Self-Assembling Decentralized Control Constructs for Large-Scale Variably-Interconnected Systems

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Corey A. Ippolito

M.S., Aerospace Engineering, Georgia Institute of Technology
B.S., Aerospace Engineering, Georgia Institute of Technology

Carnegie Mellon University
Pittsburgh, PA

December, 2016

Abstract

There is an emerging need to develop new techniques for control system design that better address the challenges posed by modern large-scale cyber-physical systems. These systems are often massive networks of interconnected and interoperating subsystems that fuse physical processes, embedded computation, automation technologies, and communication. The resulting problems are dimensionally large, exhibit significant time-varying structural variations during operation, and feature complex dynamics, constraints and objectives across local and global-system scales. These properties are difficult to address using traditional control theoretic methods without substantial loss of performance and robustness.

To overcome these limitations, this dissertation presents new concepts and methods for control of modern large-scale variably-structured systems through self-assembling and self-configuring control constructs that allow for fundamental restructuring of the control system's topology in response to the current system structure. We present the System Component Graph (SCG) formulation as a mathematical framework that generalizes and extends directed graph methods from decentralized control. We present algorithms, methods, and metrics for real-time decentralization and control-structure optimization, utilizing the inclusion principle for addressing interconnected overlapping dynamics and optimal linear-quadratic (LQ) methods for local decentralized subsystem control. Global system control and performance is achieved through a centralized planner that provides continuous real-time optimized trajectories as guidance command inputs to each subsystem. We present the method of Random Subcomplement Trees (RST) for pseudo-optimal real-time trajectory planning of large-scale systems which formalizes and extends the method of rapidly-exploring random trees in a control optimization framework. The RST method defines transformations from the higher-dimension state space into an intermediate lower-dimensional search space, where optimal transitions between subspace states are defined. In the context of this approach, the resulting decentralized topology found within the SCG framework provides the RST subspace definition and requisite transformations, and optimal transitions in the search space are found through forward evaluation of the closed-loop decentralized subsystem dynamics.

The methods developed in this thesis are applied to a set of real-world problems spanning various domains and demonstrate the application of these methods from first-principle modeling through control system analysis, design, implementation, and evaluation in experimental tests and simulation.

Acknowledgements

I would like to thank my doctoral committee members - Jason Lohn, John Dolan, Kalmanje Krishnakumar, and the committee chair Abe Ishihara - without whose input this dissertation would not have been possible. I would like to acknowledge the National Aeronautics and Space Administration for funding my academic studies and providing me the opportunity to complete this dissertation. I have been fortunate to work with an extraordinary array of remarkably talented individuals at Carnegie Mellon University and Ames Research Center, to all of whom I am tremendously grateful for their input, advice, guidance, leadership, and friendship. I would like to thank Khalid Al-Ali whose mentorship and guidance early in my studies had a profound influence on this work. I would like to thank Kalmanje Krishnakumar and Chad Frost at Ames Research Center, who provided guidance and latitude to help balance my work and academic commitments. I would like to thank the members of the Advanced Control and Evolvable Systems at Ames, all of whom have had some influence on my studies, with particular thanks to Nhan Nguyen, Ritchie Lee, Sebastian Hening, and Yoo-Hsiu Yeh. I would like to thank my unmanned aircraft systems research collaborators at Ames, particularly Mathew Fladeland, who provided leadership vision and support that made many projects in this dissertation possible, as well as Ric Kolyer and Bruce Storms for their technical contributions. I would like to thank the Sustainability Base team, particularly Steven Zornetzer, whose leadership vision and support made the building control research possible, as well as Kai Goebel and Scott Poll. I would like to thank my collaborators at the United States Geological Survey, particularly Jonathan Glen, Geoff Phelps, and John Spritzer. I would like to thanks the members of various projects I was involved with during this dissertation, including the Surprise Valley Earthquake Hazards project, the Turrialba Volcano Monitoring project, the Payload-Directed Flight project, the Elastically-Shaped Aircraft Control project, the Seedling Wing Morphing Project, and the Unmanned Traffic Management System project. Finally, I would like to thank my wife, Monica, for providing an endless source of motivation, love, and joy that made this journey possible.

Table of Contents

List of Tables

List of Figures and Illustrations

Chapter 1 Introduction

An emerging challenge for control system designers is the need to address systems that are ever-increasing in size, sophistication, and complexity. Across many industries where systems historically were independent and minimally-automated, the trend in modern systems is to instrument, automate, and interconnect across real-time communication pathways. Modern large-scale dynamical systems are becoming massively interconnected networks of interdependent and interoperating systems that fuse physical and cybernetical elements. These cyber-physical systems are an inextricable fusion of physical and dynamical processes with embedded computation, algorithms, automation technologies, and communication. The resulting control problems may exhibit simultaneous elements – objectives, constraints or dynamics, for instance – that are complex, difficult to express algebraically, and generally multidisciplinary in nature. These elements may combine continuous, algorithmic, geometric, and discrete properties. Further, the large-scale system may be composed at run-time from varying numbers of smaller sub-systems. These sub-systems may variably compose, restructure, reassemble, and detach from the large-scale system during normal operation. The resulting large-scale system exhibits extreme variations in size, dimension, and structural composition. There are often complex time-varying constraints and objectives that vary and compete across the local and global-system scale. Further, many modern systems are variably-structured compositions of smaller sub-systems, where each sub-system is well known at the system design stage, but there may be little to no a priori knowledge of the structure of the large-scale system during system operation, as the structure may result from seemingly arbitrary circumstance.

The resulting control system design problems are difficult to address using traditional control theoretic methods without substantial loss of optimality, performance and robustness. For instance, methods applying stochastic uncertainty such as H_∞ or μ-synthesis control, parametric uncertainty such as linear matrix inequality methods, or parametric structural variability such as parametric adaptive control, have been successfully applied to systems that exhibit limited structural variability and structured uncertainty. These methods unfortunately become intractable for larger systems that exhibit significant structural and dimensional variability; robust methods become increasingly suboptimal to accommodate variation, perturbation methods assume limited structural variation, and parametric uncertainty methods are infeasible as the parametric dimension grows exponentially with system dimension (e.g., n^2 possible interconnections in a system with n states). These approaches would require multiple control system solution

designs for every likely system configuration, which is only feasible for systems with a limited number of known fixed structural configurations and limited structural variability. Likewise, there are many approaches developed for control of large-scale systems, such as vector dissipative approaches for interconnected dynamical systems, decentralized control approaches such as system digraph methods, nested epsilon decomposition, convex Youla optimization methods, and quadratic invariance methods. Unfortunately, these approaches similarly become intractable as structural and dimensional variability increases. Structurally optimizing decentralization algorithms have been presented to help automate the decentralized control design process, but these methods are still offline methods that require numerical optimization and intensive human interaction, and are not readily applicable to autonomous online decentralization. Further, all these techniques provided limited abilities to handle complex system elements described above; these techniques are often formulated for linear or non-linear algebraic systems, and while computational approaches such as graph-theoretic methods have been widely applied, these methods are limited to analogous linear algebraic operations. Further, the graph representation contains limited information about the system, and is not readily amenable to inclusion of general control problem elements beyond simple structural constraint information utilized for analyzing partitions for decentralization.

There are many applications cited in the literature for large-scale system control and design. The impetus for pursing increased automation across systems is often to achieve necessary improvements in performance, safety, reliability, efficiency, and functionality. For instance, the automotive industry is seeing a transformation from independent human-operated mechanical vehicles into large-scale networks of intelligent autonomous self-driving vehicles. Advances in sensing and computational technologies provide amounts of real-time data per vehicle that is orders of magnitude larger than what had been previously available. Automotive researchers envision city-wide networks of vehicles that communicate, interact, interoperate, and coordinate to navigate through future smart-cities with greater efficiency, performance, and safety than is currently possible in the legacy system involving human-operated vehicles on the existing roadway system using the current traffic flow control mechanisms. Many systems, such as the current air traffic control network, are nearing capacity and are unable to accommodate growing demand without substantial increase in cost, safety, and performance. Other systems, such as environmental building control, have grown to such scale that suboptimal performance and inefficiencies have substantial large-scale global effect in terms of economics, pollution, and emissions. Automating, networking, and coordinating control throughout a smart building has the potential to provide substantial improvements in efficiency and performance, but this approach

unfortunately transforms the control design problem from a single-input, single-output, single-objective control problem into a large-scale control problem involving thousands of sensors, states, and actuators, and operating across communication and computational infrastructures that introduce additional simultaneous and competing constraints and objectives. The resulting problem of controlling modern cyber-physical systems overwhelm classical and modern control theory in terms of sheer magnitude, dimensionality, complexity, and scope. Developing formal methods to address the interconnection, monitoring, coordination, and control of such modern large-scale cyber-physical systems has significant promise, and this has therefore been the focus of recent research efforts in many industries, such as research into the control of wind turbine farms, regional and national control of unmanned aircraft traffic, and design and control of flexible aerospace structures for the next generation of aircraft and spacecraft systems.

1.1 Thesis Statement

> *Realization of a self-assembling self-configuring distributed decentralized control architecture that could fundamentally restructure and adapt to large-scale structurally-variable cyber-physical systems would enable both local/global system control with greater performance, efficiency, and reduced control system design effort.*

1.2 Approach

This dissertation focuses on control of large-scale distributed interconnected cyber-physical systems, focusing particularly on systems with significant uncertainty in structure – such as when the structure of the system to be controlled is not known at the time of control system design – or when the structure is expected to alter significantly during operation. Many problems in control of such large-scale systems involve: (1) providing greater autonomy at the local system levels through substantial increases in computation, networking, actuation, and sensing capabilities as compared to legacy approaches; (2) fusing and interconnecting massive numbers of local agents across a global system in a spatially-distributed, time-varying and structurally-varying environment; and (3) controlling and coordinating all agents as they interact with each other and the environment toward both local system and global system objectives. The structurally-varying and time-varying nature of these large-scale systems pose significant challenges. For instance, these systems may see significant variation during run-time operation in the number of local

sub-systems being controlled, their configuration in relation to each other, the interconnectivity between sub-systems, and the control objectives for the local and global systems.

For these problems, designing a control system for every combinatorically possible configuration is not feasible. In addition to the complexity and dimensionality issue, popular control theoretic approaches – such as robust or adaptive control – are difficult to apply when confronted with systems that exhibit large fundamental structural variability in the plant. Further, elements of the control problem (e.g., dynamics, objectives, and constraints) are inherently multidisciplinary and are difficult to pose strictly within a control theoretic construct. We take particular inspiration from recent advances in the field of decentralized control for large-scale systems that merge modern control techniques with graph theoretic constructs. We extend mathematical formulations from this field to a generalized framework that can be used to model the computation, communication, and information flow challenges of distributed cyber-physical systems, and further extend offline design techniques to allow for online reconfiguration that provides real-time structural adaptation of the control system.

This thesis presents concepts for a control system that, when presented with a large-scale system composed of a variable structure of smaller known sub-systems at run-time, automatically determines the proper structure for decentralized feedback-control to control the system toward a given global objective, then automatically determines the proper course of action for individual sub-system without any user intervention. The only a priori knowledge required for the control system is a mathematical model of the sub-systems, a global objective function, and during run-time, the structure of the large-scale system must be determinable (i.e., the number of sub-systems interacting and the nature of their interaction). In this dissertation, we assume full knowledge of the system structure, which can be fulfilled through communication or instrumentation. For instance, in a collaborative highway system of autonomous vehicles, inter-vehicle communication passing relative position information between vehicles would be sufficient to know the structure of the large-scale system at any given time. In a smart building environment, sensors placed on interfaces between rooms - such as doors and windows – would be sufficient. The assumption of full system structure knowledge could be relaxed as a future research direction to allow for various types of uncertainty in the system or interconnection structure.

1.3 Contributions

1.3.1 Self-assembling Self-Configuring Control for Structurally Variable Large-Scale Systems

This dissertation presents a new concept for control of modern massively-interconnected variably-structured cyber-physical systems through application of self-assembling and self-configuring control constructs that restructure the control system topology in response to the current system structure. These constructs can be applied throughout the system to address local and global system control challenges. These constructs can adapt to significant structural changes in topology in real-time, allowing greater performance, efficiency, and robustness over existing approaches. The methods build on recent progress in the literature from many fields, including decentralized control, optimal control, trajectory optimization, set and graph theory, and computer science. The realization of such approaches could have significant impact on the aforementioned applications, and as such, this dissertation shows novel application of these methods to various related fields, with a primary focus on analysis and experimentation for intelligent integrated building control.

1.3.2 System Component Graphs: A Mathematical Framework for Modeling, Analysis, and Synthesis of Large-Scale Control Problems

Toward realization of this concept we present a new formulation for systems as generalized graphs called System Component Graphs (SCG). Graphs are a natural representation to utilize for analysis of the structure of large-scale systems and many such descriptions have been presented in the literature. Particularly, the method of system digraphs (D. Siljak 2011) for decentralized control analysis has seen wide application, through which important properties of system structure - such as input/output reachability, structural controllability, and structurally fixed modes – can be deduced. Graph formulations and system digraphs will be reviewed in Chapter 2.3. The formulation presented in this dissertation can be seen as an extension and generalization of this approach. The system digraph formulation provides a description of a system that only captures limited structural properties for a particular class of linear time-invariant system plants. We present a formulation for a graph-space that presents a complete mathematical description of the large-scale system, and we present transformations between alternative system representations – such as between graph-space and state-space for linear time-invariant systems – that are linear, invertible, and isomorphic. The formulation presents a generalized description of any type of linear or non-linear mathematical relationship. The nature of our dissertation problem is multi-disciplinary, and the formulation proposed represents a unifying representation for the entire large-scale system for these various domains. In this way, our approach transforms the

problem once into the graph-space and then utilizes this representation throughout the presented framework all the way through to implementation. With this generalized representation, all parts of the framework can manipulate and perform operations on the system in system-graph space. The presented formulation allows the framework to decompose, manipulate, transform sub-components back and forth between representations, recompose the system, and pass systems to other parts of the framework. As such, this representation can be seen as a unifying and generalized extension of the system digraph representation.

1.3.3 Random Subcomplement Search Tree (RST) Method for Large-Scale System Optimization

A significant challenge exists in trajectory planning and optimization for large-scale systems. Even when control systems can be synthesized through approaches such as decentralization, the resulting large-scale system is high-dimensional. Further, approaches often require significantly simplified objectives with limited mathematical representations and properties, such as linear quadratic costs on control input and states, which have difficulty in capturing large-scale system objectives that may be complex, non-linear, and difficult to represent in equational form. For instance, a swarm of collaborating unmanned vehicles may be required to optimize trajectories on complex wind field data that is not amenable to reduction to such a simplified global form. At the level of vehicle control, large-scale system objectives are not typically introduced. For instance, aircraft flight control traditionally minimizes error along a flight path; it does not take into account coverage optimization for a group of collaborating vehicles. Likewise, building control typically controls temperature to a desired set-point; it does not take into account total cost optimization which might require coordinating traditional HVAC control with modern building control systems that may include window control, lighting control, radiative heating systems, geothermal cooling system, exterior shading systems and louvers, electrical window films and glazes, and plenum floor ventilation.

We present and formalize a new approach to real-time planning that extends the concepts of random tree search to larger-dimensional systems. Our method defines an intermediate reduced dimensional space for planning to occur. The user must define transformations between the system and this intermediate space, then define optimal transformations between states in this reduced dimensional space. For instance, this can be seen as a generalization of planning for a general vehicle with a general 12-state rigid-body representation, where an optimal control system has been created to control the vehicle between positions, allowing planning to occur in three dimensions. By then unfolding the control response to the resulting plan, the exact trajectory – states, inputs, and outputs – is given over time for the twelve-state system. The presented planning method can be seen as a generalization of this approach that

is then applied to the dissertation thesis of control self-configuration and self-assembly. The presented planning method is specifically amenable to control of large-scale decentralized control systems that are solved through the presented approach of self-assembling control. The presented framework decomposes the control problem into smaller solved sub-problems, then reformulates the large-scale problem through reduced dimension utilizing the sub-system control solutions as inputs. The solution presented allows complex control objectives to be directly solved through this system, without the intermediate human intervention typically required.

1.3.4 Clustering Algorithm and Metric for Online System Partitioning

We present a new clustering algorithm and associated metric that operate on the system graph representation to quickly identify system partitions. The clusters resulting from this algorithm define the structural topology of the control constructs for the local and global systems. The algorithm and metrics extend recent methods presented in the literature for partitioning of structure graphs for dynamic structure analysis, but extend the optimization criteria beyond system dynamics to include a wider range of constraints and objectives that are introduced for these problems, some of which arise from multi-disciplinary considerations (e.g., communication network bandwidth limits, as demonstrated in the UGV indoor navigation application).

1.3.5 Savant-ML: A Fluid/Thermal Building Simulation and Control Modeling Library

We derive and implement a new model for simulation and environmental control system development of building systems. This system models fluid and thermal flow through fixed finite constant volumes from first principles through the Navier-Stokes equations. The modeling library is a novel graph-based formulation derived using the mathematical system graph formulation presented. The modelling derivation was developed into an application programming interface (API) and implemented in the Savant-ML C++ programming library. Simulation utilizes a high-order Runge-Kutta integration scheme.

1.3.6 Dynamic Models and Simulation Software for Fixed-wing Aircraft, UAS, and UGV

We present several applications in this dissertation to demonstrate and evaluate the unique aspects of our approach. Toward this goal, we derive and present dynamic models for the various systems from first-principles. We develop simulations for each of the models presented, then integrate these models with the control system concepts being tested. The derivation and detail of each model, implementation, and closed-loop simulation results are presented in Chapter 6.

1.4 Dissertation Outline

The organization of this dissertation is as follows. Chapter 3 presents details of the approach, including a high-level outline of the framework and the general process for automated control system assembly. In Chapter 4, we present the System Component Graph formulation - a graph-theoretic mathematical formalism for modeling, analysis, and synthesis of the control system. We present transformations, algorithms and operations that manipulate the graph representation in the process of assembly, partitioning the problem into a decentralized set of small feasible control problems. We present an optimal control solution to solve the local decentralized control problems, then present the algorithms for reassembling a global model for analysis from these solutions. In Chapter 5 we present the approach of Random Subcomplement Search Trees for real-time trajectory planning of the reconstructed large-scale system that optimizes global system objectives and satisfies global constraints, which completes the assembly process.

In Chapter 6, we evaluate the framework and methods presented in this dissertation on various real-world example problems. The framework and novel research contributions presented have applicability to other applications and other domains. To illustrate this, we generalize and abstract the problem that each of the framework's components addresses, present solutions to the abstract problem, then apply these solutions to a real-world problem to illustrate the effectiveness and utility of these components. Each section presents a detailed application example. We derive models from first principles, analyze the problem utilizing the dissertation approaches, synthesize controllers, then implement and test the controller. We present and analyze results from simulation and experimentation to demonstrate effectiveness and performance of these methods. In section 6.1, the RST method for real-time trajectory optimization is demonstrated on a wildfire smoke plume monitoring problem on an autonomous fixed-wing aircraft using a body-fixed remote imaging sensor. In section 6.3, the framework and optimizer is applied to cooperative indoor navigation of an autonomous robotic ground vehicle interacting with distributed building sub-systems in a smart building environment. In section 6.4, the approach is applied to combined flight and structural control of a commercial transport aircraft with a flexible aeroelastic wing through distributed actuation of a variable continuous trailing-edge control surface. Section 6.5 applies the framework toward implementation of an intelligent integrated building control system. We present and derive a new mathematical physics model based on a first-principles thermal-fluid analysis that is amenable to the graph theoretic formulation presented in the framework. We present results from an experimental analysis of the model in a simple test building. We then demonstrate the process of control assembly and reconfiguration on example building configurations.

Chapter 2 Background and Related Work

2.1 Large-Scale System Control

Control of large complex systems is a topic of continued research interest. As systems grow in size and complexity, a centralized monolithic control scheme becomes intractably difficult to design and implement. Many approaches have been pursued in the literature. These approaches can generally be classified in one of the following categories – which extends the three categories presented in (Mahmoud, Hassan and Darwish 1985).

1. **Centralized Approach**: For sufficiently simple systems, this is often the preferred approach.
2. **Decentralization and Partitioning Approach**: Partition the large system into several more easily handled sub-systems, generate solutions for individual sub-systems, and compose or coordinate sub-systems to reconstruct the overall solution.
3. **Multi-Time-Scale Approach**: For systems intrinsically possessing separation in the eigenvalue spectrum of dynamic systems, the system can be partitioning along these time-scale separated boundaries, and each system partition can be treated independently.
4. **Aggregation Approaches**: Aggregate variables in the large system in order to decrease the size of the model, and design controllers for the aggregate model.
5. **Reduced Order Modeling Approach**: Generate reduced-order models for the large complex system that capture significant behavior at these various operating points, and design controllers around the reduced-order models for these operating points.

The concept of control decentralization is encapsulated by approach (2) as a fundamental technique for control of complex systems. A conceptual centralized and decentralized architecture is illustrated in Figure 2-1. Decentralized control addresses complex systems that exhibit one of three fundamental characteristics: systems of high dimensionality, system with high uncertainty, and systems with imposed constraints on information structure or resources (D. Siljak 2011). Decentralization provides the mechanism for decomposing a large complex control problem into a distributed set of smaller, tractable, feasible control problems that can be individually addressed and then combined to derive a solution for the overall system. Decentralization delegates and distributes control authority to a set of sub-system controllers which may have similar overlapping objectives, limited input into the system, limited

amount of information and sensing input, and subject to information flow constraints between sub-systems. Many applications cited in the literature pursue decentralization for the inherent benefits derived from this approach, including simplification of the problem domain, tractability of the approach and the deployed solution, and robustness to a wide range of uncertainties too complex to model. Other applications in the literature pursue decentralization out of necessity. Physical properties of many systems necessitate that control authority be partitioned and delegated between sub-systems, where information constraints are imposed on the control structure by system properties.

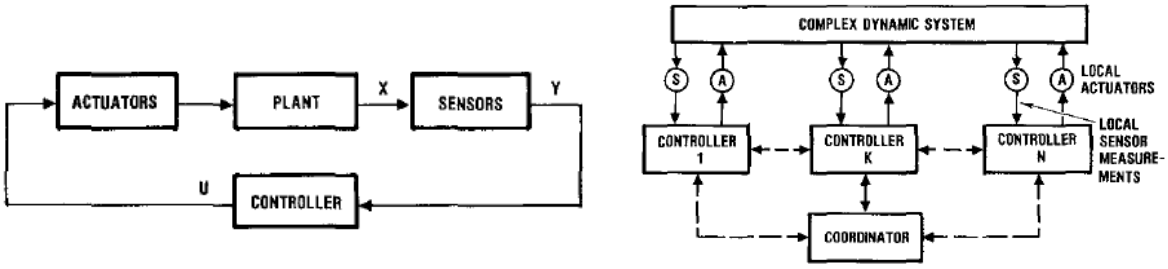


Figure 2-1. Centralized (left) Versus Decentralized Control Architecture (right).

Decentralized control system design typically follows the following general process:

1. Decompose/partition the complex system into a set of smaller sub-systems
2. Develop individual control solutions for the sub-systems
3. Compose the sub-system solutions into a global composite solution.
4. Ensure that composite solution meets the objectives and requirements of the original problem.

2.1.1 Hierarchical Approaches

Decomposition is the de facto standard when engineering complex systems, and is fundamental to constructing large aerospace structures (Wertz and Larson 1999), designing and constructing any general engineered system (Blanchard and Fabrycky 2010), and constructing software of almost any size (Pressman 2009). Historically, one of the most common architectures utilized in control of complex systems is a hierarchical architecture (Findeisen 1980) (Filipovic and Meystel 1993). Hierarchical architectures are extensively studied and represent the most, have been extensively utilized, and are the basis for most aviation control systems, from onboard modern flight control systems (Stevens and Lewis 1992) to large distributed air traffic control (Krozel and Peters 2000). The traditional systems engineering practice of top-down design requirements decomposition naturally translates to a physical hierarchical system architecture. Complexity is managed through delegation of ever finer control responsibility to lower level sub-

systems, and from a system design perspective, the decision to decompose functional responsibility of a controller into a set of multiple lower level system controllers is largely driven by some aspect of complexity. Designers of a system must trade between complexity in a single component, and simplifying the single component by defining new sub-systems and delegating responsibility from the higher-level component. This dissertation makes a distinction, however, between decentralization of control authority between peer systems, which is truly a decentralization problem, and delegation of control authority to subordinate systems.

2.1.2 Reduced-Order Approaches

A fundamental method for control of large complex systems is through model reduction. The goal of model reduction is to create a lower order model of a highly complex model - which may include nonlinearities, model uncertainty, or other difficulties that pose modeling and control difficulties – and create a lower order model that preserves the input-output behavior of the higher order model. Model reduction is well known concept for modeling, but still an active field of research (Benner, Quintana-Ort and and Quintana-Ort 2000) (Gugercin and Antoulas 2004) (Li and White 2002) (Sorensen and Antoulas 2002).

Control law development imparts special requirements on the model reduction process as opposed to model order reduction for generation of open loop models. For modeling, the low order model must preserve the input-output behavior of the actual system, and properties such as sensitivity, stability and passivity must be preserved. For control law development, the goal of model order reduction is to create a low order model of the closed-loop system that captures the behavior and characteristics in the controller and plant in a closed-loop.

Model reduction when closed-loop control laws are being generated is not easy, since lower order models must be generated based on controls laws, and control laws must be generated based on the lower order models. Further, this process requires data be generated from a highly accurate model that reflects the non-linearity and complexity of the real-world system, but modeling the complex system was part of the problem to begin with. For this reason, two approaches are popular: iterative methods to generate the controller, and adaptive methods to handle model uncertainty.

The outline of a general iterative algorithm for generating control laws using model order reduction follows, as used in several applications in the literature (Afanasiev and Hinze 2001) (Kunisch and Volkwein, 2006) (Arian, Fahl and Sachs 2000) (Wortelboer, Steinbuch and Bosgra 1999), which often use black-box type simulation environments to generate data points for analysis, and the data points are used to generate the lower order model. Methods such as

POD (Chatterjee 2000) with Galerkin projections (Holmes, Lumley and Berkooz 1996.), goal oriented optimization methods (Bui-Thanh, et al. 2007), Hankel model reduction, and other parameter estimation techniques, are used to generate the low order model is from this data. The POD method of snapshots was introduced by (Sirovich 1987) and further extended in (Afanasiev and Hinze 2001) and (Ly and Tran 1998), and has been applied to Navier-Stokes fluid-based models (Afanasiev and Hinze 2001). These approaches have the benefit of not only generating models, but also extracting shapes of the major mode, as long as sufficient excitation is available. From the low order model, an optimal or near-optimal controller is generated. The controller is hooked up to the black-box simulation, and the process repeats itself. The black-box and controller simulation is executed, generating additional data points with the latest control system in the loop. A new low order model is generated from the data points. A new controller is generated from the latest reduced order model. The new controller is tied into the black-box simulation and the process repeats until a desired convergence is achieved.

2.1.3 Decentralized Control

The definition of a centralized architecture varies in the literature, and is often defined indirectly through comparison with a decentralized scheme. Generally, a centralized control architectures centralizes information flow through a common point in the component system graph. A centralized control system will exhibit one or more of the following three properties:

1. Information Centralization: Information used for control system decision making are gathered at a single control processing location;
2. Observation Centralization: Sensor processing and state estimation for control system decision making are performed at a single control processing location;
3. Control Centralization: Control responsibility and authority for the control system is allocated to a centralized location.

The advantages and trade-offs between a centralized versus decentralized scheme have been well documented in the literature – for instance, see (Sesak and Coradetti 1979) (D. Siljak 1978) (M. Singh 1981). The main benefit of decentralized control is the ability to control large complex systems where traditional approaches have difficulty. Complexity can be managed, where complexity is characterized by high dimensionality, high uncertainty, or imposed constraints on information structure or resources (Siljak 1991). Large complex systems can be decomposed into many sub-systems, yielding a smaller dimensional space for the analysis, design, and implementation of each sub-system

(Linnermann 1984)(Singh 1981). Decentralized control can provide lower order controller solutions (Li et al. 1999). Decentralized control allows for functional independence of controllers, allowing complexity to be managed through development of controllers designed independently of each other (Siljak 1991). Decentralization offers reliability and fault tolerance. Decentralized control schemes are superior to centralized schemes in terms of reliability with respect to controller failure and structural reconfigurations (Siljak 1991). Decentralized controllers provide superior robustness to uncertainties (Siljak 1991). This includes robustness to unpredictable structural perturbations where sub-systems are disconnected and again connected during operation. Many large complex systems require fast control action in response to local input and perturbations, and decentralization provides a method for distributing control and optimizing information structures to support timely control system response. Computational burden can also be distributed, supporting the trend of toward greater distributed parallel computing (Siljak 1991). While decentralized control systems are generally and necessarily suboptimal, these schemes provide a measure of robustness. Centralization minimizes control complexity for sufficiently simple systems. Decentralization of the control requires replication of components, such as communication modems and processors, requires increased local requirements, and requires sophistication in communication sub-systems. This can translate to added weight, cost, proliferation of failure modes, and implementation time and effort. Centralization facilitates optimal performance. Allocation of control authority to a single unilateral decision making with full observation of the system facilitates performance and optimality in control implementation, minimization of possible latency and delay effects, and establishes consistent information flow though time in the system. Decentralized strategies require extra consideration and analysis by control engineers to ensure the performance, stability, and correctness of the system, placing extra burdens on the control and system designer.

In general, a decentralized solution should be considered when properties of the system render a centralized approach impractical or inadmissible. As mentioned, centralization and decentralization can each offer advantages and disadvantages unique to each approach. Centralization often provides simplicity, efficiency, and better performance, and is usually the preferred solution if the system permits it. Decentralization of the control architecture introduces a number of unique challenges to the design, analysis, and validation of the system.

Decentralized control will provide suboptimal performance due to inherent nature of decentralization, stemming from a loss of feedback signals, incomplete state estimation, and multiple competing objectives, performance degradation due to communication infrastructure, and control requirements imparted by decentralization.

Decentralization requires that controllers be subject to additional simultaneous constraints, requirements and objectives (Krtolica and Siljak 1980). Designers must account for some degree of degradation. For instance, (Li, et al. 1999) analyze their decentralization strategy as a tradeoff between controller performance in a centralized scheme versus computational performance of each sub-system in the decentralized scheme. The intrinsic suboptimality of decentralization is formally characterized in (Krtolica and Siljak 1980) over a large class of decentralized control problems under stochastic uncertainty, and show that decentralization necessarily presents a trade between optimal performance and robustness/reliability.

Several challenges stem from the imposition of information structure constraints on the solution space. Structural constraints add significant difficulty to controller design, and general forms proposed for decentralized control synthesis problems under information structural constraints are intractable (Rotkowitz and Lall 2005). Many problems involve control under limited information, where decentralized controllers must autonomously make proper control decisions with limited information and awareness of the larger system (Linnemann, 1984) (Toshikawa, 1978). Controller inputs may be confined to only local information, and naïve assumptions on stability may result in degraded performance or instability in the larger system. For instance, (Duan, Wang and Huang 2006) presents systems where the stability of a complex decentralized system necessarily requires instability in individual controller's sub-systems. Limited information can be addressed through maintaining beliefs on the state of other sub-systems, which requires additional analysis to show closed loop stability is ensured (Lavaei and Aghdam 2006). Many decentralized control applications require that no communication or information exchange occur between local sub-system controllers. For many information structures, it is important to show disturbances will not propagate through the sub-systems, or that disturbance propagation will be bounded. For instance, concepts for intelligent vehicle highway systems involve platoons of autonomous vehicles operating at high speeds and closely spaced. Controllers local to each vehicle must control the vehicle based on the car in front of it, and ensure any spacing errors do not propagate through the convoy (Swaroop and Hedrick 1995) (Levine and Athans 1966) (Melzer and Kuo 1971). Overly relaxed information constraints add demands to the communication architecture, but overly strict constraints add demands to the individual controllers, and determining the right balance is an open question. Determine the minimal constraint is often a challenging problem in itself; for instance, many local vehicle controllers were developed for decentralized spacing control in a convoy before Swaroop and Hedrick showed the necessary conditions for stability of these systems

required a particular information structure; stability in this case required information from the lead vehicle and from the vehicle immediately ahead of the controlled vehicle (Swaroop and Hedrick 1995).

System complexity and scale is often a driver to decentralize. Decentralization of control authority may be necessary when systems are inundated by the sheer number of inputs and outputs. For instance, high sensor and actuator density renders centralization impractical in many large-scale civil structural control systems (Lynch and Law 2002). Bandwidth limitation in any transmission often necessitates that control laws be physically distributed in a system and partitioned into implementable sub-systems. For instance, in distributed engine control (Culley and Behbahani 2008). In formation flight control of aerial vehicles, full-state feedback is impractical due to the sheer number of states in the system, requiring decentralized solutions (Wolfe, Chichka and Speyer 1996).

