figure 3.1: Levels of autonomous control defined by the amount of human interaction required [19].

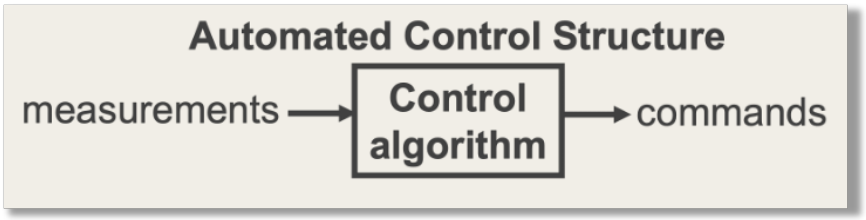

Figure 3.2: A simple automated control system which interacts with its environment by outputting commands based on comparisons made with input measurements.

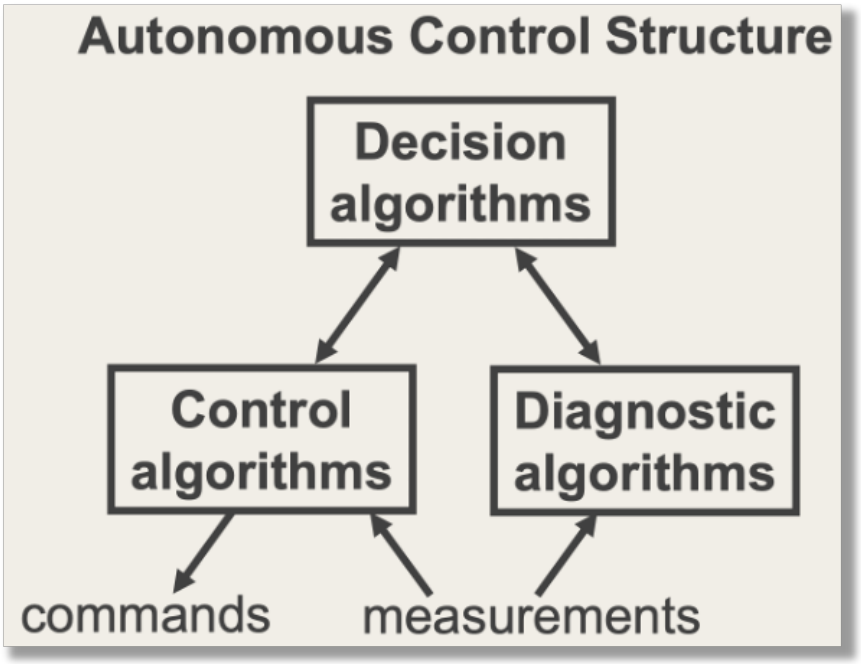

Figure 3.3: The basic structure of an autonomous control system includes diagnostic and decision capabilities, in addition to the basic functionality of an automated control system.

3.2 Embedded Decision

3.2.1 Introduction

Decision algorithms embedded within a control system, can provide greater autonomy for the system to interact with its environment. This means a system with embedded decision capabilities may be able to respond to unexpected conditions, such as fault scenarios, with reduced reliance on human intervention. This is the foundational element on which the mechanism to adapt and reconfigure is to be built.

The DOE advanced small modular reactor (ASMR) program serves as the baseline for automated decision and autonomous control in nuclear power [20][21]. The ASMR program documented the state of the art for decision methods, their applications, and their use in a terrestrial SMR supervisory control framework. This chapter is intended to be an introduction to decision algorithms and methods in the context of control systems, based on the ASMR documentation.

3.2.2 Decision Algorithms

Decision theories are developed, studied, and applied in a myriad of disciplines. From an engineering application perspective, the ASMR program found it useful to distinguish the many algorithms on the basis of time dependence and uncertainty. These distinctions are used to categorize decision methods by their general characteristics. The categories are:

- time dependence
  - static
  - dynamic
- uncertainty
  - deterministic
  - probabilistic

Static decision theories are time invariant one-time decisions performed in advance of their desired application. For a control system, this means all decisions would have been analyzed before being deployed to hardware. This approach allows known responses to be employed for known conditions. For a control system this can effectively be boiled down to a lookup table of responses for given conditions. While these decisions may have been analyzed once, on a control system, the decision process can be executed at any desired rate.

Dynamic decision theories are generally more complex time varying methods which can account for faults in real time. That is, the decision occurs in real time, as opposed to having already been performed prior to being deployed. These methods may be especially useful for applications which are too complex to characterize every possible state in advance. As such, these methods require some capacity to evaluate and diagnose their parent system in order to identify the current configuration or state of the system. This also allows for more adaptable, flexible behavior of the decision system, lending itself to being highly modular. While the control framework may need some re-design to include decision capabilities, the decision methods should exist in a standalone module as opposed to being "hard coded" into the control system. For this research, modularity is highly desirable because the approach is to build autonomous behavior into an existing control system by applying embedded decision methods from the bottom up. A dynamic method will be more adaptable to changes in the system as other decision methods are added and complexity is built into the decision module. Thus, it would be less likely to require reconfiguration as development progresses in future phases of research.

Deterministic methods are constructed using logical flow patterns, and can also be referred to as rule based. These allow for defining priorities and hard boundaries, and are good for enforcing deterministic rules and regulations on the final decision of a system. However this kind of method does not take into account the stochastic nature of most systems.

In order to take into account some stochastic system behavior, probabilistic methods make use of distributions to represent variables which can be characterized as such. For example,

an important piece of stochastic information is the probability of the risk of some unwanted outcome of an action. Purely probabilistic decision methods do exist, and the decisions they produce are said to be based on risk. However, these methods are often supported by deterministic methods. These combined approaches are referred to as risk informed methods, where risk is used to inform the decision process, but the decision is ultimately bound by explicitly imposed rules and regulations. This deterministic behavior is expected to be required for any modern government sanctioned power or flight system.

Moving forward with this work, there is a deliberate effort to maintain simplicity in the application of decision methods to the SNM ECS. Since the goal is to understand decision algorithms in NTR engine control systems, the first approach will be a decision system which is simple to implement, but still retains features which have not been applied to an NTR engine control system. The risk informed deterministic methods previously mentioned can provide these desirable attributes as a reasonable first attempt to studying decision capabilities in NTR engine control systems. A rule based framework can be designed to be simple and straight forward to implement with a probabilistic method as the core of the decision algorithm.

The BMW highly autonomous vehicle (HAV) made use of a modified multi-attribute weighted utility theory embedded in their rule based deterministic framework for emergency stop scenarios on the highway [20], and also for use in normal highway driving conditions [22]. Utility functions allowed the decision system to account for uncertainty exhibited by real world events, while the rule-based state machine framework enforced traffic laws and prevented non-deterministic driving behavior. When the system was engaged, sensor data would be used to build a picture of the surrounding environment. Then, the lanes available to the car were assessed using the utility functions, which assigned each lane a value, and suggested a course of action. The BMW HAV demonstrated that this approach is capable of providing a high degree of autonomous control on an automotive platform in a real world setting [22]. For these reasons a similar approach is being pursued as the first application of decision theory to nuclear thermal rocket engine control.

Chapter 4

NTR Engine Fault Accommodation With Embedded Decision Algorithms

4.1 Introduction

Normal operations for terrestrial reactors requires a significant amount of human oversight, and the same is true for the ground based NTR engine experience with the XE-Prime experiments. Of more critical importance, fault accommodation in nuclear reactors typically involves a significant amount of human decision because current control approaches are highly limited in addressing degradation or fault scenarios. However, availability of human oversight is severely restricted in space applications.

The solution to this limitation is to reduce the control system's reliance on human oversight and intervention for operational demands. As previously discussed, this can be achieved by developing autonomous control capabilities, and this effort focuses on fault accommodation as a more critical need. This research is focused on developing the building blocks for for increasing operational autonomy. To extend the state of the art in reactor control, a framework to embed low-level decision-making capabilities for fault accommodation is proposed as a proof of concept for increasing operational autonomy of nuclear reactors.

In order to best understand the effects of our approach, this problem has been distilled into what is perceived as the simplest problem with the potential for significant impact on the system. That is, this research is intended to focus on a critical failure scenario in order to best exercise the proposed benefits of embedded decision algorithms.

4.2 Problem Definition

The fault scenario to be addressed will be stuck control drums during the transient thrust buildup stage of engine operation. This scenario is considered critical for two reasons. Most importantly, the control drums are the only dedicated reactivity control elements provided on any historical iteration of the system, which is controlled indirectly via the nozzle chamber temperature. If the primary reactivity control mechanism is compromised, so is the functionality of the entire engine, which poses a direct threat to the mission and passengers. Additionally, for this particular engine design the other available primary control variable is nozzle chamber pressure. This control variable has an identical redundant control element, a second turbopump, designed into the system. Thus, nozzle chamber temperature control has less recourse in the event of actuator failure. Though this probably will not be the case for modern engine designs, they are still expected to have two pumps, just not identical in functionality.

In order to model this scenario, the drum fault, and also the fault detection, are simply emulated in software as logical flags. Since this research is not focused on fault modeling or fault detection, the level of fidelity required to emulate the fault and detection is not significant. However this may be of interest for future research topics. Once the fault detection signal is received, the system is then required to make a decision with the available information. The choices are whether to switch to the available backup control method, or remain in the current control configuration. This decision capability is embedded within a framework designed to function as a complement to the existing nominal control system, and simultaneously accommodate this research and future work which may follow.

4.3 General Architecture

The framework developed for this research is designed to be modular in order to have minimal effect on the original control architecture. This maintains separation between modules for a variety of reasons. The primary benefits of this approach being the original intentions to demonstrate incremental addition of autonomous behavior to a legacy control system while providing separation from the controllers for studying the decision systems within the NTR engine on their own, on a platform which is highly extensible for future researchers.

The embedded decision framework, conceptualized in figure 4.1, consists of two core capabilities in addition to the original control algorithms and necessary input/output modules. The decision architecture, being the primary focus of this research, is enabled by the ability to detect faults and draw actionable information from the available sensors. This capability to evaluate the sensed environment is defined as perception, and can encompass a wide range of functionality such as diagnostics, prognostics, signal validation, and more. These modules are informed by a set of inputs, and communicate back to the model through a set of outputs. First, data flows into the integrated control architecture via inputs. In general this input information can consist of raw sensor data, and also synthetic information such as predefined performance goals or on-line calculated values. The raw data is used directly by the available controllers to generate control actions. These control actions are no longer directly used as demands to the engine, but are provided to the perception module along with the raw input data for processing. The perceived state of the engine is then provided to the decision module along with the unprocessed control outputs. The raw input data could also be preserved and passed through to the decision module, depending on what particular information is desired for evaluating the choices at hand. When a decision is made, the choice is directly executed via the controller signal selection within the decision module. The selected control action is output as the desired actuator demands and logged for record keeping, while the perception module is also notified of the decision for future consideration. In practice, the first implementation of this conceptual architecture is somewhat simplified due to this work's focus on the decision module, as shown in figure 4.2.

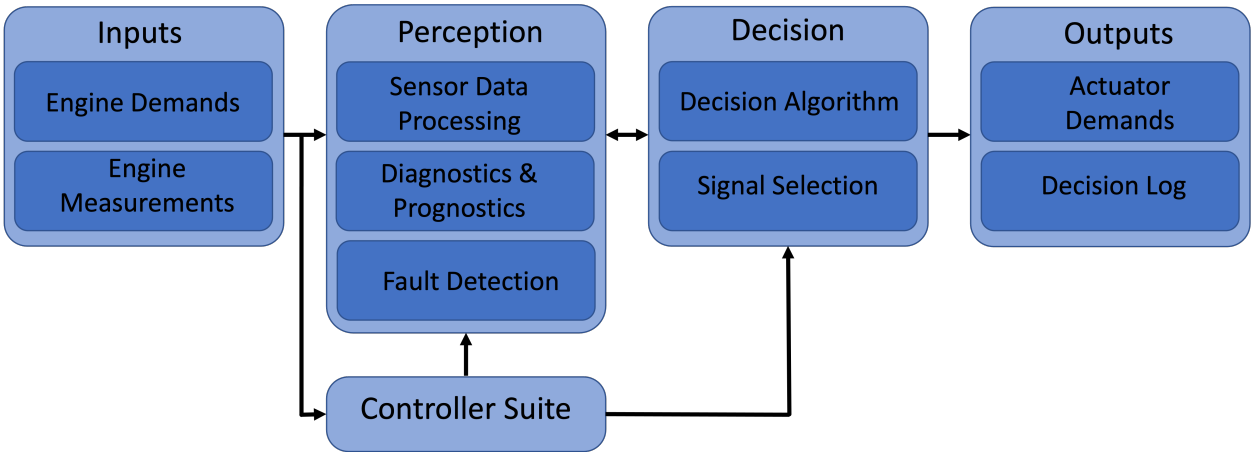

Figure 4.1: A general architecture for embedding decision algorithms into an NTR engine control system.

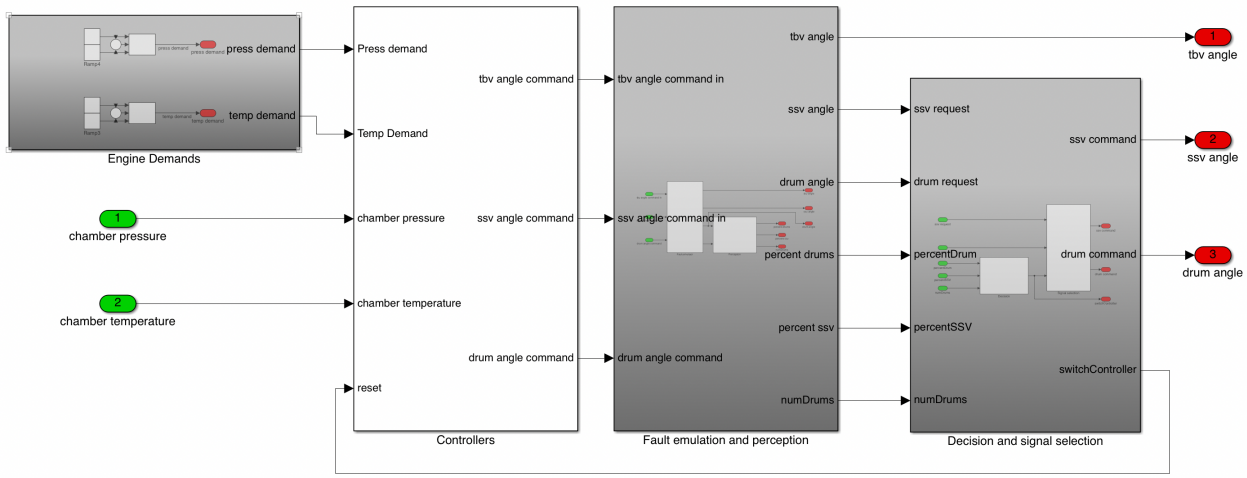

Figure 4.2: Decision architecture implemented in the SNM ECS Simulink model.

The inputs to the realized integrated control system mirror those which were proposed for the conceptual architecture. Nozzle chamber temperature and pressure were the only engine measurements needed for this implementation, and are input from the engine model located at the top level of the latest configuration depicted in figure 4.3. The engine demands as specified by the desired thrust buildup trajectory are also necessary, and originate from an engine demand block within the engine control block. These inputs are fed directly to the controller block to generate actuator position commands for following the demanded engine performance trajectory.

The controller block currently houses a collection of three controllers which make up the controller suite shown in figure 4.4, though any desired number of controllers can be maintained in this module. The turbine bypass valve controller and the drum angle controller makeup the nominal control system developed to maintain the engine temperature and pressure under normal conditions. The support structure valve angle controller is also included as the backup temperature control approach. Each controller simultaneously calculates a desired actuator position based on the inputs, and outputs position commands to the fault emulation and perception block, which serves as the perception module with some additional functionality.

The perception module shown in figure 4.5 serves two primary functions. The perceptive capability performs calculations on the actuator position measurements for use in the decision system. In addition, the perception module also houses the fault emulation capability. Because fault mechanics are not implemented, this is where drum actuator faults are injected via boolean flags, which also serve as the fault detection signals. To emulate stuck drums, the drum signals are fixed in position when a failure flag is received and the actuator position values are output for use in the perception and decision modules. The drum fault behavior can be scripted by the user via the "`drumFaults.m`" file in Appendix B. It is important to note that since actuator dynamics and their position measurements are not modeled, the controller outputs are taken directly as actuator positions.