Decentralization is often preferred for systems composed of spatially or geographically disjoint sub-systems. Spatial separation necessitates communication between sub-systems, which is subject to bandwidth constraints, latency, drop-outs, and unreliability. For instance, in coordination of mobile robotic manipulators (Khatib, et al. 1998). Often, communication links do not allow full information to be shared in a timely fashion, as seen in large flexible space structures. For instance, embedded sensors and wireless communication networks for control of civil structures (Y. Wang, et al. 2007). System dimension and scale may necessitate a decentralized solution; for instance, large scale building structural control (Loh and Chang 2008). Geographically distributed sensors and actuators often need to minimize information exchange between components due to finite power resources; for instance, in decentralized control of multi-area interconnected power systems (Davison and Tripathi 1978), or in distributed control of sensor networks for monitoring sea level (). Fault tolerance is often high priority in a system composed of unreliable sub-system components, as is often seen in decentralized control approaches over wireless sensor networks; for instance, fault tolerant network communication control. Wireless network systems are composed of physically decentralized hardware components that are often connected in peer-to-peer or ad hoc network topologies, and control over such systems is necessarily decentralized; for instance, active power control (Lewis 2004).

Large complex systems with numerous interconnected sub-systems and a large number of variables are difficult to analyze. Well-known approaches for small systems become quickly intractable. For instance, traditional rank condition tests for controllability and observability of systems is ill-posed and numerically intractable as systems become large (D. D. Siljak, Decentralized Control of Complex Systems 1991). Early developments in decentralization tried to show decentralized stability by strengthening stability of the sub-systems, but this is not only conservative to

require all closed-loop sub-systems be stable, it can be shown that certain contrived systems require destabilizing control in one or more of the sub-systems to ensure global stability (Duan, Wang and Huang 2006). Duan et al. observes that the necessary conditions for decentralized stability are often based on fixed modes of the system, whereas sufficient conditions are established based on Lyapunov function methods or optimized algorithms. Small gain theorem can be used to strengthen robust stability. Stabilized controllers can be created using output and state feedback (M. Singh 1981). Graph theoretic approaches have been developed that provide structural insight into the control problem for large systems (D. D. Siljak, Decentralized Control of Complex Systems 1991).

Decentralization is often necessary when distributed controllers are restricted from accessing the full set of possible input or output signals. A large class of decentralized control problems can be considered as information structural constraints that restrict the admissible controller space (Chen and Stankovic 2005). Another class of problems with overlapping system dynamics can be transformed into an expanded dimensional space where sub-systems appear to be disjoint, and decentralized control solutions can be developed on each sub-system, then transformed back into the original controller space (Yousuf, 1998). Many formulations restrict any communication of information between local sub-system controllers. Under this information constraint, each local controller needs to attenuate any degrading effect that interconnected dynamics and controllers will have on its system.

Decentralization also provides an inherent level of robustness to uncertainty in structure and disturbances (D. D. Siljak, Decentralized Control of Complex Systems 1991). Many researchers have utilized combinations of robust and adaptive approaches to enhance these properties. For instance, a fuzzy logic and H_∞ controller was used to provide adaptive and robust guarantees for a class of nonlinear MIMO systems (Yousef, et al. 2006), and distributed game theoretic logic was applied for decentralized control to address adaptation in wireless mobile network (Menon, et al. 2004).

Decentralization has been applied when the logistics of coordinating a centralized control strategy are difficult. For instance, formation control of vehicle convoys can be subject to large uncertainty in the information structural constraints, the formation structure may be free and dynamic, where the exact vehicles partaking in the formation may not be known ahead of time, the size of the formation may not be known, and knowledge of vehicle communication paring requirements may not be known in advance (Wolfe, Chichka and Speyer 1996).

Decentralization is often applied when systems are naturally composed of disjoint self-interested and self-optimizing agents, each with local sensors, controllers, and actuators (look-think-act) capabilities who are independent

but need to interact. Communication constraints or systems where agents are in competition require individual agents to infer information on neighboring agents (e.g., state, intentions, inputs) from limited observation subject to ambiguity, uncertainty, and possible deception. The theory of probability collectives introduced a probabilistic game theoretic framework for analysis and control of these types of systems (Wolpert 2006), which includes hybrid continuous-discrete time systems (Bieniawski, Wolpert and Kroo, Discrete, Continuous, and Constrained Optimization Using Collectives 2004) and adaptive flutter suppression on flight control surfaces (Bieniawski, Kroo and Wolpert, Flight Control with Distributed Effectors 2005). Probability collective theory was used to develop the product distribution framework for controlling multi-agent systems (Lee and Wolpert 2004). A framework using N-inference-observability for decentralized control of discrete event systems for inferencing over ambiguities was presented in (Shigemasa and Ratnesh 2008). Local stations often must protect again worst-case response from neighboring agents. Exact inferencing can be very costly, and alternatives can be considered. For instance, inference avoidance algorithms on wireless systems for adaptation (Rose, Ulukus and Yates 2002) (Menon, et al. 2004) or distributed waveform-adaptation algorithms in (Ulukus and Yates 2001) which required minimal feedback between receivers and transmitters. Many multi-agent systems must handle adversarial situations where other agents may try to minimize the objectives of other agents (Rehák, Pěchouček and Tožička 2005). Decentralized control for advanced air traffic systems is surveyed in (Tomlin, et al. 2006) and in (Krozel and Peters 2000). Air traffic management involves competing self-interested airlines or aircraft, for instance using game theoretic negotiated settlement to provide onboard separation assurance in a decentralized air traffic control system (Wollkind, Valasek and Ioerger 2004). Other air traffic applications include conflict resolution in terminal approach areas, where stability can be measured by how many requests are generated in response to a conflict notification (Iamratanakul, et al. 2004).

2.2 Decentralized Control as a Convex Optimization Problem

The problem of decentralized control formulation has been characterized in the literature as a problem in sparsity or information constraints on the controllers being designed. A canonical form for decentralized control problems follows (Rotkowitz and Lall 2005). Let $\mathcal{R}_p^{i\times j} = \{G: j\mathbb{R} \to \mathbb{C}^{m\times n} | G\ proper, real-rational\}$ be the set of all matrix-valued proper real-rational transfer function matrices. Let $S \subseteq \mathfrak{R}_p^{n_u \times n_p}$ be some subspace of admissible controllers. Given $P \in \mathfrak{R}_p^{(n_z+n_y)\times(n_w+n_u)}$, solve the following problem:

$$\begin{aligned} &\text{minimize } \|f(P,K)\| \\ &\text{subject to: } K \text{ stabilizes } P \\ &K \in S \end{aligned} \tag{2.2-1}$$

Unfortunately, even simple problems in this context can be extremely difficult (Witsenhausen 1968) or intractable (Tsitsiklis 1986) (Blondel and Tsitsiklis 2000). Solutions to special cases have been identified in the literature, including (Fan, Speyer and Jaensch 1994) (Ho and Chu 1972) (Bamieh and Voulgaris, 2002) (Bamieh and Voulgaris, 2005) (Qi, et al. 2004) (Voulgaris 1999). Recent literature (Rotkowitz and Lall 2002) (Rotkowitz and Lall 2005) has recast a large class of problems under a common framework of convex optimization.

This approach extends Youla parameterization for a class of decentralized problems where the constraint set is quadratically invariant to the system. The set of closed-loop maps that are stabilizing controllers can be parameterized given a coprime factorization of the plant. This set of controllers is affine under these parameters, which is the key result that converts the problem of synthesizing an optimal controller into a problem of convex optimization. A two-step compensation scheme can be used for strongly stabilized plants (stabilizable through a stable compensator), where an initial stable stabilizing compensator is used for the factorization, and an optimization step occurs to find the optimal controller under the parameterization.

In equation (2.2-1), $f(P,K)$ is the lower linear fractional transform, and the $\|\cdot\|$ is any norm on $\mathfrak{R}_p^{n_z \times n_w}$ and in general, the norm $\|f(P,K)\|$ may be a non-convex function of K. Here, n_u represents the size of the inputs, n_y the size of the outputs. The subspace constraint of K in S, the information constraint, makes this problem interesting; less the information constraint on K, this problem can be solved by a direct change in variables approach using the well-known Youla parameterization. No tractable algorithm exists for the general solution (Rotkowitz and Lall 2005).

The general constraint where Q is non-convex is given as

$$K_{nom} - h(h(K_{nom}, G), Q) \in S \tag{2.2-2}$$

The results below will recast the problem as a convex optimization problem, where

$$\begin{aligned} &\text{minimize } \|T_1 - T_2 Q T_3\| \\ &\text{subject to: } Q \in \mathcal{RH}_\infty \\ &Q \in S \end{aligned} \tag{2.2-3}$$

The condition of quadratic invariance on the constraint set relative to the plant is necessary and sufficient for the constraint S to be invariant under a broad class of LFT. The condition for the Youla parameter Q to be convex is established, the authors show they can recast problem of finding optimal controllers as a convex optimization problem, and further investigate the influences of delay disturbances (particularly, transmission delay, physical phenomena propagation delay and computation delay).

2.3 Graph-Based Methods for Decentralization and Partitioning

A fundamental problem for decentralized control designers is to determine an appropriate partitioning scheme. In certain applications, sub-system partitions intuitively lie on functional, logical, spatial, or temporal boundaries; for instance, formation control of a vehicles platoon provides natural boundaries for decomposing the higher dimensional platoon system into a set of individual vehicle systems that are lower dimensional, but overlapped and interoperating. In other cases, the partition is not as obvious; decomposition schemes for complex structural systems or fluid flow system is a non-trivial problem with many possible approaches.

Large complex systems with numerous interconnected sub-systems and a large number of variables are difficult to analyze. Well-known approaches for small systems become quickly intractable. For instance, traditional rank condition tests for controllability and observability of systems is ill-posed and numerically intractable as systems become large (D. Siljak 1978). Further, the rank tests do not provide useful insight, such as how to establish controllability or observability. To address these issues, graph theoretic analysis for decentralized control of large scale systems has been extensively studied in the literature (D. Siljak 2011) (Yamada and Foulds 2006).

Graph-based approaches have become a well-established method for decentralization and continue to be a subject of recent research focus, particularly utilizing directed graphs (Mesbahi et al, 2010) (Sundaram and Hadijcostic, 2011) (Gharesifard and Cortes, 2012) (Cao et al, 2008). Directed graphs (digraphs) were utilized for vehicle formation control in (Lafferriere et al, 2005). Several approaches had similarly used leader-following methods that dictate a hierarchical control structure or utilize consensus-based approaches (Sayyaadi and Soltani, 2016). Online methods for maintaining strong connectivity in directed graphs were presented in (Sabattini et al, 2013). Dynamic structure graphs were explored for robotic networks with limited sensor field of view in (Paola et al, 2012). The impact of symmetry and sparsity was explored utilizing system digraphs in (Kirkoryan and Belabbas, 2014). Similar digraph system method was used for decentralized system identification of a building in (Agbi and Krogh, 2014). In (Desai, Kumar and Ostrowski, Modeling and control of formations of nonholonomic mobile robots 2001), control of a team

of nonholonomic mobile robots navigating a terrain with obstacles was explored, while maintaining a desired formation and changing formations when required. The algorithm presented enumerates possible control graphs and coordinates transition between two formations.

Structural information constraints on a decentralized control system dictate how information can flow within the system. The structure of a complex system has precise definition in terms of directed graphs (Georgiou and Floudas 1990). Any dynamical system of the form in (4.1-23) can be represented by a directed graph G=(V,E), where the vertex set $V = U \cup X \cup Y$ contain the inputs, state, and output vertices, where $U = \{u_1 \dots u_m\}, V = \{x_1 \dots x_n\}, Y = \{y_1 \dots y_l\}$. The edge set $E = (V, V)$ is defined where an edge $e = (v_1, v_2)$, $v_1, v_2 \in V$, exists if and only if $e_{ij} = 1$, where $E = [e_{ij}]$ is the interconnection matrix of the system. For the time-invariant system form of (4.1-23) with D=0, the interconnection matrix E is a binary matrix of size $(n \times l) \times (n \times m \times l)$ is given by the following relationships, where $\tilde{A}, \tilde{B}, \tilde{C}$ are binary matrices with identical dimensions to A, B, and C.

$$E = \begin{bmatrix} \bar{A} & \bar{B} & 0 \\ 0 & 0 & 0 \\ \bar{C} & 0 & 0 \end{bmatrix} \quad where \quad \bar{a}_{ij} = \begin{cases} 1 \; if \; a_{ij} \neq 0 \\ 0 \; if \; a_{ij} = 0 \end{cases} \quad \bar{b}_{ij} = \begin{cases} 1 \; if \; b_{ij} \neq 0 \\ 0 \; if \; b_{ij} = 0 \end{cases}$$
$$\bar{c}_{ij} = \begin{cases} 1 \; if \; c_{ij} \neq 0 \\ 0 \; if \; c_{ij} = 0 \end{cases} \tag{2.3-4}$$

The structural matrices $\tilde{A}, \tilde{B}, \tilde{C}$ give insight into the connections that exist within the system's structure. A sub-system can be defined by a subgraph $G_k = (V_k, E_k)$ where $E_k = (V_k \times V_k) \cap E$. A subgraph G_k is strongly connected if every vertex in V_k is reachable under E_k from every other vertex in V_k. A subgraph G_k is a strong component if it is strongly connected, and there are no cycles that exist in G that include both vertices in V_k and vertices not in V_k. The condensation of a system is found through collapsing strong components of G into a single vertex. Condensations can be used to identify partitions in large systems, for system decomposition and order reduction (Yamada and Foulds 2006). In (Evangelatos 1995), model dimensional reduction is achieved using the condensation of partial sub-systems and the condensed reachability matrix to achieve a strongly connected component decomposition of reduced dimensions.

The concept of structural controllability on $(\tilde{A}, \tilde{B})$ and structural observability on $(\tilde{A}, \tilde{C})$ was introduced in (C. T. Lin 1974) based on the system's structural matrices. Note that two systems are structurally equivalent if their corresponding structural matrices $\tilde{A}, \tilde{B}, \tilde{C}$ are similar (equivalent under a similarity transformation). Sufficient conditions for structural observability and controllability are summarized in (D. D. Siljak, Decentralized Control of

Complex Systems 1991). The concept of system graph reachability for the design of controllers and estimators on large systems was introduced in (D. D. Siljak, On reachability of dynamic systems. 1977). A vertex v_i is reachable from v_j if there exists a directed path from v_j to v_i. The set of all vertices reachable from a vertex or a set of vertices defines the reachable set. The antecedent set of a set of vertices V_i is the set of all vertices in V from which at least one vertex in V_i is reachable. A system with graph $G = (U \cup X \cup Y, E)$ is input reachable if the state vertex set X is a reachable set of the input vertex set U. A system is output reachable if X is an antecedent set of Y. The reachability matrix $R = [r_{ij}]$ is defined as the binary matrix where $r_{ij} = 1$ if v_i is reachable from v_j, otherwise $r_{ij} = 0$. Input and output reachability can be loosely considered as necessary conditions for controllability and observability over typical decentralized systems (D. Siljak 2011) .

2.4 Decomposition and Coupling Metrics

A fundamental challenge in decentralization and partitioning when posed as a direct optimization over a topology is the need for metrics that can measure the relative interaction strength between various components of the large-scale system. Many approaches have been proposed in the literature as a metric for analyzing large structures or for decomposition. These include Rijnsdorp's interaction measure (1965),Rosenbrock's Direct Nyquist Array (1974), μ-interaction measure (Grosdidier and Morari, 1986), generalized in (Skogestad and Morari, 1989), ϵ-decomposition (Zečević and Šiljak,1995) (Siljak, 1991 and 1996), overlapped and nested epsilon decomposition (Selzer and Siljak, 1991), Balanced BBD decomposition (Zečević and Šiljak, 1994), RGA derived measures, BRGA (Samyudia, 1995). Many of the decomposition approaches require assuming the large-scale system has some predefined form. Three particular forms are presented in (Chen and Stankovic, 2005) and solved using the inclusion principle. Longitudinal (chain) structures assume each sub-system $\boldsymbol{S}_i$ are shared only by adjacent sub-systems $\boldsymbol{S}_{i-1,i}$ and $\boldsymbol{S}_{i,i+1}$. Loop (circle) structures are similar to longitudinal structures except the first and last systems are shared. Radial star structures are in which one sub-system is shared by the remaining ones (Shoenwald, 2000).

Epsilon decomposition techniques as outlined in (Siljak, 1996) decomposes a large sparse system into weakly coupled sub-systems through the following.

$$A = A_0 + \epsilon A_c \;\; ; \;\; B = B_0 + \epsilon B_c \;\; ; \;\; C = C_0 + \epsilon C_c \tag{2.4-5}$$

Here, ϵ is a small positive number, the matrices A_0, B_0, and C_0 have block-diagonal structure, and all the elements of A_c, B_c, C_c are smaller than one in magnitude. One advantage of this approach stems from the fact that discarding

the ϵA_c term greatly increases the sparsity of A. An algorithm for epsilon decomposition was described in (Zečević and Šiljak, 1995) (Taoka, 1992). (Amano 1996) expanded this work using block-parallel Newton iterative method to accelerate convergence. Overlapping epsilon decomposition assumes A_0, B_0, and C_0 have overlapping block diagonal forms that is amenable to methods such as the inclusion principle. (Sezer and Šiljak, 1991) (Šiljak, 1991) (Zečević and Šiljak, 1994).

The potential methods listed above are summarized in Table 1. The method of ϵ-decomposition (Zečević and Šiljak,1995) and expanded in (Sezer and Šiljak, 1991) is applied in the dissertation approach. However, application of other metrics may provide a better basis for optimized partitions depending on the application. Further research into applying alternative metrics would be needed to understand the trade-off and effectiveness of utilizing various metrics.

Table 1. Alternative Metrics for Graph Connectivity Analysis and Partitioning

Rijnsdorp's interaction measure (1965)
Rosenbrock's Direct Nyquist Array (1974)
μ-interaction measure (Grosdidier and Morari, 1986), and generalized in (Skogestad and Morari, 1989)
ϵ-decomposition (Zečević and Šiljak,1995) (Siljak, 1991 and 1996), overlapped and nested epsilon decomposition (Selzer and Siljak, 1991)
Balanced BBD decomposition (Zečević and Šiljak, 1994)
RGA derived measures, BRGA (Samyudia, 1995), DRGA
Structured singular values (SSV)
J-spectral factorization (Seo at al, 1999)

2.5 Dynamic System Graphs

The popularity of wireless networks has given rise to recent interested into decentralized control on dynamic information structural topologies, where similar concepts of controllability over these structures have recently been explored through graph theoretic formulation (Ji, Lin and Lee 2008). (Tanner 2004) derives necessary and sufficient conditions for controllability on agent-based networks with nearest neighbor laws with follow the leader rules. (Zamani and Lin 2009) utilize similar formulations to show necessary and sufficient conditions for multi-agent structural controllability in a dynamic network, where connectivity is shown as necessary and sufficient for structural controllability. (Rahman, et al. 2009) considers control algorithms on multi-agent networks with similar leader-

follower relationships, and shows that symmetry in the network, as characterized by automorphism groups, directly relates to the controllability properties of the system.

Utilizing a simpler graph definition that presented above, wireless network of mobile robots were modeled as an undirected graphs in (Winfield 2000). The graph definitions were used to generate decentralized control laws for robotic formation control in (Desai, Ostrowski and Kumar, Controlling formations of multiple mobile robots 1998). This approach was extended in (Desai, Kumar and Ostrowski, Modeling and control of formations of nonholonomic mobile robots 2001) to controlling a team of robots navigating terrain with obstacles while maintaining a desired formation. The robotic team is allowed to change formations when needed. Handling changing formations requires a dynamic graph formulation. In this case, the transition matrix between the current adjacency matrix and all possible control graphs are evaluated.

Directed graph theory was used to analyze control of a team of robots navigating terrain with obstacles while maintaining a desired formation and changing formations when needed (Desai, 2001). At each time, the transition matrix between the current adjacency matrix and all possible control graphs must be considered, and the authors present a detailed analysis for this particular problem. The constraints proposed can be seen as specification of necessary conditions that must be imposed on the controller graph, such as those defined for structural reachability and structural observability in (D. Siljak 2011). The graph approach in robotics often builds approaches that are difficult to rigorously analyze. For comparison, consider the same problem of decentralized control of cooperative robotic vehicles and formations as presented in (Feddema, Lewis and Schoenwald 2002).

2.6 Overlapping Partitions

The inclusion principle (Ikeda, Siljak and White 1984) is a mathematical framework for dealing with large complex systems with overlapping interconnected sub-system and overlapping information structure constraints. A complex system S can be transformed to a higher dimensional expanded space, where the expanded system $\tilde{S}$ maintains essential information about the initial system. Designers must define the relationship between the initial contracted system and the expanded system through selection of the appropriate linear transformation. A set of complementary matrices can be selected to satisfy a set of necessary and sufficient conditions to ensure the existence of solutions, and the choice of these matrices will influence the controllability, observability, stabilizability and detectability of the system. Controllers can be designed on $\tilde{S}$. The final solution is obtained by transforming

(contracting) the controller back to the original space. The inclusion principle provides a method to expand a complex overlapping system into a higher dimensional space in which the overlapped sub-systems appear disjoint. The method has been applied a wide range of applications involving large complex systems with continuous, discrete, and stochastic elements. This theory was applied in (Li, et al. 1999) for an H_∞ robust controller design for a segmented telescope where computational resources were limited, and decentralization reduced the order of the controller at the cost of performance. (Iftar and Ozguner 1998) and (Stankovic, Stanojevic and Siljak 2000) applied the inclusion principle to formation control for a platoon of vehicles. Formation control for autonomous aerial vehicles was presented in (Wolfe, Chichka and Speyer 1996) and (Stipanovic, et al. 2004). Automated optimization of decentralized control structural was invested in (Palacios, Pujol, Rodellar, & Rossell, 2006), which presented a numerical approach to iteratively determine the M and N matrices for aggregation systems where all other parameters are given a priori. The inclusion principle was extended to distributed time-delay systems in (İftar 2014). Similarly, the inclusion principle was extended for robustness to time-delay in (Ahmadi and Aldeen 2016). (Chen and Xeubo 2014) applied the inclusion principle to coordinated consensus control of a multi-agent system. In (Stanković et al. 2014), the expansion/contraction paradigm based on the restriction conditions is applied to decentralized tracking control with a sequential LQ optimization process, assuming information structure constraints are given.

2.6.1 Inclusion Principle Formulation

Consider a pair of two dynamical **S** and $\tilde{\mathbf{S}}$

$$\mathbf{S}: \quad \begin{matrix} \dot{x}(t) = Ax(t) + Bu(t) \\ y = Cx \end{matrix} \quad ; \quad \tilde{\mathbf{S}}: \quad \begin{matrix} \dot{x}(t) = \tilde{A}\tilde{x}(t) + \tilde{B}\tilde{u}(t) \\ \tilde{y} = \tilde{C}\tilde{x} \end{matrix} \tag{2.6-6}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, and $y(t) \in \mathbb{R}^l$ are the state, input, and output **S** at time t, and $\tilde{x}(t) \in \mathbb{R}^{\tilde{n}}$, $\tilde{u}(t) \in \mathbb{R}^{\tilde{m}}$, and $\tilde{y}(t) \in \mathbb{R}^{\tilde{l}}$ are for $\tilde{\mathbf{S}}$, let the input $u(t) \in \mathbb{R}^m$. Let the matrices $A \in \mathbb{R}^{n\times n}, \tilde{A} \in \mathbb{R}^{\tilde{n}\times\tilde{n}}, B \in \mathbb{R}^{n\times m}, \tilde{B} \in \mathbb{R}^{\tilde{n}\times\tilde{m}}, C \in \mathbb{R}^{l\times n}, \tilde{C} \in \mathbb{R}^{\tilde{l}\times\tilde{n}}$ be constant. The state, input, and output vectors of S are smaller (or at most equal to) S (so that $n \leq \tilde{n}$, etc.).

The systems are related by the transforms shown below. U and V defined as $V: \mathbb{R}^n \to \mathbb{R}^{\tilde{n}}$ and $U: \mathbb{R}^{\tilde{n}} \to \mathbb{R}^n$, where $V \in \mathbb{R}^{\tilde{n}\times n}, U \in \mathbb{R}^{n\times\tilde{n}}, rank(V) = n, UV = I_n$. The right pseudo-inverse of V (by Moore-Penrose) can be found from V by $U = (V^TV)^{-1}V^T$. Similar for R and Q (for inputs), and for S and T (for outputs).

$$\tilde{x} = Vx \quad , \quad x = U\tilde{x} \quad , \quad \tilde{u} = Ru \quad , \quad u = Q\tilde{u} \quad , \quad \tilde{y} = Ty \quad , \quad y = S\tilde{y} \tag{2.6-7}$$

Definition: (Inclusion Principle) A system **S** *includes* the system $\tilde{\mathbf{S}}$, denoted by $\tilde{\mathbf{S}} \supset \mathbf{S}$, if there exists a quadruplet of matrices (*U,V,R,S*) such that for any initial state x_0 and any fixed input $u(t)$ of **S**, the choice $\tilde{x}_0 = Vx_0$ and $\tilde{u}(t) = Ru(t)$ for all $t \geq 0$ of the initial state $\tilde{x}_0$ and the input $\tilde{u}(t)$ of the system $\tilde{\mathbf{S}}$ implies $x(t; x_0, u) = U\tilde{x}(t; Vx_0, u)$ and $y[x(t)] = S\tilde{y}[\tilde{x}(t)]$ for all $t \geq 0$.

If $\tilde{\mathbf{S}} \supset \mathbf{S}$, then $\tilde{\mathbf{S}}$ is said to be an *expansion* of **S**, and **S** is a *contraction* of $\tilde{\mathbf{S}}$.

Definition: (Restriction) A system **S** is a restriction of $\tilde{\mathbf{S}}$ if there exists a pair of matrices (U, V) satisfying $UV = I$ and such that for any initial state x_0 and any fixed input $u(t)$ of **S**, the choice $\tilde{x}_0 = Vx_0$ implies $\tilde{x}(t; \tilde{x}_0, u) = Vx(t; x_0, u)$ for all $t \geq 0$.

Definition: (Aggregation) A system **S** is an aggregation of $\tilde{\mathbf{S}}$ if there exists a pair of matrices (U, V) satisfying $UV = I$ and such that for any initial state $\tilde{x}_0$ and any fixed input $u(t)$ of $\tilde{\mathbf{S}}$, the choice $x_0 = U\tilde{x}_0$ implies $x(t; x_0, u) = U\tilde{x}(t; \tilde{x}_0, u)$ for all $t \geq 0$.

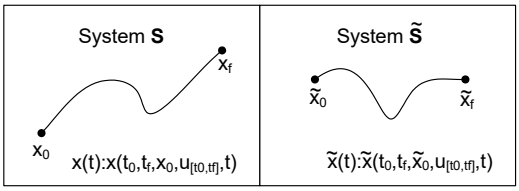


Figure 2-2. Inclusion Principle Definitions.

***S** includes $\tilde{\mathbf{S}}$ implies* $(\tilde{x}_0 = Vx_0) \Rightarrow \left(x(t)|_{x_0,u} = U\tilde{x}(t)|_{\tilde{x}_0,u}\right)$

***S** is a restriction of $\tilde{\mathbf{S}}$ implies* $(\tilde{x}_0 = Vx_0) \Rightarrow \left(\tilde{x}(t)|_{\tilde{x}_0,u} = Vx(t)|_{x_0,u}\right)$

***S** is an aggregation of $\tilde{\mathbf{S}}$ implies* $(x_0 = U\tilde{x}_0) \Rightarrow \left(x(t)|_{x_0,u} = U\tilde{x}(t)|_{\tilde{x}_0,u}\right)$

The matrices $\tilde{A}$, $\tilde{B}$, and $\tilde{C}$ can be expressed as

$$\tilde{A} = VUA + M \quad ; \quad \tilde{B} = VBQ + N \quad ; \quad \tilde{C} = TCU + L \tag{2.6-8}$$

Here, M, N, and L are complementary matrices. Typically, V, R, and T are defined apriori to establish the structural relationships between the two systems, and the complementary matrices M, N, and L give freedom to build an expanded system from the original system to meet certain specifications.

Theorem 2.6-1. $\tilde{\mathbf{S}} \supset \mathbf{S}$ if and only if $UM^iV = 0$, $UM^{i-1}N = 0$, $SLM^{i-1}V = 0$ and $SLM^{i-1}NR = 0$ for all $i = 1, 2, \ldots, \tilde{n}$

Proposition 2.6-1. **S** is a restriction of $\tilde{\mathbf{S}} \iff MV = 0 \ \wedge \ N = 0$

Proposition 2.6-2. **S** is an aggregation of $\tilde{\mathbf{S}} \iff UM = 0 \ \wedge \ UN = 0$

Theorem 2.6-2. If N=0 then the expanded system $\tilde{\mathbf{S}}$ is uncontrollable. A proof is proved in (Bakule, Rodellar and Rossell 2001) (L. Bakule, J. Rodellar and J. Rossell, et al. 2001) and (Malinowski and Singh 1985)

Proposition 2.6-3. **S** is a restriction of $\tilde{\mathbf{S}}$ if one of the following is true, as proven in (Stipanovic, et al. 2004).

a) $\tilde{A}V = VA$, $\tilde{B}R = VB$, and $\tilde{K}V = RK$.

b) $\tilde{A}V = VA$, $\tilde{B} = VBQ$, and $K = Q\tilde{K}V$

2.6.2 Overlapping System Structure

The general form of an overlapped system below has been studied extensively in the literature. A system in this form can be extended to a system of n overlapping and coupled sub-systems. Consider the following system **S** defined as

$$\mathbf{S}: \quad A = [A_{i,j}] \quad , \quad B = [B_{i,j}] \quad , \quad C = [C_{i,j}] \quad , \quad i,j = 1,2,3 \tag{2.6-9}$$

The dimensions of $x=(x_1^T,x_2^T,x_3^T)^T$ are n_1, n_2, and n_3, respectively, where $n_1+n_2+n_3=n$. Similarly, the dimensions of $u=(u_1^T,u_2^T)^T$ are m_1 and m_2, such that $m_1+m_2=m$. Consider **S** is composed of two sub-systems $\mathbf{S}_1$ and $\mathbf{S}_2$, with overlapping state, control, and output.

$$\begin{aligned} \mathbf{S}_1: &\quad \{[A_{i,j}],[B_{i,j}],[C_{i,j}], \ \ i,j = 1,2\} \\ \mathbf{S}_2: &\quad \{[A_{i,j}],[B_{i,j}],[C_{i,j}], \ \ i,j = 2,3\} \end{aligned} \tag{2.6-10}$$

The overlap occurs in A_{22}, B_{22}, and C_{22}. A standard selection for V, R, T which has been extensively studied in the literature is (Palacios, et al. 2006) (Bakule, Rodellar and Rossell 2001):

$$V = \begin{bmatrix} I_{n_1} & & \\ & I_{n_2} & \\ & I_{n_2} & \\ & & I_{n_3} \end{bmatrix} \ ; \ R = \begin{bmatrix} I_{m_1} & & \\ & I_{m_2} & \\ & I_{m_2} & \\ & & I_{m_3} \end{bmatrix} \ ; \ T = \begin{bmatrix} I_{l_1} & & \\ & I_{l_2} & \\ & I_{l_2} & \\ & & I_{l_3} \end{bmatrix} \tag{2.6-11}$$

The expanded system has weaker interconnections but has repeated state, inputs, and outputs, given by $\tilde{x} = (x_1^T, x_2^T, x_2^T, x_3^T)$ where $\tilde{n} = n_1 + 2n_2 + n_3$, and similarly for y and u. The inverse transform V is found to be $U = (V^TV)^{-1}V^T = \begin{bmatrix} I_{n_1} & & & \\ & \frac{1}{2}I_{n_2} & \frac{1}{2}I_{n_2} & \\ & & & I_{n_3} \end{bmatrix}$, and similar for R and T. This yields:

$$\tilde{A} = \begin{bmatrix} A_{11} & \frac{1}{2}A_{12} & \frac{1}{2}A_{12} & A_{13} \\ A_{21} & \frac{1}{2}A_{22} & \frac{1}{2}A_{22} & A_{23} \\ A_{21} & \frac{1}{2}A_{22} & \frac{1}{2}A_{22} & A_{23} \\ A_{31} & \frac{1}{2}A_{32} & \frac{1}{2}A_{32} & A_{33} \end{bmatrix} + M \quad ; \quad \tilde{B} = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix} + N \tag{2.6-12}$$

Theorem 2.6-3. Consider a system **S** as given above with the structure (2.6-10), and a transform V,R, and T specified in (2.6-11). Then $\tilde{\mathbf{S}} \supset \mathbf{S}$ iff the complementary matrices M, N, and L have the following form

$$M = \begin{bmatrix} 0 & M_{12} & -M_{12} & 0 \\ M_{12} & M_{22} & M_{23} & M_{24} \\ -M_{12} & -(M_{22} + M_{23} + M_{33}) & M_{33} & -M_{24} \\ 0 & M_{24} & -M_{24} & 0 \end{bmatrix} \tag{2.6-13}$$

This satisfies the following conditions for all i=2,…,$\tilde{n}$.

$$\begin{aligned} &\begin{bmatrix} M_{12} \\ M_{23} + M_{33} \\ M_{42} \end{bmatrix} [M_{22} + M_{33}]^{i-2} [M_{21} \quad M_{22} + M_{23} \quad M_{24}] = 0 \\ &\begin{bmatrix} M_{12} \\ M_{23} + M_{33} \\ M_{42} \end{bmatrix} [M_{22} + M_{33}]^{i-2} [N_{21} \quad N_{22} + N_{23} \quad N_{24}] = 0 \\ &\begin{bmatrix} L_{12} \\ L_{23} + L_{33} \\ L_{42} \end{bmatrix} [M_{22} + M_{33}]^{i-2} [M_{21} \quad M_{22} + M_{23} \quad M_{24}] = 0 \end{aligned} \tag{2.6-14}$$

This satisfies the following conditions for all i=2,…,$\tilde{n} + 1$.

$$\begin{bmatrix} L_{12} \\ L_{23} + L_{33} \\ L_{42} \end{bmatrix} [M_{22} + \mathrm{M}_{33}]^{i-2} [N_{21} \quad N_{22} + N_{23} \quad N_{24}] = 0 \tag{2.6-15}$$

where N, L have the same structure as M.

Substituting the results of (2.6-13) into (2.6-12) yields the form for $\tilde{A}$ (similar for $\tilde{B}$ and $\tilde{C}$).

$$\tilde{A} = \begin{pmatrix} A_{11} & \frac{1}{2}A_{12} + M_{12} & \frac{1}{2}A_{12} - M_{12} & A_{13} \\ A_{21} + M_{21} & \frac{1}{2}A_{22} + M_{22} & \frac{1}{2}A_{22} + M_{23} & A_{23} + M_{24} \\ A_{21} - M_{21} & \frac{1}{2}A_{22} - (M_{22} + M_{23} + M_{33}) & \frac{1}{2}A_{22} + M_{33} & A_{23} - M_{24} \\ A_{31} & \frac{1}{2}A_{32} + M_{42} & \frac{1}{2}A_{32} - M_{42} & A_{33} \end{pmatrix} \tag{2.6-16}$$

2.6.3 Controllability and Observability

The following results concern the transmission of controllability and observability from the contracted system **S** to the expanded system $\tilde{\mathbf{S}}$. The observability and controllability of $\tilde{\mathbf{S}}$ can be maintained in situations explored in (L. Bakule, J. Rodellar and J. Rossell, et al. 2001), which corrects earlier assertions in (Malinowsk and Singh 1985) that claimed that both observability and controllability of the contracted system could not be preserved in the expanded system. Bakule presents a method for synthesis that has more degrees of freedom than the typical definitions of restriction and aggregation.

In particular, Bakule looks at two sub-cases. In case (a), the matrices on the left in (2.6-14) are zero, then

$$\begin{aligned} \text{Case (a):} \quad & M_{12} = 0, \quad M_{23} + M_{33} = 0, \quad M_{42} = 0 \\ & L_{12} = 0, \quad L_{23} + L_{33} = 0, \quad L_{42} = 0 \end{aligned} \tag{2.6-17}$$

In case (b), the matrices on the right in (2.6-14) are zero, then

$$\begin{aligned} \text{Case (b):} \quad & M_{12} = 0, \quad M_{23} + M_{33} = 0, \quad M_{42} = 0 \\ & N_{12} = 0, \quad N_{23} + N_{33} = 0, \quad N_{42} = 0 \end{aligned} \tag{2.6-18}$$

Theorem 2.6-4. (Case a) Consider systems **S** and $\tilde{\mathbf{S}}$ given in (2.6-6), where the initial system S is controllable-observable and the expanded system S is of the form (2.6-16), and satisfying (2.6-17). Then there always exists submatrices $M_{i,j}, N_{i,j}, L_{i,j}$ ensuring that $\tilde{\mathbf{S}}$ is controllable-observable.

Theorem 2.6-5. (Case b) Consider systems **S** and $\tilde{\mathbf{S}}$ given in (2.6-6), where the initial system S is controllable-observable and the expanded system S is of the form (2.6-16), and satisfying (2.6-18). Then there always exists submatrices $M_{i,j}, N_{i,j}, L_{i,j}$ ensuring that $\tilde{\mathbf{S}}$ is controllable-observable.

For the systems in *Theorem 2.6-4* and *Theorem 2.6-5*, selection of $M_{21} = A_{21}$, $M_{22} = \frac{1}{2}A_{22}$, and $M_{24} = A_{23}$, with the appropriate selection of matrix M and P, guarantees controllability and observability.

2.7 LQ Synthesis with Optimal Complementary Matrix Selection

Within the inclusion principle framework, methods for automated selection and optimization of the M and N matrices have been explored. The following case is studied in (Palacios, et al. 2006). Consider a pair of optimal control problems which share a common input u ($u = \tilde{u}$).

$$\begin{aligned} &\min_u J(x_0, u) = \int_0^\infty [x^T(t)Q^*x(t) + u^T(t)R^*u(t)]dt \\ &s.t.\ \mathbf{S}: \dot{x}(t) = Ax(t) + Bu(t) \end{aligned} \tag{2.7-1}$$

and

$$\begin{aligned} &\min_u \tilde{J}(\tilde{x}_0, u) = \int_0^\infty \left[\tilde{x}^T(t)\tilde{Q}^*\tilde{x}(t) + u^T(t)\tilde{R}^*u(t)\right]dt \\ &s.t.\ \tilde{\mathbf{S}}: \dot{\tilde{x}}(t) = \tilde{A}\tilde{x}(t) + \tilde{B}u(t) \end{aligned} \tag{2.7-2}$$

Let weighting matrices $\tilde{Q}^*, Q^*$ be symmetric positive semi-definite, let $\tilde{R}^*, R^*$ be symmetric positive definite. Define the following relationships between **S** and $\tilde{\mathbf{S}}$, where M and N are free to be selected by the designer.

$$\begin{aligned} \tilde{A} = VAU + M \ \ ;\ \tilde{B} = VB + N \\ \tilde{Q}^* = U^TQ^*U + M_{Q^*} \ \ ;\ \tilde{R}^* = R^* + N_{R^*} \end{aligned} \tag{2.7-3}$$

Matrices M_{Q^*}, N_{R^*} must be chosen in such a way that $\tilde{Q}^*$ and $\tilde{R}^*$ are symmetric positive semi-definite and symmetric positive definite, respectively. Let system (2.7-1), the matrices A,B,Q^*,R^*, and the transforms U and V be given, where A,B are given by (2.6-10). Consider here that the complementary matrices M_{Q^*}, N_{R^*} are fixed apriori, so the cost function $\tilde{J}$ is completely defined. Determine the complementary matrices M and N, defined in (2.7-3), that minimizes $\tilde{J}$.

The algorithm takes two stages

1. Select an initial M_0 and N_0 such that $\tilde{\mathbf{S}} \supset \mathbf{S}$ and the expanded system $\tilde{\mathbf{S}}$ is controllable.
2. Use an iterative routine seeking the optimal complementary matrices M and N such that $\tilde{J}$ in the expanded space $\tilde{\mathbf{S}}$ is minimized.

To select initial M and N, first note that if $\tilde{\mathbf{S}} \supset \mathbf{S}$, then **S** is an aggregation of $\tilde{\mathbf{S}}$. *Proposition 2.6-2* holds. The matrices M_{Q^*}, N_{R^*} are constructed in this manner in such a way that $\tilde{Q}^*$ and $\tilde{R}^*$ are symmetric positive semi-definite and symmetric positive definite, respectively. The results are used to select the initial M_0 and N_0 with structure given in (2.6-11). From these results, the expanded matrices ($\tilde{A}_0, \tilde{B}_0, \tilde{Q}^*, \tilde{R}^*$) are determined from M_0 and N_0 through (2.7-3).

The iterative step proceeds as follows. At any iteration i, given M_i and N_i:

1. Compute $\tilde{A}_i = VAU + M_i$ and $\tilde{B}_i = VB + N_i$
2. Verify the pair $(\tilde{A}_i, \tilde{B}_i)$ is controllable.
3. Minimize $\tilde{J}$ over M and N vary, using a numerical routine (Matlab function *fmincon*, which determines minimum of constrained nonlinear multivariable function). This determines the new optimal M_{i+1} and N_{i+1}.
4. Calculate $\tilde{P}_i$ using a linear quadratic state-feedback regulator design (the unique solution to the resulting algebraic Riccati equation).
5. Compute the value of the cost function at iteration i through $\tilde{J}_i = tr(\tilde{P}_i)$.

2.8 Planning and Trajectory Optimization for Constrained Dynamic System

Constrained trajectory optimization for autonomous vehicles is a fundamental problem in robotics and aerial vehicle control. Mathematical frameworks from the optimization and control literature typically require significant computation and human interaction when addressing nonlinear plants and com-plex constraints. The field of robotics has developed fast computational algorithms to quickly generate feasible paths with complex boundaries and relatively high dimension. These methods include A*, probabilistic roadmaps, Voronoi graphs, and rapidly-exploring random trees (RRT) (Jaillet, Cortés and Siméon 2010) (Geraerts and Overmars 2007) (Diankov and Kuffner 2007). Unfortunately, these algorithms often neglect dynamic in their formulation or require simplification of the problem. Many of the approaches result in 'low quality' trajectories, for instance using greatly simplified motion assumptions or returning solutions with unnecessary and 'jerky' motion (Geraerts and Overmars 2007). For instance, Diankov in (Diankov and Kuffner 2007) applies pattern matching to cubic polynomials that have been regressed to identify primitive motion paths for robotic trajectory planning. Heuristic algorithms are often applied to improve path quality by encoding some a priori knowledge about the environment. Path pruning and shortcut heuristics, such as the partial shortcut method, assume dynamics can be neglected (Geraerts and Overmars 2007). Further, complex robotic algorithms often encode a large number of heuristics to accurately capture sufficient domain knowledge, but tuning heuristics is an open problem (Diankov and Kuffner 2007).

Chapter 3 Approach Overview

Designers of modern large-scale cyber-physical systems are increasingly confronted with the need to control a system composed of variable numbers of variably-interconnected systems. Consider as example a large-scale system, as shown in Figure 3, composed of a variable number of masses connected through springs and dampers, where the number of masses and their particular interconnections are not known in advance and change as the system evolves in time. The system shown consists of four masses in Figure 3 (a) and eight masses with different interconnections in Figure 3 (b).

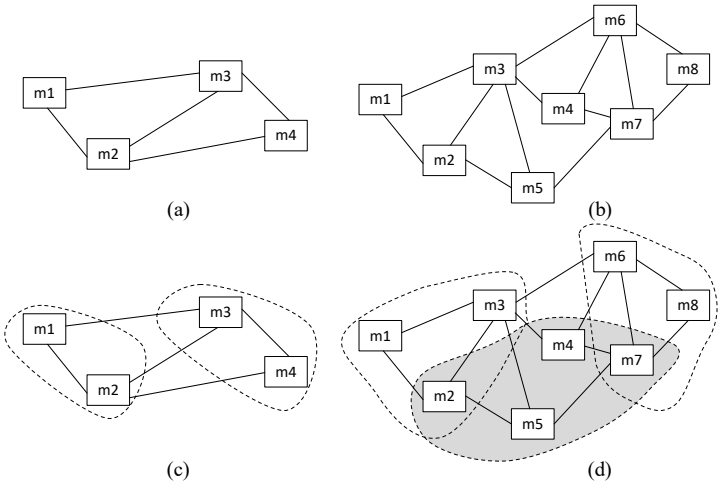


Figure 3. Variably-Connected Spring Mass Example.
System of variably-interconnected masses at two different time instances (a,b), and an assembly of control structures at each time instance (c,d).

While traditional adaptive or robust control techniques are well suited to addressing uncertainty in a particular interconnection, such as the spring strength between m1 and m3 which effects individual terms within the system dynamic matrices, these techniques would not be well suited toward systems that exhibit fundamental structural uncertainty such as this, where most elements in the system matrices, the size of the matrices, and the definition of the state variables may change. The decentralized control literature provides techniques for control systems designers to analyze and construct control systems for specific configurations of these types of large-scale systems, but largely do not address online reconfiguration or deal with objectives or constraints which are introduced by implementation

specific challenges, such as bandwidth or latency limitations that restrict the form of structures across multiple computational elements. Further, decentralized control techniques often utilize modern control methods in the solution formulations and as such are limited by these techniques; for instance, high-dimensional trajectory optimization is a challenge.

We present a new concept for self-assembling self-configuring control to address the challenges faced in control of modern large-scale cyber-physical systems that exhibit substantial variability in system structure. This approach utilizes control system constructs that can morph by adjusting the control structure in real-time to match the current structure of the large-scale interconnected system at any given time.

3.1 Three-Layer Architecture

The approach presented can be conceptually divided into three distinct layers: the transport layer, the topological layer, and the implementation layer.

The *transport layer* is a distributed plug-and-play avionics architecture that allows for immediate on-the-fly reconfiguration of local and global vehicle systems. The transport layer comprises of a number of traditional physical layers, including avionics buses in the system such as CAN, MIL-STD-1553, Firewire, serial, etc., the computing systems on the bus, which is internally composed of memory, processors and other computing components over a typically high bandwidth high speed bus on the chip, wireless communication bus which might include 802.11, RF frequency communication, or other communication bands, as well as the protocols which operate over all these disparate layers that allow the system to function. The implementation utilizes the Reflection Architecture as the main protocol for inter-component communication among system components that have the requisite constitutionality and where is suitably appropriate. Reflection provides a set of capabilities and functionality for real-time component-based plug-and-play which is sufficient for this investigation.

The *topological layer* provides a mathematical description of a plug-and-play architecture as conceived for the purposes of a self-assembling control system implementation. The analysis layer is a topological construct capable of describing a large class of closed-loop system plants and control configurations, which must also allow for modeling and analysis of real-world properties of distributed computing architectures such as latency, bandwidth limitations, and errors and uncertainty in the data signals. This layer models component-based intercommunications largely

through a signal-routing architecture model, which is compatible with the physical implementation of Reflection as the core transport layer.

The *implementation layer* poses the main problems, provides the analytical and synthesis engines required to solve the large-scale assembly problems - whose operation is largely confined to the topological layer - and provides the controlling algorithms and procedures to implement these results - which operate on the domain of the transport layer.

3.2 Process for Control Synthesis

The general process for generating self-assembling self-configuring control is outlined in Figure 4 below. A system dynamics model must first be developed. The system is then transformed into the graph space model. In graph space, the system is analyzed, partitioned, and decomposed into sub-graphs which represent decentralized control sub-problems. In the approached presented, we utilize method for overlapping dynamic partitions outlined in section 2.6, and present a new clustering algorithm for identifying these partitions. Each sub-problem is transformed back into state-space and an optimal control solution is generated. Once the control system is generated for all subgraphs, the plant and control systems are transformed back into the graph domain. Each local decentrally controlled sub-system systems is replaced with equivalent subgraphs to define the search space for optimizing using the Random Search Tree method. The RTS optimizer then analyzes this reduced search space for the best trajectory that optimizes the global objectives and constraints. The resulting trajectories, which found in the search space, and transformed back to the time domain which provides the full time-based trajectory solution for the global system.

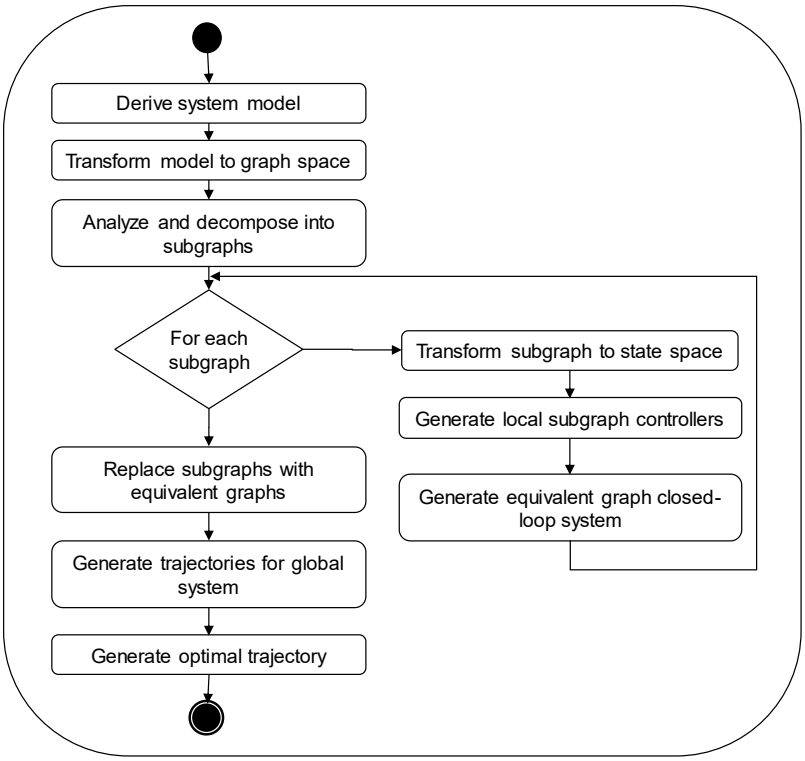


Figure 4. Outline of the General Procedure for Real-Time Control of Large-Scale Structurally-Variant Systems

In this dissertation, we focus on the topological layer definitions and algorithmic solutions for the implementation layer. The topological layer is presented in Chapter 4. The RST optimization method for the implementation layer is presented in Chapter 5. Details and examples are presented in Chapter 6 to show application of the architecture and framework to relevant problems.

This approach utilizes a *pseudo-time-varying* assumption on various elements in the problem formulation. This refers to the assumption that, although an element is time-varying and unknown at design time, during run-time this element will be fixed, known, and constant for a sufficiently long enough period of time that it can be considered constant over this interval. We assume pseudo-time-varying elements can make discrete and instantaneous changes between constant time intervals. We further assume control reconfiguration occurs instantaneously and assume dynamics resulting from transitions can be ignored.

Chapter 4 System Component Graph Formulation in the Topological Layer

For this formulation, we utilize a directed edge-weighted B-hypergraph as illustrated in Figure 5. A directed hypergraph can be considered a generalization of a directed graph where edges connect two sets of vertices, a head set and a tail set. B-hypergraphs, or B-graphs, are a specific form of hypergraph where the head set is restricted to a size of 1. Hypergraphs are chosen for this work to represent general equations that relate a variable to set variables as will be encountered in the analysis of this system.

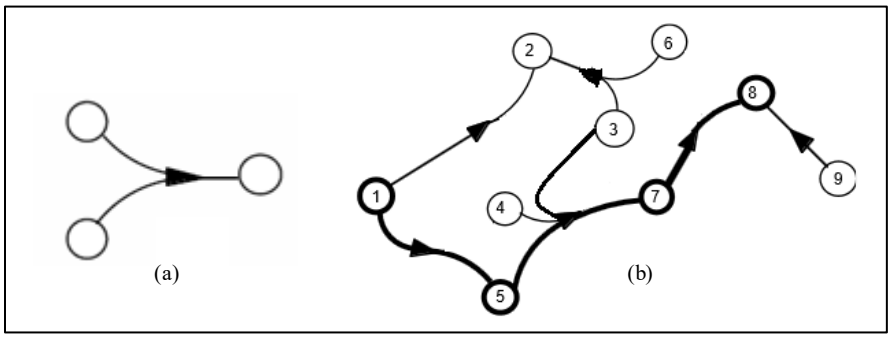


Figure 5. Directed edge-weighted B-arc (a), B-Graph and B-Path (b).

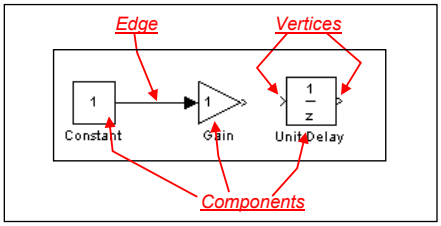


Figure 6. Component Graph and Atomic Definitions

S.1 Let a **vertex** $v \in \mathcal{V}$ is an indivisible unit in the component graph. Let $\mathcal{V}$ the set of all vertices. Vertices in the graph represent a data attribute of a component.

S.2 Let an **edge** $e \in \mathcal{E}$ be a directed b-hyperarc, defined as an ordered pair $e \coloneqq \langle S, t \rangle$, where $S \subseteq \mathcal{V}$ is a set of vertices referred to as the starting or tail vertices, and where $t \in \mathcal{V}$ is a singleton vertex referred to as the terminal or head vertex. Let $\mathcal{E}$ be the set of all edges.

For notational convenience we define the following operators. Given a set A, let the operator $\mathcal{P}(A)$ return the set of all possible subsets of A, often defined as the **power set** of A. Given an edge $e \in \mathcal{E}$ where $e = \langle S, t \rangle$, define the

operators $S: \mathcal{E} \to \mathcal{P}(\mathcal{V})$ and $t: \mathcal{E} \to \mathcal{V}$ to return the starting vertex set $S(e) \coloneqq S$ and the terminal vertex $t(e) \coloneqq t$, respectively.

S.3 Let a **system component graph** $G \in \mathcal{G}$ be defined as a tuple $G \equiv (V, E, C)$, where $V \subseteq \mathcal{V}$ is a set of vertices $V = (v_1, \ldots, v_{|V|})$ in graph G, where $E \in \mathcal{E}$ is a set of edges $E = (e_1, \ldots, e_{|E|})$ in the H, and where $C \in \mathcal{G}$ is the set of components in the graph, to be defined recursively below. Let $\mathcal{G}$ be the recursive space of all possible graphs of the form $\mathcal{G} = \mathcal{V} \times \mathcal{E} \times \mathcal{G}$.

For notational convenience, given a system graph $G = (V, E, C)$, let the operator $V: \mathcal{G} \to \mathcal{V}$ be defined as $V(G) = V$ to return the set of vertices in H. Let $E: \mathcal{G} \to \mathcal{E}$ be defined as $E(G) = E$ to return the set of edges in G. Let $C: \mathcal{G} \to \mathcal{C}$ be defined as $C(G) = C$ to return the set of components in G.

S.4 Given two graphs $H, G \in \mathcal{G}$, the graph H is a **subgraph** of a graph G, denoted as $H \subseteq G$, if an only if $V(H) \subseteq V(G)$, $E(H) \subseteq E(G)$, and $C(H) \subseteq C(G)$. The graph H is a **proper subgraph** of G, denoted as $H \subset G$, if and only if $H \subseteq G$ and either $V(H) \neq V(G)$, $E(H) \neq E(G)$, or $C(H) \neq C(G)$.

S.5 Let a **component** $K \in C(G)$ in graph G be defined with the following properties:

S.5.1	$K \subseteq G$	(Subgraph)
S.5.2	If $e \in E(K)$, then $S(e) \subseteq V(K)$ and $t(e) \in V(K)$	(Edge Containment)
S.5.3	If L∈C(K) then L⊆K	(Subcomponent Containment)
S.5.4	K∉C(K)	(Monotonicity)

For a component K, an element of $C(K)$ is a referred to as a ***subcomponent*** of K. The definition of a component in (S.5) defines what graphs are permissible as a component. For a graph K to be a valid component of graph G, K must be a subgraph of G (S.5.1). If a component contains an edge, it must start and end in vertices contained within the vertex set of that component (S.5.2). All subcomponents are proper subgraphs of that component S.5.3. A component cannot contain itself as a subcomponent (S.5.4). Note that component monotonicity facilitates induction and recursion.

The definition of a component as a subgraph given in (S.5) allows system components graphs to be collapsed or expanded to any arbitrary levels of detail when analyzing and manipulating components in the system, as illustrated in Figure 7. Non-trivial systems can be described as a single component that contains the entire component graph G.

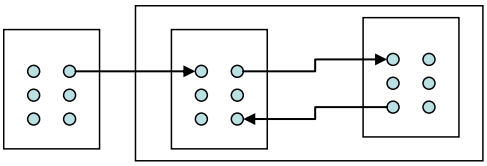


Figure 7. Component Level of Detail

S.6 A component K **subsumes** a component D if and only if $D \subset K$ and either (1) $D \in C(K)$, or (2) there exists a subcomponent $K' \in C(K)$ where K' subsumes D. Note that a component can therefore never subsume itself. Let the *power set* of K, denoted as $\mathcal{P}(K)$, be the set of all possible components subsumed by a component K.

The definition in (S.6) has several implications. A component can never subsume itself. The set of all possible subsumed components is finite. Further, traversing subsumed subcomponents will always result in a component graph that is monotonically decreasing in size - in terms of number of vertices and edges.

S.7 Let a **trace path** P of length q in a system graph $G = (V, E, C)$ to be an ordered sequence of vertices and edges given by $P = (v_1, e_1, v_2, e_2, \dots, v_q, e_q, v_{q+1})$, subject to the following properties: (i) $v_j \in T(e_j) \ \forall\, j = (1..q)$, and (ii) $v_j = H(e_{j-1}) \ \forall\, j = (2..q+1)$. These properties state that any vertex v_j in the path must be the head vertex of the previous edge in the sequence, and must also be in the tail set of the following edge. We refer to v_1 as the source vertex of path P, and v_{q+1} as the terminal vertex of path P.

For notational convenience, given a trace path $P = (v_1, e_1, v_2, e_2, \dots, v_q, e_q, v_{q+1})$, define the operators $src(P) \coloneqq v_1$ and $ter(P) \coloneqq v_{q+1}$ to return the source and terminal vertices, respectively. Define the operators $V(P) \coloneqq (v_1, \dots, v_{q+1})$ and $E(P) \coloneqq (e_1, \dots, e_q)$ to return the set of all vertices and edges, respectively.

S.8 A trace path P is **simple** if all hyperarcs in P are distinct. A trace path P is **elementary** if all vertices in P are distinct.

S.9 A trace path P in a system graph $H \in \mathcal{H}$ is a **cycle** if $src(P) = ter(P)$. A **cyclic** path contains one or more sub-paths that are a cycle. A path that contains no cycles is **acyclic**.

S.10 A vertex $t \in \mathcal{V}$ is **connected** to vertex $s \in \mathcal{V}$ in a system graph $H \in \mathcal{H}$ if and only if there exists a path P_{st} from s to t, where $s = src(P)$ and $t = ter(P)$.

S.11 A trace path P in a system graph H **induces** a minimal subgraph $H_P = (V_P, E_P, W_P)$ such that (i) $E_P \subseteq \mathcal{E}$, (ii) $s, t \in V_p$, where $V_p = \bigcup_{e_i \in E_P} e_i \subseteq \mathcal{V}$, and (iii). $x \in \mathcal{V} \Rightarrow x$ is connected to s in H_P by means of a cycle-free simple path.

Operations and analysis on the system graph are facilitated through assignment of an edge weighting function to edges in the graph.

S.12 Let $w \in \mathcal{W}$ be an **edge weighting function** $w: \mathcal{E} \to \mathbb{R}$ that maps a hyperarc to a real scalar value in the set $\mathcal{W}$ of all such functions.

We define a notion of a configuration space that defines the set of all possible configurations. We defined the configuration space graph below to contain the set of all possible directed hyperarcs between vertices in G.

S.13 Let the **configuration space** $\tilde{C}(G)$ of a graph $G \in \mathcal{G}$ be defined as the following: (1) $V(\tilde{C})$ is the set of all vertices in G, C(G), and all subcomponents of C(G); (2) $E(\tilde{C})$ is defined as the set of all edges $e \in \mathcal{E}$, $e = \langle S, t \rangle$, for all $S \subseteq V(G)$ and $t \in V(G)$; and (3) $C(\tilde{C}) = C(G)$.

S.14 A components C **contains** a set of vertices V if V$\subseteq V$(C).

S.15 Let the **parent component** of a vertex v be defined as follows. For every vertex v$\in$V in $\tilde{C}$, there exists a unique c$\in C(\tilde{C})$ such that v$\in$c, E(c)=$\emptyset$, and C(c)=$\emptyset$. This unique component c is said to be the parent of v, and is written *parent*(v) = c.

A vertex has one and only one parent component. As a result of this rule, the set of parent components in C($\tilde{C}$) are set of components with no edges or subcomponents, and this set is unique and disjoint. Further, there are components with no edges or subcomponents that are inadmissible in $\tilde{C}$ because of violation of the parent component definition.

The concept of connectiveness from traditional graph theory is extended here for components in a graph. Note that these definitions do not consider edge direction in the definitions for component-connectivity.

S.16 Given a graph G, two components A,B$\in C$(G) are **neighbors in G** if there exists an edge e$\in E$(G) such that (s(e)$\in V$(A) $\wedge$ t(e)$\in V$(B)) $\vee$ (s(e)$\in V$(B) $\wedge$ t(e)$\in V$(A)).

S.17 A **component-path** P_c is a defined as a set of two or more components where either:

S.17.1 There are two components in P_c, and these components are neighbors, or

S.17.2 There exist two components $A,B \in P_c$ which are neighbors, and $(P_c\text{-}A)$ is a component-path.

S.18 A component graph G is **component-connected** if there exists at least one component-path between every pair of components in G.

Given these definition we can define a particular configuration in configuration space. A configuration represents a set of components with edges and vertices.

S.19 Let a **configuration graph** G in $\mathcal{C}(G)$ be defined as a graph G=(V,E,C) that represent a specific configuration implementation, subject to

S.19.1 $G \in C(\mathcal{C})$ (G is a Component)

S.19.2 For all $v \in V(G)$ there exists one and only one $K \in C(G)$ such that $v \in K$ (Disjoint Subcomponents)

S.19.3 For all $e \in E(G)$, let $init(e) \in V(L)$ and $ter(e) \in V(M)$ are contained in different components. (Pruned Edges)

S.19.4 G is component-connected. (Component-Connected)

For a configuration graph, all components in the representation are disjoint; in other words, vertices are contained by only one component in G, which is not the parent of the vertex if $parent(v \in G) \notin C(G)$.

S.20 Define an **output variable** as follows. Let $isoutput: V \rightarrow B$ be a mapping such that $isouput(v)=b$ where $v \in V$ and $b \in \mathcal{B}$ if and only if there exists $e \in E(\mathcal{C})$ such that $init(e)=v$.

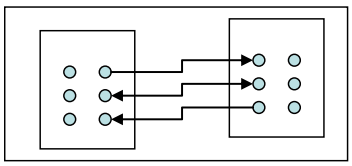


Figure 8. Input/Output Variables

S.21 Component Contraction: There exists a unique mapping $CV: G \rightarrow G_v$ where $G_v=(V,E)$ s.t. $CV(g)=g_v$, defined as follows: For every $g \in \mathcal{C}$, $\exists g_v=G_v$ such that every $c \in C(G)$ has a corresponding $v \in V(G_v)$, where $v=CV(e)$ defines a mapping $CV: E \rightarrow V$, and for every edge $e \in E(G)$, there exists a corresponding $e_v \in E(G_v)$ between the associated $CV(ancestor(init(e)))$ and $CV(ancestor(ter(e)))$.

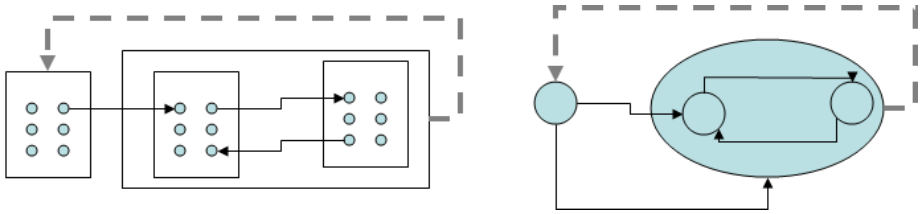


Figure 9. Isomorphic Projection: Controller Space Graph to Component-Vertex Space.

S.22 Component Space (C_v): The C_v component space represents the isomorphic projection of the controller space through component contraction.

S.23 System Boundary Partitions: Every component is an element of one system boundary partition.

S.24 System Boundary Partition Space (C_P): The system boundary partition space (C_P) is defined as a projected space of a component across partitions.

S.25 Flow Definition: Between two sets of vertices let $f(X,Y):=\{ e\in F \mid x\in X, y\in Y; x\neq y; X,Y\subseteq V(G)\}$. A mapping $f:E\rightarrow R$ is a flow in N if it satisfies there conditions:

(i) $f(\{<x,y>\}) = -f(\{<y,x>\})$

(ii) $f(x,V) = 0$ for all $v\in V\backslash\{s,t\}$

(iii) $f(e)\leq c(e)$ for all $e\in E$

S.26 Kirchhoff Network Map: Let a mapping $f:E\rightarrow R$ be a Kirchhoff network mapping if the sum of all flows into a component is equivalent to the sum of all flows coming out of a component.

S.27 Min-Flow Max-Cuts Theorem (Ford & Fulkerson): In every network, the maximum total value of a flow equals the minimum capacity of a cut.

Note that Kirchoff's law does not apply for general costs (e.g., bandwidth constraints), so network theory is not quite appropriate in the general case. Latency constraints in component space, however, can be addressed using a Kirchhoff mapping.

4.1 Transformation Between State-Space Dynamics to System Component Graphs

Consider a large-scale linear time-invariant dynamic system $S \in \mathcal{S}$ in the set $\mathcal{S}$ of all possible systems given by the following.

$$S \;:\; x(t) = Ax(t) + Bu(t)$$
$$y(t) = \mathrm{C}x(t) \tag{4.1-4}$$

Here, $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, and $y(t) \in \mathbb{R}^l$ are the are the state, input, and output. The system matrices are constants given by $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{l\times n}$. Let x(t), u(t), and y(t) be vectors of dimension n, m, and l, respectively.

Without loss of generality, consider $S \in \mathcal{S}$ be composed of n interconnected overlapping sub-systems, where $\mathrm{N} = \{1,2,\dots,n\}$. This system is defined in equation (4.1-23). Interconnections between each system are given by the off-diagonal elements Aij, Bij, and Cij where $i \neq j$.

$$S \;:\; \dot{\mathrm{x}}_i(t) = \sum_{j=1}^{N}(A_{ij}x_j + B_{ij}u_j)$$
$$\mathrm{y_i}(t) = \sum_{j=1}^{N}(C_{ij}x_j) \qquad for\ all\ \ i \in \mathrm{N} \tag{4.1-23}$$

Graph operations are defined to analyze, modify, and partition the system hypergraph into isolated subgraph components. The goal of these operations is to expose structural features of the model that will facilitate decentralization of the control scheme. The isolated subgraphs are individually transformed back into the time-domain representation for control synthesis to occur, then transformed back through the process to arrive at the final decentralized control system.