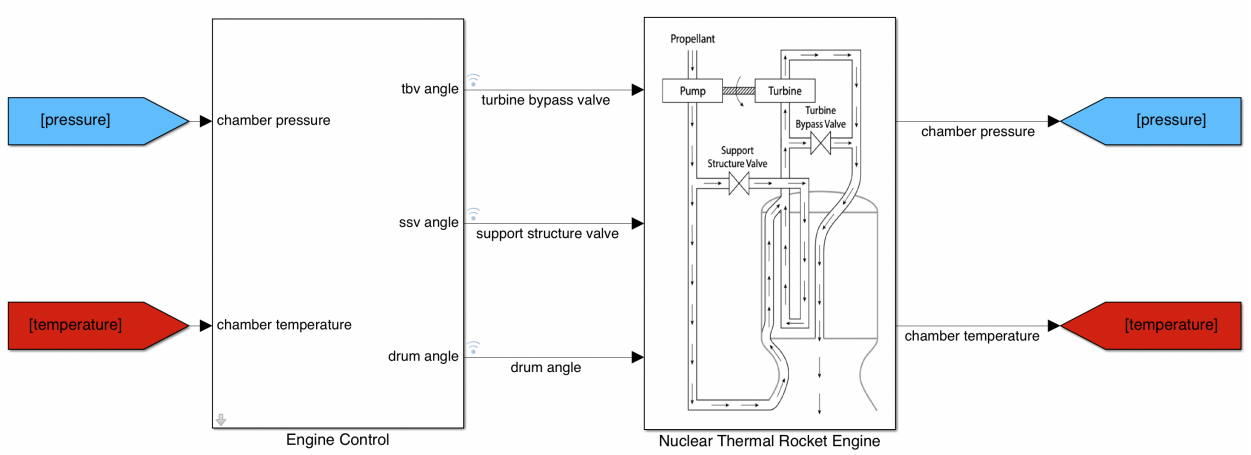

Figure 4.3: Top level of the Simulink model with the SNM engine model outputs used as feedback for the engine control and decision architecture.

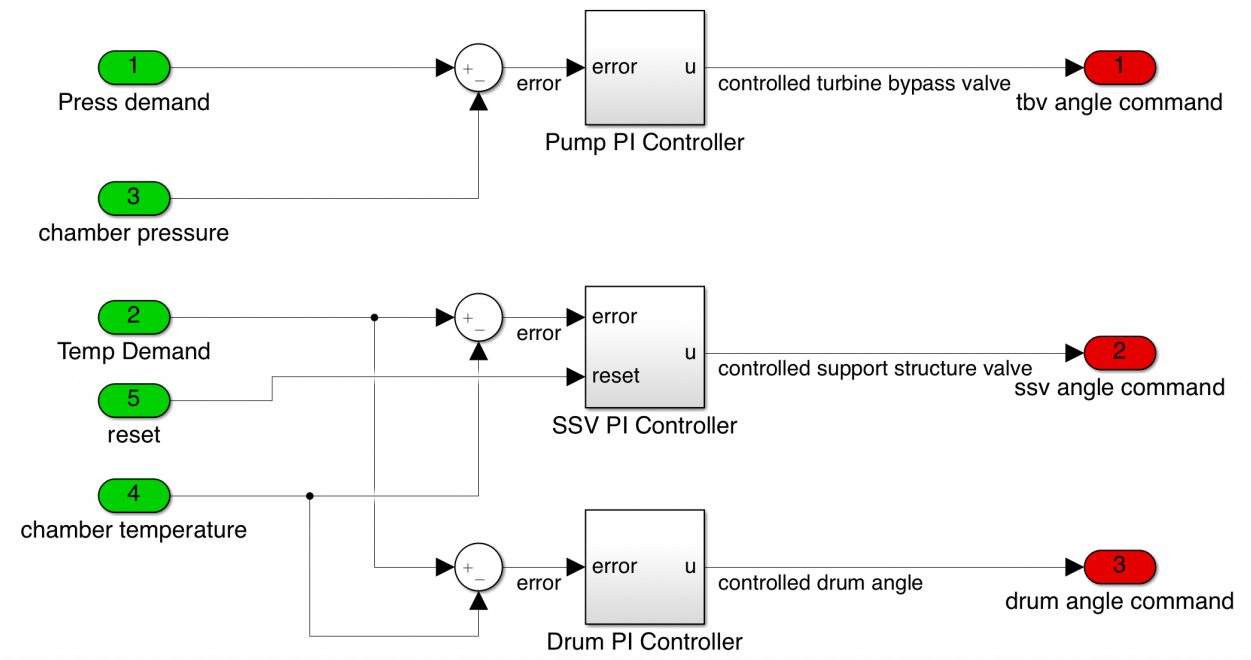

Figure 4.4: The engine controller suite is a collection of PI controllers for controlling the nozzle chamber temperature via control drum, and pressure via turbine bypass valve. Backup temperature control is provided by PI control of the support structure valve.

Once actuator dynamics are modeled, their position measurements can also be inputs to the engine control block, and used by any of the necessary modules. Any additional desired inputs can be made accessible at this level as can be seen in figure 4.2 as green input ports. In the proposed architecture, actuator and sensor faults would be modeled local to the relevant system, and the detection algorithms implemented in the sensing module would be responsible for detecting anomalies in the signals and notifying the decision module of any perceived fault. Since the SNM includes chamber sensor dynamics, these measurements may be good candidates for first attempts at including sensor or signal anomalies. After all necessary calculations, the actuator positions and additional perception data are output to the decision and signal selection block to be used as consideration in evaluating the choices at hand.

The implemented decision module also performs two functions, as shown in figure 4.6 . First, since the current approach is specifically designed for control drum fault accommodation, the SSV and drum positions along with the perception data are taken into account in the decision module. This decision block functionality will be described in more detail in section 4.4. Once a decision is made about which temperature control approach to use, the decision outputs a controller switch flag. If a decision is made to switch controllers, then the signal selection block will output the controller generated SSV position and fix the control drums in the positions they were when the switch signal was received. Otherwise, the SSV is maintained at the nominal predetermined position, and the control drums remain commanded by the respective controller. The controller switch flag is also output for use in the controller suite. This signal is used to reset the SSV controller integrator and proportional outputs in order to produce a bumpless transfer effect when switching from drum control to SSV control.

Finally, the BCV, SSV, and drum positions are output from the engine control block to the engine model. Again, since actuator dynamics are not modeled, the engine model takes these values as actual actuator positions. However, once actuators are modeled, these position values will be taken as control commands for the actuators to follow.

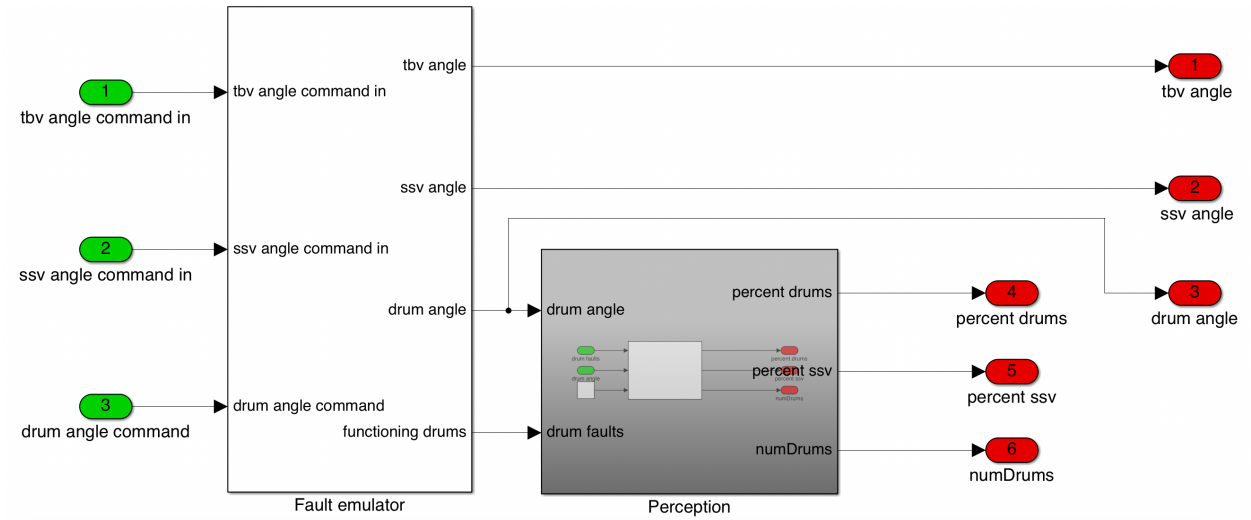

Figure 4.5: The fault emulation and perception block from figure 4.2 combines the fault emulation and detection with the perception capabilities required for this work.

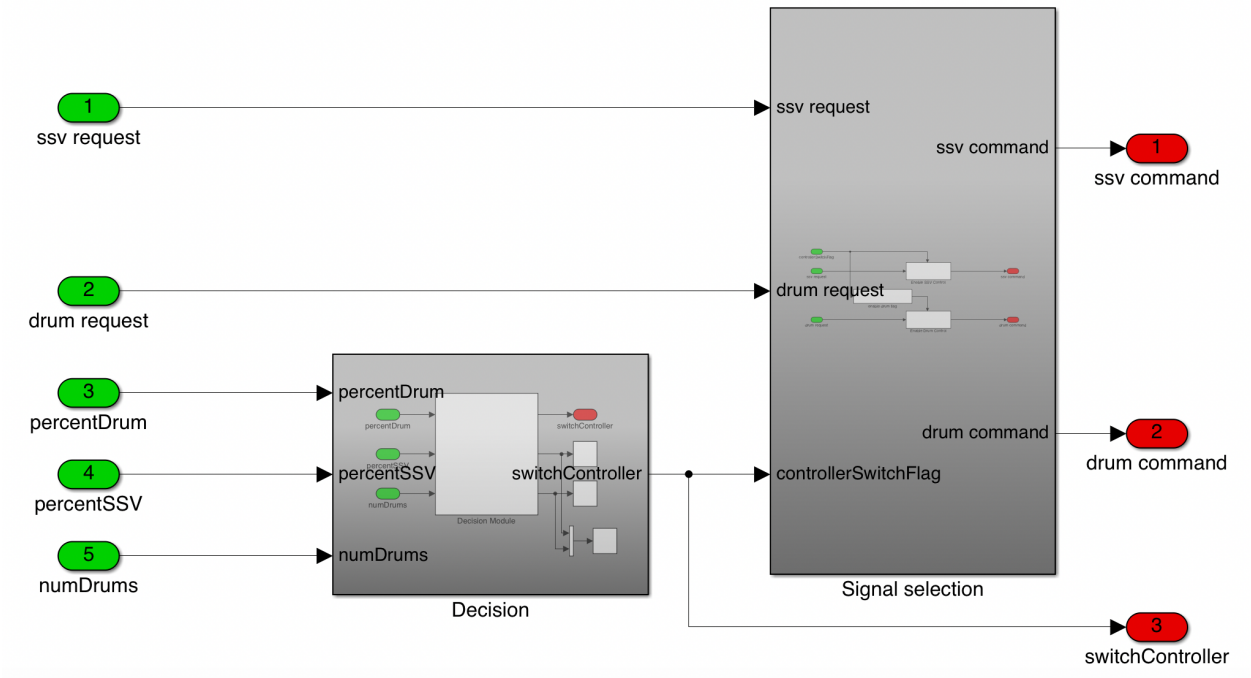

Figure 4.6: The decision signal and selection block from figure 4.2 contains the decision algorithms along with machinery for selecting the signals chosen by the decision block.

4.4 Applied Decision Approaches

While evaluating decision algorithms for this research, three primary features were identified to be the target attributes for defining the first approach. Since this is the first attempt to apply decision algorithms to NTR engine control, simplicity was specified as the primary desirable attribute to ensure a feasible approach. Second, deterministic behavior is expected to be desired for rocket engine applications, so applied decision algorithms should at least be compatible with deterministic frameworks. Finally, while determinism may be desirable, it is important that decision methods have the ability to account for uncertainty in the decision itself. From these desired attributes, the target for this work was chosen to be a risk informed approach bound by a deterministic framework, with a focus on developing the core functionality of the uncertainty based decision approach. For the sake of simplicity, uncertainty will be taken into account via single attribute utility theory, which is compatible with various deterministic frameworks. Additionally, a simple deterministic rule based decision approach was also implemented for comparison.

The integrated control architecture presented in section 4.3 allows the application of multiple decision algorithms to NTR engine control to be studied by embedding them in standalone functions, which are made accessible to the model decision architecture via decision mode selector embedded in the decision block. The user can choose the desired decision algorithm to be executed during runtime by selecting it from a dropdown menu, shown in figure 4.7, provided by double clicking on the top level engine control block in figure 4.3. Figure 4.8 demonstrates how to add new decision approaches to the dropdown list by adding a label to the "`decisionAlgorithm`" mask parameter type options. This dialog is accessed by right clicking the engine control block and selecting "Mask"→"Edit Mask" from the menu. The new decision algorithm can be made accessible to the model by adding a corresponding switch case to the the "`decisionModeSelector.m`" file in Appendix C. Any additional necessary inputs will need to be made available to the selector for passing to the new algorithm by including input ports in Simulink, and updating the decision mode selector function signatures in the embedded block function call and the standalone file.

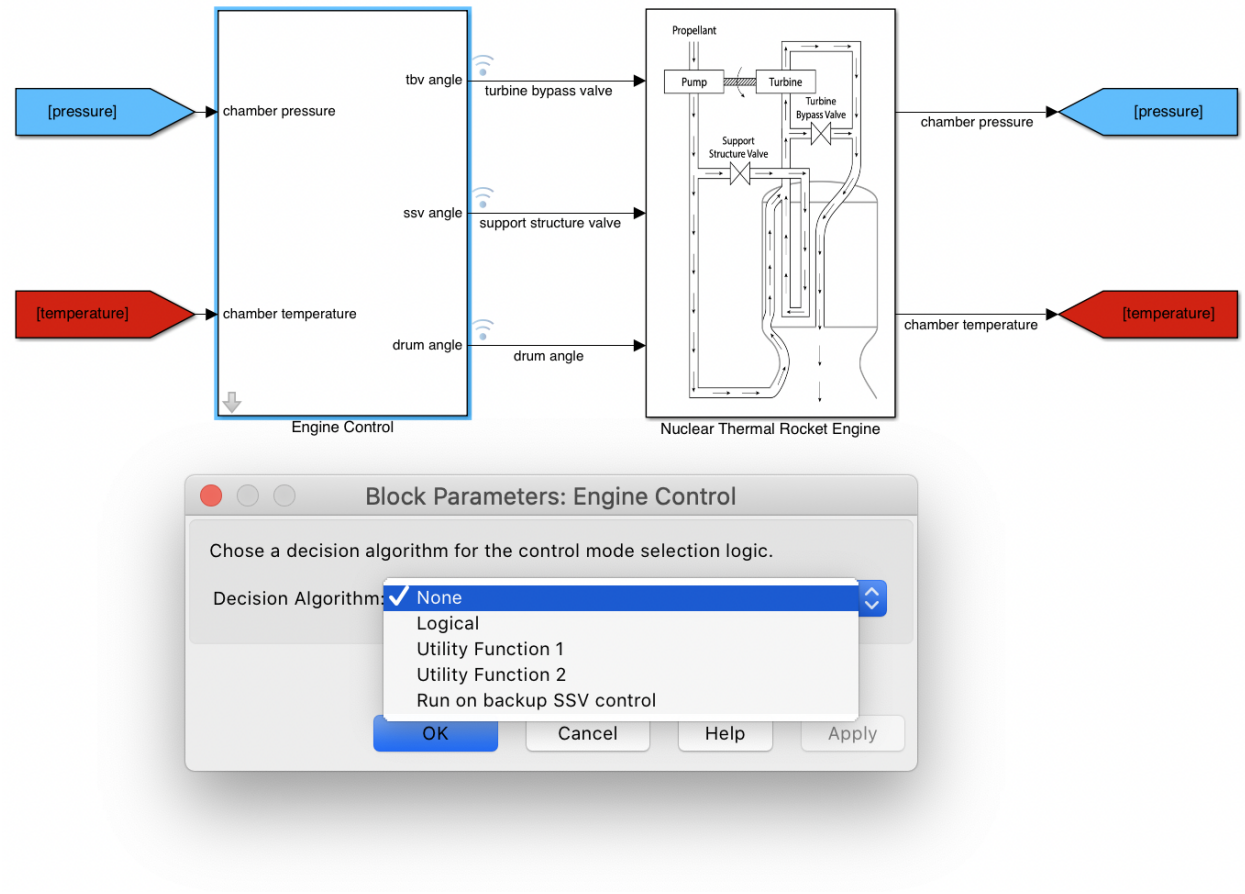

Figure 4.7: The decision mode for control drum fault accommodation can be user selected by double clicking the engine control block at the top level of the model.