Given a dynamic system $S \in \mathcal{S}$ as defined in (4.1-23), consider a transformation T to be a transformation $T: \mathcal{S} \rightarrow \mathcal{H}$ that transforms $S \in \mathcal{S}$ into the system graph $H \in \mathcal{H}$ where the vertices $V(H) = (x_1,..,x_n,u_1,\dots,u_m)$ are defined as the set of all state variables and input variables, and the edges E(H) of the graph are induced by a weighted adjacency matrix A_w defined as

$$A_w(S,\delta) = \min\left(\begin{bmatrix} A & B & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \delta\right)$$

S.28 A transformation F is a linear transformation on a,b if the following holds: F(a+b)=F(a)+F(b).

S.29 A transformation F is invertible if there exists a transformation H() such that a=H(F(a)).

S.30 A transformation F is an isomorphic transformation if F(x) is invertible and linear.

S.31 Proposition: The transformations $T_{SH}: \mathcal{S} \rightarrow \mathcal{H}$ and $T_{HS}: \mathcal{H} \rightarrow \mathcal{S}$ define an isomorphic transformation between $\mathcal{S}$ to $\mathcal{H}$.

This can be shown inductively. This property holds for a graph with a single element. Assuming this property holds for graph with some given number of elements, adding a new element to the graph will still maintain this property.

Isomorphism is an enabling property of the transformation T_{SH} and T_{HS} between $\mathcal{S}$ and $\mathcal{H}$. The algorithms presented utilize this property to freely transform between the state space representation and the system graph representation. The system graph representation can be manipulated, the graph can be broken into smaller pieces, smaller subgraphs can be transformed back and forth between the system graph and state space representations, and subgraphs can be reassembled and transformed back into the state space representation through this property.

4.2 Graph Partitioning and Clustering

Transforming the system dynamics into graph space provides an alternative representation of the system that is amenable to structural topological analysis. A critical step in decentralized control design is the identification of a decomposition that can fully exploit the sparsity of the system matrix. In this section, we build upon Epsilon decomposition techniques as outlined in (Siljak, 1996), which decomposes a large sparse system into weakly coupled sub-systems through the following.

$$A = A_0 + \epsilon A_c \;\; ; \;\; B = B_0 + \epsilon B_c \;\; ; \;\; C = C_0 + \epsilon C_c \tag{4.2-6}$$

Here, ϵ is a small positive number, the matrices A_0, B_0, and C_0 have block-diagonal structure, and all the elements of A_c, B_c, C_c are smaller than one in magnitude. For this approach, we define the following general algorithm for iterative reduction of the system graph.

1. Form a metric for closeness of two vertices, mapping c:VxV->R
2. Identify the closest vertices
3. Collapse the graph between the two vertices to form the topological graph minor
4. Repeat until convergence criteria is met

The goal of graph clustering is to divide the graph into spanning disjoint sets of vertices, or clusters, such that the elements assigned to a particular cluster are more similar in some mathematically defined sense to the vertices in its assigned cluster than vertices in another cluster. In our sense, we define clustering as the following problem.

Given a graph G=(V,E). Let a cluster function $f_c(V,V)$: $V \times V \rightarrow \mathbb{R}^+$ be defined over every pair-wise vertex.

Define a cluster $C = (C_1, \ldots, C_k)$ of size k on G as a partition of the vertices V(G) such that $C = (C_1, \ldots, C_k), C_i \subseteq V, C_i \cap C_j = \emptyset$ that minimizes the sum of the within-cluster sum

$$C_{f_c}(G) = \underset{C}{\operatorname{argmax}} \sum_{s=1}^{k} \left(\sum_{vi,vj \in C_i} f_c(v_i, v_j) \right) \tag{4.2-7}$$

4.3 Clustering Definitions and Metrics

We propose a clustering function and algorithm defined as follows.

S.32 Let the function $E(u,v) = \{e \in E(G): e = (u,v) | e(v,u); u,v \in V(G)\}$ return the set of all edges between two vertices u and v.

S.33 Let the operator Γ: $\mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n \times n}$ return the diagonal elements of A, defined as follows.

$$\Gamma(\mathrm{A}) = [\gamma_{ij}] \ where \ \gamma_{ij} = \begin{cases} a_{ij} \text{ if i = j} \\ 0 \ otherwise \end{cases}$$

S.34 The following operator defines the 'binary' adjacency matrix from A_w. Let the operator $F(\cdot,\cdot)$: $\mathbb{R}^{n \times n}, \mathbb{R} \rightarrow \mathbb{Z}^{n \times n}$ be defined as

$$F(A,\delta) = [f_{ij}] \ where \ f_{ij} = \begin{cases} 1 \ if \ |a_{ij}| > \delta \\ 0 \ otherwise \end{cases}$$

S.35 Let the operator $|\cdot|$: $\mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$ return a matrix with the element-wise absolute values of the original matrix.

S.36 *(Sw: Sum of the normed weights matrix*) Given the weighted adjacency matrix A_w, the second term in (4.3-10) can be found by looking at the elements of a $S_w = [s_{w_{i,j}}]$ matrix, where $s_{w_{i,j}}$ contains the sum of the absolute values of the weights of the edges between vertices i and j. Let the matrix-valued operator $S_w(\cdot)$: $\mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n \times n}$ be defined as returning this matrix over a weighted adjacency matrix Aw, defined as

$$S_w(A_w) = |A_w| + |A_w^T| - \Gamma\,(|A_w|) \tag{4.3-26}$$

Note that the matrix *Sw* is symmetric, and is analogous to an undirected version of *Aw*.

S.37 (*Se: Edge count matrix*) Given the weighted adjacency matrix A_w, the first term in (4.3-10) can be found by looking at the elements of a $S_e = [s_{e_{i,j}}]$ matrix, where $s_{e_{i,j}}$ represents the number of edges between vertices i and j.

Let the sum of the edges matrix operator be defined as

$$S_e(A_w) = F(A_w, \delta) + F(A_w, \delta)^T - \Gamma\big(F(A_w, \delta)\big) \tag{4.3-27}$$

Note that the matrix *Se* is symmetric, analogous to an undirected version of *Aw*.

S.38 Define a clustering function f_1 be the sum of the number of edges between u and v given by the following, where $k_e, k_w \in \mathbb{R}$ are constant scalar values.

$$f_1(u, v) = K_1 \cdot |E(u, v)| + K_2 \cdot \sum_{e \in E(u,v)} |W(e)| \tag{4.3-10}$$

S.39 Define the F1 cluster function matrix as follows

$$F_1 = k_e S_e(A_w) + k_w S_w(A_w) \tag{4.3-29}$$

Note that the diagonal elements of the F_1 matrix are zero.

Given a graph G and the f_1 cluster function defined in (4.3-10) , we propose an algorithm to determine the clusters C_{f1} that maximizes the clustering equation (4.2-7).

S.40 Define the cluster-graph, C=(V,E), as a directed edge-weighted multi-graph. Cluster graphs will generally allow for multiple edges between vertices, and can no longer be represented with a standard weighted edge adjacency matrix.

S.41 Define cluster-graph contractions as the following operator for merging two clusters, shown visually in Figure 10, as similarly defined for edge contraction. The difference is that all the edges and weights must be maintained, and thus a multi-graph may result in the general case.

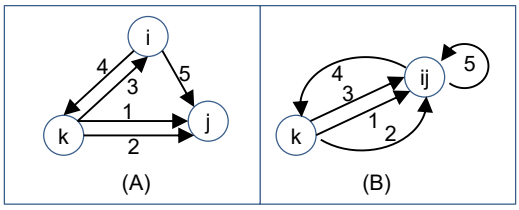


Figure 10. Contractions on Cluster Graphs.
(A) shows the uncontracted graph, (B) shows the graph after contracting vertices i and j.

The desire for cluster graph contraction is to merge two vertices that are connected by an edge into a, and each vertex or super-vertex. All edges pointing to one of the two contracted vertices must end up pointing to the contracted component. All edges between the two contracted vertices become self-edges. The contracted graph is a cluster graph minor by the following definition.

S.42 A graph H is a minor of the graph G if H can be formed from G by deleting edges and vertices and by contracting edges.

The two following definitions are for implementation in Matlab that is efficient in terms of matrices. We desire a way of storing and manipulating graphs (particularly, forming minors through contractions) that can be implemented in Matlab quickly. Since the cluster-graph is a multi-graph, can't store the entire graph using an adjacency matrix of size N by N, so these two matrices will store only the information about the graph that is necessary to implement the algorithm below, but won't store complete information about the minors.

S.43 Let the cluster-graph edge count matrix be defined a matrix We=[aij] such that aij is the number of edges from vertex j to vertex i.

S.44 Let a cluster-graph sum-of-normed-edge-weights matrix be defined as a matrix Ws=[aij] such that aij is the sum of the weights of all edges from vertex j to vertex i.

Given a system graph Aw, and the cluster-graph edge count matrix, the cluster graph can now be formed with each vertex initially in its own cluster, utilizing We=Se(Aw) as the edge count matrix and Ws=Sw(Aw) as the sum-of- normed-edge-weights matrix for the cluster graph.

4.4 F1 Graph Clustering Algorithm

Utilizing the definitions presented, a simplified algorithm can be presented for formation of clusters. The following constructive algorithm that forms graph clusters through iteratively combining clusters.

Algorithm: FormF1Clusters(G,\delta)
Input: G=(V,E) is the directed graph, \delta is the minimum value to be considered an edge

1. C = { $\{v_1\}$, $\{v_2\}$, …, $\{v_N\}$ } // initialize with N clusters, each vertex in a cluster
2. Aw = Aw(G)
3. Ws=Ws(Aw,\delta)
4. We = We(Aw,\delta)
5. While stopping criteria is not reached (k>k', or something else?)
6. Wsum = K1*Ws + K2*We // form Wsum
7. Find (i,j) = min(Wsum) where i,j are the indices of the minimum entry in Wsum
8. We,Ws = MergeClusters(We,Ws,i,j)

4.5 Linear Quadratic Control Synthesis for Subgraphs

For notational convenience, let $\mathcal{S}$ be the set of all dynamical systems as defined in (4.1-23), let $\mathcal{S}_C \subset \mathcal{S}$ be the set of all controllable dynamical system as defined where (A,B) is controllable. Let $\mathcal{J}: (\mathbb{X}\times\mathbb{U}) \to \mathbb{R}$ be the set of admissible quadratic performance index functions of the form in (4.5-30), where $Q \geq 0$ and $R > 0$ are constant symmetric matrices, and $x_0 = x(0)$ is the initial state at time t=0. In this formulation, we set $\mathcal{T} = (0, \infty]$.

$$\mathcal{J} = \big(J(x_0,u)\big) \quad : \quad J(x_0,u) = \int_0^{+\infty} (x^T Q x + u^T R u)dt \tag{4.5-30}$$

Here, $Q \geq 0, R > 0, Q = Q^T, R = R^T$. Let $\mathcal{O}(\mathbf{S},J): (\mathcal{S}_C\times\mathcal{J}) \to \mathbb{U}^r$ be a mapping from a dynamical system $\mathbf{S} \in \mathcal{S}_C$ and performance index $J \in \mathcal{J}$ to a control strategy $u(t) \in \mathbb{U}$. Let $\mathcal{O}_{LQ} \in \mathcal{O}$ map the pair ($\mathbf{S}$,J) to the standard optimal control problem solution over the nominal system, denoted $u^{\odot}_{\mathbf{S},J}$ and defined as follows.

$$\begin{gathered} \mathbf{S}: \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \\ \mathcal{O}_{LQ}(\mathbf{S},J) \equiv u^{\odot}_{(\mathbf{S},J)} \equiv \underset{u}{\operatorname{argmin}} J(x_0,u) \\ J(x_0,u) = \int_0^{+\infty} (x^T Q x + u^T R u)dt \end{gathered} \tag{4.5-31}$$

The solution to this problem is given by the following, where $P > 0$ is the unique symmetric solution to the Riccati equation shown below, and $(A, Q^{\frac{1}{2}})$ as determined by $\mathcal{O}_{LQ}$ is observable.

$$\begin{gathered} u^{\odot}_{(\mathbf{S},J)} = -Ky \\ K = R^{-1}B^T P \\ A^T P + PA - C^T P B R^{-1} B^T P C + Q = 0 \end{gathered} \tag{4.5-32}$$

Closing the loop on the control law $u^{\odot}$ yields the closed loop system $\hat{\mathbf{S}}^{\odot}$ and optimal cost $J^{\odot}(x_0) \equiv J\left(x_0, u^{\odot}\right)$, as follows.

$$\hat{\mathbf{S}}^{\odot} \;:\; \dot{x} = (A - BKC)x$$
$$J^{\odot}(x_0) = x_o^T P x_o - \lim_{t\to\infty}\left(x^T(t)Px(t)\right) = x_o^T P x_o \quad if \quad \lim_{t\to\infty} x(t) = 0 \tag{4.5-33}$$

4.6 Three-Mass System Example

Consider the following example involving a system of three masses (m_1, m_2, m_3) interconnected by springs and dampers. Two masses have control forcing inputs are strongly coupled to a third mass without direct input, and weakly coupled to each other, such that $k_1 \gg k_2$ and $b_1 \gg b_2$.

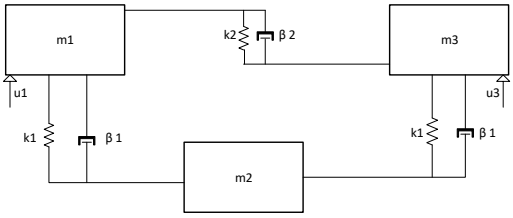


Figure 11. Coupled Three-Mass System

The system dynamics S_1 are given by the following model.

$$S_1 : \dot{x} = Ax + Bu$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -(k_1+k_2) & -(b_1+b_2) & k_1 & b_1 & k_2 & b_2 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ k_1 & b_1 & -2k_1 & -2b_1 & k_1 & b_1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ k_2 & b_2 & k_1 & b_1 & -(k_1+k_2) & -(b_1+b_2) \end{bmatrix} \tag{4.6-34}$$

$$B = \begin{bmatrix} 0 & 0 \\ f & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & f \end{bmatrix}$$

We wish to find a control solution for this system that minimizes the following.

$$\min_{u} J(x_0, u) = \int_0^{\infty} [x^T(t)Q^*x(t) + u^T(t)R^*u(t)]dt$$
$$s.t.\ \mathbf{S}: \dot{x}(t) = Ax(t) + Bu(t) \tag{4.6-35}$$

Transforming system S_1 into a system graph form yields the following graph in Figure 12.

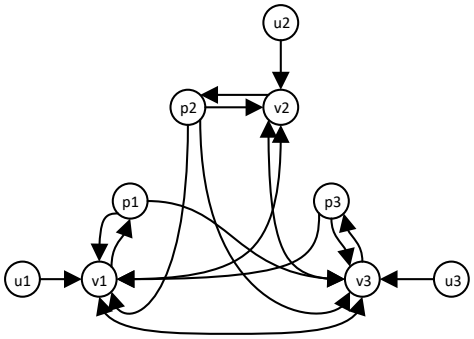


Figure 12. System Graph Representation of the Three Mass System S_1.

The F1 cluster weight function on S1 creates a symmetric N by N weighting matrix, which can be interpreted as an undirected graph overlaid on S1. The clustering function metric applied is given by the following equation.

$$C_{f_c}(H) = \underset{C}{\text{argmax}} \sum_{s=1}^{k} \left(\sum_{vi,vj \in C_i} f_c(v_i, v_j) \right) \tag{4.6-36}$$

Applying the clustering metric to the graph results in the weighted graph in Figure 13.

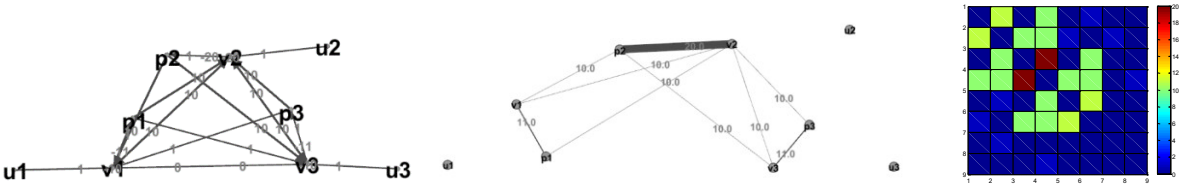


Figure 13. System Graph Representation of the Three Mass System S_1.
Original system graph (left), cluster function matrix (middle), cluster function graph (right).

The clustering function was applied to the transformed system in Figure 13 with clusters collapsed, resulting in the following contracted system in Figure 14.

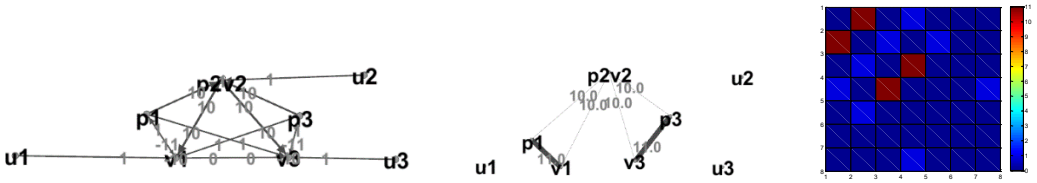


Figure 14. Post contraction graph of the three-mass system S_1. .

Note the structure of the contracted cluster graph shows strong linkage between P1V1 and P2V2, and also between P2V2 and P3V3. There are weak linkages between P1V1 and P3V3. The clustered representation identifies candidate decomposition of overlapping expansion and contraction. Applying overlapping decentralization yields the expanded system in Figure 15.

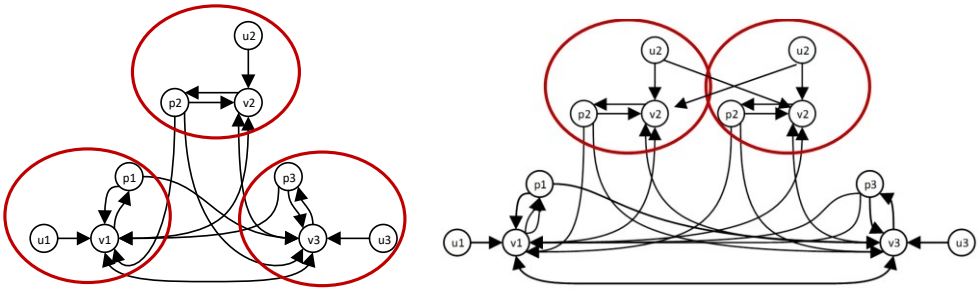


Figure 15. Overlapped system identification (left) and expanded system after expansion (right).

The clustered graph provides the following form for the expansion and contraction transformation matrices.

$$V = \begin{bmatrix} I_{n_1} & & \\ & I_{n_2} & \\ & I_{n_2} & \\ & & I_{n_3} \end{bmatrix} \; ; \quad R = \begin{bmatrix} I_{m_1} & & \\ & I_{m_2} & \\ & I_{m_2} & \\ & & I_{m_3} \end{bmatrix} \; ; \quad T = \begin{bmatrix} I_{l_1} & & \\ & I_{l_2} & \\ & I_{l_2} & \\ & & I_{l_3} \end{bmatrix} \tag{4.6-37}$$

The generation of a decentralized controller for this system would proceed as described in (Siljak, D. 1992) if performed on the state space model. Utilizing this approach would yield the following.

```
A = [    0        1      0      0       0        0;
    -(k1+k2) -(b1+b2)  k1     b1      k2       b2;
        0        0      0      1       0        0;
        k1       b1   -2*k1  -2*b1     k1       b1;
        0        0      0      0       0        1;
        k2       b2     k1     b1   -(k1+k2)  -(b1+b2) ];
```

$$\mathbf{S}_1: \quad \{[A_{i,j}], [B_{i,j}], [C_{i,j}], \; i,j = 1,2\}$$

$$\mathbf{S}_2: \quad \{[A_{i,j}], [B_{i,j}], [C_{i,j}], \; i,j = 2,3\}$$

The transformed system yields

$$\tilde{A} = \begin{bmatrix} A_{11} & \frac{1}{2}A_{12} & \frac{1}{2}A_{12} & A_{13} \\ A_{21} & \frac{1}{2}A_{22} & \frac{1}{2}A_{22} & A_{23} \\ A_{21} & \frac{1}{2}A_{22} & \frac{1}{2}A_{22} & A_{23} \\ A_{31} & \frac{1}{2}A_{32} & \frac{1}{2}A_{32} & A_{33} \end{bmatrix} + M \quad ; \quad \tilde{B} = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix} + N$$

$$M = \begin{bmatrix} 0 & M_{12} & -M_{12} & 0 \\ 0 & M_{22} & -M_{22} & 0 \\ 0 & M_{32} & -M_{32} & 0 \\ 0 & M_{42} & -M_{42} & 0 \end{bmatrix} \quad N = \begin{bmatrix} 0 & 0 \\ N_{21} & N_{22} \\ -N_{21} & -N_{22} \\ 0 & 0 \end{bmatrix}$$

$$\tilde{A} = \begin{bmatrix} A_{11} & A_{12} & 0 & A_{13} \\ A_{21} & A_{22} & 0 & A_{23} \\ A_{21} & 0 & A_{22} & A_{23} \\ A_{31} & 0 & A_{32} & A_{33} \end{bmatrix}$$

Given the isomorphic properties of the graph space transform, the operations can analogously be performed utilizing graph operations. Performing the minimum cut isolates the graph into two distinct sub-graphs, shown in Figure 16.

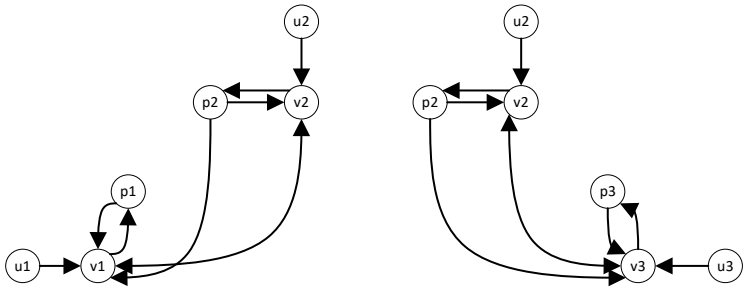


Figure 16. Isolated systems.

The optimal LQR synthesis algorithm detailed in section 4.5 can then be applied to the isolated systems.

4.7 Variably-Coupled Inverted-Pendulum Example

The following example of a large-scale variably-interconnected cyber-physical system illustrates some of the challenges faced by modern control system designers. The elements of this example problem are derived from the applications in Chapter 6.

Consider a system composed of a pseudo-time-varying number of variably-interconnected vehicle systems. The number of vehicles, and thus the system size, is not known ahead of time and will vary with time. Each vehicle is composed of an inverted pendulum on a sliding cart with 1 degree of translational freedom and 1 degree of rotational

freedom. Each vehicle contains sensing and computational processing elements to implement feedback control. Each vehicle can sense the cart position x_i and the pendulum orientation θ_i.

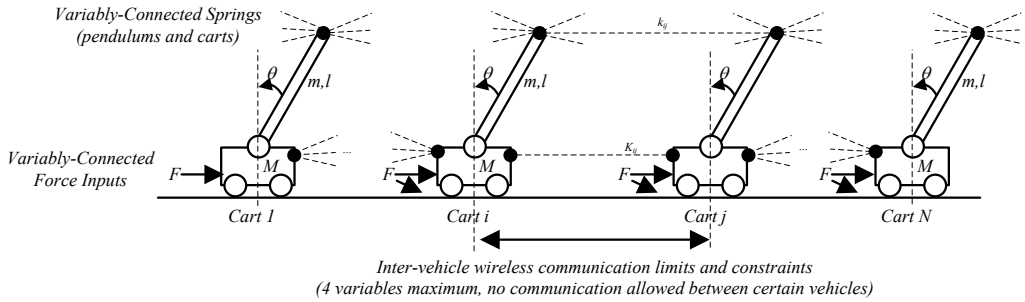


Figure 17. Variably-Large Variably-Coupled Inverted-Pendulum System Example.

At any point in time, the pendulum of any vehicle i may be connected to the pendulum of any vehicle j by a spring of known spring-constant k_{ij}. Likewise, cart i may be connected to cart j by a spring with constant K_{ij}. Let $k = [k_{ij}]$, $K = [K_{ij}]$, where $K \in \mathbb{R}^{n \times n}$ and $k \in \mathbb{R}^{n \times n}$, be pseudo-time-varying matrices containing the spring constants that interconnect pendulum i and pendulum j, and between cart i and cart j, respectively. Note that k and K are symmetric, pseudo-time-varying, variably-sized, and are unknown at design time.

The system is controlled by a set of n force inputs, $u \in \mathbb{R}^n$, which are variably connected to the cart of each vehicle in a pseudo-time-varying configuration. Consider an input mapping matrix $K_u \in \mathbb{R}^{n \times n}$, $K_u = \left[K_{u_{ij}}\right]$, where an element $K_{u_{ij}}$ determines how an input j effects a system i. The force at each cart Fi is given by $F_i = K_u u$. The matrix K_u is pseudo-time-varying, variably-sized, and unknown at design time.

Inter-vehicle communication occurs through limited bandwidth wireless links. In this example, we constrain the maximum number of variables that any vehicle can send to another vehicle, and set this constraint to $C_{com} = 4$.

The objective for control is to stabilize the unstable inverted pendulums while controlling vehicle location to optimize a global objective that is a function of cart positions. The global objective is given in terms of a time-varying cost map on the positions of each cart. The position of each cart is constrained to a limited range specified for each vehicle i, $x_i \in \left[x_{i_{min}}, x_{i_{max}}\right]$. The cost and constraint maps are not known during the control system design and are pseudo-time-varying. Local vehicle objectives are provided in terms of standard Q and R matrices on the states and inputs.

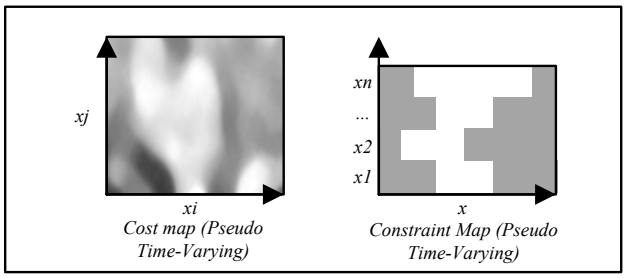


Figure 18. Global Objective Map and Position Constraints.

We model the cart and pendulum system from the free-body diagram shown in Figure 19. The system can be linearized about the unstable equilibrium condition where the pendulum is vertical, at $\theta_i = 0$. The reaction forces are given by R and N in the figure, which couple the rotational and translational equations of motion.

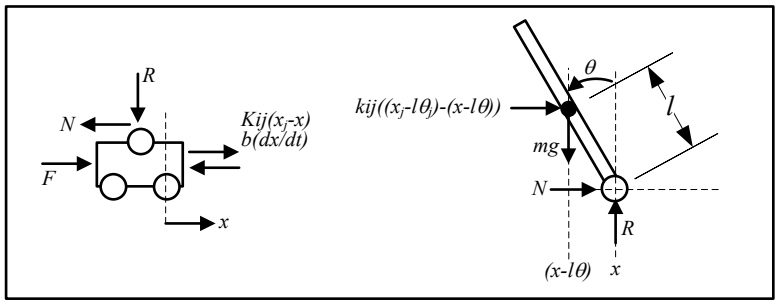


Figure 19. Free-Body Diagram for the Cart-Pendulum System.

The dynamics for this system are given by the following equation.

$$\begin{aligned}(I+ml^2)\ddot{\theta} - mgl\theta + k_{ij}l\left(\left(x_j - l\theta_j\right) - (x - l\theta)\right) &= ml\ddot{x} \\ (M+m)\ddot{x} + b\dot{x} - K_{ij}(x_j - x) - ml\ddot{\theta} &= u\end{aligned} \tag{4.7-38}$$

Substituting and solving for $\ddot{x}$ and $\ddot{\theta}$ yields

$$\begin{aligned}\ddot{x} = &\left(\sum_{k_{ij}}\left(\frac{\frac{l^2}{m}k_{ij}}{P}\right) - \sum_{K_{ij}}\left(\frac{(I+ml^2)K_{ij}}{P}\right)\right)x - \left(\frac{b(I+ml^2)}{P}\right)\dot{x} + \left(\frac{l^2 g}{P} + \sum_{k_{ij}}\left(\frac{\frac{l^3}{m}k_{ij}}{P}\right)\right)\theta + \left(\frac{(I+ml^2)}{P}\right)u \\ &+ \left(\sum_{K_{ij}}\left(\frac{K_{ij}(I+ml^2)}{P}\right) - \sum_{k_{ij}}\left(\frac{\frac{l^2}{m}k_{ij}}{P}\right)\right)x_j + \left(\sum_{k_{ij}}\left(\frac{\frac{l^3}{m}k_{ij}}{P}\right)\right)\theta_j \\ \ddot{\theta} = &\left(\sum_{k_{ij}}\left(\frac{(M+m)lk_{ij}}{P}\right) - \sum_{K_{ij}}\left(\frac{mlK_{ij}}{P}\right)\right)x - \left(\frac{mlb}{P}\right)\dot{x} + \left(\frac{mgl(M+m)}{P} - \sum_{k_{ij}}\left(\frac{(M+m)l^2k_{ij}}{P}\right)\right)\theta + \left(\frac{ml}{P}\right)u \\ &+ \left(\sum_{K_{ij}}\left(\frac{mlK_{ij}}{P}\right) - \sum_{k_{ij}}\left(\frac{ml^2(M+m)k_{ij}}{P}\right)\right)x_j + \left(\sum_{k_{ij}}\left(\frac{ml^3(M+m)k_{ij}}{P}\right)\right)\theta_j\end{aligned} \tag{4.7-39}$$

Converting this to state space yields the full n-body system of equations as given by the following, where $\bar{x} \in \mathbb{R}^{4n}$, $\bar{u} \in \mathbb{R}^{n}$, $\bar{y} \in \mathbb{R}^{2n}$, $A \in \mathbb{R}^{4n\times 4n}$, $B \in \mathbb{R}^{4n\times n}$, $C \in \mathbb{R}^{2n\times 4n}$, $D \in \mathbb{R}^{2n\times n}$. In this example, we consider C and D to be fixed constant block-diagonal matrices.

$$\begin{gathered}\bar{x} = [\bar{x}_1 \dots \bar{x}_n]^T \;\; ; \;\; \bar{x}_i = [x_i \quad \dot{x}_i \quad \theta_i \quad \dot{\theta}_i] \;\; ; \;\; \bar{u} = [u_1 \dots u_n]^T \\ \dot{\bar{x}} = A\bar{x} + B\bar{u} \\ \bar{y} = C\bar{x} + D\bar{u}\end{gathered} \tag{4.7-40}$$

The structure of the large-scale system can be seen in the following equation. Note that the A and B matrices are variably sized and may be dense at any given point in time, and the structure follows from the interconnectivity matrices K, k, and Ku. When there is no interconnectivity, the A and B matrices are diagonal. Further note that the size of this system is pseudo-time-varying.

$$\begin{gathered}\frac{d}{dt}\begin{bmatrix}\bar{x}_1 \\ \vdots \\ \bar{x}_i \\ \vdots \\ \bar{x}_n\end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{in} & \cdots & A_{1n} \\ \vdots & \ddots & & & \vdots \\ A_{ni} & & A_{ii} & & A_{in} \\ \vdots & & & \ddots & \vdots \\ A_{n1} & \cdots & A_{ni} & \cdots & A_{nn}\end{bmatrix}\begin{bmatrix}\bar{x}_1 \\ \vdots \\ \bar{x}_i \\ \vdots \\ \bar{x}_n\end{bmatrix} + \begin{bmatrix} B_{11} & \cdots & B_{in} & \cdots & B_{1n} \\ \vdots & \ddots & & & \vdots \\ B_{ni} & & B_{ii} & & B_{in} \\ \vdots & & & \ddots & \vdots \\ B_{n1} & \cdots & B_{ni} & \cdots & B_{nn}\end{bmatrix}\begin{bmatrix}u_1 \\ \vdots \\ u_i \\ \vdots \\ u_n\end{bmatrix} \\ \begin{bmatrix}\bar{y}_1 \\ \vdots \\ \bar{y}_i \\ \vdots \\ \bar{y}_n\end{bmatrix} = \begin{bmatrix} C_i & & & & 0 \\ & \ddots & & & \\ & & C_i & & \\ & & & \ddots & \\ 0 & & & & C_n\end{bmatrix}\begin{bmatrix}\bar{x}_1 \\ \vdots \\ \bar{x}_i \\ \vdots \\ \bar{x}_n\end{bmatrix} + \begin{bmatrix} D_1 & & & & 0 \\ & \ddots & & & \\ & & D_i & & \\ & & & \ddots & \\ 0 & & & & D_n\end{bmatrix}\begin{bmatrix}u_1 \\ \vdots \\ u_i \\ \vdots \\ u_n\end{bmatrix}\end{gathered} \tag{4.7-41}$$

The elements of the system matrices in (4.7-41) can be derived from (4.7-39), and are given by the following.

$$A_{ii} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \left(\sum_j \frac{l^2}{mP}k_{ij} - \sum_j \frac{(I+ml^2)}{P}K_{ij}\right) & \left(-\frac{b(I+ml^2)}{P}\right) & \left(\frac{l^2 g}{P} + \sum_j \frac{l^3}{mP}k_{ij}\right) & 0 \\ 0 & 0 & 0 & 1 \\ \left(\sum_j \frac{(M+m)l}{P}k_{ij} - \sum_j \frac{(M+m)l}{P}K_{ij}\right) & -\left(\frac{mlb}{P}\right) & \left(\frac{mgl(M+m)}{P} - \sum_j \frac{(M+m)l^2}{P}k_{ij}\right) & 0 \end{bmatrix}$$

$$A_{ij} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \left(\sum_j \frac{(I+ml^2)}{P}K_{ij} - \sum_j \frac{l^2}{mP}k_{ij}\right) & 0 & \sum_j \frac{l^3}{mP}k_{ij} & 0 \\ 0 & 0 & 0 & 0 \\ \left(\sum_j \frac{ml}{P}K_{ij} - \sum_j \left(\frac{(M+m)ml^2}{P}k_{ij}\right)\right) & 0 & \left(\sum_j \frac{(M+m)ml^3}{P}k_{ij}\right) & 0 \end{bmatrix} \tag{4.7-42}$$

$$B_{ii} = \begin{bmatrix} 0 \\ \left(\frac{(I+ml^2)}{P}\right)K_{u_{ii}} \\ 0 \\ \left(\frac{ml}{P}\right)K_{u_{ii}} \end{bmatrix} \;\; ; \;\; B_{ij} = \begin{bmatrix} 0 \\ \left(\frac{(I+ml^2)}{P}\right)K_{u_{ij}} \\ 0 \\ \left(\frac{ml}{P}\right)K_{u_{ij}} \end{bmatrix}$$

$$C_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \;\; ; \;\; D_i = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

The following table summarize the variables and constants used in analysis of this example problem.

Table 2. Definitions, Variables and Constants for the Interconnected Pendulum System.

Variable	**Description**	**Value/Units**
M	Mass of the cart	0.5 kg
m	Mass of the pendulum	0.2 kg
b	Coefficient of friction	0.1 N/m/sec
l	Distance from pendulum CG to pivot point	0.3 m
I	Mass moment of inertia of the pendulum	0.006 kg m^2
F	Force input applied to cart	N
θ	Pendulum angle	radians
x	Position of the cart	m

Many elements of this problem are difficult to address using existing control theoretic techniques as outlined in Chapter 2. For instance, small changes in K, k, and Ku, can result in instability or loss of controllability. Consider the case when N=4, and between time intervals t1 and t2, Ku changes from $K_u(t_1) = \begin{bmatrix} I_8 & 0 \\ 0 & I_8 \end{bmatrix}$ to $K_u(t_2) = \begin{bmatrix} 0 & I_8 \\ I_8 & 0 \end{bmatrix}$. Though a simple change, this would result in loss of controllability using the existing methods. Further, the communication constraint is very difficult to address using these methods, as they require fixed and structured constraints that must be known during control system design time, and these methods are not trivially extended to the case where constraints are unstructured and unknown until run-time. Combining time-varying system dimension with all these challenges is further debilitating to existing techniques.

Consider the case of four uncoupled and independent systems, where n=4, k=0n, K=0n, and Ku=In. The resulting large-scale system has 4n states. The A and B matrices are block diagonal, as can be seen by the sparsity pattern of the system matrices as shown in Figure 20.

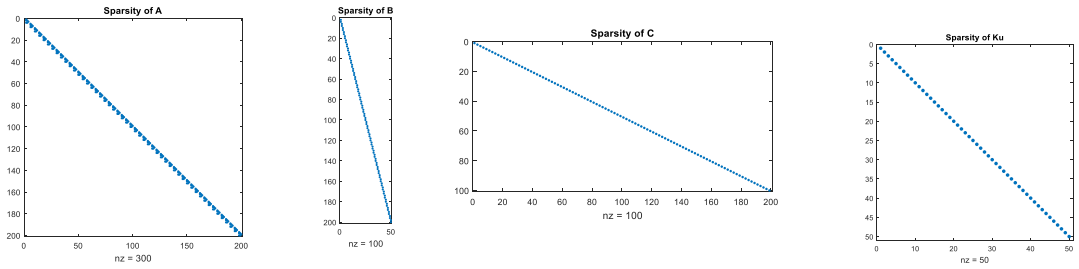


Figure 20. Sparsity Structure of A, B, C, and Ku matrices.

The controllability and observability matrices are of full rank for this system. The open-loop system is at an unstable equilibrium when $\bar{x} = [0]$ with four unstable roots at around +5.8 on the positive real axis. Open-loop step response and impulse response is identically unstable for each system, as shown in Figure 21 below.

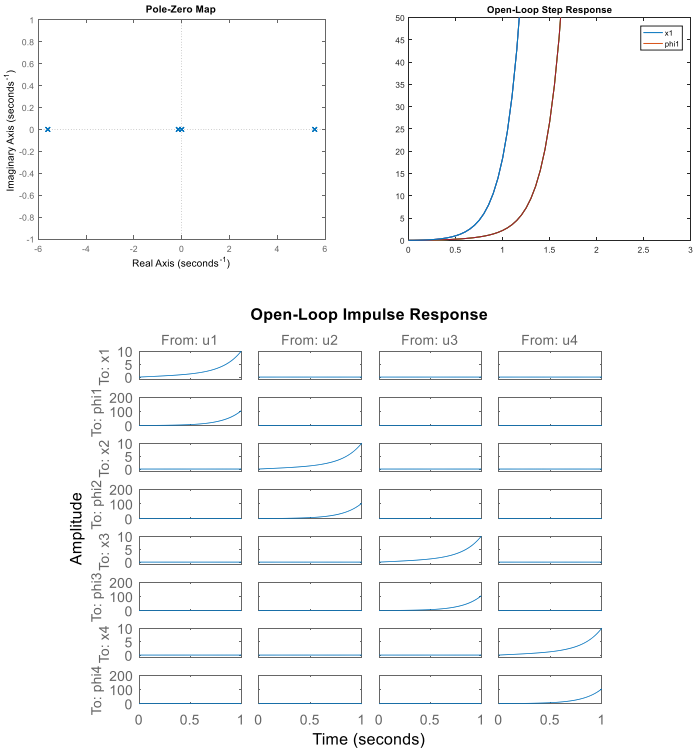


Figure 21. Pole Location, Open-Loop Step Response, and Open-Loop Impulse Responses.

As a first step in our process, the system in (4.7-40) is transformed into the system component graph space. The resulting graph is visualized in Figure 22, which illustrates the decoupled and independent nature of the four independent and isolated subsystems.

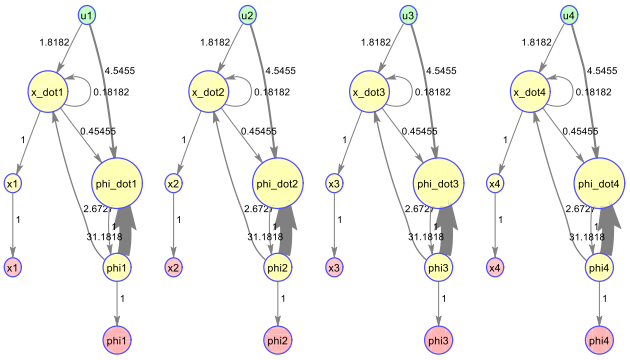


Figure 22. System Component Graph for the Decoupled 4-Vehicle System.

The system is next analyzed to determine local subsystems that are strongly connected. The clustering algorithm and metric are applied to the system graph, as defined in sections 4.3 and 4.4. The graph collapses into subgraph components (Figure 23, left), and the clustering returns four individual subgraph components (Figure 23, right).

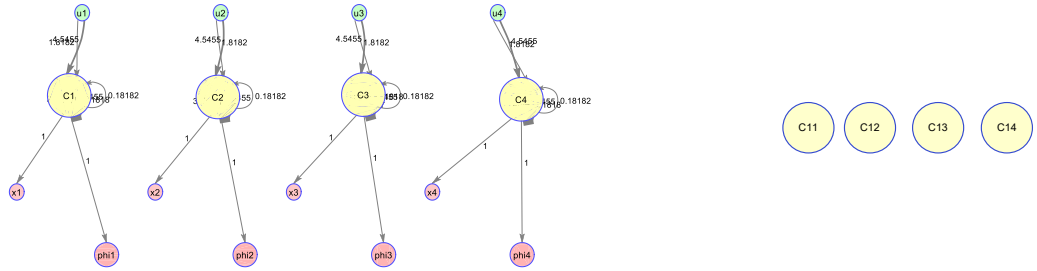


Figure 23. Clustering the Decoupled System.
An intermediate contraction is shown on the left, and the resulting system component graph is shown on the right.

The set of resulting components (C11, C12, C13, and C14) define local subsystems for this configuration. Each component is then transformed from graph-space to state-space for control system design and subcomplement system generation, then the resulting system is transformed back to graph-space. This process is shown in Figure 24 for component C11. The given component C_i, as shown in Figure 24 (a), is fully expanded through the operation defined in section 4.2, which terminates when the graph contains no further components C that can be expanded. The resulting expanded graph, shown in Figure 24 (b), is transformed from graph-space to state-space through the isomorphic transform defined in section 4.1. A local control is generated on the state-space representation through the LQR approach defined in section 4.5. The resulting closed-loop state-space system is transformed from state-space to graph-space, as shown in in Figure 24 (c). The resulting graph, C_{cl_i}, becomes part of the subcomplement branch generation system in the RST system. A first-order simplified system is also generated, C_{eq_i}, which defines the subcomplement search space for the RST planning system, as shown in Figure 24 (d).

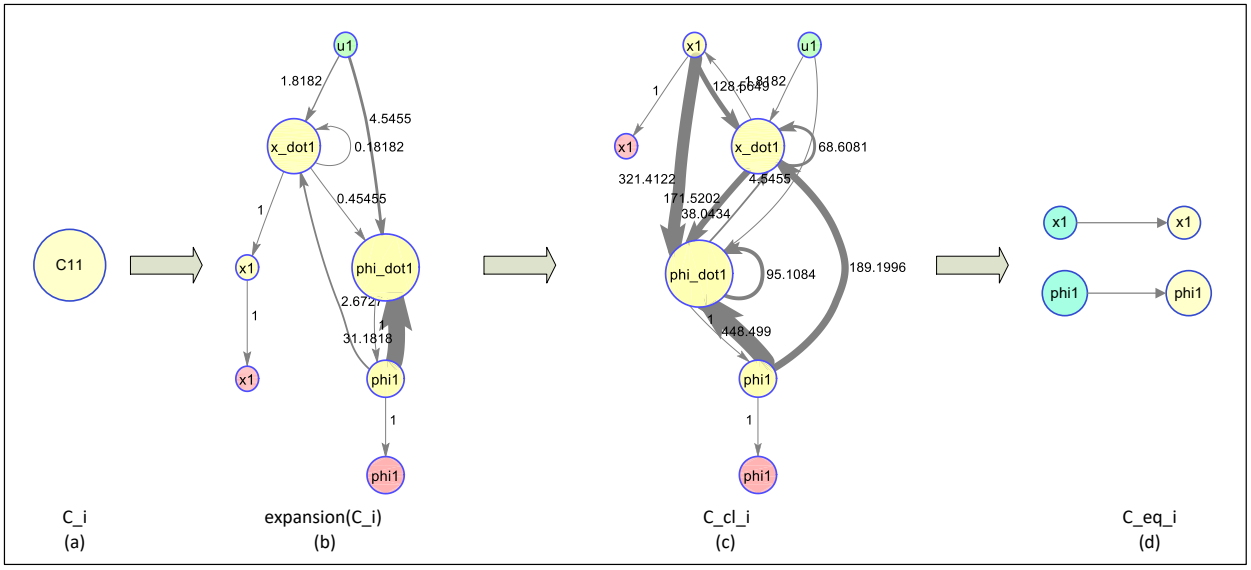


Figure 24. Control-System and Subcomplement-System Generation.

The resulting control response is shown in Figure 25 to a step and impulse command input.

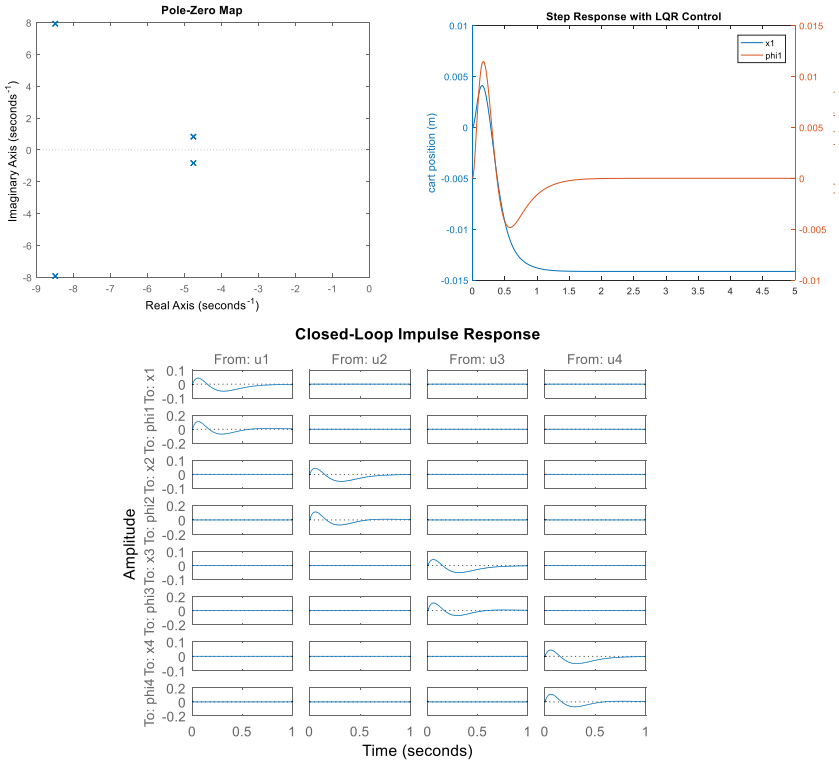


Figure 25. Local Subsystem Control Response.
The poles (top-left), step response (top-right), and large-scale system response (bottom) are shown.

The subcomplement branch-generation system and trajectory planning system is then generated through accumulation of the closed-loop system, shown in Figure 26. The closed-loop graph system, C_{cl}, and equivalent

system, C_{eq}, is constructed by aggregating all the closed-loop systems and equivalent systems from the local subgraphs.

$$C_{cl} = \bigcup_i C_{cl_i} \quad ; \quad C_{eq} = \bigcup_i C_{eq_i} \tag{4.7-43}$$

The cost maps are incorporated directly into the cost generation system, and the constraint map is utilized in the feasibility evaluation and pruning of branches. The construction of the subcomplement system components is illustrated in Figure 26. The inputs to C_{eq} define the subcomplement search space, while the output of C_{eq} is used as guidance input to C_{cl}, which is forward-integrated to generate full trajectories. The details and definitions are described in Chapter 5.

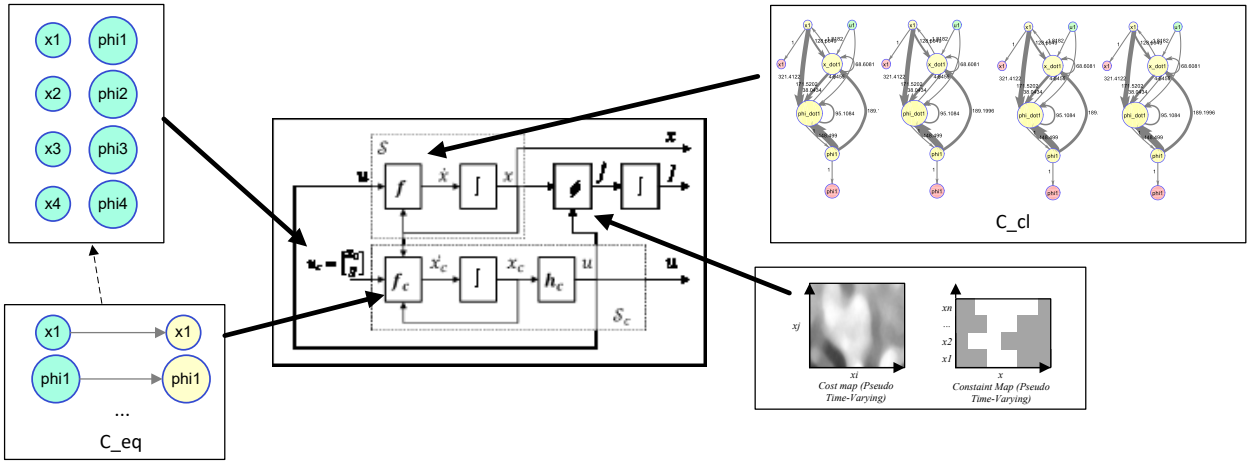


Figure 26. Generation of the subcomplement system for path generation.

In the decoupled case, the resulting control strategy is exactly the same as the global optimal strategy, which is derived by developing a single LQR solution for the entire large-scale system. Consider now the effect of a change to the interconnections of the pendulum. This structure modification to the system is represented as a discrete change to the pseudo-time-varying matrices k, K, and Ku, as shown in Figure 27.

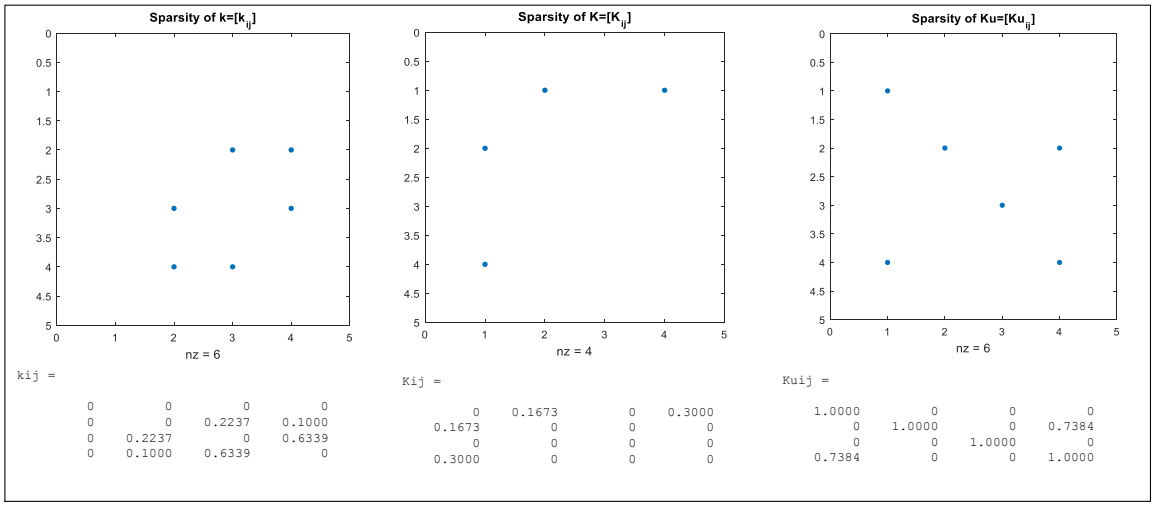


Figure 27. Changes to the Intercoupling Matrices.
Matrix k (left), K (middle), and Ku (right).

The structural modification results in loss of control for this system. The result of this change on the system A and B matrices are shown in Figure 28 (left). The resulting matrices are more densely populated, and the modified interconnections have destabilized several closed-loop modes. After this modification occurs, the existing feedback control law can no longer maintain stability of the pendulums and the system goes unstable, as shown in Figure 28 (right).

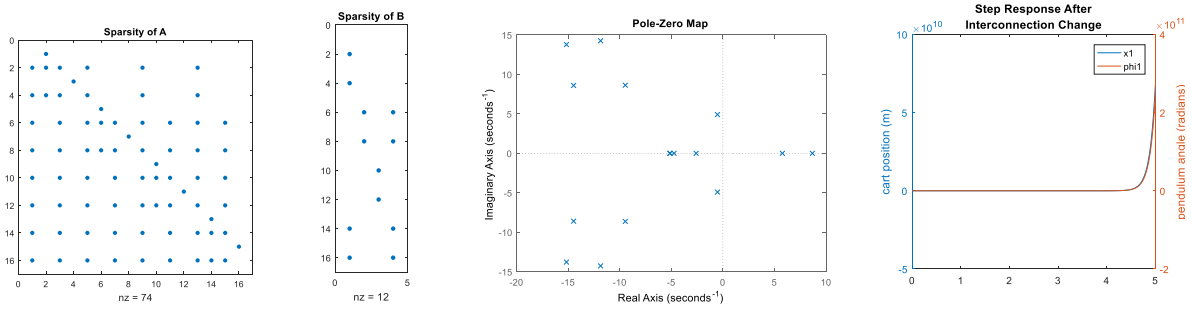


Figure 28. Effect of Changes in the Intercoupling Matrices.
System matrices A (left) and B (left-middle), and closed-loop pole-zero map (right-middle), and unstable closed-loop step-response after the change (right).

When confronted with this structural change, our proposed system automatically adjusts the structure of the local closed-loop controllers. The modified open-loop system component graph is shown in Figure 29.

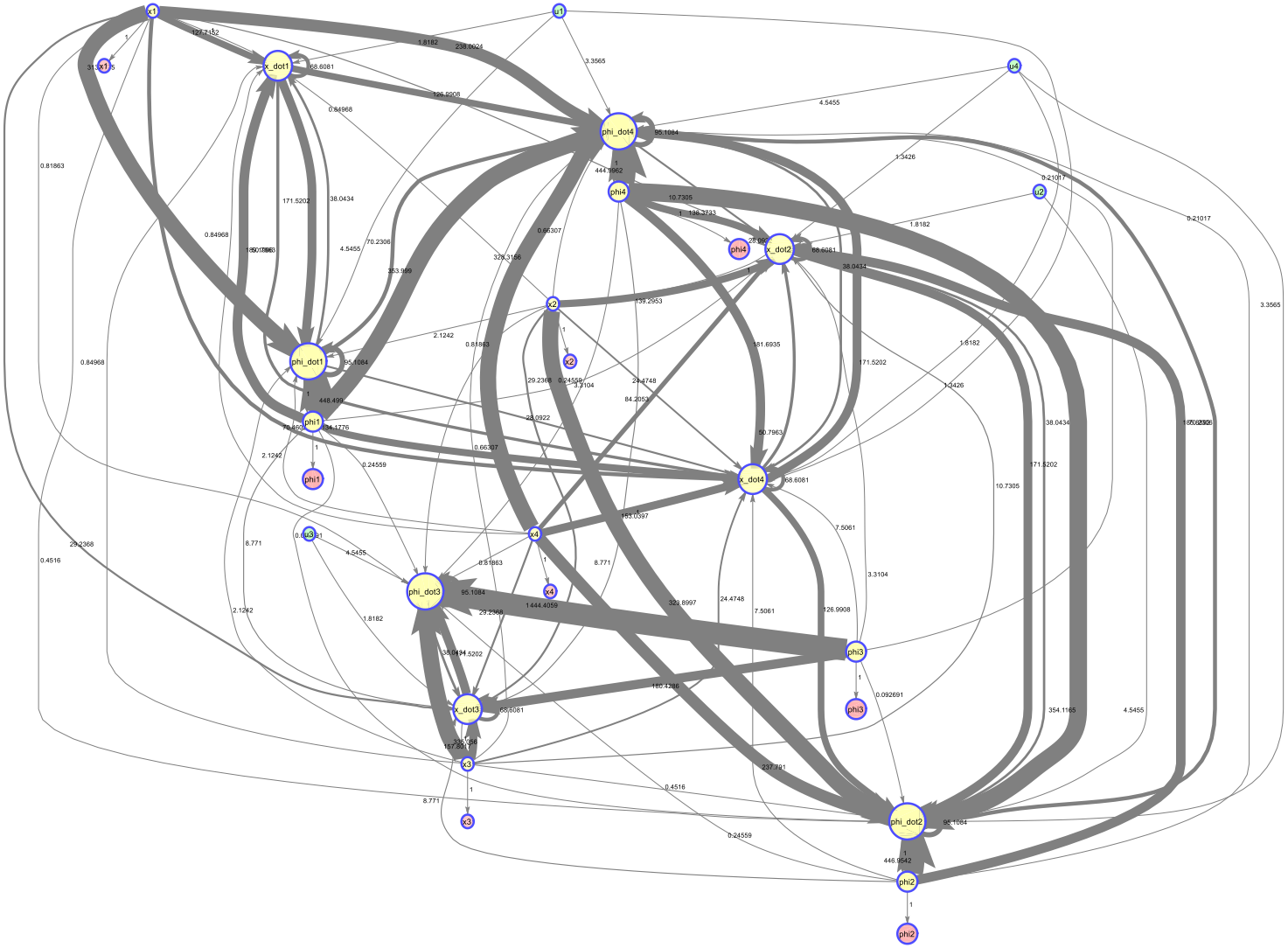


Figure 29. System graph for the modified system.

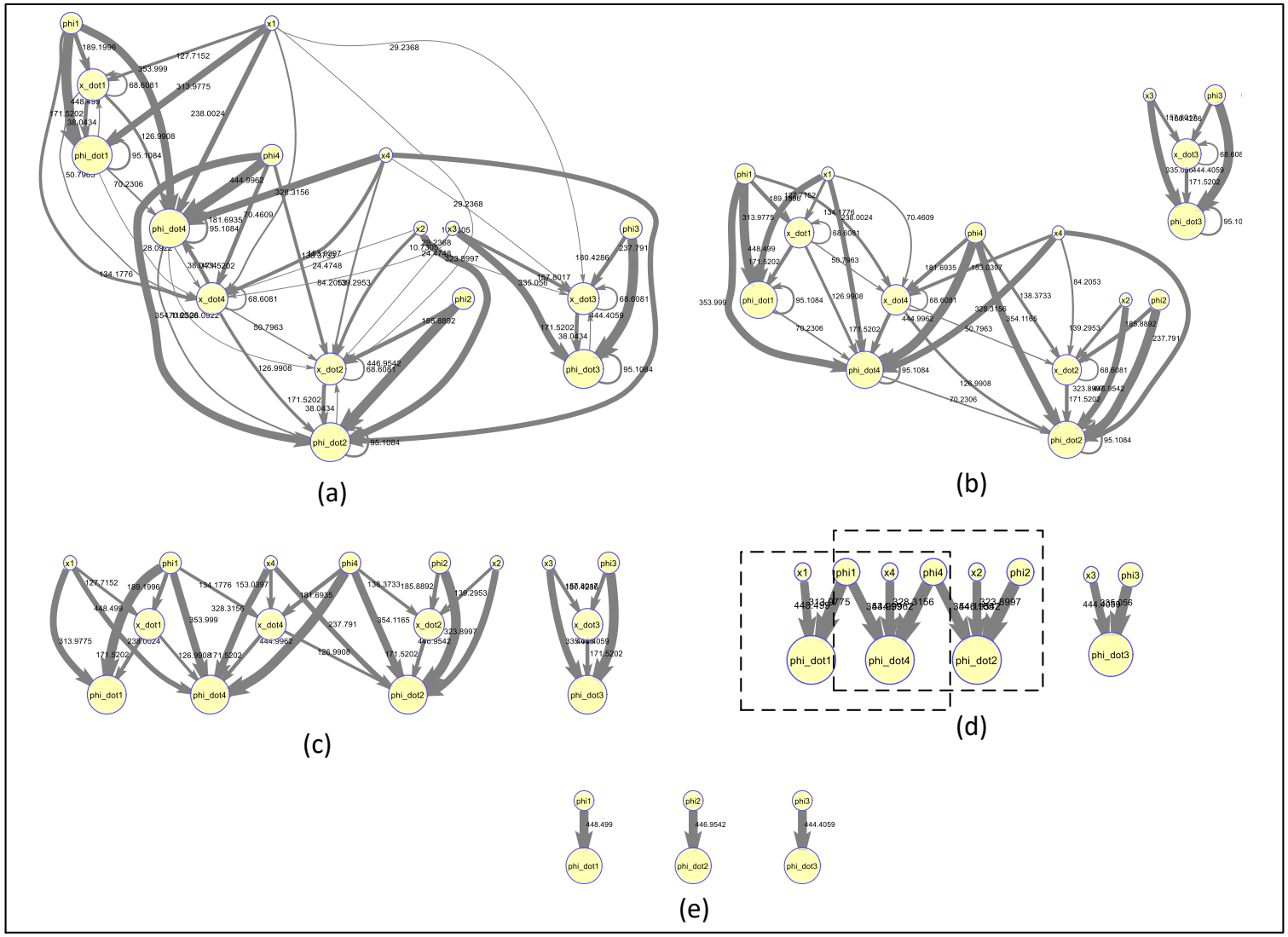


Figure 30. Clustering Algorithm Progression on the Modified System Component Graph

.

The modified graph is analyzed through the graph clustering method. Progression through the graph clustering process is shown in Figure 30. The collapsed system graph after the first iteration is shown in Figure 30 (a), with successive iterations shown (b)-(d). System structure emerges as the algorithm progresses, with a weakly-coupled subgraph appearing in (b), and further reduction occurring through (d). The algorithm at (d) identifies a balanced overlapping condition, and the expansion/contraction method for overlapping systems replicates the shared system containing the fourth cart-pendulum system. After expansion, the algorithm continues to decompose the graph until the penultimate system of three systems is identified at (e).

Continuing the same process, the resulting three decentralized system component graphs are transformed into state-space, new local control systems are generated through the LQR synthesis method, the resulting closed-loop systems are transformed back into the system component graph space, and the full controller is then synthesized. The resulting control structures allow the local systems to maintain stability, while performance of the global system in the face of variable restructuring is still achieved. The resulting stable pole locations and response to commanded inputs are shown in Figure 31.

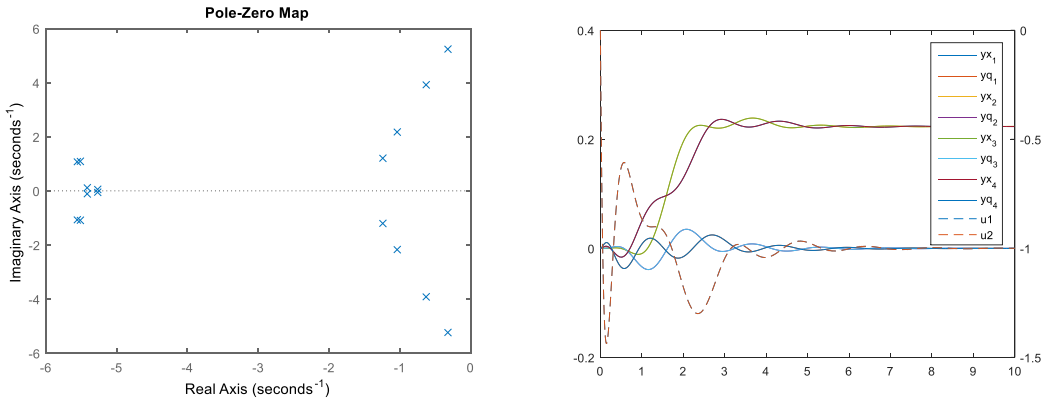


Figure 31. Controller Response to the Structurally Modified System.
Closed-loop system pole locations are shown on the left. Closed-loop system response shown on the right.

Chapter 5 Random Subcomplement Search Trees (RST) Method for Real-Time Trajectory Optimization on Large Variably-Structured Systems

Consider a dynamical system $\mathcal{S}$ given by

$$\mathcal{S}: \begin{cases} \dot{x}(t) = f(x,u,t) \\ y(t) = h(x,t) \end{cases} \tag{4.7-1}$$

where $f: \mathcal{X}\times\mathcal{U}\times\mathcal{T} \to \mathbb{R}^n$, $h: \mathcal{X}\times\mathcal{T} \to \mathcal{Y}$, state space $x \in \mathcal{X} \subseteq \mathbb{R}^n$, input space $u \in \mathcal{U} \subseteq \mathbb{R}^m$, output space $y \in \mathcal{Y} \subseteq \mathbb{R}^p$, and time is defined over the interval $t \in \mathcal{T} \subseteq (0..t_f)$.