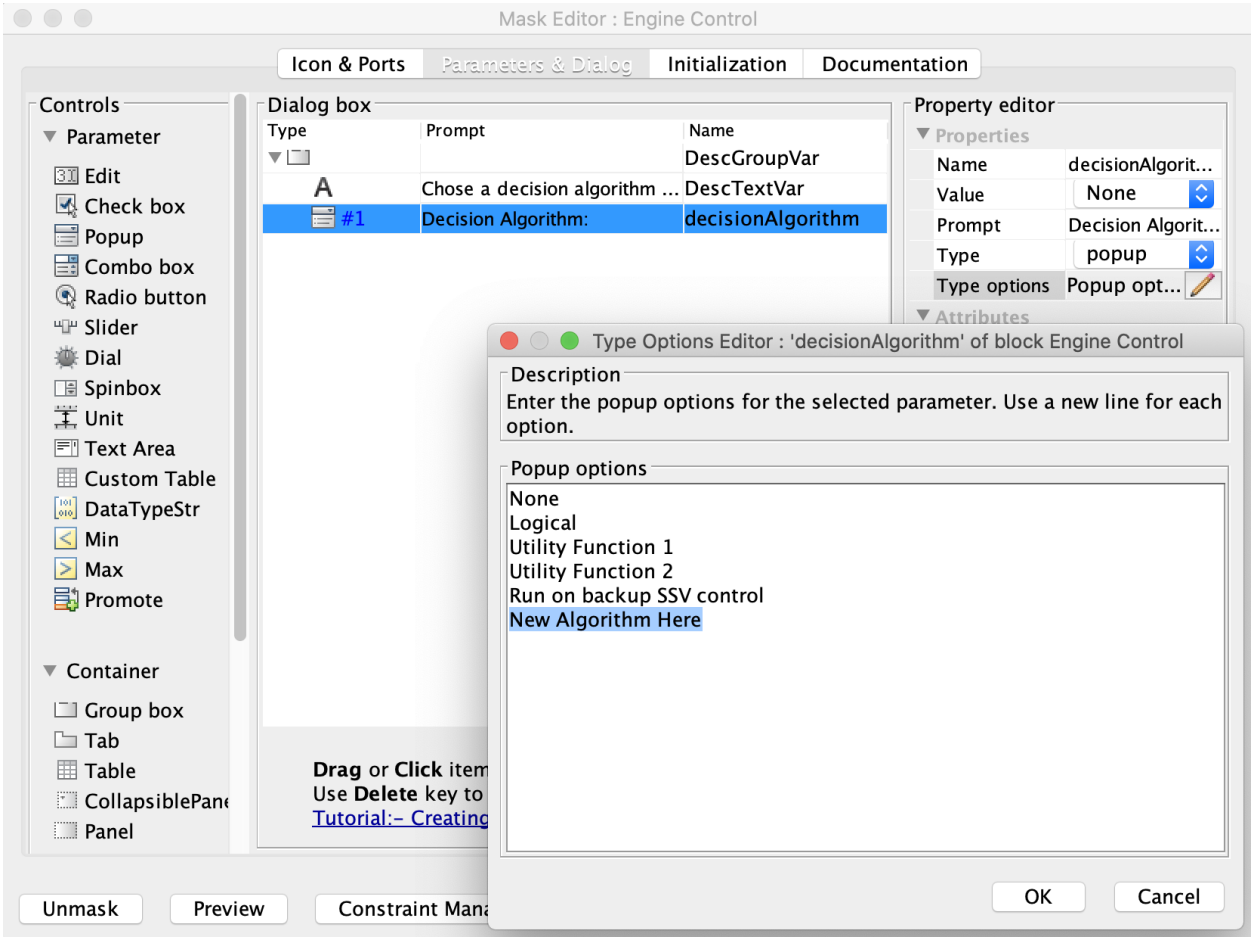

Figure 4.8: New algorithm choices can be added to the dropdown menu in figure 4.7 by editing the **decisionAlgorithm** parameter's Type options in the Engine Control block mask editor.

4.4.1 Logical Decision

With the full research platform complete, including the integrated control architecture, the next step in this investigation was to implement a simple logical decision approach. This method, labeled 'Logical' in the decision selector drop-down menu in figure 4.7, was implemented first to help gain a sense of limiting factors with minimal reliance on formal decision theories. It also serves as a baseline for comparison with additional algorithms.

The logical decision approach is a simple implementation with the decision based solely on the perceived number of stuck drums. The rationale behind this method is that, awareness of drum failure suggests that the primary reactivity insertion mechanism is compromised. Naturally, one conclusion to be drawn from this information is that the system controllability is at risk. With this information, the decision can be made to switch to the backup control in hopes of preserving, or regaining control over the engine reactivity, and thus nozzle chamber temperature. To determine the switching criteria, a study was performed to evaluate engine thrust buildup performance with failed control drums. This was done by introducing the full range of possible number of drum faults at 0, 5, 10, 15, 20, and 25 s into the thrust buildup maneuvers, shown in figures 4.9, 4.10, 4.11, 4.12, 4.13, and 4.14, respectively. The study demonstrates the degradation of the engine performance under stuck drum conditions at almost any point during the thrust buildup phase of operation. The runs which deviated significantly from the demanded trajectory are highlighted with legend entries. It is immediately evident that the degree of degraded functionality is associated with the time the drum faults occur during thrust buildup. For example, figure 4.10 shows that there is some minimal degradation in engine performance introduced for every additional control drum failure, with a significant decrease in ability to maintain demanded performance observed after more than 9 control drums are stuck. This significant departure from demanded performance is shown to be reduced in figures 4.11, 4.12, 4.13, and 4.14 as the drum faults are introduced later into the thrust buildup maneuver. However, figure 4.9 shows that this significant degradation appears with the fewest number of drums stuck at 0 s. This is because the failed drums contribute no reactivity in this scenario.

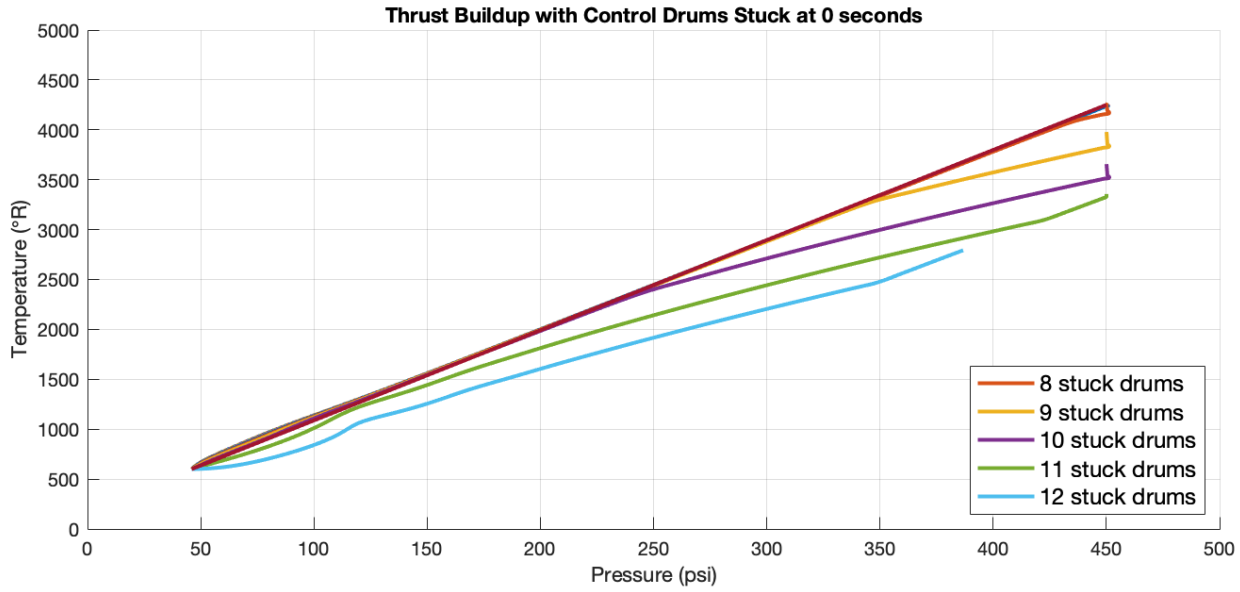

Figure 4.9: Engine thrust buildup map with full range of drum faults injected at 0 s into each run. Maneuvers with 8 to 12 drum failures are shown to exhibit significant deviation from demanded conditions.

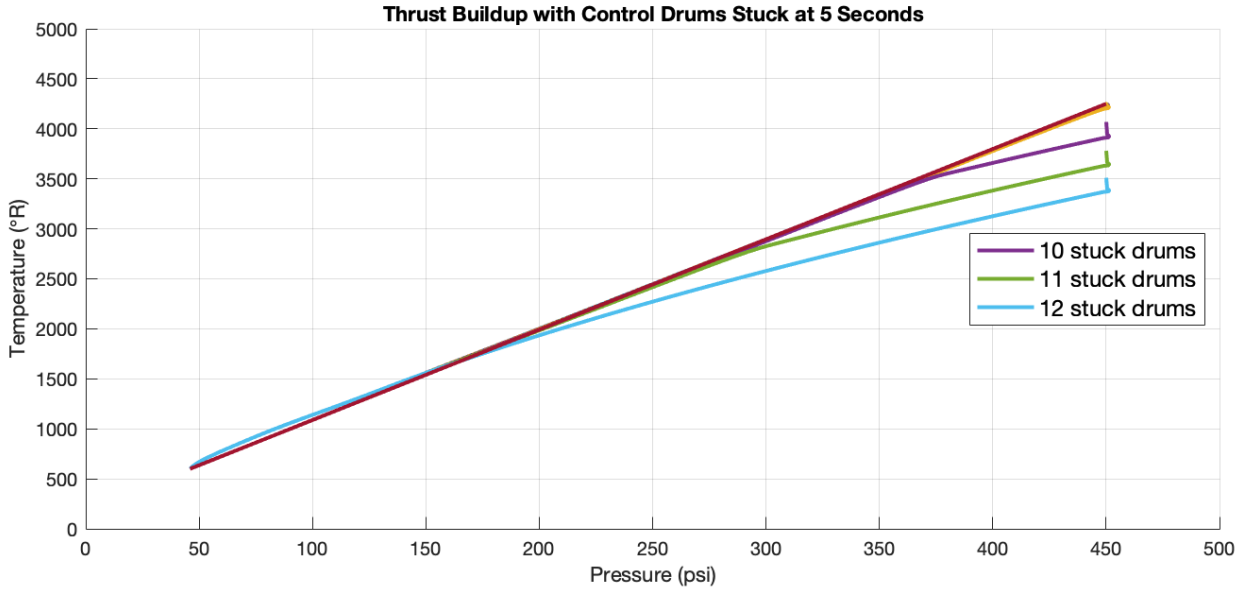

Figure 4.10: Engine thrust buildup map with full range of drum faults injected at 5 s into each run. Maneuvers with 10 to 12 drum failures are shown to exhibit significant deviation from demanded conditions.

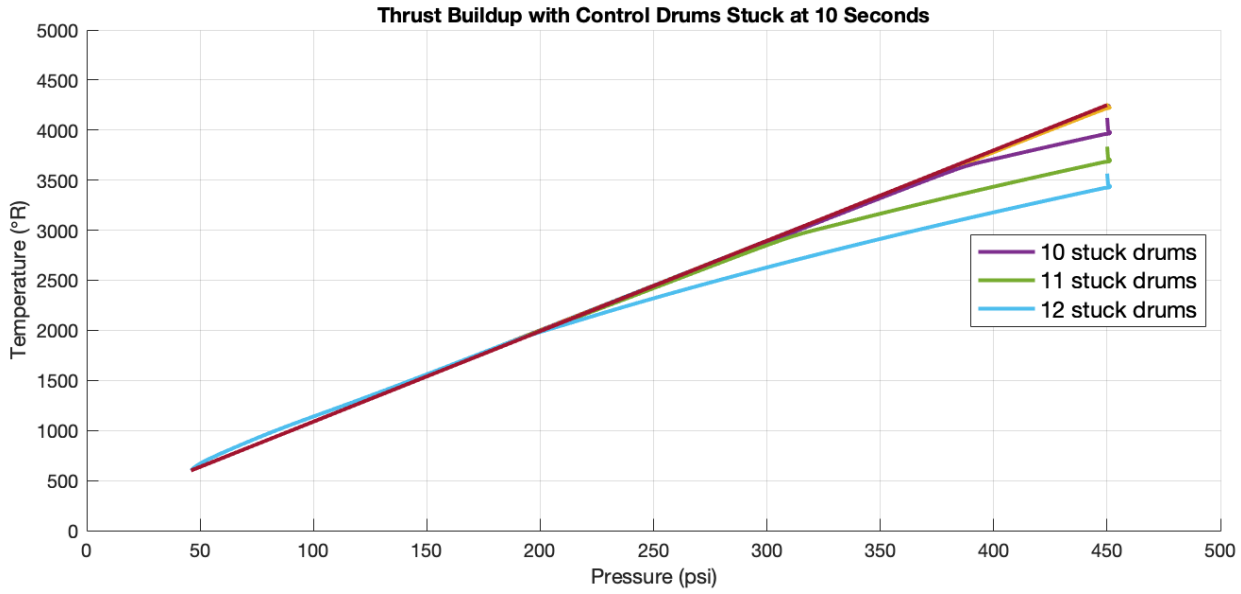

Figure 4.11: Engine thrust buildup map with full range of drum faults injected at 10 s into each run. Maneuvers with 10 to 12 drum failures are shown to exhibit significant deviation from demanded conditions.

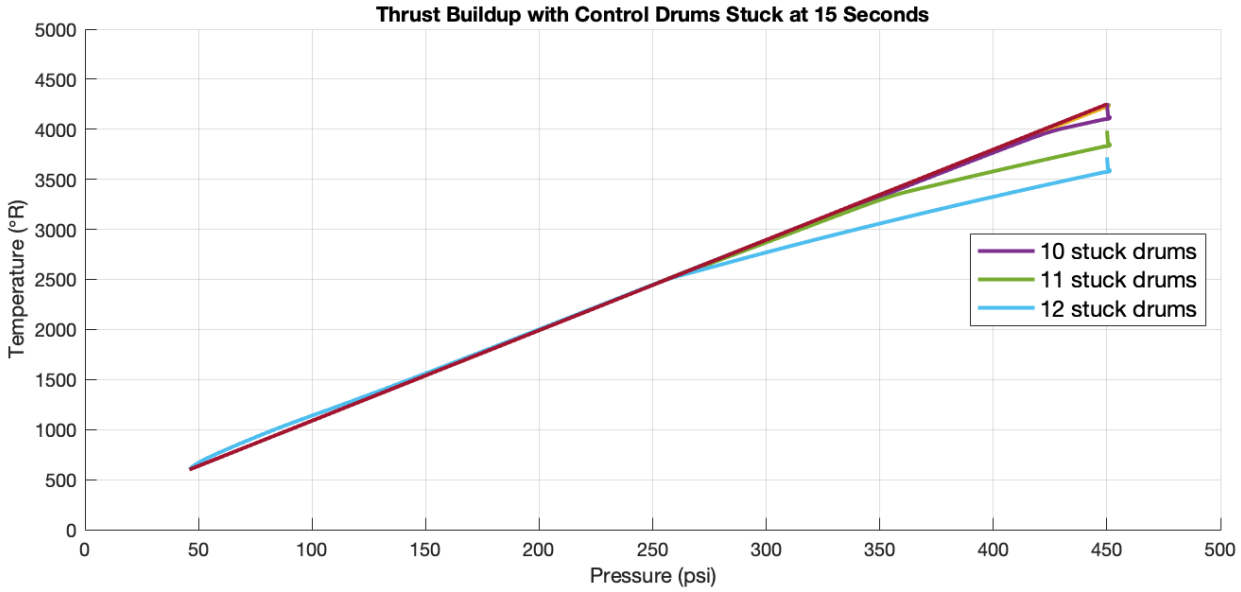

Figure 4.12: Engine thrust buildup map with full range of drum faults injected at 15 s into each run. Maneuvers with 10 to 12 drum failures are shown to exhibit significant deviation from demanded conditions.

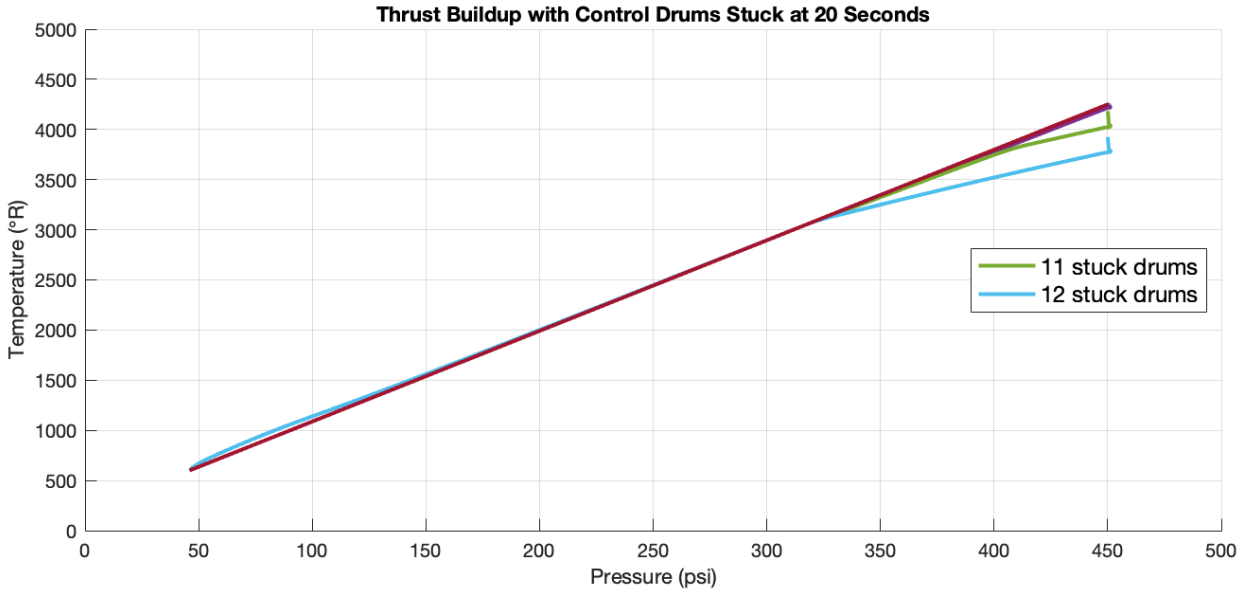

Figure 4.13: Engine thrust buildup map with full range of drum faults injected at 20 s into each run. Maneuvers with 11 and 12 drum failures are shown to exhibit significant deviation from demanded conditions.

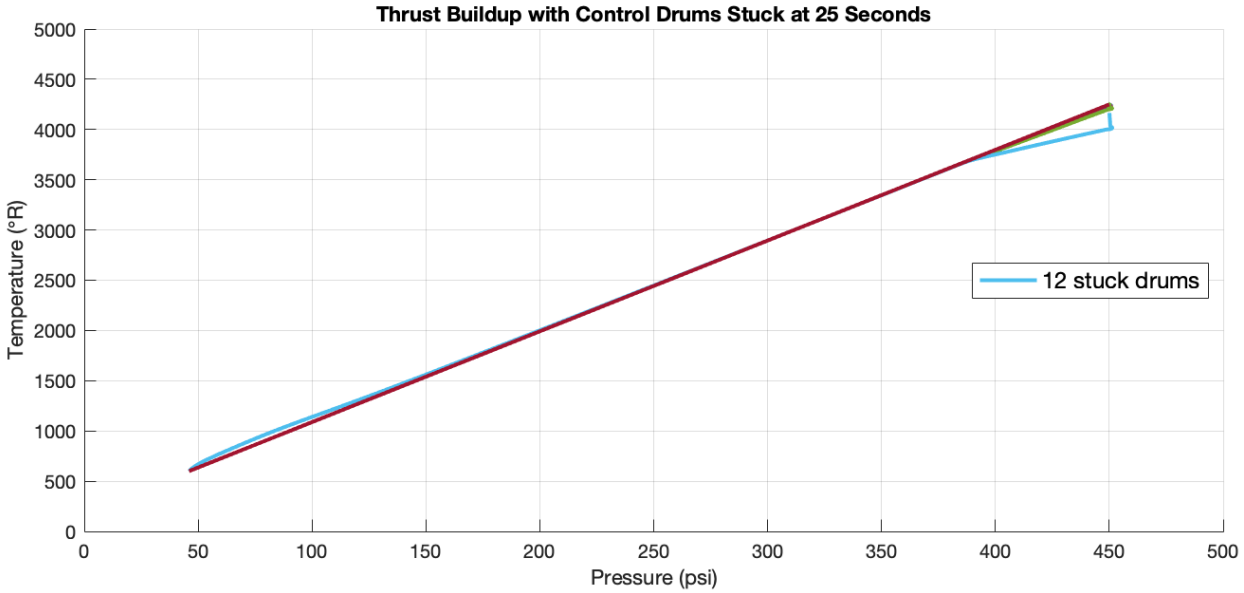

Figure 4.14: Engine thrust buildup map with full range of drum faults injected at 25 s into each run. Only the maneuver with 12 drum failures is shown to exhibit significant deviation from demanded conditions.

The engine performance can generally be maintained with minimal impact due to the remaining functional drums compensating for the reduced available reactivity with increased rotation. However, the available drums are ultimately limited by a maximum actuation range of 180°. When faults occur earlier in the thrust buildup maneuver, the stuck drums contribute less reactivity, and the available drums are required to rotate further to achieve the same compensation effect, thus approaching this limit sooner. This is demonstrated in figure 4.15 by comparing drum positions from fault scenarios where 8 failed drums were introduced at different times during the thrust buildup. Figure 4.15 shows that the drums stuck 5 s into thrust buildup are rotated 40°, versus near 0°for the drums failed at 0 s into the maneuver.

With these considerations in mind, the logical decision threshold was chosen to switch control approaches if more than 6 control drums are considered to have failed. This number was selected to prevent significantly degraded engine behavior, even in the worst case scenario of failure at the beginning of thrust buildup, while including a buffer to prevent the control drums actuators from approaching their rotation limits. This implementation was demonstrated on the same representative scenarios which were previously chosen to demonstrate the impact of fault timing on the behavior of remaining operable control drums in figure 4.15. Figure 4.16 shows the engine behavior and the demanded trajectory during thrust buildup with 8 drum faults at 5 s into the maneuver. This demonstrates that the engine performance was maintained without disturbance during thrust buildup. The actuator responses which achieved this effect once the decision was made to switch controller are shown in figure 4.17. The drum angles behave normally during the initial seconds of thrust buildup, ramping up quickly to insert enough reactivity at low power. Once the faults were injected, 8 of the drums were stuck, which triggered the decision logic to switch controllers. At this moment, the rest of the control drums were maintained at their last positions so that the SSV controller could take over. The SSV control takeover initiated about 5 s after the thrust buildup began at 30 s into the simulation. The remainder of the thrust buildup was then successfully maintained by the SSV and TBV.

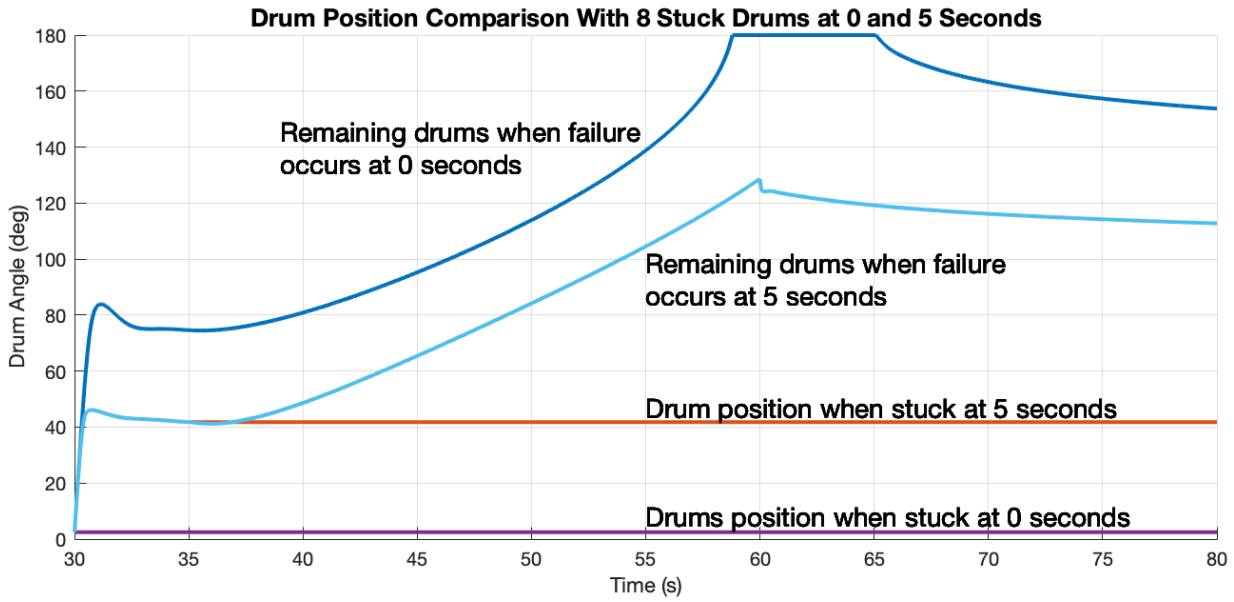

Figure 4.15: A comparison of drum positions when failure occurs at 0 and 5 s, showing the contribution of failed drums to be significant when failure occurs further into thrust buildup.

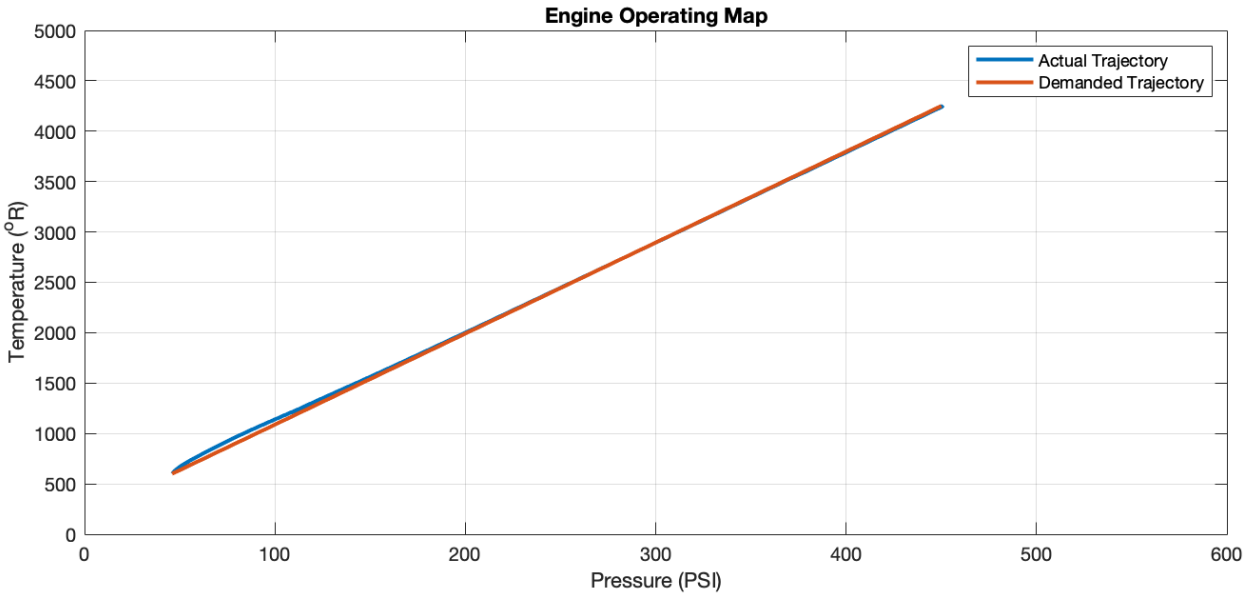

Figure 4.16: Engine operating map with actual and demanded trajectories demonstrating maintained engine performance with 8 stuck drums at 5 s into thrust buildup maneuver.

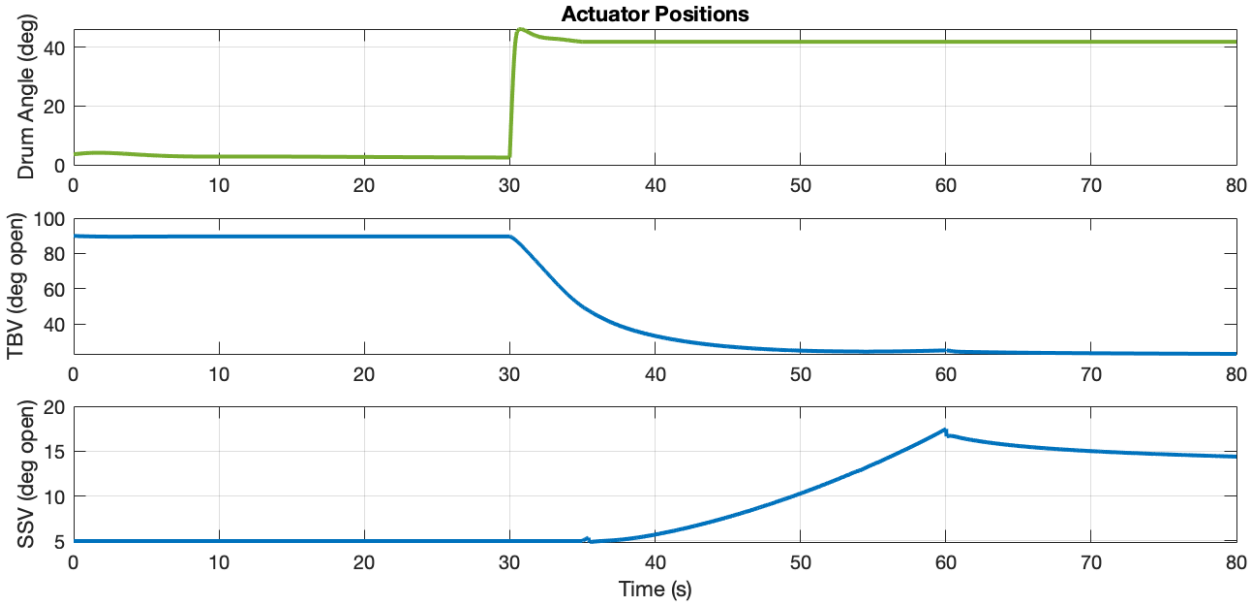

Figure 4.17: Control drum angles, turbine bypass valve angle, and support structure valve angle during idle, with thrust buildup initiated at 30 s, and 8 drums stuck at 5 s into the maneuver.

Figure 4.18 shows the engine performance during thrust buildup with eight drum faults introduced at zero s into the maneuver, along with the demanded trajectory. This figure shows that engine performance is quite similar to figure 2.16 because the entire maneuver was performed using only SSV control, due to the fault occurring at zero seconds into the maneuver. Arguably, the performance during this scenario is worse than if drum control were allowed to remain until later in the thrust buildup, when trajectory deviation due to lack of drum input is expected. This suggests that timing may be an important factor for the switch decision logic. Figure 4.19 shows that the drums had turned less than half of a degree before the fault occurred, and were paused in place. Notably, the SSV angle exhibits a sharp initial response with overshoot and some minor oscillation. It is speculated this may be cause for the notable disturbance in the TBV angle response as well. Additionally the SSV is briefly commanded to close before being commanded to open in response to the demands. While this kind of blip may be smoothed out by actuator dynamics, it is not desirable and the SSV control system behavior may need to be further investigated for early thrust buildup switching scenarios.

This study demonstrates that it is possible to manage drum faults during thrust buildup using simple logical switching to adjust the control approach to an available backup method. However, there are some immediate drawbacks to be considered. The most evident problem with this current approach is that there are scenarios where the logic will switch controllers when it is not yet desirable. Of course this can be remedied by addressing specific scenarios on a case by case basis. This approach is extensible in that any particular effect can be addressed individually with additional rules. Specifically for this implementation, time during thrust buildup has been identified as an important factor in the decision to switch. However, this limits the decision system to the specific cases explicitly addressed by the software. As the list of desired specific scenarios grows, manually accounting for all possible conditions makes complex considerations cumbersome, and it becomes more difficult to ensure that the desired behavior is achieved in all circumstances. Furthermore, the decision approach demonstrated here has only been developed for one alternative choice.