Define two sets of constraints $C = \{C_e, C_i\}$, $C_e = \{C_{e_1}..C_{e_{n_e}}\}$, $C_i = \{C_{i_1}..C_{i_{n_1}}\}$ of the following form, where $C_{e_i}, C_{i_i}: \mathcal{X}\times\mathbb{R}^n\times\mathcal{U}\times\mathcal{T} \to \mathbb{R}$.

$$\begin{aligned} C_{e_i}(x,\dot{x},u,t) &= 0 \\ C_i(x,\dot{x},u,t) &\leq 0 \end{aligned} \tag{4.7-2}$$

Define a set of performance objective functions $L = [L_1..L_{n_L}]^T$, where $L_i: \mathcal{X}\times\mathcal{U}\times\mathcal{T} \to \mathbb{R}$, $\mathcal{L}: \mathcal{X}\times\mathcal{T} \to \mathbb{R}$. The overall objective function is defined by

$$J(x,u,t) = \left\| \int_0^{t_f} L(x,u,\tau)d\tau \right\|_1 \tag{4.7-3}$$

Given a system $\mathcal{S}$ subject to constraints C starting at time t=0 with initial state x_0. Find the optimal control u^* which satisfies

$$Find\ \ u^* = \underset{u}{\operatorname{argmin}}\big(J(x,u,t)\big) \tag{4.7-4}$$

Combining the objective J with the system $\mathcal{S}$ yields the augmented system $\tilde{\mathcal{S}}$, defined as follows.

$$\tilde{\mathcal{S}}: \begin{cases} \begin{bmatrix} \dot{x} \\ \dot{J} \end{bmatrix} = \begin{bmatrix} f(x,u,t) \\ \|L(x,u,t)\|_1 \end{bmatrix} \\ [J] = [J(x,u,t)] \end{cases} \tag{4.7-5}$$

Let the output of a system $\tilde{\mathcal{S}}$ integrated from time t=0 to t_f with input u and initial state x_0 be written in shorthand notation as $y_{\tilde{\mathcal{S}}}[t_f: x_0: u]$. Solving equation (4.7-4) is equivalent to solving the following problem.

$$Find\ \ u^* = \underset{u}{\operatorname{argmin}}\big(y_{\mathcal{S}}[0{:}\,t_f{:}\,x_0{:}\,u]\big) \tag{4.7-6}$$

5.1.1 Subcomplement Problem

Given a system $\mathcal{S}$, define a goal subspace $\mathcal{X}_G$. Let $u_c = (x_s, g)$ where $\mathcal{U}_c = \mathcal{X}\times\mathcal{X}_G$ and let $y_c = (x, u, J)$ where $\mathcal{Y}_C = \mathcal{X}\times\mathcal{U}\times\mathbb{R}$. Let a system $\mathcal{S}_c$ be defined as

$$\mathcal{S}_c{:}\begin{cases}[\dot{x_c}] = [f_c(x_c, u_c, t)]\\ [u] = [h_c(x_c, u_c, t)]\end{cases} \tag{4.7-7}$$

Here, $f_c{:}\,\mathcal{X}_C\times\mathcal{U}_c\times\mathcal{T}\to\mathbb{R}^n$, $h_c{:}\,\mathcal{X}_C\times\mathcal{T}\to\mathcal{Y}_c$, $x_c\in\mathcal{X}_c\subseteq\mathbb{R}^{n_c}$, $u\in\mathcal{U}_C$, $y\in\mathcal{Y}_c$. The system $\mathcal{S}_C$ can be combined with system $\mathcal{S}$ and the objective function J to form the augmented system $\tilde{\mathcal{S}}_C$ as follows.

$$\tilde{\mathcal{S}}_c{:}\begin{cases}\begin{bmatrix}\dot{x}\\ \dot{x_c}\\ \dot{J}\end{bmatrix} = \begin{bmatrix}f(x,u,t)\\ f_c(x_c,u_c,t)\\ \|L(x,u,t)\|_1\end{bmatrix}\\ \begin{bmatrix}u\\ x\\ J\end{bmatrix} = \begin{bmatrix}h_c(x_c,u_c,t)\\ x\\ J\end{bmatrix}\end{cases} \tag{4.7-8}$$

The output of a system $\mathcal{S}_c$ is the *subcomplement* of the original system $\mathcal{S}$. If optimizing over the subcomplement system $\mathcal{S}_C$ yields the exact solution to equation (4.7-4) for the original system $\mathcal{S}$, then $\mathcal{S}_C$ can be used to find the exact solution. More formally, a system $\mathcal{S}_C$ defined in (4.7-7) is a *perfect subcomplement* of a system $\mathcal{S}$ if the following holds for some $x_c(0)$:

$$y_{\tilde{\mathcal{S}}_C}\left[t_f{:}\begin{bmatrix}x_0\\ x_c(0)\\ 0\end{bmatrix}{:}\begin{bmatrix}x_0\\ x(t_f)\end{bmatrix}\right] = \underset{u}{\operatorname{argmin}}\big(J(x,u,t)\big) \tag{4.7-9}$$

Graphically, the systems $\mathcal{S}$ and $\mathcal{S}_C$ are shown in Figure 32.

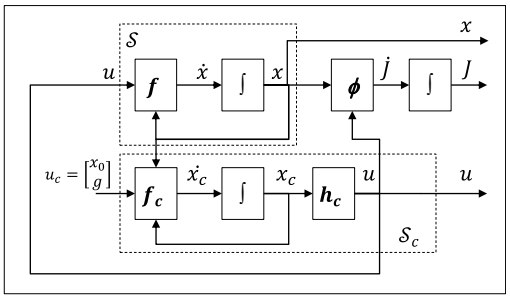


Figure 32. Augmented Subcomplement System $\mathcal{S}_C$ of a system $\mathcal{S}$

Given the optimization problem in (4.7-4) for a system $\mathcal{S}$ and a subcomplement system $\mathcal{S}_C$ for $\mathcal{S}$. Let $x_{C0} = [x_0\ 0_{nc}\ 0]^T$. The *approximate subcomplement solution* to (4.7-4) is $u^{\tilde{*}}$ given by solutions to the following problem.

$$Find\ \ u^{\tilde{*}}(t) = y_{\tilde{S}_c}\left[t_f : x_{c_0} : [x_0\ \ g^*(t)]^T\right][\mathbf{1}_m\ \mathbf{0}_n\ 0]^T \tag{4.7-10}$$

The optimal g* is given by

$$g^*(t) = \underset{g(t)}{\operatorname{argmin}}\left\{y_{\tilde{S}_c}\left[t_f : x_{c_0} : [x_0\ \ g(t)]^T\right][\mathbf{0}_m\ \mathbf{0}_n\ 1]^T\right\} \tag{4.7-11}$$

The subcomplement system will be used to generate fast high quality but suboptimal solutions to the general problem using a random search tree algorithm. Defining a subspace $\mathcal{X}_G$ may help reduce the dimensionality of the problem in (4.7-4) and yield a more tractable solution with higher quality solutions. Intuitively, the reduction of dimension will cause a loss of information, and achieving optimality in the original system is not possible for the general case. The degree of suboptimality can be characterized through evaluation of the error between the resulting approximated solution and full-system optimal solution. This error is given by the following.

$$e^{\tilde{*}} = \left\| \int_0^{t_f} \left(L(x^{\tilde{*}}, u^{\tilde{*}}, \tau) - L(x^*, u^*, \tau) d\tau\right) \right\|_1 \tag{4.7-12}$$

5.1.2 Optimization Algorithm

Let the search tree $\mathcal{T} = (V, E)$ be defined as a set of vertices $\mathcal{V} = (\mathcal{X}, \mathcal{U}, \mathcal{T}, \mathbb{R})$ where a vertex $v_i \in \mathcal{V}$ given by $v_i = (x(t_i)\,, u(t_i), J(x_i, u_i, t_i), t_i)$ represents the exact value of the state x, input u, and objective function value J in a trajectory at time instance t_i. Let an edge $E =< V, V >$ be an ordered set of vertices. Informally, let the square bracket operator $a[\cdot]$ return an element a from any set variable. Let the parent function $p_G(v)$ be defined as $p_G(v) = u \mid < u, v > \in E(G)$, which is unique and exists for all $v \in V(G)$ except for the root vertex of the tree. Note that integrating the trajectory from $p_G(v)$ to v will result in a very small bounded error. Let this be specified by the shorthand notation $e_x(t[p[v]] : t[p])$, given by

$$\begin{aligned} e_x\left(t\left[p[v]\right] : t[p]\right) &= \|x[v] - \tilde{x}[v]\|_2 \\ &= \left\| x[v] - x\left[p[v]\right] + \int_{t[p[v]]}^{t[v]} f\left(x\left[p[v]\right], u\left[p[v]\right], t\left[p[v]\right]\right) d\tau \right\|_2 \end{aligned} \tag{4.7-13}$$

This error - solely the result of this data representation - may be bounded for certain classes of controllers that can keep this residual zero along an infinite sequence of points. The search algorithm is shown below, and is inspired by the search algorithm in RRT algorithms.

Table 3. BuildOptimizationTree() Algorithm

Algorithm 1. BuildOptimizationTree (x_0, $\mathcal{G}$, C)

Input: x_0: Start state, $\mathcal{G}$: Branch generation system, C: Constraint set

Variables: $\mathcal{T}$: Tree, (v, v_l, v*): Vertex (current, leaf, best)

1. $\mathcal{T} \leftarrow$ InitTree(x_0)
2. v*←∅
3. while (not *StopCondition*()) do
4. g←RandomGoalPoint()
5. v←RandomTreeVertex()
6. v_l←*GenerateBranch*($\mathcal{T}$, $\mathcal{G}$, v, g, C)
7. v*←*StoreBest*(v*,v_l)
8. End while

Table 4. GenerateBranch() Algorithm

Algorithm 2. GenerateBranch ($\mathcal{T}$, $\mathcal{G}$, v, g, C)

Input: $\mathcal{T}$: Tree, v: Start vertex, g: Goal vertex, C: Constraint set

Variables: Tree $\mathcal{T}$, Vertex v', Branch b

1. b←FwdIntegrate ($\mathcal{G}$, v', g)
2. b←*Trim*(b,C)
3. if (b ≠ ∅)
4. *TreeAdd* ($\mathcal{T}$, v, b)
5. End if

5.1.3 Synthesis Methodology

The subcomplement definitions provide a methodology to find approximate solutions to the optimization problem in (4.7-4). The augmented subcomplement system shown in Figure 32 will be utilized to quickly generate feasible branches using the following process.

1. Implement the original system S in the solver.
2. Augment the plant model with constraints.
3. Implement the cost metrics J.
4. Define the goal space $\mathcal{X}_C$ and the form of the augmented subcomplement system in (4.7-8).

5. Design the augmenting control system f_c and h_c to control the augmented plant from an initial state $x_0 \in \mathcal{X}$, to a destination point in the goal space $g \in \mathcal{X}_g$.
6. Implement G' in the online trajectory search algorithm outlined above.

Chapter 6 Applications

In this chapter, we provide a select set of examples that illustrate application of the approach in Chapter 3, system graph model formulation presented in Chapter 4, and optimization algorithm presented in Chapter 5. These examples have arisen from real-world research problems that these approaches have addressed. The following sections provide examples of how to derive models and transform them into the graph space, how to formulate these problems into this framework, and how to apply the algorithms presented toward solving these issues. The first two examples in sections 6.1 and 6.2 demonstrate application of the RST real-time optimization approach for unmanned flight systems. The example in section 6.2 demonstrates application of this framework toward decentralized swarm control of a fleet of unmanned aerial systems that optimizes mission objectives for monitoring and tracking of volcanic plumes. This application presents new model formulations for the RST optimization, presents hardware developed for vehicle test evaluation, and presents new CFD models developed for analysis of wind fields around the target volcano – the Turrialba volcano in San José, Costa Rico. The application in 6.3 demonstrates analysis and application of the mathematical formulation to allow on-the-fly control system reconfiguration to allow a ground robotic vehicle to traverse through a smart-building while closing control loops around various sensors and actuators within the building. The application in 6.4 demonstrates application of this framework and optimization toward lateral and longitudinal flight control with structural stabilization of a flexible aircraft wing with continuous distributed trailing edge actuators. The final example, presented in 6.4.5, demonstrates application of this framework toward developing an intelligent and integrated building control system that takes advantage of overlapping sensors and actuators from traditionally independent control systems in the NASA Sustainability Base - a modern smart building structure. This section presents a new graph-based mathematical model for building control simulation that was developed specifically for transformation into the graph space model framework presented in Chapter 4, with results from validation experiments. This section further shows application of the mathematical framework and optimization algorithms, followed by simulation and control results presented for various small configurations of building systems.

6.1 Pseudo-Optimal Real-Time Path Planning for Automated Smoke-Plume Monitoring from an Unmanned Aerial Vehicle

6.1.1 Model Description

Let $\mathcal{F}_B, \mathcal{F}_W, \mathcal{F}_S$ represent the vehicle body axis frame, the world frame, and the sensor axis frame respectively. Let the rotation between a frame $\mathcal{F}_A$ to a frame $\mathcal{F}_B$ be given by $R_{AB}(x) \in \mathbb{R}^{n\times n}$.

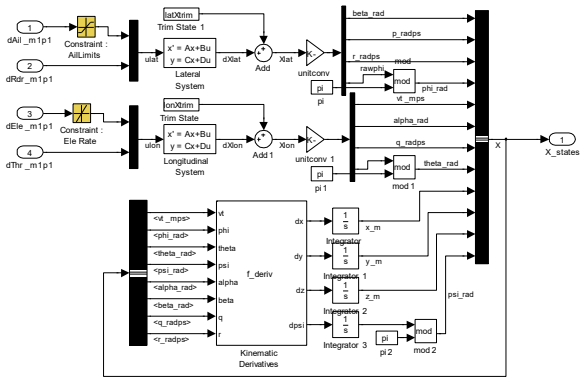


Figure 33. Simulink Implementation of the Model

The UAV plant model needs to be able to resolve the sensor. The state is given by $x = [x_{lat}\ x_{lon}\ x_{kin}]^T$, n=12, $x_{lat} = [\Delta\beta\ \Delta\mathrm{p}\ \Delta\mathrm{r}\ \Delta\phi]$, $x_{lon} = [\Delta v_t, \Delta\alpha, \Delta\theta, \Delta q]$, $x_{kin} = [P_w\ \psi]$. Control input is given by $u = [u_{lat}\ u_{lon}]^T \in \mathbb{R}^{12}$, m=4, $u_{lat} = [\delta_{ail}, \delta_{rdr}]$, $u_{lon} = [\delta_{ele}, \delta_{thr}]$ where $x_{lat}, x_{lat}, x_{kin} \in \mathbb{R}^4$, $u_{lat}, u_{lon} \in \mathbb{R}^2$, world position $P_w \in \mathbb{R}^3$. The dynamics equations are given by the form

$$\begin{aligned}
\dot{x}_{lat} &= A_{lat}x_{lat} + B_{lat}(u_{lat}) \\
\dot{x}_{lon} &= A_{lon}x_{lon} + B_{lon}(u_{lon}) \\
\dot{P}_W &= R_{BW}R_{W_i2B}[1\ 0\ 0]^T\left(v_{T_0} + \Delta v_T\right) \\
\dot{\psi} &= \begin{bmatrix} \frac{sin\phi}{cos\theta} & \frac{cos\phi}{cos\theta} \end{bmatrix}\begin{bmatrix} \mathrm{q}_0 + \Delta q \\ \mathrm{r}_0 + \Delta r \end{bmatrix}
\end{aligned} \tag{6.1-1}$$

The dynamics for the aircraft were estimated from flight data from the NASA EAV (Exploration Aerial Vehicle) at 400ft altitude and 40kts airspeed.

$$A_{lat} = \begin{bmatrix} -1.0176 & 0.0728 & -0.8887 & 0.2918 \\ -16.7704 & -5.2969 & 1.2992 & 0 \\ 13.7306 & -0.3138 & -1.3475 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix};\quad B_{LAT} = \begin{bmatrix} 0.3404 & 0.5297 \\ 55.2736 & 4.6948 \\ -1.4038 & -9.7679 \\ 0 & 0 \end{bmatrix}$$

$$A_{lon} = \begin{bmatrix} -1.409 & -0.040515 & 0 & -0.096803 \\ -0.0281 & -4.0540 & 1.2896 & 0 \\ -0.0566 & -20.9832 & -1.6784 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix};\quad B_{lon} = \begin{bmatrix} -0.0699 & 3.24 \\ -0.8405 & -0.03057 \\ -25.99 & -0.07424 \\ 0 & 0 \end{bmatrix} \tag{6.1-2}$$

The control surfaces for the EAV are rate limited, added as constraints given by

$$C_1: \left|\dot{\delta}_{ail}\right| < 0.5\ rad/s$$
$$C_2: \left|\dot{\delta}_{ele}\right| < 0.5\ rad/s \tag{6.1-3}$$

The constraints given in (6.1-3) were incorporated into the branch generation system.

In addition to the input constraints, the UAV must avoid certain regions of space that are too dangerous to fly into, while at the same time must remain in the approved airspace. These constraints can be written as

$$C_{obj_i}: P(t) \notin P_{obj_i} \quad for\ i = 1\ to\ n_{obj}$$
$$C_{AA}: P(t) \in P_{AA} \tag{6.1-4}$$

Consider three objective functions to minimize:

1. L_1: Sensor pointing accuracy to target
2. L_2: Aircraft distance to target
3. L_3: Sun angle relative to camera direction

The geometry of the sensor frame relative to the target is shown in Figure 34, where the imager is oriented along $[0\ 1\ 0]^T$ in $\mathcal{F}_s$. Let the position of the vehicle and the target observation point be given by P_v, and P_{to}, respectively, and define $v_{vo} = P_{to} - P_v$. The L_2 objective function is modeled as a curve that falls off with the square of the distance to the target position. Let r_d be the desired distance from the sensor to the phenomena, and let r_Δ be the deviation distance. Let $\hat{v}_{sun}$ be the normalized sun pointing vector.

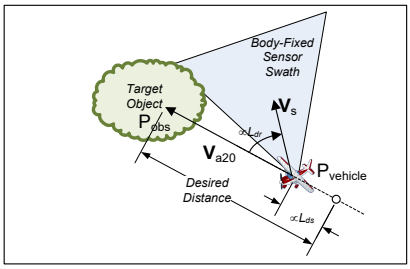


Figure 34. Geometry for Sensor Pointing Accuracy

The costs were modeled as follows.

$$L_1 = \frac{1 - \left(\frac{v_{vo}}{\|v_{vo}\|_2} \cdot (R_{b2w} R_{s2l} [0\ 1\ 0]^T) \right)}{2}$$
$$L_2 = \min\left(1, \frac{(\|v_{vo}\|_2 - r_d)}{r_\Delta^2}\right) \tag{6.1-5}$$
$$L_3 = 1 - \left(\hat{v}_{sun} \cdot (R_{b2w} R_{s2l} [0\ 1\ 0]^T)\right)$$

The subcomplement system in (4.7-7) must be designed so that the approximate solution of (4.7-10) can be generated. For this problem, the goal space was chosen as $\mathcal{X}_G = P_w \times \psi$, where $P_w \in \mathbb{R}^3$ is the world position and ψ is the Euler yaw angle. The control problem for a vehicle to go from an initial position to a final position and orientation is analogous to a 'track-to' mode in a vehicle autopilot system. A 'track-to' controller was designed for this example; the autopilot design is shown in Figure 35. The controller guides an aircraft from one waypoint to the next by tracking the line between the waypoints. The crosstrack error (XT_{err}) is used to calculate a delta heading angle (ψ_Δ) which is the deviation from the nominal heading angle (ψ_{nom}). This relationship is shown graphically below.

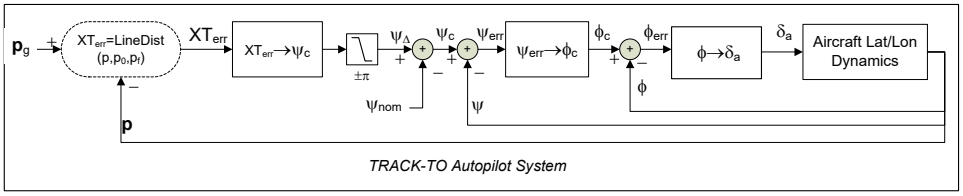


Figure 35. Track-To Autopilot Design

The Simulink block diagram for the complete augmented subcomplement system is shown in Figure 36.

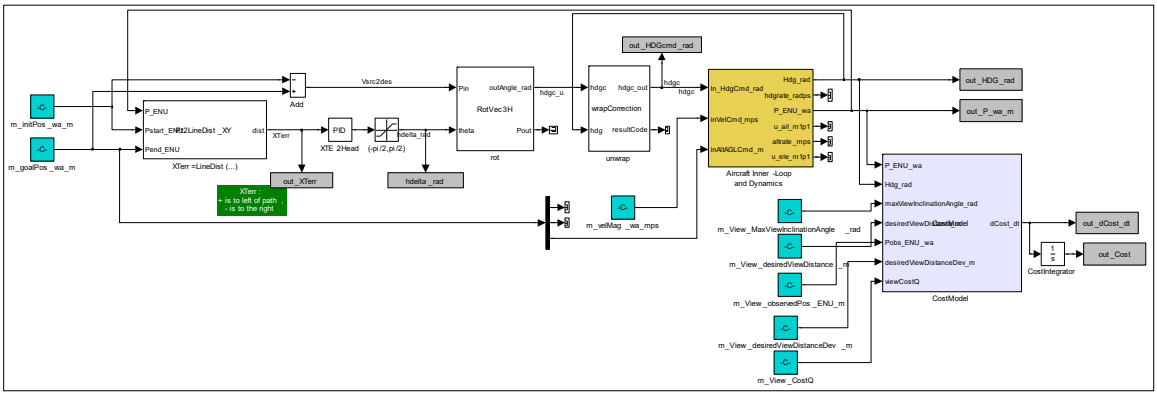


Figure 36. Simulink Model for Trajectory Control System

6.1.2 Simulation Testing

A simulation framework was created to investigate the wildfire monitoring problem. A dynamic smoke plume simulation was created with a particle-based rendering visualization, and vehicle fixed cameras were simulated by

rendering the scene from the point of view of the cameras to offscreen bitmaps. The UAV is required to field sensors around this smoke plume whose position is not known a priori. The obstacle constraints are provided by a plume estimation system that processes images in real time to determine the likely position of the plume. These dynamic environment maps are passed to the optimizer as the constraint set C_{obj_i} that the trajectory planner cannot violate. The centroid of the plume is provided to the optimization performance objectives L_1, L_2 as the position of the target observation point P_{to}. If the planner calculates poor trajectories around the plume, the image processing algorithm will never identify the plume and the aircraft may violate the constraints.

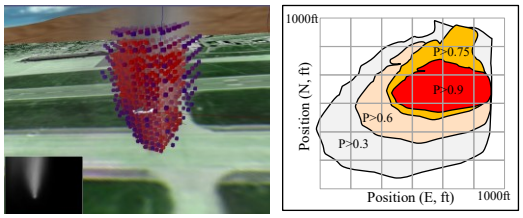


Figure 37. Vision-Based Smoke Plume Detection.
Simulation (left) and a 2D Cross-Section (right)

A nonlinear 6-DOF simulation model was created for the vehicle based on aerodynamic coefficient lookup tables. This model is shown in Figure 38. The controller's C++ code was autogenerated from the Matlab Simulink models, and the controller was tested simulation using the Reflection Architecture. The scenario was tested on an Intel Core 2 X9650/3GHZ/ 3GB.

States:

$\mathrm{P}_e = [X_e\ Y_e\ Z_e]^T$; $\mathrm{V}_b = [u\ \ v\ \ w]^T$; $\mathrm{q} = [q_0\ q_1\ q_2\ q_3]^T$; $\omega_b = [p\ \ q\ \ r]^{\mathrm{T}}$

Kinematics:

$$\dot{\mathrm{P}}_e = \left(\Omega_{E_e}\mathrm{P}_e\right) + \mathrm{R}_{b2e}\mathrm{V}_b$$
$$\dot{\mathrm{q}} = -0.5\widetilde{\omega}_{\mathrm{b}}\mathrm{q}$$
$$\dot{\mathrm{V}}_b = -(\omega_b \times \mathrm{v}_b) - \left(\mathrm{R}_{e2b}\Omega_{E_e}^2 + \mathrm{R}_{e2b}\Omega_{E_e}\mathrm{R}_{b2e}\omega_b\right) + \mathrm{R}_{e2b}\mathrm{g}_e + m^{-1}\left(\mathrm{F}_{PR_b} + [-Drag\ Y - Lift]R_{wi2b}\right)$$
$$\dot{\omega}_b = -\mathrm{J_b}^{-1}\widetilde{\omega}_b\mathrm{J_b} + \mathrm{J_b}^{-1}\left(\mathrm{T}_{PR_b} + [L\ M\ N]\right)$$

Dynamics:

$$\mathrm{Lift} = \mathrm{QSC}_{\mathrm{l}_\alpha}(\alpha, \delta_{\mathrm{flp}}) + \mathrm{Q} * \mathrm{S}_{\mathrm{ht}} * \frac{\mathrm{dC_l}}{\mathrm{d}\delta_{\mathrm{ele}}}\delta_{\mathrm{ele}} + \left(\frac{\mathrm{QS}\bar{\mathrm{c}}}{2\mathrm{V}}\right)\frac{\mathrm{dC_l}}{\mathrm{d}\dot{\alpha}}\dot{\alpha} + \left(\frac{\mathrm{QS}\bar{\mathrm{c}}}{2\mathrm{V}}\right) * \frac{\mathrm{dC_l}}{\mathrm{dq}}\mathrm{q}$$

$$\mathrm{Drag} = \mathrm{QSC}_{\mathrm{D}_0} + \mathrm{QS} * \mathrm{C}_{\mathrm{D}_\alpha}(\alpha, \delta_{\mathrm{flap}}) + \mathrm{QSC}_{\mathrm{D}_{\delta_{\mathrm{flap}}}}\left(\delta_{\mathrm{flap}}\right) + \mathrm{QS}_{\mathrm{HT}}\left(\frac{\mathrm{dC_D}}{\mathrm{d}\delta_{\mathrm{ele}}}\right)\delta_{\mathrm{ele}} + \mathrm{QS}\left(\frac{\mathrm{dC_D}}{\mathrm{d}\beta}\right)\beta$$

$$\mathrm{Y} = \mathrm{QSC}_{\mathrm{Y}_\beta}(\beta) + \mathrm{QS}\left(\frac{\mathrm{dC_Y}}{\mathrm{d}\delta_{\mathrm{ail}}}\right)\delta_{\mathrm{ail}} + \mathrm{QS}\left(\frac{\mathrm{dC_Y}}{\mathrm{d}\delta_{\mathrm{rdr}}}\right)\delta_{\mathrm{rdr}} + \mathrm{QS}\left(\frac{\mathrm{b}}{2\mathrm{V}}\right)\left(\frac{\mathrm{dC_Y}}{\mathrm{dp}}\right)\mathrm{p} + \mathrm{QS}\left(\frac{\mathrm{b}}{2\mathrm{V}}\right)\left(\frac{\mathrm{dC_Y}}{\mathrm{dr}}\right)\mathrm{r}$$

$$\mathrm{L} = \mathrm{QSC}_{\mathrm{L}_\beta}(\beta) + \mathrm{QS}\frac{\mathrm{b}^2}{2\mathrm{V}}\left(\frac{\mathrm{dC_L}}{\mathrm{dr}}\right)\mathrm{r} + \mathrm{QS}\left(\frac{\mathrm{b}^2}{2\mathrm{V}}\right)\left(\frac{\mathrm{dC_L}}{\mathrm{dp}}\right)\mathrm{p} + \mathrm{QSb}\left(\frac{\mathrm{dC_L}}{\mathrm{d}\delta_{\mathrm{ail}}}\right)\delta_{\mathrm{ail}} + \mathrm{QS}_{\mathrm{vt}}\overline{\mathrm{MA}}_{\mathrm{vt}}\left(\frac{\mathrm{dC_L}}{\mathrm{d}\delta_{\mathrm{rdr}}}\right)\delta_{\mathrm{rdr}}$$

$$\mathrm{M} = \mathrm{QS}\bar{\mathrm{c}}\left(\frac{\mathrm{dC_M}}{\mathrm{d}\delta\alpha}\right)\alpha + \mathrm{QS}\bar{\mathrm{c}}\mathrm{C}_{\mathrm{M}_0} + \mathrm{QS}\bar{\mathrm{c}}\mathrm{C}_{\mathrm{M}_{\mathrm{flap}}} + \mathrm{QS}\left(\frac{\bar{\mathrm{c}}}{2\mathrm{V}}\right)\frac{\mathrm{dC_M}}{\mathrm{dq}}\mathrm{q} + \mathrm{QS}\frac{\bar{\mathrm{c}}^2}{2\mathrm{V}} * \dot{\alpha} * \frac{\mathrm{dC_M}}{\mathrm{d}\dot{\alpha}} + \mathrm{QS}_{\mathrm{ht}}\overline{\mathrm{MA}}_{\mathrm{ht}}\left(\frac{\mathrm{dC_M}}{\delta_{\mathrm{ele}}}\right)\delta_{\mathrm{ele}}$$

$$\mathrm{N} = \mathrm{QSbC}_{\mathrm{N}_\beta}\beta + \mathrm{QS}\left(\frac{\mathrm{b}^2}{2\mathrm{V}}\right)\frac{\mathrm{dC_N}}{\mathrm{dp}}\mathrm{p} + \mathrm{QS}\left(\frac{\mathrm{b}^2}{2\mathrm{V}}\right)\frac{\mathrm{dC_N}}{\mathrm{dr}}\mathrm{r} + \mathrm{QSb}\left(\frac{\mathrm{dC_N}}{\delta_{\mathrm{ail}}}\right)\delta_{\mathrm{ail}} + \mathrm{QS}_{\mathrm{vt}}\overline{\mathrm{MA}}_{\mathrm{vt}}\left(\frac{\mathrm{dC_N}}{\mathrm{d}\delta_{\mathrm{rdr}}}\right)\delta_{\mathrm{rdr}}$$

Figure 38. Nonlinear Simulation Model

The tree search algorithm was scheduled to run every 20 seconds for 5 seconds. The performance results from a 30-minute test are shown in Table 5. A visualization of the search graph is shown in Figure 39. This image shows the 500 lowest cost trajectory branches with color coded costs shown along the 3D trajectories (red represent higher costs, blue represents lower costs). Figure 40 shows the cost derivatives L_1 and L_2 for six branches starting from a common node in the tree. Figure 41 shows the cost function evaluated over nodes for the best (lowest-cost) paths.

Table 5. Simulation Performance Results from RST Optimization on Smoke Plume Problem

N Trees	Search Time (avg)	Number of Constraints (avg)	Number of Violations (total)	Vertices Added (avg)	Branches Explored (avg)	CPU Cycles (avg)
90	4.99s	142	0	48,202	3,435	15.0E9
90	4.99s	148	0	49,526	3,620	15.0E9

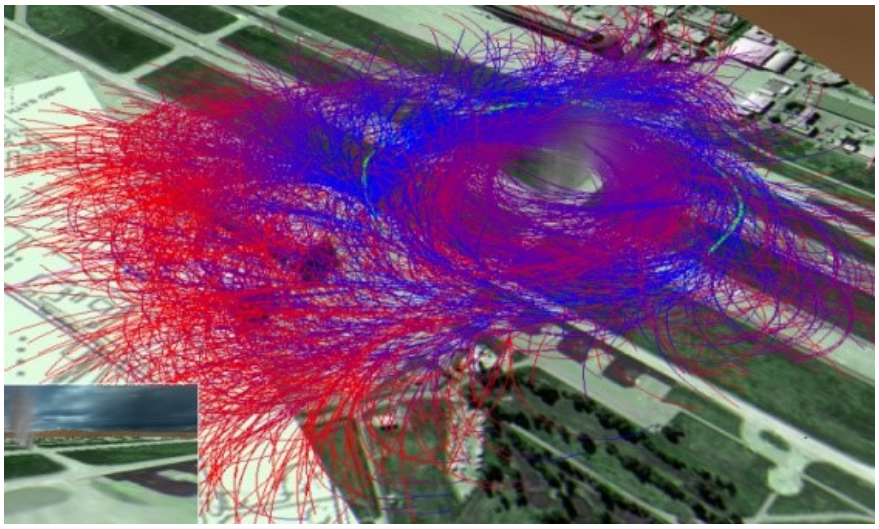


Figure 39. Visualization of Search Tree

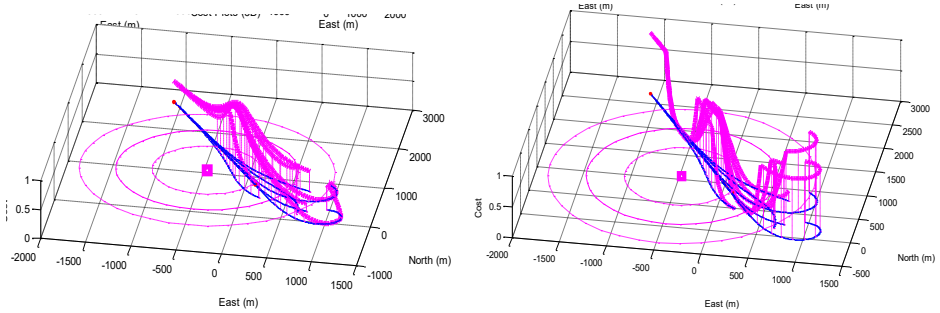


Figure 40. Trajectory Cost Derivatives L_1 (left) and L_2 (right).
Trajectories shown from a branch in the tree.

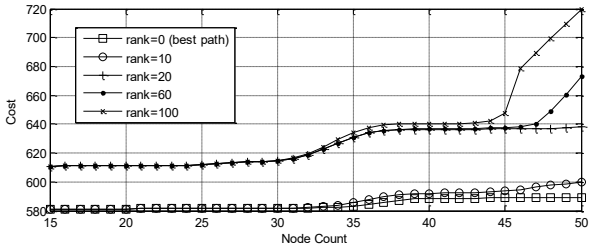


Figure 41. Cost Increase over Best Paths.