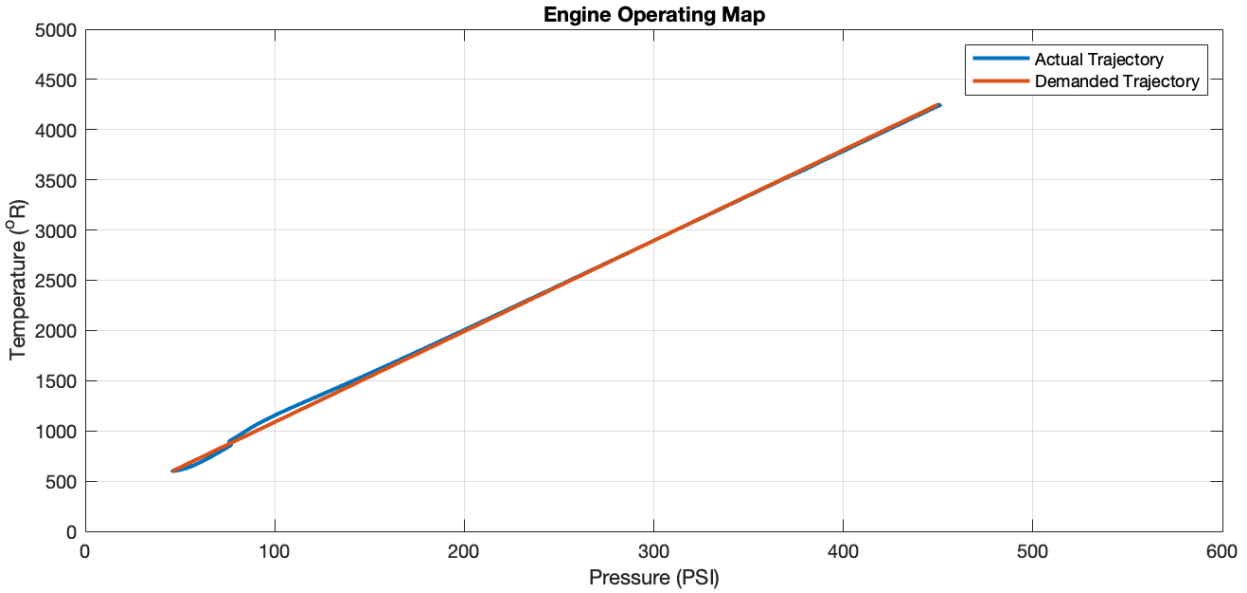

Figure 4.18: Engine operating map with actual and demanded trajectories demonstrating degraded engine performance with 8 stuck drums at 0 s into thrust buildup maneuver.

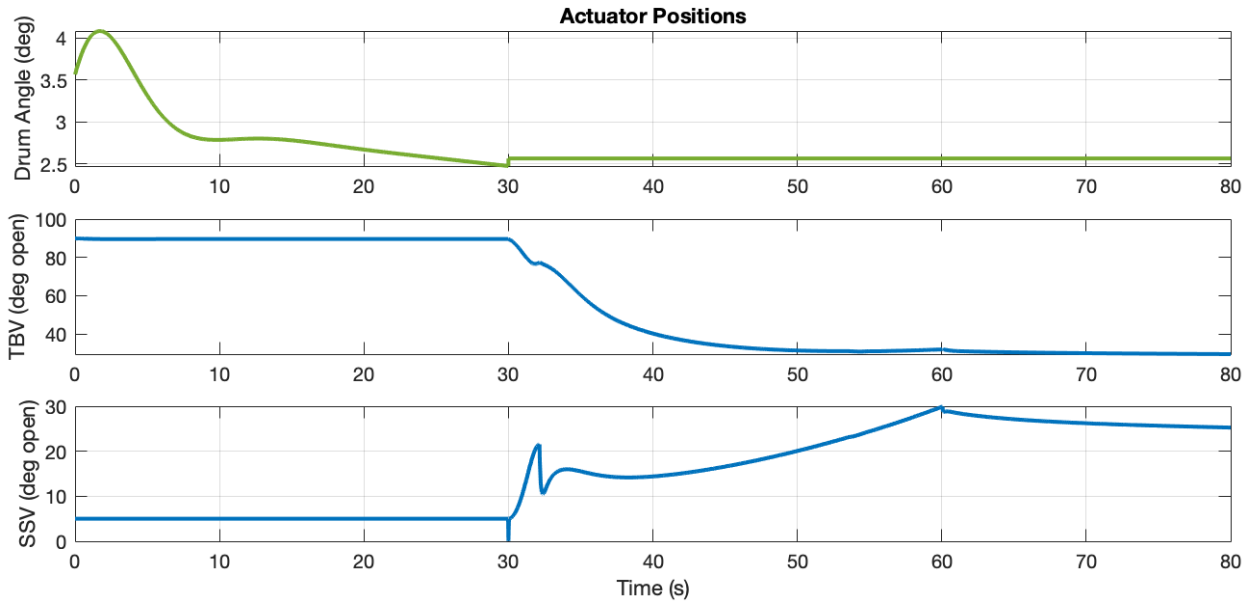

Figure 4.19: Control drum angles, turbine bypass valve angle, and support structure valve angle during idle, with thrust buildup initiated at 30 s, and 8 drums stuck at 0 s into the maneuver.

Extending this method for additional choices in the future would require modification of every single implemented rule for each additional choice option, or an additional decision implementation to choose an alternate option once the decision to switch has been made. While this is possible, it is a severely limiting factor with regard to the extensibility and simplicity of this approach.

To summarize, this implementation was particularly useful in understanding the impact of fault timing on the decision to switch to backup controllers during thrust buildup. It is possible to use this approach for the fault scenario of interest, but a more nuanced decision based on additional considerations would likely be desirable. Improving this approach for use as the sole decision method for control drum fault accommodation requires additional rules to be developed based on any new variable of interest. This is straight forward for adding one or two more cases, but this method is not suitable for particularly complex decisions. The complexity of the approach is further compounded if additional choice options are to be added in the future. A more modular approach is much more desirable in order to serve as a research platform which can be readily extended to accept additional backup choice options for every given decision as they become available.

Therefore, the logical decision method presented in this section is not considered to be desirable for application as a primary decision algorithm for for accommodating NTR engine control drum faults during the thrust buildup operating regime. However, this kind of approach is considered much more useful for more rigidly defined situations where decisions can be made to satisfy clearly defined requirements, such as enforcement of high level rules and regulations on final decisions. Further development as a sole decision algorithm is not recommended, but could be pursued as future work in order to investigate framework implementations for defining the final deterministic behavior of an engine control system. Further investigation of control system decision algorithms requires the study and application of approaches which are more accommodating to diverse combinations of scenarios and choices, as well as to the needs of future researchers.

4.4.2 Utility Theory

Utility theory is a highly developed subject which has been studied and applied across a wide range of fields [23]. This approach was chosen as the first application of a formal decision theory to NTR engine control fault accommodation because it satisfies the primary desirable attributes laid out in section 4.4. Most importantly, it is a simple theory which was expected to translate to simple implementation, with the built-in capability to be extended as desired. This enables the current implementation to continue to be used for research beyond this initial investigation. Additionally, a decision approach based on utility theory can be formulated with uncertainty, and can be easily embedded within a deterministic rule based framework.

More specifically, utility theory is a mathematical framework for describing relationships between choices, and preferences or sentiments about the outcomes of those choices [23]. This relationship is defined by a utility function, allowing a group of alternatives to be evaluated and compared on a numerical basis called utility. For a given scenario, a utility value is calculated for each of the available choices, based on their expected outcomes. The alternative with the highest utility is the preferred choice, on the basis of the definition of rational choice. The outcomes are defined by properties of the system called attributes, which can be measured or derived. These attributes are often a measure of a desirable quantity which is affected by the decision at hand. For example, in a system where the decision maker wants to maximize monetary gain, a primary attribute of interest would be in units of currency, such as dollars. If one action is known to result in a 100 dollar gain, and another action results in 50 dollars, then it may seem trivial to choose the action which nets the greatest profit. However the decision becomes more difficult when there are risks associated with the choice options. For example, there may be a risk that the first option could instead result in zero dollars, and the second choice guarantees 50 dollars. Utility functions allow this risk to be taken into account by defining a relationship between utility and the attribute values, based on our sentiments of the risks involved. Thus, the first step

in the process to develop a utility based decision approach is to choose the attributes which best represent the system outcomes, because they will become the basis of the decision.

4.4.2.1 Attribute Selection

This first implementation of utility based decision applied to NTR engine fault accommodation is to be based on a single attribute approach to serve as a foundation for further research with minimal initial complexity. This framework can be used to investigate additional standalone attributes on their own, while extension of this method is feasible via multi-attribute utility functions [24][25][26][27]. This is encouraged for evaluating fault accommodation decisions with more complex and nuanced considerations.

When evaluating attributes for use in the utility approach, it was initially considered particularly important to identify parameters which are most representative of the engine performance, because that is what we are trying to preserve with a fault accommodation feature. To that end, deviation of nozzle chamber temperature from demanded conditions was the first parameter identified to be valuable as an attribute for use in the utility function because it is directly affected by the actuators in question. In addition, deviation of the other primary control variable, chamber pressure, was considered as a secondary performance indicating attribute. While it is not controlled by the control drums, there is a direct relationship between chamber temperature and pressure, which suggests that it is possible to derive some benefit from including pressure in the decision algorithm. Thus, the deviation of the primary control variables from demanded conditions were the first parameters considered to for evaluation as attributes in the utility based decision approach. Of course it is also possible that additional system parameters may also be useful in the decision making process. However, one parameter needed to be selected for the single attribute approach.

The natural choice from the initial parameter selection would be nozzle chamber temperature deviation, due to the direct connection to drum and engine performance. However, some brief consideration reveals that this is counter to the original intention to maintain simplicity in this approach. This decision approach requires a utility value to be calculated for all

outcomes in the decision space simultaneously for on-line comparison. The nozzle chamber conditions are available as direct system measurements, but only for the active configuration. Under normal conditions, these measurements are unavailable for the backup controller, and vice versa, and thus cannot be used to generate a utility value for both choice options at the same time. The solution to this issue could be a model based approach, where additional faster than real-time models are embedded to generate predicted parameter values for each of the control options for comparison. This is possible to implement as separate modules in the current framework, but this level of complexity is undesirable at this stage in the investigation. Thus it was determined that attributes which could be measured or calculated for all choice options concurrently without significant additional complexity are most desirable to maintain simplicity for this approach.

With these constraints in mind, it was necessary to identify useful parameters which are accessible regardless of the current engine configuration. Without model based predictions, there do not appear to be many simultaneously accessible attributes shared between choice options for the drum failure fault scenario. However, referring back to the decision criterion from the logical decision method in section 4.4.1, it may be possible to ascertain some information about the level of actuator availability for both the control drums and the backup actuator SSV at the same time. The position measurements from both of these actuators are available regardless of control configuration. Since absolute positions are not comparable, the approach taken was to determine the percentage of either actuator which is available to contribute to the reactivity beyond engine idle conditions. This may not necessarily directly reflect engine performance, but it is an indicator which may be used to ascertain the severity of a failure, or to some degree, the health of the system. For the SSV, the availability is calculated as the difference between the maximum angle, $\theta_{SSV_{max}}$, and initial position required to maintain engine idle, $\theta_{SSV_{IC}}$, divided by $\theta_{SSV_{max}}$.

$$\theta_{SSV_{\%avail}} = 100\left(1 - \frac{\theta_{SSV_{IC}}}{\theta_{SSV_{max}}}\right) \tag{4.1}$$

Similarly, the drum availability is calculated as a percentage of the maximum contribution of stuck drums, $\theta_{D_{stuck}}$, plus unaffected drums, $\theta_{D_{maxUn}}$, relative to the maximum total drum rotation $\theta_{D_{max}}$.

$$\theta_{D_{\%avail}} = 100\left(\frac{\theta_{D_{maxUn}} + \theta_{D_{stuck}}}{\theta_{D_{max}}}\right) \tag{4.2}$$

This accounts for reactivity which is still being contributed by stuck drums, beyond idle conditions, and the availability of healthy drums to rotate freely. Unlike the drum availability calculation, there is currently no consideration for the health of the SSV actuator or the likelihood it may be stuck upon being commanded to move, but if it becomes available, it is likely a desirable consideration to be included in future actuator availability calculations. The current calculations are simple and extensible, and are located in the "`attribute_percentActuator.m`" file provided in Appendix D. The "`attribute_`" file naming convention followed here is recommended as a best practice for code maintenance and consistency when extending this research with additional attributes.

With the utility approaches' defining attribute determined and implemented for both actuators, the next step was to develop and implement a single attribute utility function for defining the decision behavior of the utility based fault accommodation system.

4.4.2.2 Utility Functions

As discussed in section 4.4.2, utility is a concept for quantifying the value of situations to a decision maker, so that they can be compared numerically. A utility function is used to calculate a utility value, u_i, for the expected outcomes of each available choice, c_i, in the decision space, C. The utility function is designed to be a representation of beliefs or sentiments toward outcomes of an action, and is based on one or more attributes of the system in question. For this application, the utility function will be based on a single

attribute, actuator availability, denoted a_i in equation 4.3.

$$u_i(c_i) = f(a_i) \tag{4.3}$$

This relationship is defined by evaluating the decision maker's sentiments on outcomes and risks in terms of the chosen attributes. An introductory description of this process is provided by North [28], and a deeper more rigorous treatment with emphasis on multi-attribute approaches provided by Keeney [24]. First, endpoints of the utility function must be established by determining the maximum and minimum values of the attribute, and assigning the appropriate utility values. For actuator availability, 100% is the maximum possible value, and is the most desirable state of an actuator. Thus, 100% is assigned the maximum utility value of 1. Likewise, 0% is the minimum possible value of actuator availability, and is assigned the minimum utility of 0. Next, the general behavior of the rest of utility function can be defined by assessing risk preferences to gain and loss scenarios using lotteries like the one shown in figure 4.20. When the probability of an outcome P is equal to 0.5, the lottery becomes a 50/50 gamble, and can be used to define points which represent the decision maker's indifference to the risk of loss relative to the potential gain in functionality due to the outcome of a choice being made. Figure 4.21 demonstrates this assessment to find the first indifference point for this utility function, at the midpoint between the established sentiment values on the utility axis. This was accomplished in a subjective manner, by determining the value for which the decision maker is indifferent to the outcome of a 50/50 gamble between 0% and 100% actuator availability. In this case, 25% actuator availability was chosen as the point where the decision maker would be indifferent to the outcome. That is, in a scenario where the system has only 25% remaining drum functionality, the decision maker is willing to accept the risk of losing all functionality for a 50% chance of gaining full functionality of a backup system.

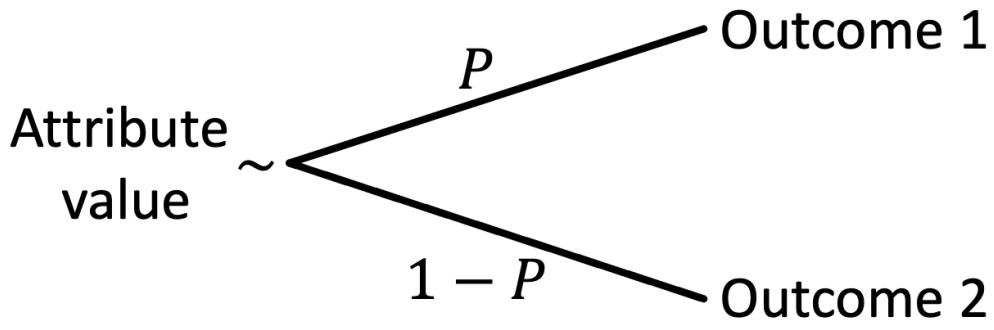

Figure 4.20: Lottery diagram for assessing the value of an attribute where a decision maker would be indifferent to the outcome of an action given probability P and $1 - P$ of the expected outcomes.

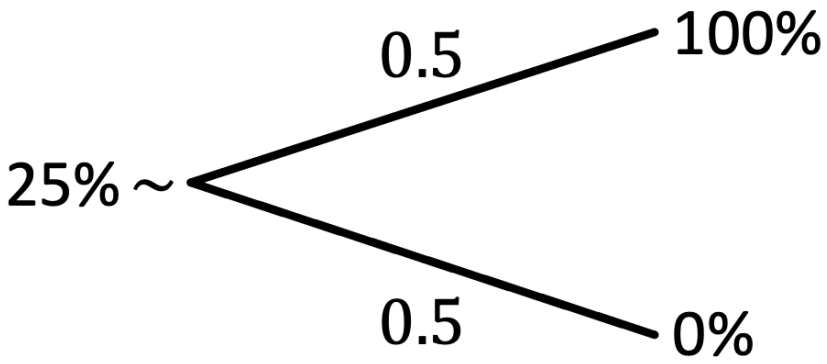

Figure 4.21: Lottery diagram showing that the decision maker is indifferent to the 50/50 chance of receiving either 0% or 100% functionality of a secondary control actuator when the existing control actuator is only capable of 25% capacity.

This point was assigned a utility value of 0.5, and the process was repeated two more times to establish a trend. Using the previous point as the maximum, the next indifference value was determined to be 10% actuator availability for a 50/50 lottery between 0% and 25% actuator availability, shown in figure 4.22, and was assigned a utility of 0.25. The first indifference point was then used as the minimum boundary for the final lottery between 25% and 100% availability, shown in figure 4.23, resulting in a final indifference point at 50% actuator availability at a utility value of 0.75. To generate a utility function, the sentiment data was fit using an exponential function in the form of equation 4.4.

$$u = a + be^{cx} \tag{4.4}$$

The parameters a, b, and c were found by minimizing the euclidean distance between the desired function, and the previously defined sentiment points, using Matlab's $fminsearch()$ function. This was done in the "`genfit.m`" script provided in Appendix E, which can be updated with alternate sentiment data to generate new exponential curve fits. The resulting utility function was implemented in "`utility_01.m`", provided in Appendix F, which takes percent actuator availability as an input, and outputs a utility value. The function itself, labeled 'original utility function', is plotted along with the sentiment data in figure 4.24. The convex shape is referred to as a risk averse utility function, because it prioritizes actions with lower risk outcomes over high returns. A linear utility function is considered neutral, and a concave function is considered risk loving. While this function is obviously convex, an additional utility function was made to be more strongly risk averse. Instead of performing lotteries to develop a new trend, a single sentiment point, in addition to the endpoints, was chosen simply to ensure a strongly risk averse utility function, rather than reflecting true sentiments of a human decision maker. This secondary utility function is plotted alongside the original in figure 4.24 for comparison. The curve fit was done again using "`genfit.m`", and the resulting function was implemented in "`utility_02.m`", provided in Appendix G.

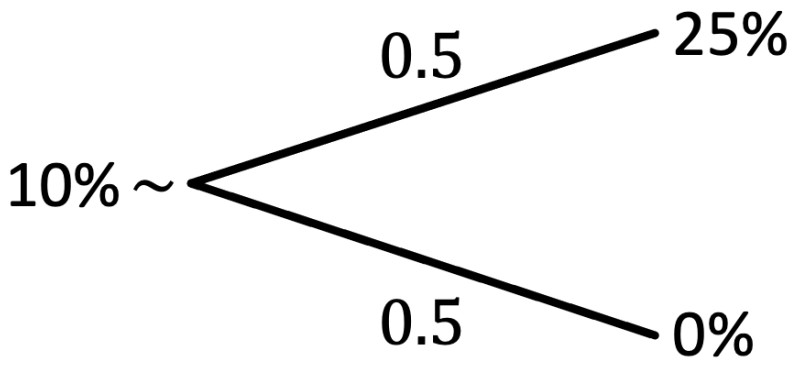

Figure 4.22: Lottery diagram showing that the decision maker is indifferent to the 50/50 chance of receiving either 0% or 25% functionality of a secondary control actuator when the existing control actuator is only capable of 10% capacity.

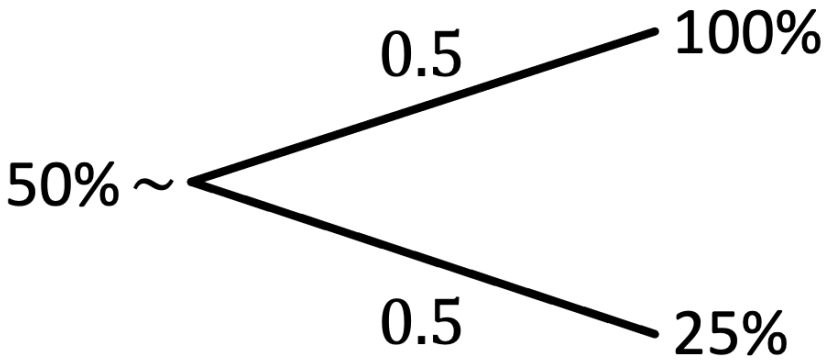

Figure 4.23: Lottery diagram showing that the decision maker is indifferent to the 50/50 chance of receiving either 25% or 100% functionality of a secondary control actuator when the existing control actuator is only capable of 50% capacity.

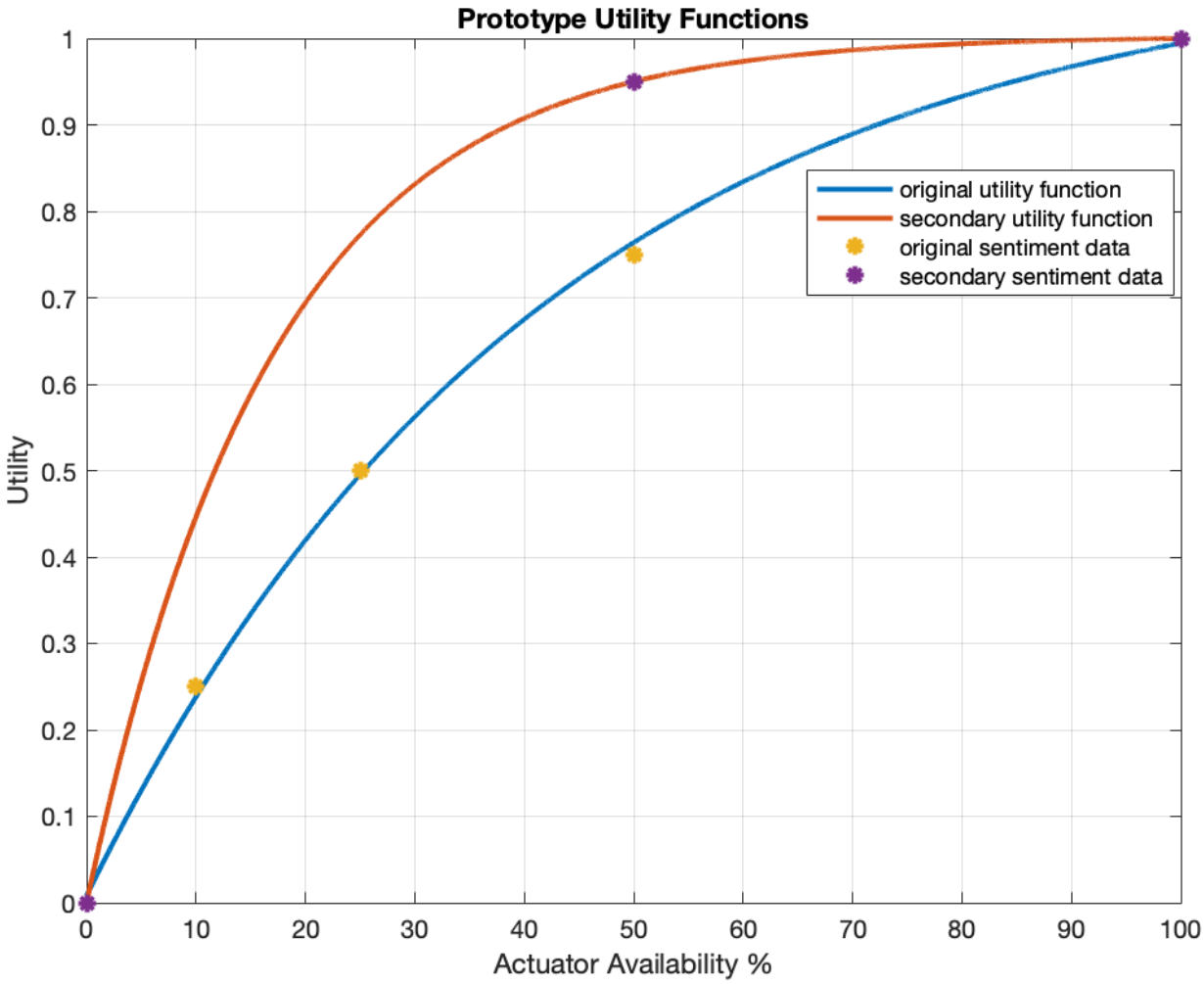

Figure 4.24: Risk averse utility functions for the actuator availability attribute, fit to sentiment data using exponential functions.

This demonstrates the simplicity in generating additional utility functions using the tools developed for this work. Additional research on the application of utility functions will require their implementation, and these tools are intended to enable such work. Additionally, this can serve as an initial comparison on the sensitivity of this system to the shape of the utility function for actuator availability.

4.4.2.3 Utility Based Decision making

The utility functions developed for this work can be used to generate decisions based on the outcomes of the available choices. However, the utility values are not intended to be directly compared. The decision maker's appetite for risk is codified into the utility functions, but the uncertainty of the outcomes were not taken into account. To accomplish this, the decision mechanism was implemented to select the option with greater *expected* utility, using probabilities to reflect the confidence in the expected actuator availability values.

For an available action, the expected utility is calculated by multiplying the utility of each expected outcome of that action by their respective probabilities. Equation 4.5 demonstrates this calculation for an action A which has two possible outcomes represented by their attribute values x and y.

$$U(A) = P(x)u(x) + (1 - P(x))\,u(y) \tag{4.5}$$

For this application, it is assumed that only the measurable outcomes are known, and there is no information available for evaluating additional possible outcomes of a given action. This results in the utility of all other possible outcomes to be zero, reducing equation 4.5 to equation 4.6.

$$U(A) = P(x)u(x) \tag{4.6}$$

This was implemented in "`decision_utility.m`" with simple hard coded values to act as representative probabilities that the measured actuator values are accurate. This utility based decision generator works by calculating a utility value for each of the available choices by taking into account the known possible outcomes and their probabilities as described by equation 4.6, and the option with the greatest expected utility is chosen. This file is provided in Appendix H, and can be edited by the user to perform studies. However, for use in actual decision scenarios, it is recommended to update this implementation with more comprehensive uncertainty buildups, or even online calculations which can give real time assessments of the uncertainty associated with the processes and measurements being modeled.

4.4.2.4 Results

To understand the new behavior produced by this approach, the first step was to determine the decision system's sensitivity to control drum failure. Drum faults were again introduced to the system during the thrust buildup phase at 0, 5, 10, 15, 20, and 25 s into the maneuver. It was quickly discovered that this approach was highly sensitive to drum failures when the probability of the two outcomes are the same, regardless of the utility function used. The system generates a decision to switch to backup control when two or more drum failures are detected at any time during thrust buildup. Furthermore, the decision to switch would even be triggered by a single drum fault if it occurred before 21.2 s into thrust buildup. That means the system will only maintain primary control if one drum failure is detected on or after 21.2 s into thrust buildup. This sensitivity was found to be due to the nature of the actuator availability attribute itself. Looking at the actuator availability calculations from equations 4.1 and 4.2, under normal conditions, the SSV and drum availabilities are 94.4%, and 100% respectively. This means that a single drum loss at zero seconds into thrust buildup results in drum actuator availability of 91.67%, immediately resulting in a controller switch decision. The loss of a drum would have to occur after it had rotated 33.3% of its range, or 60°, in order for the total drum availability to remain above 94.4%. Without consideration for uncertainty in the possible outcomes, this approach strongly favors reacting to almost any sense of disruption of the drum actuators, by switching to the backup SSV

controller. This is generally considered undesirable due to the engine's ability to maintain reasonable performance under degraded conditions early in the thrust buildup maneuver, up to 9 stuck drums in the worst case scenario of failure at 0 s, illustrated by the engine performance study carried out in figure 4.9. This also demonstrated that the same issue with early failures identified for the logical decision method in section 4.4.1 was still present with this method, only more so. Decisions are generated to switch to backup control for scenarios where any number of faults happen before 5 s into thrust buildup, resulting in the same undesirable engine performance and actuator dynamics shown in figures 4.18 and 4.19, regardless of the number of failed drums.

Although it was determined that this approach is far more sensitive to drum failures than desired, it was still necessary to gain some understanding of how uncertainty of outcomes would affect the decisions generated by the system. Extrapolating from the initial study, it is easy to see how any uncertainty in the outcome of maintaining primary control would result in a pure favoring of operation under SSV control. Hence, a study was performed to evaluate how uncertainty in the expected outcome due to switching to backup control affects the decision behavior. To do this, a relationship was identified to describe the point where the probability of outcome would be low enough to affect the decision. This relationship, defined by equation 4.7, was found analytically by setting the expected utility calculations for each choice equal to each other and solving for the probability of expected outcome due to switching to backup SSV control, $P_{\theta_{SSV}}$.

$$P_{\theta_{SSV}} = \frac{u\left(\theta_{D_{\%avail}}\right)}{u\left(\theta_{SSV_{\%avail}}\right)} \tag{4.7}$$

Since the SSV availability is known for normal conditions, this relationship was used to generate decision lines of probability versus time over thrust buildup, using the drum position dynamics for failures from 1 to 12 drums. This was then validated by performing simulation

experiments mirroring the engine performance study from 4.4.1. The full range of drum faults were introduced at 0, 5, 10, 15, 20, and 25 s into thrust buildup, using the SSV outcome probabilities determined using equation 4.7.

The decision lines for the original utility function are plotted in figure 4.25, along with the experimentally generated validation points. These lines highlight how the decisions are affected by the uncertainty of the outcome of a choice, in particular, the choice to switch to backup SSV control. They are another way to represent the risk preferences encoded in the utility functions, but in terms of the model dynamics and expected outcome probabilities. Equation 4.7 provides a mapping of the utility functions to the time domain via control drum dynamics, normalized by the utility of available SSV in nominal conditions, and represent the limit between the available choices. For any given scenario, if the expected outcome probability is greater than the decision line, then the decision will be made to switch to backup control. If the probability is below the decision line, then the algorithm will choose to maintain primary control, up to the specified number of failed drums. As the failure scenarios grow in severity, the decision algorithm becomes less sensitive to uncertainty in the expected outcome of switching to the backup control actuator. The secondary utility function demonstrates this same behavior in figure 4.26, but with much more sensitivity to uncertainty of the outcome. This was expected because the secondary utility function was deliberately designed to be highly risk averse, and therefore much more affected by the perceived presence of risk. Though, due to the nature of the actuator availability attribute formulated for this study, the utility function shape did not have a strong impact on the behavior of the decision system when outcome risks were similar for both choices.

These studies demonstrated that a single attribute utility function approach based on actuator availability is highly sensitive to control drum actuator faults. The current implementation will generate decisions to switch from primary drum control of the nozzle chamber temperature to backup SSV control in undesirable situations. However, it is important to continue the study of embedded decision techniques with considerations for uncertainty as an enabling technology for NTR engine autonomous control.