6.2 Decentralized Volcanic Plume Monitoring from Unmanned Aerial Vehicle Platforms

Volcanologists face an urgent need for greater in situ sampling capabilities in and around plumes and drifting ash clouds resulting from explosive volcanic eruptions. Current modeling efforts to detect, characterize, and track volcanic emissions are hindered by very sparse in situ validation data, a chronic and pervasive problem identified

within the NASA's Earth Surfaces and Interior (ES&I) focus area and more generally within the volcanological literature. Collection of data is of crucial scientific importance for understanding the dynamics and chemistry of volcanic activity, particularly for validation of existing ash plume detection, retrieval, and transport algorithms (Pieri and Abrams 2004) (Yamaguchi, et al. 1998) (Simpson, et al. 2000) (Xi, et al. 2014).

Although unmanned aerial systems (UAS) are well-suited for performing pre-programmed remote surveys in safe locations far away from hazardous phenomena and where the survey area is fixed and well-known, collecting meaningful in situ data in and around uncertain, dynamic, and dangerous phenomena such as volcanic plumes is exceedingly difficult with current state-of-the-art technology. These data would be more readily collectable if vehicles had the ability to autonomously find and track complex non-stationary earth science phenomena, the evolution of which is usually difficult to predict ahead of time. Further hindering the utility of UAS in this application domain is the inability of on-vehicle autonomy to estimate, sense, and avoid hazardous environmental conditions, making in-situ sampling near dangerous sources impossible without risking the vehicle platform. This unfortunately drives in-situ solutions to small disposable platforms which limits the scientific significance of the collected data.

In this section we address these deficiencies, presenting an intelligent automated system for coordinating multiple unmanned aerial vehicles to locate, characterize, track, and monitor emission, transport, composition, and in situ properties of aerosol emissions from active volcanic sources. Particularly, we address two limitations of current UAS autonomy that hinder greater in situ measurement capabilities. First, UAS currently have no ability to autonomously sense, understand, and react to the hazards in the environment or the observation targets associated with payload sensors. Second, UAS are currently unable to coordinate flight activities in a scalable manner. The approach presented incorporates model-based predictions based on computational fluid dynamic (CFD) models of the system which are used as a priori predictions of the wind vector field and plume state as functions of environmental conditions. These offline CFD predictions are incorporated into an online probabilistic model that produces real-time estimates of aerosol plume location and density based on real-time data streamed from a distributed network of airborne and ground-based sensors. Predictions from this model are utilized for real-time trajectory optimization of the airborne vehicle platforms around the volcano, generating flight plans that place sensors in locations that will maximize scientific data return, identify and avoid dangerous areas of adverse winds and turbulence, take advantage of local wind patterns, and adhere to real-time constraints such as synchronizing observations with the sensor swath of thermal infrared (TIR) sensors of the NASA-MITI Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) instrument

onboard the NASA Terra satellite (Simpson, et al. 2000). We describe the requirements and challenges involved with autonomous in situ volcanic plume monitoring. An outline of the approach is presented with a description of the architectures being built for planned test-flights over and around the Turrialba volcano in Costa Rica. Preliminary results from analysis, simulation, and initial flight testing are then presented.

6.2.1 Background

The ability for autonomous airborne systems to collect systematic in situ data of eruption plumes and drifting ash clouds would address several identified urgent needs for the science community. For instance, the initial phase of volcanic eruptions are often unobserved due to considerable uncertainty in prediction models, yielding considerable uncertainty in even basic predictions, such as plume geographic extent. Remote sensing from spaceborne platforms, for instance the NASA ASTER instrument, provides crucial information for monitoring these processes but by itself is not sufficient to answer all the science questions without additional sensor data measurements provided from in situ measurements. In situ measurements provide calibration and validation, provide data not available from the vantage of a remote sensor, and are necessary for interpretation of remote sensor data. For instance, it is currently difficult to directly relate ASTER TIR measurements and UV ash-loading retrieval models to infer actual in-plume concentrations (Simpson, et al. 2000) (Xi, et al. 2014).

This section presents methods toward enabling safe and reliable collection of high-quality in situ data around complex natural phenomena such as volcanic plumes. For this problem, we characterize these natural phenomena as large, dynamic, hazardous, uncertain, and time-varying. Uncertainty in the phenomena requires responsiveness of measurement platforms and adaptability in mission execution. Timely sampling requires systematic data collection focused around the phenomena of interest, but many of these target phenomena - such as aerosol plumes - are difficult to sense and invisible to pilots. Given the large-scale nature and rapidly evolving dynamics of volcanic eruptions, in situ sensor measurements should be taken as near simultaneously as possible throughout the entire plume, with high-density uniform samples needed at key locations of concentration that are not known a priori and will vary as the plume evolves. Sampling capability should also be persistent, allowing complete measurements to be made at regular and frequent time intervals throughout the life cycle of the volcanic process. Preprogrammed sampling flight plans would be necessarily conservative, and would not meet the desired time frequency and spatial sampling density even with a large number of aircraft. Scaling the mission to multiple independently controlled aircraft would help increase coverage, but would require a large number of aircraft and flight teams, posing significant challenges to logistics,

mission execution, coordination, and separation assurance. Real-time adaptability in mission execution based on real-time payload sensor data is likely necessary to conduct this mission, directing the aircraft to focus flight trajectories around areas of important plume concentration. The extent of plume concentrations is unknown and difficult to estimate from individual sensor readings, requiring advanced processing to estimate the plume extent. The resulting behavior must satisfy the constraints imposed for scientific sampling, such as near orthogonal transects across the phenomena with uniformity in flight line spacing. Unfortunately, the environment around volcanic plumes contains inherent dangers to manned and unmanned aircraft alike, and the measurement targets (e.g., aerosol and ash plumes) are located near to or are collocated with these environmental hazards. For example, volcanic ash clouds pose a danger to aircraft air-breathing engines, and the areas in and around the plumes experience regions of adverse winds and extreme turbulence. Similar to the volcanic plumes of interest, the environmental hazards are also large-scale, time-varying, difficult to sense, and difficult to model and predict.

6.2.2 Related Work

Both manned aircraft and unmanned aerial system (UAS) platforms are increasingly utilized to conduct remote Earth observing science missions. While prohibitively dangerous for manned aircraft, volcanic plume sampling from small disposable low-cost UAS has been demonstrated (Corrales, et al. 2012) (Caltabiano, et al. 2005) (Saggiani, et al. 2007) (Pieri, et al. 2013), where disposable aircraft are preprogrammed to fly directly through potential hazards to collect data with low-fidelity sensors. These examples show tremendous potential for autonomous in situ measurement capabilities. Unfortunately, these approaches are still limited to small, low-cost, low-accuracy, disposable sensors on small disposable aircraft, where aircraft are preprogrammed to fly blindly through hazardous areas that place the vehicle and sensor payload at risk, and where useful data may not be collected. Many challenges remain toward enabling autonomous airborne in situ sampling mission concepts that require higher-fidelity instruments on more-capable UAS platforms to be reliably, safely, repeatedly, and systematically operated. Limitations in current UAS autonomy restricts these missions to pre-programmed surveys in safe locations well-clear of hazardous phenomena, where the survey area is relatively fixed and well-known, and where there is little urgency for temporal/spatial coverage. UAS missions are currently defined by flight plans composed of preprogrammed sequences of georeferenced waypoints that the aircraft is programmed to blindly fly through. Mission plans are developed by ground operators and uploaded to the vehicle's Flight Management System (FMS), which manages execution and provides appropriate input into the vehicle's automatic flight control (autopilot) system. Some level of

adaptability can be achieved by having scientists in the loop. For instance, sensor data can be streamed in real-time from the aircraft to ground science observers who can monitor the data and request modified flight plans to the flight manager; in turn, the changes are preprogrammed by the ground station operators, validated on the ground, sent to the aircraft, revalidated on the aircraft, then commanded to execute by the ground operator. Unfortunately, this is a relatively slow process that is difficult to achieve in practice, for instance requiring significant human involvement, reliable communication, imposing additional logistics and timing challenges, and is not easily scalable (e.g., frequent long-term flight operations or multiple simultaneous flights).

Given these challenges, the areas around volcanic activity where plumes are located are unsafe and inaccessible to both manned and unmanned aircraft alike when value is assigned to these platforms. UAS platforms have a few advantages over manned aircraft; for instance, unmanned platforms can generally accept more risk than manned platforms, and electric-powered propulsion options for UAS will alleviate the risk associated with ash ingestion. However, UAS flight control systems are currently designed to fly preprogrammed patterns and are less likely to achieve mission success than a human pilot in this scenario, who can assess and adapt in real-time and can respond more appropriately to off-nominal situations and flight conditions. Unfortunately, assuming platform expendability drives mission concepts to small low-cost vehicles that have severely limited range, endurance, and reliability. Payload capabilities are then highly constrained in terms of size, weight, and power, limiting sensor selection to disposable low-quality sensors and limiting potential scientific significance.

Many concepts in the research literature have been proposed to provide autonomous mission plan adaptability to environmental conditions based on payload sensor data. This Payload Directed Flight (PDF) capability would enable many future applications for airborne science (Ippolito, Fladeland and Yeh 2009). Several approaches for processing sensor data through statistical or mathematical models have been proposed (Pisano, Lawrence and Mohseni 2006) (Bachmayer and Leonard 2002) (Peng, Doug and Mohseni 2014). These approaches make little attempt to justify the underlying mathematical model form that are used to estimate the phenomena from sensor samples, and closing-the-loop around these models will provide no assurance of correctness in behavior, especially given the inherent extrapolation which must occur to plan behaviors beyond the current sensor limits. However, the complexity of phenomena models and uncertainty of predictions in this domain likely make attempts to incorporate precise models into the control system infeasible. For instance, in similar science missions, raw sensor data collected by all aircraft and platforms during a mission must be analyzed post-mission and reprocessed based on emerging underlying

hypothesis and assumptions. These underlying assumptions may not be completely known until the entire data set is collected and analyzed in the context of all the information available, and changes in these assumptions can greatly alter the sensor processing models and resulting data products. Since aircraft must make automated decisions in real-time, the aircraft will likely not be able to rely on the accurate or reliability of onboard processing models. The approach presented in this chapter incorporates first-principle models derived in coordination with the science domain experts specifically for autonomy, allowing the vehicle to process sensor information and make conservatively accurate predictions in real-time. We accept that onboard sensor processing pipelines and real-time model predictions generated by the autonomous control system will likely have no scientific significance beyond adapting real-time flight behaviors, and derive autonomy sensing models accordingly. For instance, it would be better to use a generally conservative estimation model that assumes more areas of sampling interest exist, rather than using a more detailed prediction model that directs the aircraft away from an area of interest and results in sensor coverage gaps. The resulting models have limited scientific utility beyond autonomy.

There are several approaches in literature for onboard autonomous decision making that use simple behaviors to direct aircraft behavior, such as gradient-following, optimal-peak-finding (Bachmayer and Leonard 2002) (Kovacina, et al. 2002) (Vincent and Rubin 2004). These approaches are also difficult to extrapolate to this application as the resulting behaviors do not meet requirements or constraints derived from the scientific data collection objectives. Earth science sensor sampling trajectories typically needs to provide regular, uniform, high density flight lines, providing sufficient coverage to ensure that the phenomena has been sufficiently mapped, and appropriate for later offline processing and investigation by the science team. Simple random, optimizing, or gradient-following behaviors do not meet these requirements.

Many approaches in the literature provide methods for autonomously controlling behavior of groups of vehicles, including swarm-based autonomy, for instance mapping chemical clouds (Kovacina, et al. 2002) and multi-agent search (Vincent and Rubin 2004). Here, we generally characterize swarms as scenarios where multiple, cooperative, self-organizing agents autonomously react to each other and to the environment based on simple interaction rules, designed such that collective emergent behavior results in global system behavior that robustly achieves specified global system objectives. Swarming concepts have a number of limitations. For instance, any concept requiring multiplicity of vehicles is inherently challenging, adding complexity and increasing overall risk. Most of the challenges faced in a single UAS science mission concept will be multiplied in a swarm mission concept. Swarm

missions based on current technology introduce many unique challenges; for instance, logistics is difficult, costs are multiplied, multiple autonomous systems have shown to be difficult to monitor and control, and project resources will be dispersed and diluted across multiple instruments and platforms. Swarms themselves are a form of decentralized control strategy which are necessarily suboptimal, especially considering the utilization of simple interaction control rather than rigorously synthesized decentralized control laws. However, the tradeoff in optimality versus robustness is well suited for these mission concepts where vehicles must operate under great uncertainty, complexity, and risk. Further, given the model complexity and uncertainty for underlying phenomena models, deriving decentralization strategies that optimize over these models may not be as appropriate as designing simple interaction rules to guide swarming behavior.

In this section, we present an approach toward a self-assembling, self-organizing, self-separating swarm of vehicles that will autonomously – without user input – optimally structure itself in such a way as to detect, track, and characterize volcanic plumes. The vehicles will automatically generate in situ sampling flight lines, build maps of the environment, and re-plan to optimize sampling flight lines based on the latest plume estimates based on payload sensors. Flight lines will be planned that avoid predetermined hazardous areas of high winds and turbulence, and flight trajectories optimize over the predicted wind field to take advantage of winds aloft for greater safety and fuel efficiency. Our approach extends the Payload Directed Flight architecture developed at NASA Ames Research Center to a multi-vehicle swarm configuration. Mission designers need only provide the vehicles with the mission parameters that consist of (1) a set of mission objective functions, (2) mission constraints, and (3) mission data that the objective functions or constraints need to reference. These mission parameters codify mission objectives and science goal for the complete mission. Based on these mission parameters, each agent will automatically plan and execute the mission without the need user input. Behavior of individual aircraft is determined through automated onboard trajectory generation that optimizes the aggregate mission objective functions and constraints. Sensor data is processed by a probabilistic estimation model which provides each aircraft with an estimate of the global plume state. A CFD model is developed for the environment and used to evaluate the expected wind, hazards, and plume states given a specified condition. In this initial implementation, limited set of conditions were analyzed in the CFD study, and results were limited to steady-state condition based on constant emission rate of SO_2 and steam into the environment. The CFD model provided a priori expectations of plume location for the estimation model. The CFD results were also processed into a wind field database onboard each aircraft which was evaluated as part of the cost function for each trajectory.

The implementation is an extension of previous applications in PDF, adding CFD model integration into prior distribution of the phenomena estimation, adaptation of the mission objective function, and extension to a multi-agent swarm architecture utilizing simple control laws to control behavioral interaction between agents. The concepts for swarm autonomy are implemented on an appropriate demonstration platform to show feasibility of this approach.

6.2.3 Mission Design and Architecture

The mission concept is shown in Figure 42. As each vehicle is launched, the automated planner will determine the best flight lines to reach the sampling area. The a priori plume state, determined by offline CFD studies, is used to guide the aircraft to the expected area of the plume, while the mission cost function captures the sampling behavior desired. New aircraft that are introduced into the mission transmit their positions to surrounding vehicles. These positions are used to separate the aircraft through virtual repulsion forces, defined as part the aggregate mission cost function. For this initial implementation, sampling planes are predefined, and repulsion forces are defined strong enough to move vehicle flight trajectories to surrounding sampling planes, or to further regions of the sampling region on occupied planes. This approach will scale well until the number of aircraft becomes larger than the number of sampling planes, which is a constraint for this initial implementation.

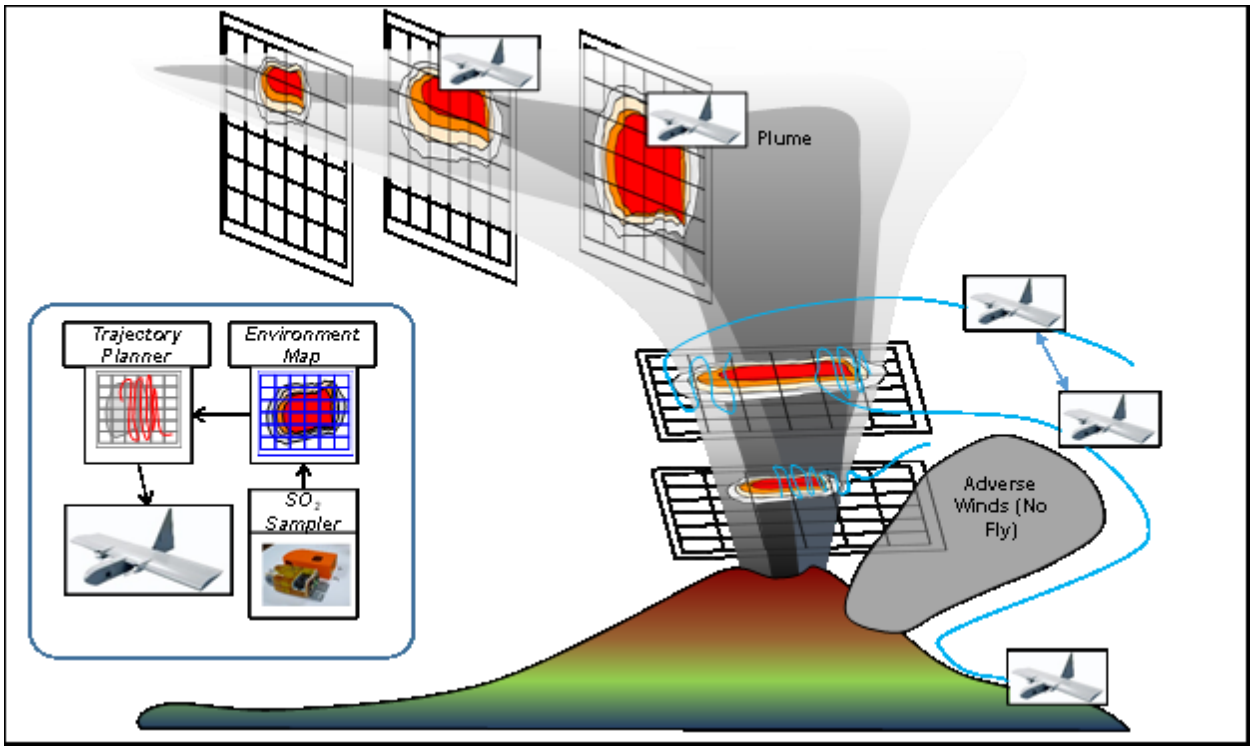


Figure 42. Mission Concept

The mission concept data processing model for swarming volcanic plume monitoring is shown in Figure 43.

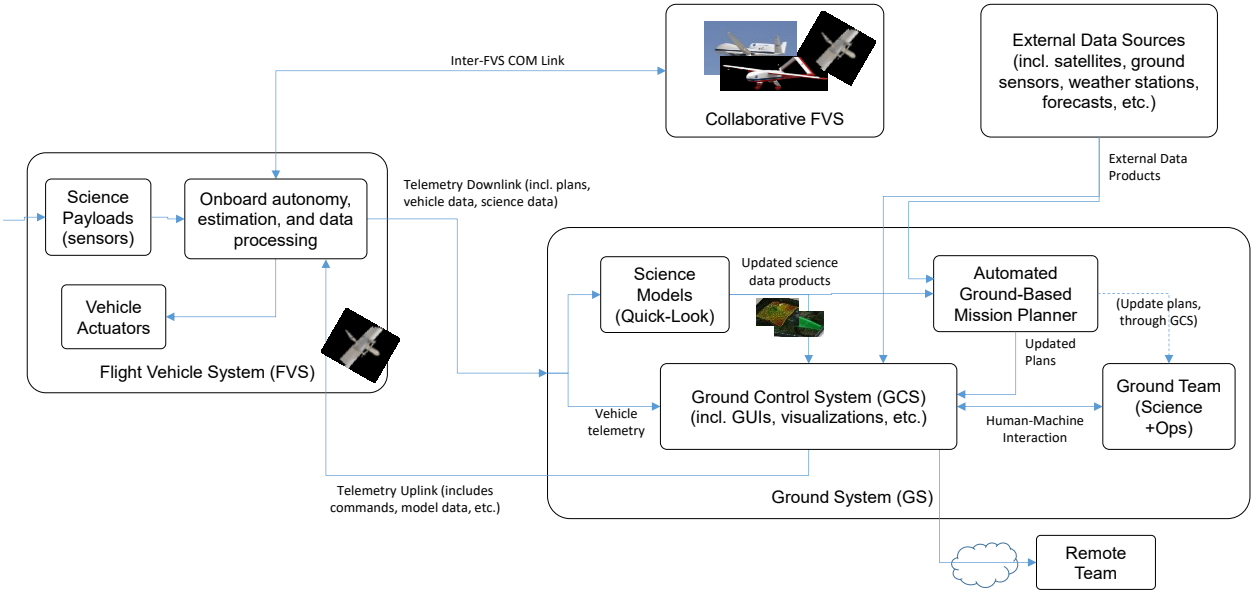


Figure 43. Data Processing Architecture

A general data flow architecture for coordinating multiple Dragon Eye vehicle systems is shown in Figure 44. An adaptive mesh radio modem allows vehicles to address other vehicle directly with point-to-point communication, allowing low-latency data sharing between vehicles during the flight mission over a remote location that does not rely on a reliable data link to a ground control station, as is expected in the given flight condition. Further, with scaling to multiple vehicles operating at large distances to the ground station, communication from each vehicle to the ground becomes an issue that an airborne mesh network may address. In previous flights over Turrialba conducted with the Dragon Eye, the vehicles experience degraded communication link with point-to-point RF modems due to environmental conditions over the Turrialba that included high humidity, cloud and fog cover. The flight operations model in Figure 44 show the major ground components and staffing/personnel requirements.

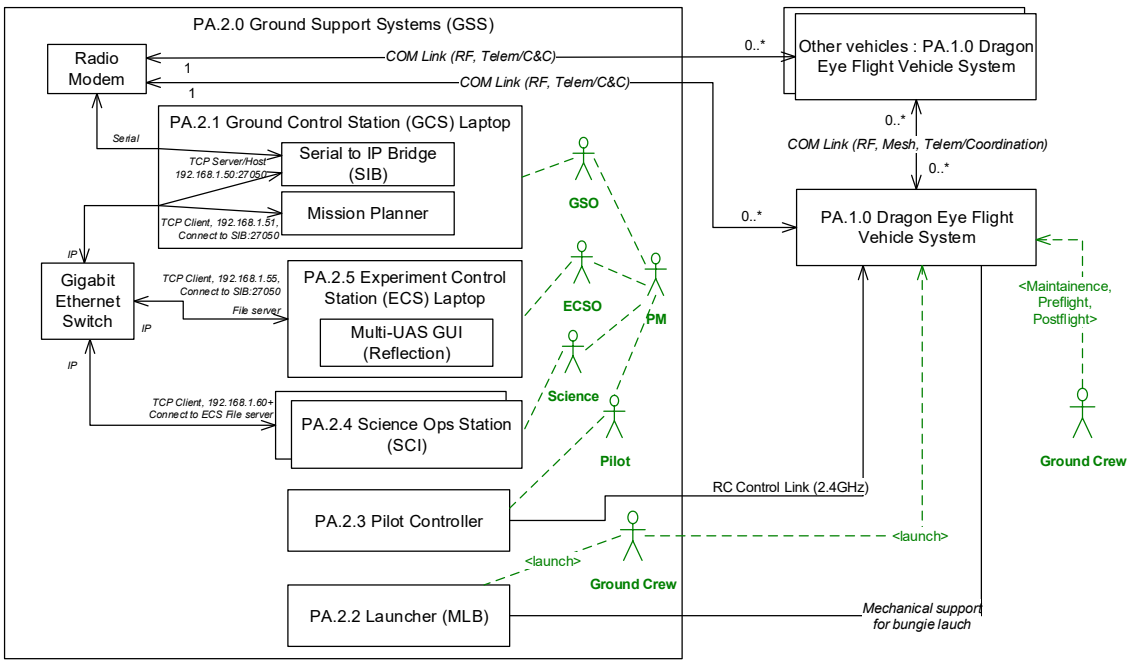


Figure 44. Flight Operations Model

6.2.4 Demonstration Flight Vehicle Design

An appropriate demonstration platform was selected based on a number of criteria. The vehicle had to be able to survive the volcanic plume environment while demonstrating low-TRL research autonomy. We desired rugged, compact, dynamically stable, and must use an electric propulsion system. The vehicle had to be capable of peer-to-peer communication, which was designed and implemented as part of this project's development. The vehicle platform had to be capable of fielding a representative sensor, which was selected and built as described below. The vehicle had to be capable of hosting the autonomy algorithms and be representative of larger platforms, which led to the team designing custom avionics based on an open-source hobby-grade autopilot system with a secondary processor. The autonomy demonstrator vehicle had to be disposable, replicable, fieldable within the constraints of the project budget, already operational, and should have low design/development costs to field the demonstration.

The Dragon Eye UAS platform, developed by Aeroenvironment, Inc., was selected as the demonstration platform for this mission as shown in Figure 45 below. A large number of Dragon Eye UAS had been acquired by NASA Ames Research Center as military surplus items, and they are small, compact, rugged, all-electric, and feature a sufficient nose section for hosting payloads. The flight time and range was sufficient to demonstrate at the selected test site. Unfortunately the vehicles featured closed military-grade export-controlled avionics, limited single-string/limited redundancy, limited end-user extensibility, and no peer-to-peer communication functionality.

Figure 45. NASA Ames Dragon Eye UAS and Payload System.

NASA Ames Dragon Eye UAS deployed at the Turrialba Volcano in Costa Rica in 2013 (left), and the SO2 sensor payload developed at NASA Wallops Flight Facility (right).

To meet the demonstrator requirements, Dragon Eye vehicle avionics were redesigned. The closed export-controlled hardware was removed. The replacement control system utilized the open-source Ardupilot autopilot system with custom software and integrated with a high-power mesh communication radio modem operating in the ISM 900Mhz band at 1 watt transmission power. The Adapteva Parallella-16 computer was installed as a secondary processor for autonomy and higher-level processing. The swarming vehicle avionics hardware architecture (Rev C) is shown in Figure 46.

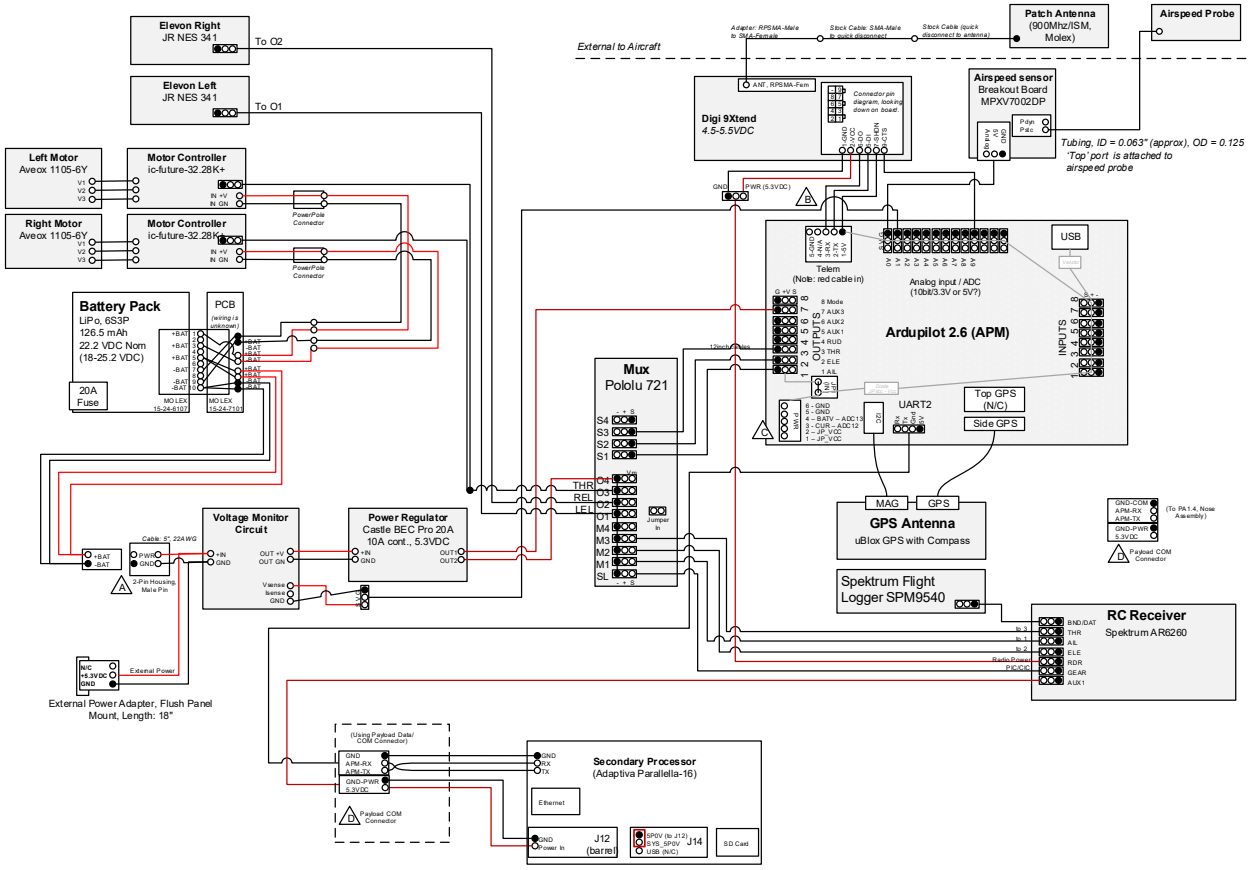


Figure 46. Electrical Schematic of the Dragon Eye Vehicle, Rev C.

The sensor payloads were selected, designed and built at NASA Wallops Flight Facility to fit into the nose of the Dragon Eye platform. The payload system consists of a Pace XR440-M data logging system with a CityTech SO2 sensor in addition to temperature, humidity, and pressure sensors. The payload system schematics are shown in Figure 47.

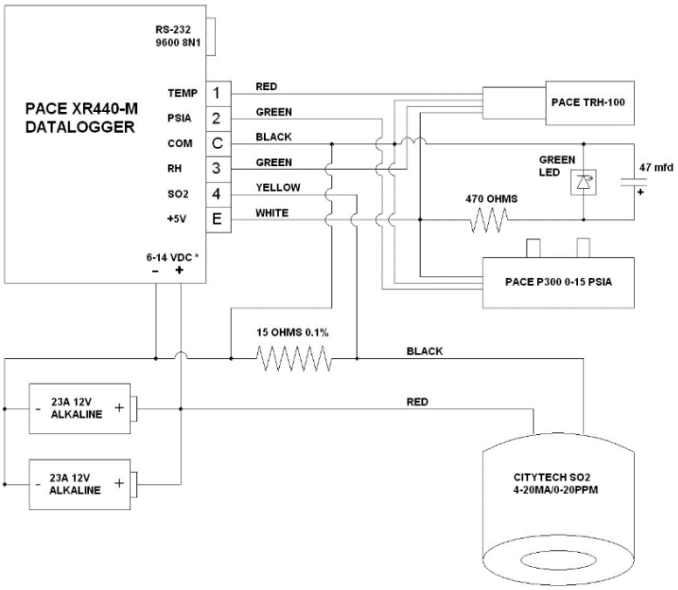


Figure 47. SO2 Sensing Payload System

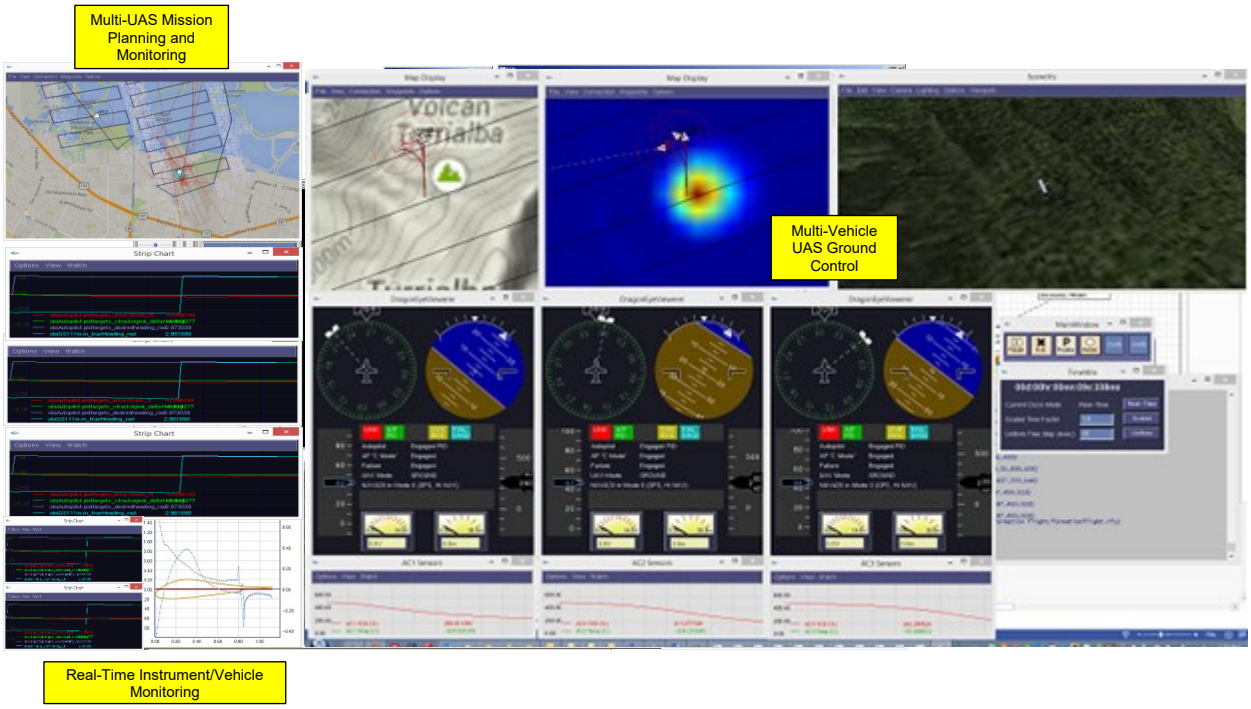


Figure 48. Multi-Vehicle Simulation Environment

A multi vehicle simulation environment was created for the evaluation of the system, as shown in Figure 48. The simulations assumed three Dragon Eye vehicles operating over Turrialba. A ground control station was developed that allows a single operator to monitor the resulting trajectories of each vehicle, with the ability to manually override

the onboard autonomy with manual commands. The CFD results for a given condition were compiled into a wind field database and added to vehicle simulation dynamics.