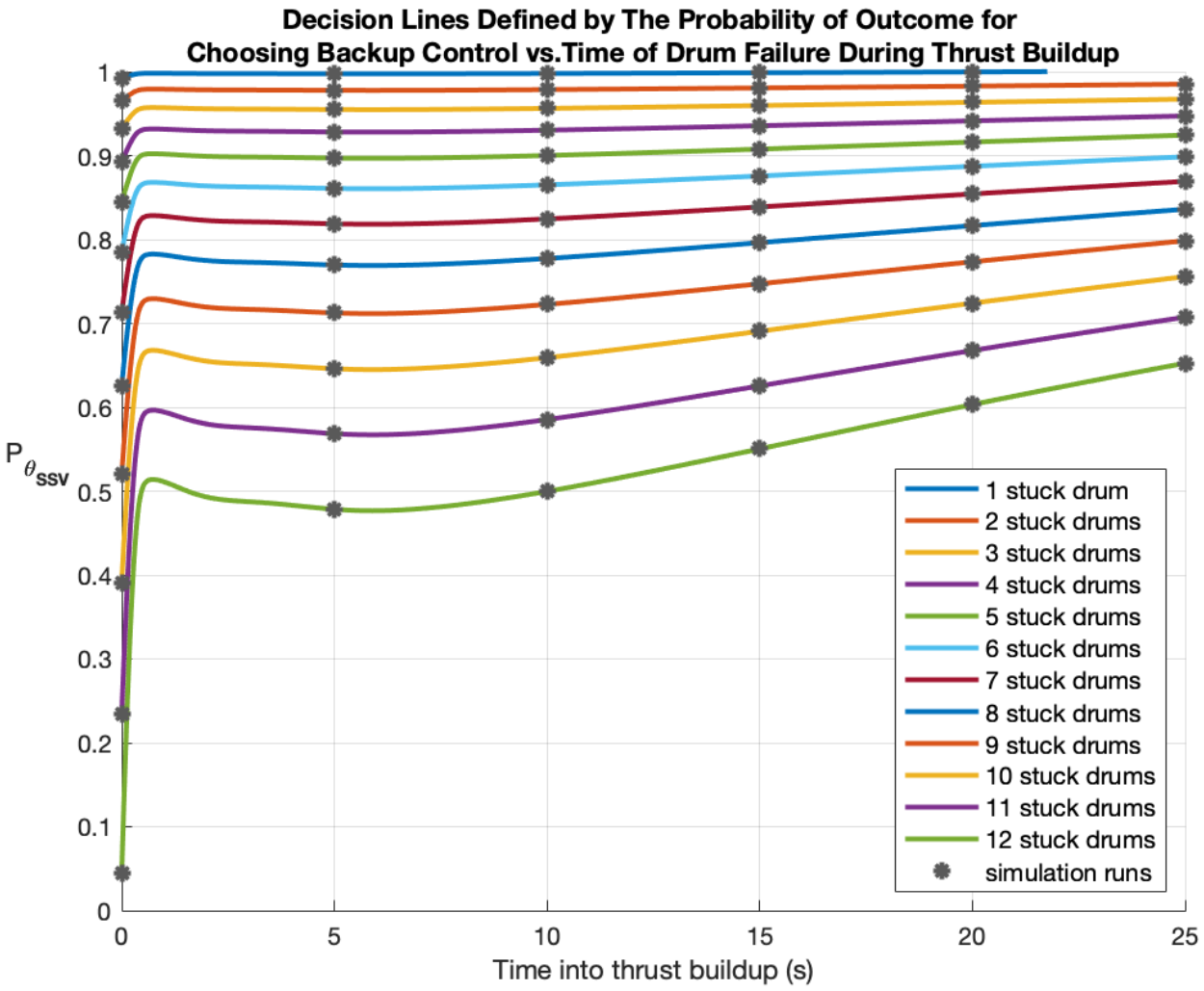

Figure 4.25: The probability of outcome, $P_{\theta_{SSV}}$, versus time over thrust buildup for 1 to 12 drum failures, using the original utility function. The results of simulations performed to confirm the relationship derived in equation 4.7 are provided as grey asterisks.

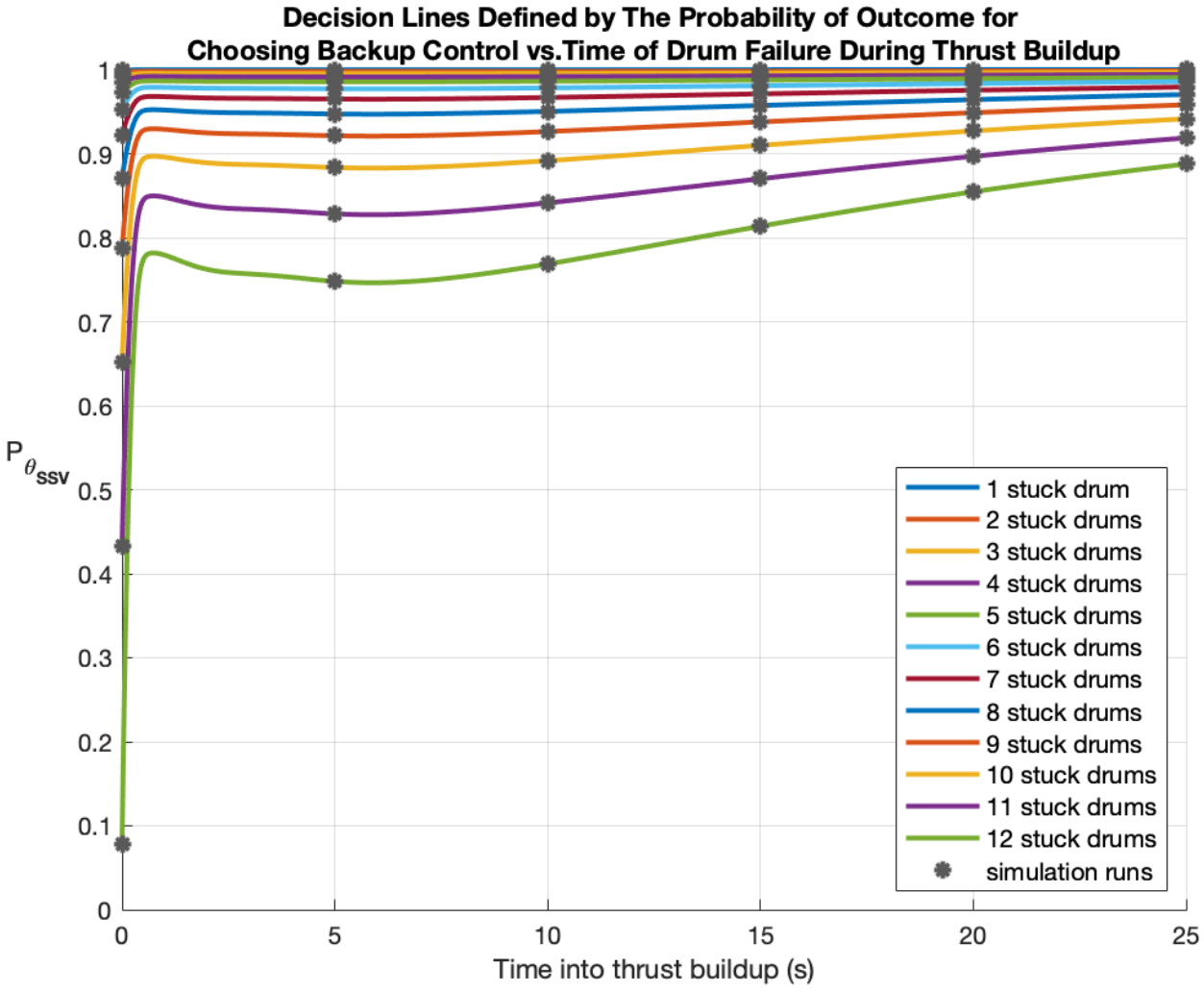

Figure 4.26: The probability of outcome, $P_{\theta_{SSV}}$, versus time over thrust buildup for 1 to 12 drum failures, using the secondary utility function. The results of simulations performed to confirm the relationship derived in equation 4.7 are provided as grey asterisks.

Based on the observations of the behavior of the utility approach presented in this work, attributes which can be better differentiated without significant differences in perceived risk are more desirable for future development. Additionally, there should be further investigation to determine whether this is the primary factor in the premature switching behavior, or if there is some additional time based attribute, or other consideration to be made. To that end, engine performance based attributes such as nozzle chamber temperature and pressure, or their error values, are proposed for further investigation. It is hypothesized that attributes which directly indicate the expected engine performance impacts due to a choice outcome may improve the behavior of the decision algorithm and produce more desirable outcomes more consistent with human decision makers. This is because attributes based on engine performance are expected to eliminate the need for considerations of the timing of the fault during thrust buildup. This will require model based predictions of engine behavior to be available, parallel to the current model, during simulation. Since the current model may be duplicated for use in generating predictions, the added complexity for calculating such attributes will likely be minimal in comparison to the potential for improved decision behavior. A multi-attribute approach with model based engine performance indicating attributes is recommended for further development.

The purpose of this approach was to investigate the implementation and usefulness of an algorithm which could provide decision capabilities under uncertain conditions. The actuator availability attribute was found to produce undesirable behavior, and is not recommended for future development. However, the utility based decision approach itself was considered to be straight forward to design and simple to implement. Furthermore, the tools developed for this work should provide a foundation for future researchers developing embedded decision based fault accommodation features.

Chapter 5

Conclusions

5.1 Summary

The motivation behind this research was to contribute to the investigation of autonomous control for nuclear reactors. Since autonomy is defined as the ability to govern one's own actions, and nuclear reactors are historically dependent on human operators to function, reducing the need for human interaction was identified as a key area of interest for addressing this challenge. To that end, embedded decision was identified as a key capability for introducing some reasoning capacity, otherwise provided by human operators, and thus is a foundational element of autonomous control. To define the scope of work, the approach was taken to address more urgent need applications of autonomous control which may be in development in the near term. Ultimately, due to NASA's active interest in nuclear thermal propulsion, and the strong need for autonomous operation of such space based systems, the decision was made to investigate embedded decision capabilities for fault accommodation in NTR engine control systems.

This required a significant background investigation effort to establish a technical basis from which to continue research efforts. First, a literature review was conducted to establish the state of the art in NTR engine control, and dynamic modeling. This portion of the review provided in chapter 2 resulted in the documentation of the American historical experience with NTR engine control and engine modeling via Project Rover, the NERVA program, and

some follow-on efforts. As a part of this effort, a reduced order model of the NERVA E1 engine, called the simplified nonlinear model, developed by Kendrick [16] was rediscovered. In this work, the SNM was recaptured as a digital model and implemented in Matlab and Simulink to serve as a research and development platform for this and future work. As an additional feature of the research platform, an engine control system was also developed for this model in accordance with conventional engine control approaches developed under project Rover. This allows for research to be conducted on engine control systems and more complete representative behavior of the controlled NERVA E-1 engine. The literature review was continued in chapter 3 to provide a basis for autonomy and applied decision theory as a foundation for autonomous control. This foundation was used to inform the development of a proof of concept application of decision theories to fault accommodation in NTR engine control presented in chapter 4. As a specific research target, the problem to address was then defined to be stuck control drums during thrust buildup. A general architecture for ECS embedded decision research was developed, and the implementation details for the current application were outlined. Finally, two decision approaches were implemented and evaluated on their applicability to the defined failure problem. Of particular significance is the first application of utility based embedded decision to the NTR ECS.

5.2 Findings

The primary goal of this work was to demonstrate embedded decision as a fault accommodation feature as a proof of concept for reducing the reliance on human interaction, and thus increasing the level of autonomy. As a result, this raised the question of how to evaluate the efficacy of the decision methods under consideration. The answer to this question is that there may not be reliable parameters for evaluating the efficacy of the decision method implementation, but rather it is more important to evaluate how well the problem is characterized in the framework of the decision algorithm, such that the outcome is the desired decision behavior. This study resulted in the identification of desirable behavior of a decision system applied to control drum fault accommodation in NTR engine control, and important features which may help achieve those desirable traits.

The engine performance study from section 4.4.1 demonstrated that this engine model exhibits degraded behavior for almost any drum failure condition. However, it also showed that degraded engine performance became more pronounced as a function of time. The result is that engine performance can be best maintained early in the thrust buildup maneuver with minimal impact for most scenarios, because remaining healthy drums can compensate with additional rotation until their rotational limits are reached. When this is compared to sole SSV control in figure 2.16, drum control can provide better early thrust buildup engine performance for up to 9 drum failures before 5 s into the maneuver. Thus, a situation can exist where a fault occurs in the primary control mechanism, yet the backup option would produce undesirable engine performance for some portion of the thrust buildup maneuver. In this situation, the decision logic should be able to determine that maintaining primary control is still preferred over switching to backup. However, the conditional decision logic implemented in section 4.4.1 produces a decision to switch any time the number of failed drums has exceeded the limit. In this study, this was shown to produce desirable engine performance during failure conditions most of the time, except for when enough faults occur too early in the thrust buildup phase. This means that at least for early failures, this logic is not sufficient to produce the most favorable engine performance, and suggests that more factors should be considered to prevent unnecessary controller switching. Within the confines of this decision approach, this behavior can only be remedied via additional logic which addresses the time during thrust build up that the failure occurs. However, there could be other factors at play, and another simple rule may also risk introducing the opposite and equally undesirable outcome of switching controllers too late. Situations such as this one, where overly simplistic and competing rules may generate sensitive and rigid behavior are undesirable and may be remedied with a more nuanced approach to decision making such as utility theory.

The single attribute utility approach presented in section 4.4.2, was found to be even more sensitive to control drum failures than the previous simpler logical decision method. This was determined to be due to the actuator availability attribute, which serves as the decision basis for this implementation. This requires further investigation to establish attributes which can

produce better decision behavior. This approach also cannot account for the time a fault occurs during thrust buildup, but has the capacity to be extended to do so with additional attributes. However, this approach does provide the ability to account for uncertainty in the knowledge of the outcomes of available choices. It was shown that in the event of a significant failure, the decision module would not switch controllers when confidence in the availability of the backup control actuators was low.

To summarize, the primary engine control mode can maintain engine performance with a significant number of stuck drums, and is likely to remain the desired controller, even in failure scenarios, as long as enough drum actuators are functional and not in danger of further malfunction. This means that the two approaches studied in this research are too sensitive to drum actuator failures to generate decisions consistent with how human operators would prefer to operate. Instead, further development of the utility approach using multi-attribute utility theory is suggested to improve decision system behavior. The goal should be to avoid premature switching by better informing the decision algorithm with considerations for engine performance based attributes such as chamber temperature and pressure, or their error values, and attributes for evaluating expected actuator health.

In addition to the findings produced by this study, these efforts ultimately result in a research framework for continuing the study of NTR ECS embedded decision. This includes a collection of tools supporting the engine modeling, control, and decision application development.

5.3 Future Work

The immediate follow-on efforts to continue this work should be focused on completing the recommendations to develop a multi-attribute decision approach based on engine performance attributes. This includes investigation of engine performance attributes, additional attributes, such as a reimagined take on actuator health, and the implications of additional embedded engine models for generating engine performance predictions. For

longer term future developments, the most significant areas of interest are improved engine modeling, and further investigation of decision algorithms applied to NTR engine autonomous control.

The digitally recaptured SNM used in this work provided a simple, computationally efficient representative model for performing the required studies. However, some improved fidelity in certain high need areas are considered necessary for a more complete picture of the engine dynamics. The most important new feature to improve the model behavior would be actuator dynamics for valves and control drums. In addition, modeling failure mechanisms and adding fault detection and prediction would result in a much more complete research platform, which would warrant their own research. There is also a need for more in-depth modeling efforts. Most importantly, engine startup and shutdown dynamics are essential for covering the full scope of engine operation, which was not addressed by any of the dynamic models available in the literature. Additionally, it is unclear whether spatial effects should be considered in future fault accommodation studies, and should be investigated.

Further investigation of applied decision algorithms beyond the scope of this study could be separated into three primary efforts. First, development of the multi-attribute method suggested at the beginning of this section could be continued to further mature the approach. A suggested feature for added functionality would be the ability to include more choice options to diversify the available responses to fault scenarios. Second, research into new decision algorithms which have not been applied to NTR engine control should be performed to gain some sense of the algorithms which best fit the problem at hand. Of particular interest are methods which can perform decisions under uncertain conditions, ranging from well understood Bayesian decision theory [20], to techniques more recently applied to autonomous control such as Markov decision process [20], to the much more recent developments such as quantum decision theory [29]. Finally, further development of NTR engine autonomy can be pursued by the broader application of decision theories to NTR engine control. New applications can be investigated for addressing additional fault scenarios, and applications to more general operational decision making needs.

Bibliography

[1] S. K. Borowski, "Nuclear thermal rocket (ntr) propulsion: A proven game changing technology for future human exploration missions," *Global Space Exploration Conference*, 2012. 6

[2] D. R. Koenig, "Experience gained from the space nuclear rocket program (rover)," tech. rep., Los Alamos National Laboratory, 1986. 7, 8, 10, 11, 12, 13, 14

[3] S. V. Gunn, "Development of nuclear rocket engine technology," *AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, January 1989. 8, 9, 11

[4] J. L. Finseth, "Rover nuclear rocket engine program: Overview of rover engine tests. final report," February 1991. 8

[5] W. ROBBINS, "An historical perspective of the nerva nuclear rocket engine technology program," *Conference on Advanced SEI Technologies*, July 1991. 8

[6] "Xe-prime engine final report vol.1, engine and facilities description," Tech. Rep. RN-S-0510, Aerojet General, 1969. 11, 16, 17, 20, 26

[7] "Xe-prime engine final report vol.2, assembly, test, and disassembly operations," Tech. Rep. RN-S-0510, Aerojet General, 1970. 11, 26

[8] M. G. Hauts, "Nasa's nuclear thermal propulsion project," *AIAA Space 2015 Conference*, 2015. 11

[9] "Xe-prime engine final report vol.3, book 2, test data analysis: Startup systems," Tech. Rep. RN-S-0510, Aerojet General, 1970. 21, 26

[10] E. F. Dowling, "Multivariable controller design process," tech. rep., Aerojet Nuclear Systems, April 1971. 23, 26

[11] E. A. Parziale and R. F. Schenz, Jr, "Thrust buildup state variable feedback control system for the e-1 cam revision 6.1," tech. rep., Aerojet Nuclear Systems, July 1971. 23

[12] "E-1 common analog model," Tech. Rep. RN-S-0469, Aerojet Nuclear Systems, July 1969. 24, 27

[13] "E-1 common analog model revision 6.1," Tech. Rep. RN-S-0469, Aerojet Nuclear Systems, December 1970. 25, 27

[14] "Xe-prime engine final report vol.3, book 1, test data analysis," Tech. Rep. RN-S-0510, Aerojet General, 1970. 26

[15] J. D. Rader, M. B. Smith, M. S. Greenwood, and T. Harrison, "Nuclear thermal propulsion dynamic modeling with modelica," *Nuclear and Emerging Technologies for Space*, February 2019. 26

[16] L. E. Kendrick, *On-Line Digital Computer Control of the NERVA Nuclear Rocket Engine*. PhD thesis, The University of Arizona, 1972. 27, 30, 86

[17] M. R. Johnson, *Precomputed State Dependent Digital Control of a Nuclear Rocket ENgine*. PhD thesis, The University of Arizona, 1972. 27, 35

[18] R. T. Wood, "Enabling autonomous control for space reactor power systems," *Proceedings of the Fifth International Topical Meeting on Nuclear Plant Instrumentation, Controls, and Human Machine Interface*, July 2006. 40, 42

[19] T. B. Sherridan, "Human and computer control of undersea teleoperators," tech. rep., Massachusetts Institute of Technology Man-Machine Systems Laboratory, 1978. 41, 43

[20] S. M. Cetiner, "Technical basis for automated decision making, a survey on the state-of-the-art of decision making and existing analytical tools," tech. rep., Oak Ridge National Lab, February 2014. 45, 47, 89

[21] S. M. Cetiner, "Transformational challenge reactor autonomous control system framework and key enabling technologies," tech. rep., Oak Ridge National Lab, February 2019. 45

[22] M. Ardelt, "Highly automated driving on freeways in real traffic using a probabilistic framework," *IEEE Transactions on Intelligent Transportation Systems Vol. 13*, 2012. 47

[23] P. C. Fishburn, "Utility theory," *Management Science*, vol. 14, no. 5, pp. 335–378, 1968. 69

[24] R. L. Keeney, "The art of assessing multiattribute utility functions," *Organizational Behavior and Human Performance*, vol. 19, pp. 267–310, 1977. 70, 73

[25] G. W. Fischer, "Four methods for assessing multi-attribute utilities: An experimental validation," tech. rep., The University of Michigan, 1972. 70

[26] D. V. Winterfeldt, "Multi-attribute decision theory: Models and assessment procedures," tech. rep., The University of Michigan, 1973. 70

[27] G. W. Fischer, "Multidimensional value assessment for decision making," tech. rep., The University of Michigan, 1972. 70

[28] D. W. North, "A tutorial introduction to decision theory," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 3, pp. 200–210, 1968. 73

[29] J. R. Busemeyer, *Quantum Models of Cognition and Decision*. Cambridge: Cambridge University Press, 2014. 89

Appendix

A Simplified Nonlinear Model Embedded System of Equations

```
function [dT15,dT2,dT1,dS,dC1,dC2,dC3,dN,dT7,P13,P11,dT11,P2,P15,dWt] = ...
    snm_001(T15,T2,T1,S,C1,C2,C3,N,T7,P13_in,P11_in,T11,P2_in,Wt,...
    thetaBcv,thetaSsv,drumAngle)
%% Introduction
% This function provides a reduced order representation of the Common
% Analog Model(CAM) - a high order representation of the NERVA E-1 Engine,
% completed sometime in the early 1970's.
%
% This Simplified Nonlinear Model (SNM) is an eleventh order reduced system
% of differential and algebraic equations developed by Larry Ellis Kendrick
% for his 1972 PhD dissertation "On-Line Digital Computer Control of the
% NERVA Nuclear Rocket Engine", further modified and implemented by Morris
% Ray Johnson for use in his later 1972 dissertation "Precomputed State
% Dependent Digital Control of a Nuclear Rocket Engine".
%
% This function implements a combination of Johnson's updated equation set
% and Kendrick's orignal set as a starting point for developing a
% computationally efficient dynamic model for rapid studies of nuclear
% thermal rocket engines, and to serve as a research and development
% platform.

%% Nomenclature
%%%%%%%%%%%% Pressures (psi)%%%%%%%%%%%%
% P15 = Nozzle chamber pressure
% P13 = turbine exit pressure
% P11 = turbine inlet pressure
% P2  = pump outlet pressure
```

```
%%%%%%%%%%% Reactivity ($) %%%%%%%%%%%
% DKT = total reactivity
% DKD = drum input reactivity

%%%%%%%%%%% Temperatures (deg R)%%%%%%%%%%%
% T15 = exit propellant temperature
% T2  = Pump outlet propellant temperature
% T1  = temperature of propellant station 1

% Tc  = average propellant temperature
% T7  = average hydrogen temp in skirt
% T11 = turbine inlet hydrogen temp

%%%%%%%%%%% Flow Rates (lb/s) %%%%%%%%%%%
% Wn  = core hydrogen flow rate
% WBcv= turbine bypass valve hydrogen flow rate
% Wt  = turbine hydrogen flow rate
% W11 = pump hydrogen flow rate
% Wno = skirt hydrogen flow rate
% Wss = support structure hydrogen flow rate

%%%%%%%%%%% Valve Positions(%open) %%%%%%%%%%%
% thetaBcv = turbine bypass valve position (Range: 0-90 deg)
% thetaSsv = support structure valve position (Range: 0-90 deg,
%           fixed at 11.7 degrees for the validation studies)

%%%%%%%%%%% Precursor Densities %%%%%%%%%%%
% C1 = first precursor
% C2 = second precursor
% C3 = third precursor

%%%%%%%%%%% Other %%%%%%%%%%%
```

```
% N = turbopump speed (RPM)
% S = percent nuclear power
% thetaD = control drum turn-in angle (Range: 0-180 deg)

%% Propellant flow equations
% WBcv = 0.04*thetaBcv*sqrt((P11_in/T11)*(P11_in-P13_in)); %DMS: Morris
WBcv = 0.0182*thetaBcv*sqrt(abs((P11_in/T11)*(P11_in-P13_in)));%DMS:kendrick

Wno  = 1.5*sqrt(abs(((P2_in+P11_in)/(2*T7))*(P2_in-P11_in)));
Wss  = 0.0184*thetaSsv*sqrt(abs(((P2_in+P11_in)/(2*T7))*(P2_in-P13_in)));
W11  = Wno + Wss;

dWt  = 10*(W11 - WBcv - Wt);
Wn   = Wt + WBcv;
P15  = 0.076*Wn*sqrt(abs(T15));
P13  = P15*(1.0+33.3/sqrt(abs(T15)));
P11  = 0.346*Wt*sqrt(abs(T11)) + 0.786*P13;
P2   = (3.434E-6)*N^2 - (2.967E-4)*W11*N + 30;

dN   = 9.6E+5*(Wt*T11/N)*(1-(P13/P11)^0.25) - (8.46E-5)*(N^2) - 0.021*W11*N;
dT7  = -0.011*W11*(T7-95) + 1.59*S; % kendrick
dT11 = -0.011*W11*(T11-95) + 2.53*S; % kendrick
% dT7  = -0.011*W11*(T7-45) + 1.59*S; % Morris
% dT11 = -0.011*W11*(T11-45) + 2.53*S; % Morris

%% Reactor equations
Tc   = 0.13*(3.057*T15 + 2.0*T2 + 3.0*T1);
thetaD = drumAngle*pi/180; % thetaD is restrained to 12degrees/second or
                           % less and must be between 0 and 180 degrees

nDrum = 12; % number of control drums
nWork = 12; % number of working control drums
```

```
frac  = nWork/nDrum; % fraction of functional drums

DKD  = frac.*(4.5*sin(thetaD/2)^2) -1.85; % Kendrick
DKT  = 0.55*sqrt(abs(Wss)) + 3.25*(P13+P15)/Tc + 0.0244*(P2+P11)/T7 - ...
    7.7E-4*(Tc-500) + DKD; % Kendrick

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DKD  = 4.5*sin(thetaD/2)^2 -4.5; % Morris
% DKT  = 0.55*Wss + 3.25*(P13+P15)/Tc + 0.0244*(P2+P11)/T7 - ... % Morris
%     7.7E-4*(Tc-500) + DKD;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dT15 = 0.02019*Wn*(T2-T15) + 16.25*S;% measured value including tc response
dT2  = 0.02671*Wn*(T1-T2) + 31.28*S;
dT1  = -0.01105*Wn*T1 + 21.31*S;

dS   = 0.0276*C1 + 0.235*C2 + 1.65*C3 - (1-DKT)*660*S;
dC1  = 181.0*S - 0.0276*C1;
dC2  = 369.0*S - 0.235*C2;
dC3  = 110.0*S - 1.65*C3;

end
```

B Drum Fault Flag Generator

```
function drumEnable = drumFault(time)


% init
drumEnable = ones(1,12);


% specify drum faults
numFailures = 1; % number of drums to fail at the specified time
failureTime = 35; % seconds


% inject faults
if numFailures > 0 && time > failureTime
    drumEnable(1:numFailures) = 0;
end


end
```

C Decision Mode Selector

```
function [switchController,drumUtil,ssvUtil] = decisionModeSelector(percentDrum,
    percentSSV,numDrums, decisionAlgorithm)
%init
switchController = 0;
drumUtil = 0;
ssvUtil = 0;

switch decisionAlgorithm
    case 1 % no switching capability - primary control will be maintained
        % do nothing

    case 2 % simple logical decision
        switchController = decision_logical01(numDrums);

    case 3 % single attribute utility function 1
        [switchController,drumUtil,ssvUtil] = decision_utility(percentDrum,
            percentSSV,1);

    case 4 % single attribute utility function 2
        [switchController,drumUtil,ssvUtil] = decision_utility(percentDrum,
            percentSSV,2);

    case 5 % run engine on SSV instead of drums
        switchController = 1;

    otherwise
        % in case of an undefined input, the model will still function as
        % if there is no decision capability. Primary control is maintained
end
end
```

D Actuator Availability Calculation

```
function [percentAvailable,percentSSV,numDrums] = attribute_percentActuator(
    drumEnabled,drumAngle,ssvIC)

maxDrums   = 180*12; % maximum available drum rotation
maxSSV     = 90;    % maximum available SSV rotation
percentSSV = 100*(1 - ssvIC/maxSSV);

% calculate number of drums considered to be functioning properly
numDrums = sum(drumEnabled); % number of functioning drums

% Determine loss of drum functionality by calculating available
% functionality remaining for thrust buildup. (note: shutdown will be a
% separate but similar operation that is unecessary for this research)

drumStuck = ~drumEnabled; % logical vector of stuck drums

stuckDrumContribution = sum(drumAngle(drumStuck)); % drum angle of stuck
                                                   % drums still
                                                   % contributing
                                                   % reactivity to thrust
                                                   % build up, i.e. where
                                                   % they got stuck.

percentAvailable = 100*(numDrums*180 + stuckDrumContribution)/maxDrums;

end
```

E Exponential Curve Fit Parameter Optimization

```
% This script is used to generate parameters for exponential curve fits to
% be used as utility functions

% Include sentiment evaluation data here:

% x = [0 10 25 50 100];
% y = [0 0.25 0.5 0.75 1];

x = [0 50 100];
y = [0 0.95 1];

% Initial guess for parameter optimization
Po = [1 -1 -0.1];

% The desired utility function f() an exponential equation
f = @(p,x) p(1) + p(2).*exp(p(3).*x);

% The goal is to find parameters p(1), p(2), and p(3) so that f best fits
% the sentiment data provided in (x,y). This is done by minimizing the
% euclidian distance between f() and (x,y) with fminsearch()
P = fminsearch(@(p) norm(y - f(p,x)), Po);
```

F Utility Function 1

```
function u = utility_01(x)

if x < 0
    x = 0;
elseif x > 100
    x = 100;
end

% exponential function parameters
a =  1.095811502346911;
b = -1.087445231303666;
c = -0.023736497361493;

% exponential curve fit
u = a + b.*exp(c.*x);

end
```

G Utility Function 2

```
function u = utility_02(x)

if x < 0
    x = 0;
elseif x > 100
    x = 100;
end

% exponential function parameters
a = 1.002762447776910;
b = -1.002749803797863;
c = -0.058905408719796;

% exponential curve fit
u = a + b.*exp(c.*x);

end
```

H Utility Based Decision

```
function [switchController,drumUtil,ssvUtil] = decision_utility(percentDrum,
    percentSSV,utilityOption)
% This function calculates the utilities of the expected outcomes for the
% two choices currently available to the engine control system.
%
% Action 1: Maintain primary control mode using drums for reactivity
%          control
%
% Action 2: Switch control to backup mode using support structure valve for
%          reactivity control
%
% For a single action, the expected utility is calculated using the utility
% of the expected outcome multiplied by the probability of that outcome,
% plus the utilities of the alternate outcomes(s) multiplied by their
% respective probabilities. This is repeated for the expected outcome(s)
% of the alternative action and compared to determine the choice with the
% highest utility. This paradigm is referred to as rational decision,
% because the choice is made based on whichever outcome provides the most
% utility.
%
% Example: The utility U of action A with two possible outcomes x and y
%         would be calculated as follows:
%
%               U(A) = P(x)*u(x) + (1-P(x))*u(y)
%
% The implementation in this function assumes a diagnostic module will
% supply only the probability of an outcome, and no additional possible
% outcomes for one given action. For example, you may have the probability
% that valve #1 is at 30 degrees is 0.9.
%
```

```
% Since there is incomplete information about what other
% outcomes are possible, they are assumed to be undesirable and are thus
% assigned zero utility, resulting in the following:
%
%                       U(A) = P(x)*u(x)
%
% If such functionality is made available, this function can be updated to
% handle more expected outcomes of a given action.
%
% Measurement uncertainties will need to be provided by a diagnostic module
% Since this work is not focused on diagnostics, the uncertainties in
% position measurements will be hardcoded here for now. If time permits,
% these inputs may be built into the model to reduce the workload on future
% researchers who implement diagnostic capabilities.
drumAngleProb = 1.0; % probability that the drum angles are as measured
ssvAngleProb = 1.0; % probability that the ssv angle is as measured

% calculate the utilities of the two available choice outcomes
if utilityOption == 1
    drumUtil = drumAngleProb*utility_01(percentDrum);
    ssvUtil = ssvAngleProb*utility_01(percentSSV);
end
if utilityOption == 2
    drumUtil = drumAngleProb*utility_02(percentDrum);
    ssvUtil = ssvAngleProb*utility_02(percentSSV);
end
if ssvUtil > drumUtil
    switchController = 1;
else
    switchController = 0;
end
end
```

Vita

David is originally from Davenport, Iowa in the Quad Cities region of the Iowa-Illinois border. During high school he worked as a welder, and continued to do so after graduation until leaving the Quad Cities to study Aerospace Engineering. He received his Bachelor of Science degree in Aerospace Engineering from Iowa State University in 2012, followed by a Master of Engineering degree in Aerospace Engineering in 2014. He then moved on to work as an Aerospace Engineer before deciding to return to academia. In 2016 David joined the University of Tennessee to pursue his Doctor of Philosophy degree in Nuclear Engineering. There he worked to establish research on autonomous control of nuclear thermal rocket engines. His research interests include autonomous control, artificial intelligence, and advanced air and space propulsion.