6.2.5 Computational Fluid Dynamics Model

Model based estimation of the wind fields and expected dispersion pattern of the emitted aerosol plume was developed through CFD evaluation. A computational fluid dynamics (CFD) model of the Turrialba volcano was develop as shown in Figure 49 below, which was generated from a digital elevation map with elevations at 3 meter spacing. The resulting wind vector field is shown in Figure 49 (b) which showing the complex wind patterns which matched expectations for this area. Two species, SO2 and CO2, were modeled as emitted from a point source from the approximate caldera location, and the predicted plumes are shown in Figure 49 (c). The model results were processed and used the prior expectation in the belief map.

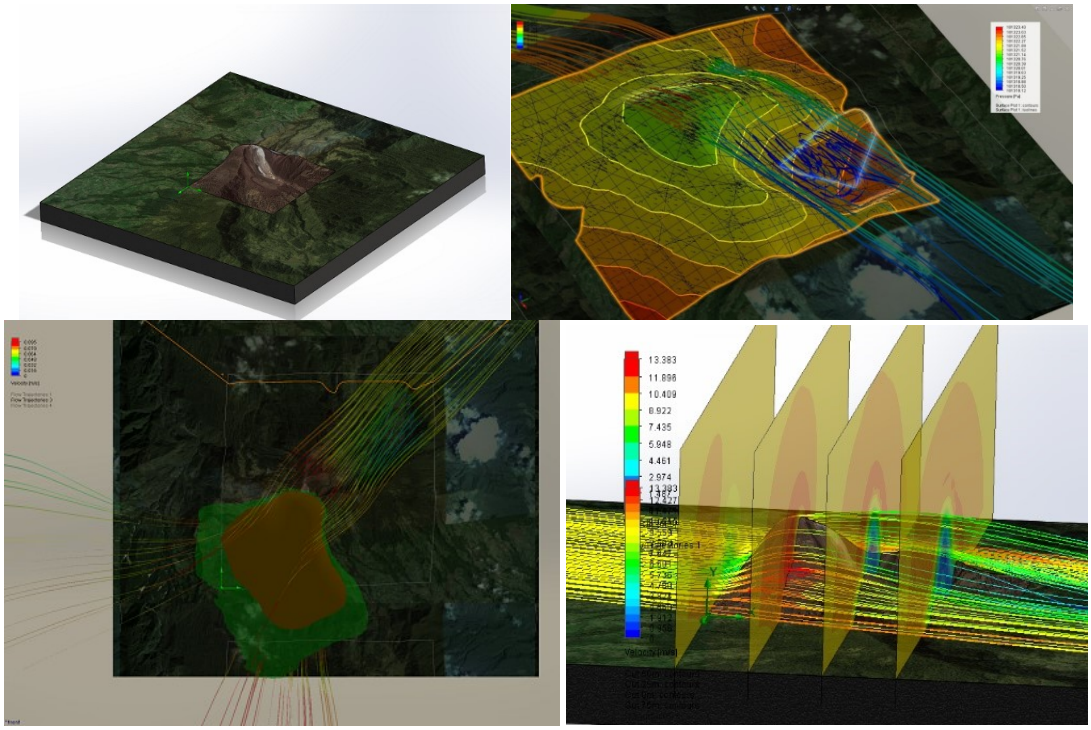


Figure 49. CFD Analysis of the Turrialba Volcano.

(a) Turrialba Volcano in the CFD Environment (top-left). (b) CFD predictions for wind vector (top-right). (c) SO2 and CO2 plume concentration predictions (bottom-left). (d) Vertical cross section expectations (priors for belief map).

6.2.6 Autonomous Control System Architecture

The high level autonomy loop is shown in Figure 50. Data collected at the UAS is filtered locally to estimate the local plume concentration, then integrated into a probabalistic occupancy grid model of the plume, where a priori distributions were loaded from the CFD model estimates. This provides a belief map of the plume for each vehicle. Based on the latest belief map, the vehicle constructs a 2D cost map that is used for trajectory generation based on a random search tree optimization. Neighbor vehicle positions are transmitted between vehicles. At this time, a simple repulsion force law was implemented to push the vehicle to distant sections of the map and keep vehicles from coming within in the vicinity of each other. The repulsion force is added to the cost function in the trajectory planner. The resulting trajectory is used to generate a sequence of waypoints that are sent to the primary vehicle control system. Trajectories are generated at a rate of once every 10-30 seconds.

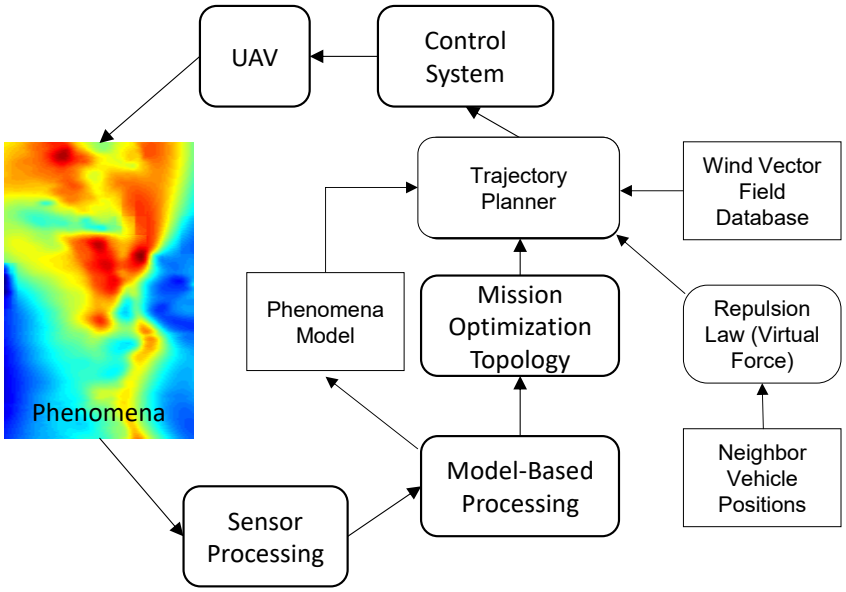


Figure 50. Optimization Loop

The vehicle autonomous control system architecture is built on the PDF architecture developed at NASA Ames Research Center (Ippolito and Yeh, 2009). The PDF architecture allowing vehicles to sense, predict, and control relative to complex phenomena through processing of payload sensor data injected at various feedback paths. The architecture is shown in Figure 52.

The PDF inner layer in Figure 52 represents the fastest response control law, and updates at a rate of 50 Hz. The inner layer must simultaneously control the aircraft to follow the trajectory commands provided by the middle layer while meeting the fast-time response tracking control requirements needed to track the phenomena. The nature of the

control system at this layer depends on the requirements for both of these objectives. General trajectory-based control is needed when aircraft must follow precision 6DOF trajectories; this is the case, for instance, trajectories are calculated to maximize the placement of a body-fixed imager viewing a target. Generally, a trajectory-based control scheme is required, such as the trajectory-linearized control outlined in (Adami, et al. 2009). For this application the control requirements are less stringent. The sensors in the current payload system (Figure 47) include an SO_2 gas sampler, temperature sensor, and pressure sensor, none of which place constraints on the vehicle attitude, relaxing the requirements to position 3DOF control. We further assume the phenomena dynamics are slowly evolving relative to the vehicle dynamics, we assume inter-vehicle separation assurance can be assured at the trajectory planning (strategic) level, and we assume there are no sensor pointing requirements on the vehicle attitude dynamics. Therefore, real-time sensor control feedback to the inner-layer was not needed, and the inner layer control mode is a simple "track-to" waypoint-following feedback control mode, fed by waypoints computed by the middle layer trajectory. Waypoint management and mission execution is managed by the FMS. The lateral mode control system structure is shown in Figure 51 below.

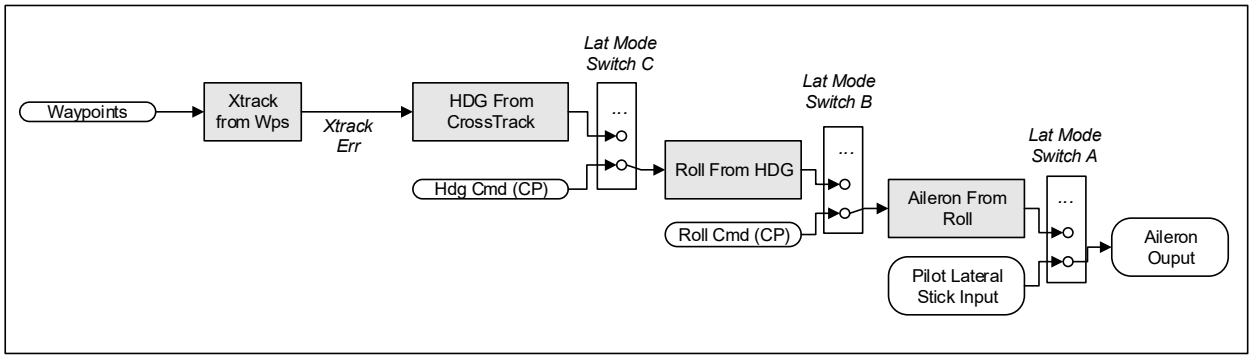


Figure 51. Inner Layer Lateral Mode Control System

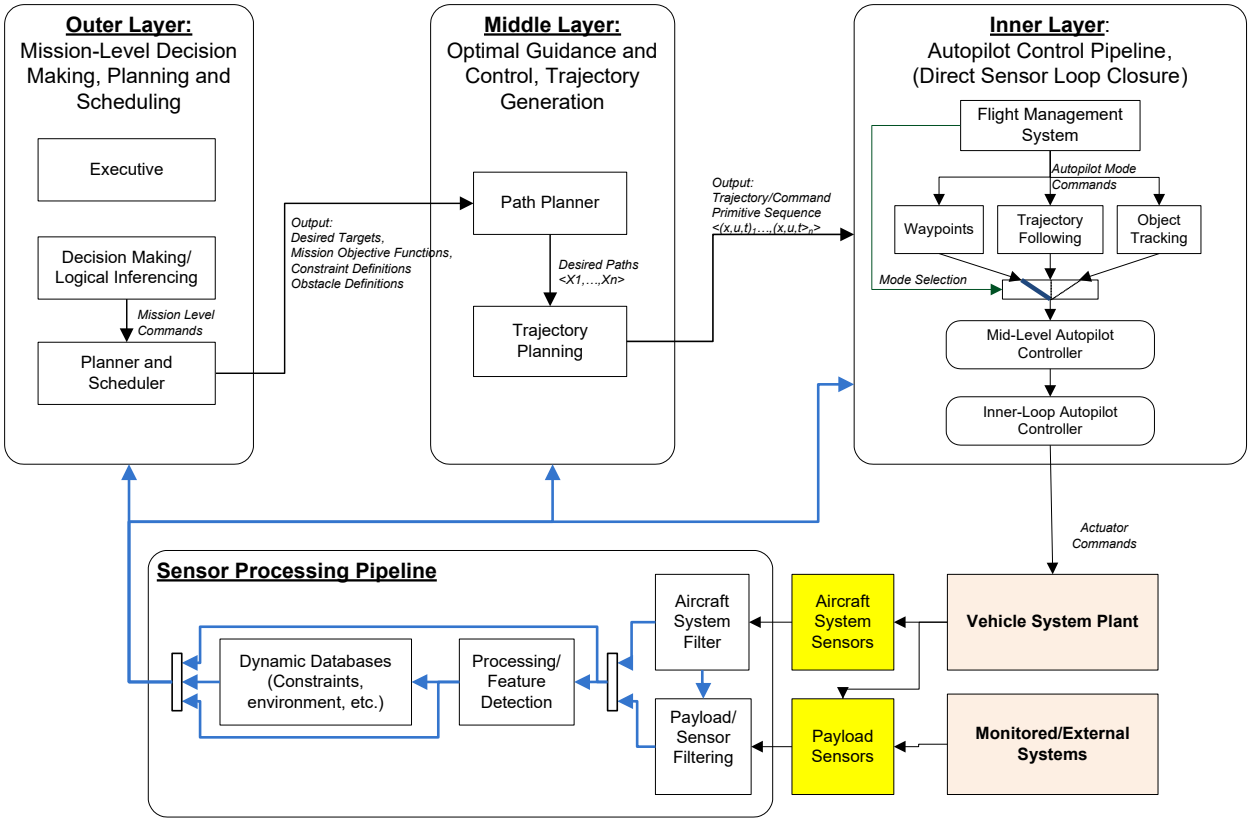


Figure 52. Payload Directed Flight Control System Architecture

The PDF architecture's middle layer (see Figure 52) is responsible for generating trajectories in the form of waypoints for the inner layer and updates at a period of 10-30 seconds. The optimization algorithm utilizes an optimization-based framework and numerical optimization algorithm first presented in (Ippolito and Yeh, 2009) and conceptually shown in Figure 53. The middle layer optimization engine (Ippolito and Yeh, 2009) (Ippolito, Khalid and Dolan, 2005). For this project, all control objectives for PDF control and inter-vehicle swarm coordination are implemented as additional constraints to this middle layer.

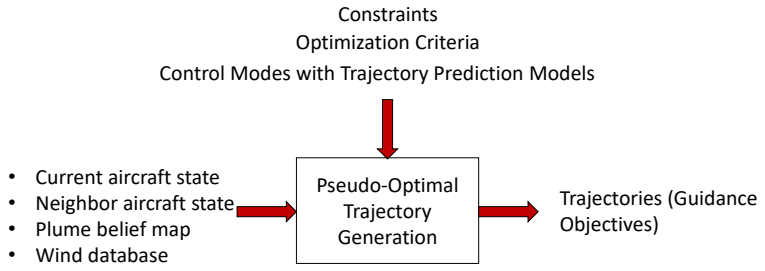


Figure 53. Middle Layer Optimization Engine

The original cost functions in (Ippolito and Yeh, 2009) were augmented with a mission objective cost function c(s) of the following form.

$$c(s) = \left(\frac{v_{max} - V(closest(s))}{v_{max} - v_{min}} - 1 \right) * [0.5 * cos(A * distance(s, closest(s)) + B) + 0.5) + 1) \quad (6.2\text{-}1)$$

Here, the function closest(s) returns the closest visited node in a 2D spatial partition grid. The function distance(a,b) returns the distance between two nodes. The cost function map from (1) returns the topology shown in Figure 54, which is added to the other cost functions. The constant parameters A and B can be adjusted to the desired grid spacing, as specified by the science mission requirements.

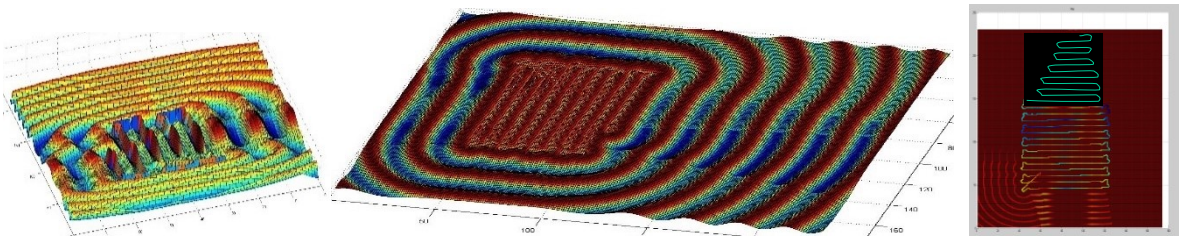


Figure 54. Mission Cost Function Topologies.
(a) Initial topology, (b) shown for case 3 in Figure 55, (c) showing optimization trajectory output.

The plume estimation model is based on the Lee model for vision-based plume tracking (Lee and Ippolito 2009). This algorithm was implemented as described with a few modifications. For this demonstration, the plume model was limited to several 2D cross-sections at various distances above the volcano. The prior probabilities on the model is seeded with the results from CFD evaluation. The estimation model has a number of limitations, including a quasi-static environment assumption, stubbornness to modifications once a solution has locked in, and currently utilizes a single static CFD solution for this initial implementation. Extension for plume dynamics are additional CFD solutions are planned for future research.

6.2.7 Results

Preliminary simulation results are shown in Figure 55. As the mission progresses and SO2 intensity data is collected, an environment map is generated based on this previously collected data. This environment map is processed into a cost function which encodes mission level objectives to map and follow the extent of the plume with a minimum spacing dictated by the mission requirements (10 meters). The resulting trajectory optimization algorithm

generates a plan that guides each individual aircraft along the path that maximizes the mission objective to maps areas of highest SO2 concentration.

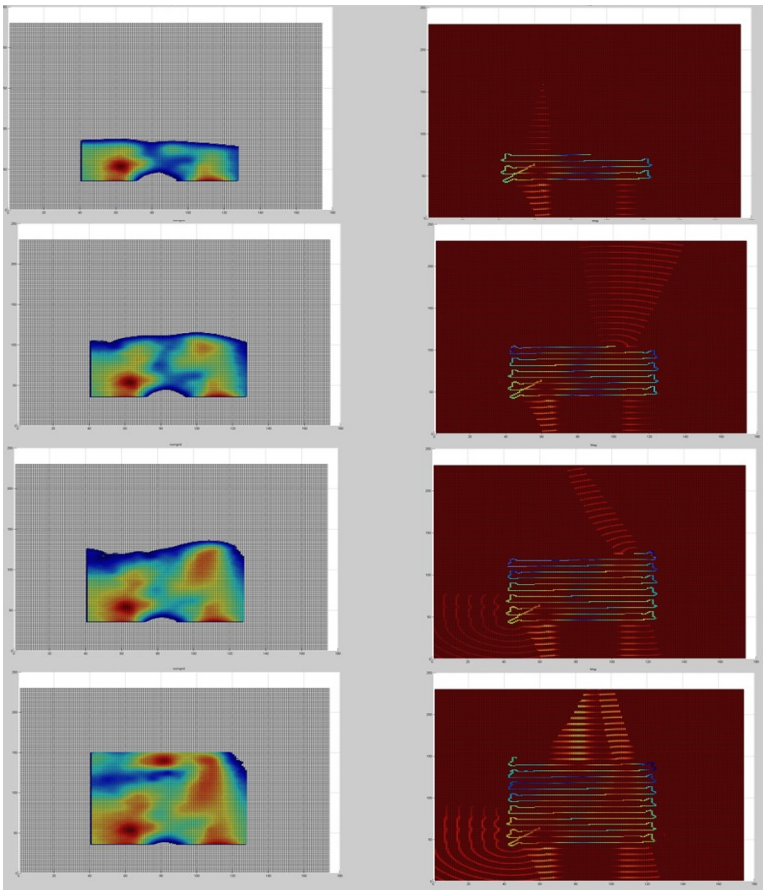


Figure 55. Evolution of the plume belief state and optimized trajectory.
Evolution of the plume belief state (left) and the resulting cost map with optimized trajectory (right) during simulation evaluation.

The results shown are still preliminary based on the initial simulation and flight system implementation. The project planned flight testing at the Turrialba volcano in 2015. This test site was selected based on a consistent low-level of activity and SO2 emission, making it an ideal candidate for this demonstration. These tests have been postponed due to renewed volcanic activity at Turrialba in early 2015, making the area surrounding the volcano inaccessible. The simulation results have been promising, and the project is continuing local flight testing with the SO_2 payload sensors and swarming Dragon Eye system. A number of challenges have emerged in the hardware system. Utilization of IEEE 802.15.4 mesh sensor network as an airborne peer-to-peer communication system has proven difficult, as the all communication is currently serviced through these modems, and the system in flight tests experienced bandwidth saturation, high packet loss and error rates, high latency, and inability for the mesh network

to adapt in flight. Additional research would be needed to address these issues, but this project is moving to a new revision (Rev D) that forgoes peer-to-peer mesh in factor of a static point-to-multipoint (ground-to-multi-aircraft) topology that provides more reliable communication. The system originally planned to broadcast information on the belief map between vehicles, but given limitations of Rev D, this was reduced to simple position updates. This information could be served with small low-cost ADS-B type receivers that are currently available for small UAS. Additionally, the secondary processor experienced overheating issues despite the addition of active cooling in the payload bay, reducing effective flight experiments of the autonomy system to around 20 minutes. Future revisions of the aircraft hardware will likely replace the processor with a lower-power processing board to address this issue.

6.3 Real-time Decentralized Reconfigurable Control for Mobile Robotics in Smart Environments

6.3.1 Introduction

The Exploration Aerial Vehicles (EAV) laboratory at NASA Ames Research Center operates a set of MAX 5A UGV platforms. The MAX 5A specifications are shown in Figure 56 (right). These ground vehicles were custom designed specifically for precision outdoor GPS-based navigation. The goal for these vehicles is to support science missions where multiple UGV's autonomously navigate and record data measurements over very long durations (several weeks or months in length) over large areas (such as the NASA Ames Research Center campus) without direct surveillance of a human operator, with similar mission design as (R. Lee, et al. 2010). To enable this goal, these autonomous UGV's must navigate autonomously through indoor environments and recharge.

Figure 56. Senseta Corporation Max 5A UGV, Specifications (right) and Simulation Model (left)

The Max 5A UGV's were custom designed for outdoor navigation and operation, with a MEMS-based inertial measurement unit (IMU) for orientation, angular velocity, and linear acceleration measurements, magnetometers for orientation measurements, and differential Global Positioning System (GPS) receivers for sub-meter positioning. Unfortunately, GPS signals within most buildings are too weak for effective localization, and the IMU dead reckoned solutions without absolute position corrections will not remain accurate long enough to allow the rover to navigate within the building. Further, additional the science payloads for these ground vehicles utilize the entire allowed payload in terms of size, weight, and power (R. Lee, et al. 2010). Outfitting these vehicles with additional indoor navigation equipment would require an extensive redesign and development effort beyond the scope of our research.

Indoor navigation for insufficiently instruments vehicles is a challenging problem. The hypothesis for this investigation is that PCS provides a compelling alternative solution to the indoor navigation problem, utilizing existing sensing equipment already embedded in the NASA campus environment to localize the vehicle. Through a PCS infrastructure, the vehicles can restructure control systems around environment sensors in real-time as needed, allowing indoor navigation without the need for hardware modification.

The goals of the experiment are as follows.

1. Develop the control laws and high level control algorithms that implement a robust polymorphic solution to the problem of indoor navigation in a realistic scenario involving multiple competing constraints with large structural uncertainty.
2. Develop a moderately high fidelity simulation infrastructure to test UGV operation in a realistic indoor/outdoor environment in order to evaluate the PCS alternative.
3. Demonstrate the feasibility of utilizing polymorphism to solve this problem.
4. Develop a solution to the two-part optimization problem that meets the requirements of this problem.

6.3.2 Experiment Configuration

In order to investigate the feasibility of using polymorphic control to enable indoor navigation for the MAX 5A UGV in a smart space environment, a concrete simulation scenario was developed around the conceptual mission's indoor segment. This scenario was then implemented in a moderate fidelity simulation, which was needed to explore systems level issues with this proposed approach. In this scenario, the UGV is performing an autonomous data gathering experiment outdoors in the NASA ARC / CMU-SV campus. The onboard power levels have just passed a

low value threshold that triggers the control system to switch into a low power recharging mode. In this mode, the UGV is required to perform the following activities:

1. Terminate the current experiment. This involves closing data files, archiving data, documenting the end of the experiment.
2. Switch to low-power operations mode. Power down non-critical systems, such as the science payloads and other systems that aren't required for the remaining tasks.
3. Perform a search for nearby accessible charging locations.
4. Identify the most appropriate charging location, and construct a plan to navigate to this location. The plan should identify specific performance objectives to be considered, which may include maximizing reserve power levels, maximizing safety, maximizing success probability, and minimizing the time to accomplish the task.
5. Autonomously navigate to the selected location satisfying the required performance objectives.

The indoor mission segment requires timely completion and deterministic power consumption. The vehicle is required to detect and avoid unexpected obstacles in the occupied buildings, and localize and navigate in dynamic indoor environments. Completion success rate is a key performance objective for the indoor navigation task.

The simulation is initialized with the rover at a random outdoor position. A fictitious library building was added to the virtual campus for this simulation, the layout is shown in Figure 57. This building is PCS enabled and outfitted with four security cameras mounted on pan and tilt actuators throughout the building. Each camera is under control of its own autonomous security system agent. These security agents are responsible for surveillance of the library, and command the camera actuators to provide acceptable coverage. The simulation environment provides monitoring of the environment, including the camera systems and UGV, as shown in Figure 58. (Figure 57, left)

Figure 57. Virtual Smart-Building in Campus.
Smart building location (left), UGV navigating towards building (middle), building layout (right)

Figure 58. Virtual Smart-Building Environment.
UGV and security camera views are shown along the bottom of the screen. The UGV's plan and current position is shown in the map view on the left.

The Reflection Architecture (Ippolito, Pisanich and Al-Ali, 2005) is a component-based plug and play environment for simulation and control of embedded vehicle systems. Simulations and embedded systems can be created by loading instances of modules into the architecture at run-time. When loaded, modules register interfaces through a dynamic run-time type identification and binding system, and establishing data routes between interfaces of the modules. The Reflection architecture was previously extended to support the requirements for a Polymorphic Control System transport layer, capable of functionality such as transferring compiled executable libraries over

wireless links during run-time, run-time rerouting of information and data flow, run-time construction and deletion of modules, run-time discovery and reflection of data variables, and communication synchronization of multiple distributed Reflection kernels through a transparent reflective communication layer. The onboard MAX 5A UGV system, ground control station, and simulations are implemented in this architecture.

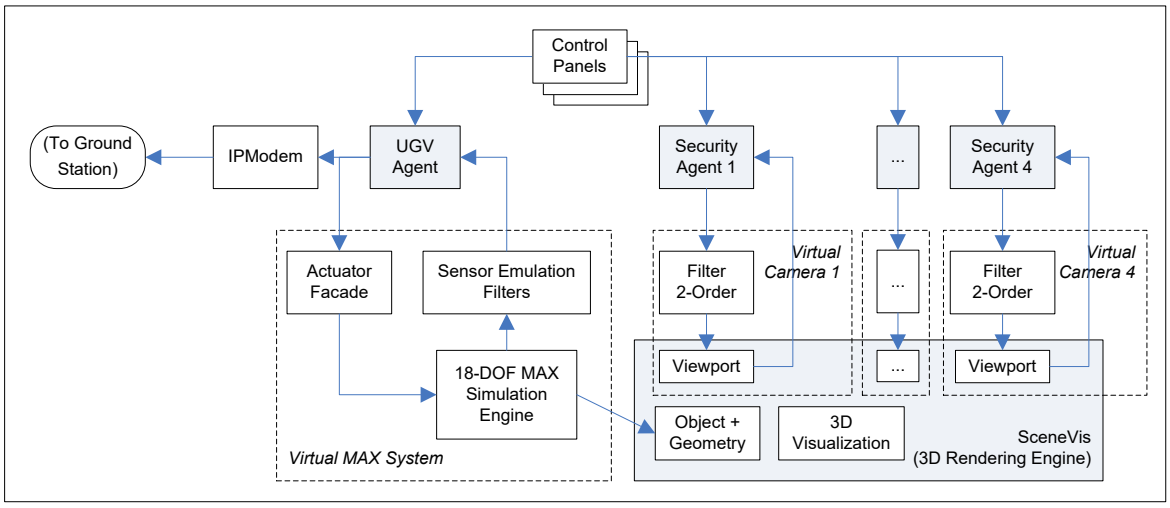


Figure 59. Reflection Architecture Simulation Configuration.
Not all components are shown. Composite components are shaded. Edges represent static data routes. Dashed boxes represent emulation of real-world hardware.

A simplified representation of the Reflection configuration for this simulation is shown in Figure 59. Boxes represent Reflection component instances. Shaded components represent multiple objects for the purpose of this drawing. For instance, the UGV Agent is composed of multiple components in Reflection as described below. The MAX 5A simulation model was created previously, implementing an 18 degree of freedom mathematical model that can simulate various nonlinear friction models over complex virtual terrains. The new modules created for this simulation are the various agents.

6.3.3 Polymorphic Reconfiguration Objective

The solution approach for this problem is shown conceptually in Figure 60. In this scenario, the rover must utilizing onboard actuation to traverse the indoor environment through areas that allow the rover to be observed by the smart security system cameras. The video images from the smart system cameras can be used to identify, track, and then localize the UGV. The data must be received by the UGV's control system in real-time, with sufficient update frequency and latency to allow for real-time autonomous navigation through its feedback control system. The

camera systems are controlled individually by Security Agents, components of vision input, processing and control components. The UGV Agent is responsible for controlling the MAX 5A UGV system, and is also compose of sensors, processing, and actuation.

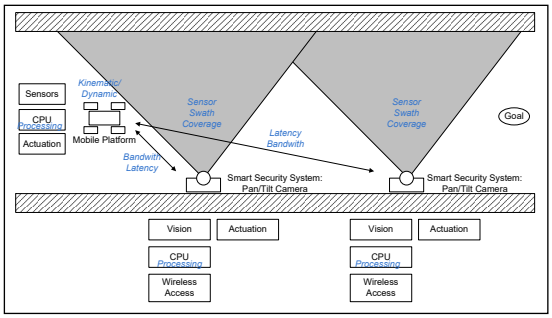


Figure 60. Problem Configuration: UGV Traversing the Smart Building Environment

The mathematic description of PCS is based on graph theoretic formulations that model the information structure of a control system. The complete definitions can be found in (Ippolito and Al-Ali, 2007). The controller graph is defined as G=(V,E,C), where a vertex $v \in V$ represents data variables in the system, the set of edges are defined as an ordered pair of vertices in V, $E \subseteq \{ < u, v > \mid u, v \in V \ \wedge u \neq v \}$, and C is a set of components of G. To simplify notation, let V(G), E(G), and C(G) represent the set of vertices, edges, and components in a graph G, respectively. A graph G'=(V',E',C') is a subgraph of graph G=(V,E,C) if V'$\subseteq$V, E'$\subseteq$E, and C'$\subseteq$C. A component $K \in C(G)$ is a subgraph of G with the properties that $K \neq$G, $V(K) \neq \emptyset$, and $e = \langle V(K), V(K) \rangle \; \forall \; e \in E(K)$ (i.e., every edge in K connects two vertices that are also in K).

The initial control configurations for the UGV and security camera systems are shown in Figure 61. Under these information structure topologies, control is decentralized and no communication occurs between systems. Each system is an autonomous agent that acts according to their objective functions. The rover's objective function is based on optimizing the path to the power port with sufficient battery power. Battery power draw is a function of processing load: each block that is hosted on board the UGV will increase the power draw of the system. The UGV's motor system also draws power from the main battery; a trajectory that minimizes the power draw will be similar to the trajectory that minimizes distance. The power constraint is added to the final state term in the cost function (L), and it must be above a pre-specified threshold.

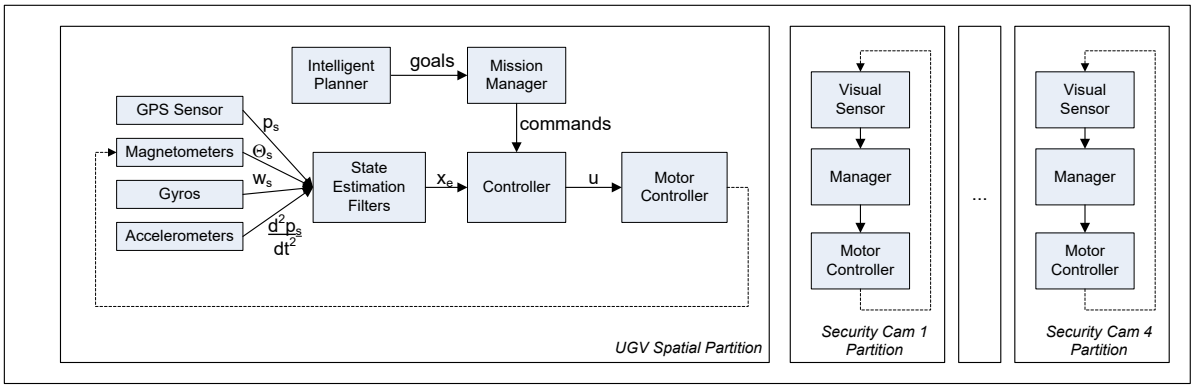


Figure 61. Initial Disjoint Control Structure for the UGV and Security Camera Systems

Initial configuration for UGV and security cameras systems feature completely disjoint control structures.

In order for the UGV to traverse the environment shown in Figure 60, the UGV Agent must be provided a set of candidate topologies that achieve the navigation goal by utilizing embedded resources in the smart environment. In addition to the initial control topology in Figure 61, four additional candidate control configurations were designed. Each candidate configuration identically closes the loop between the UGV and one of the security camera systems, as shown in Figure 62.

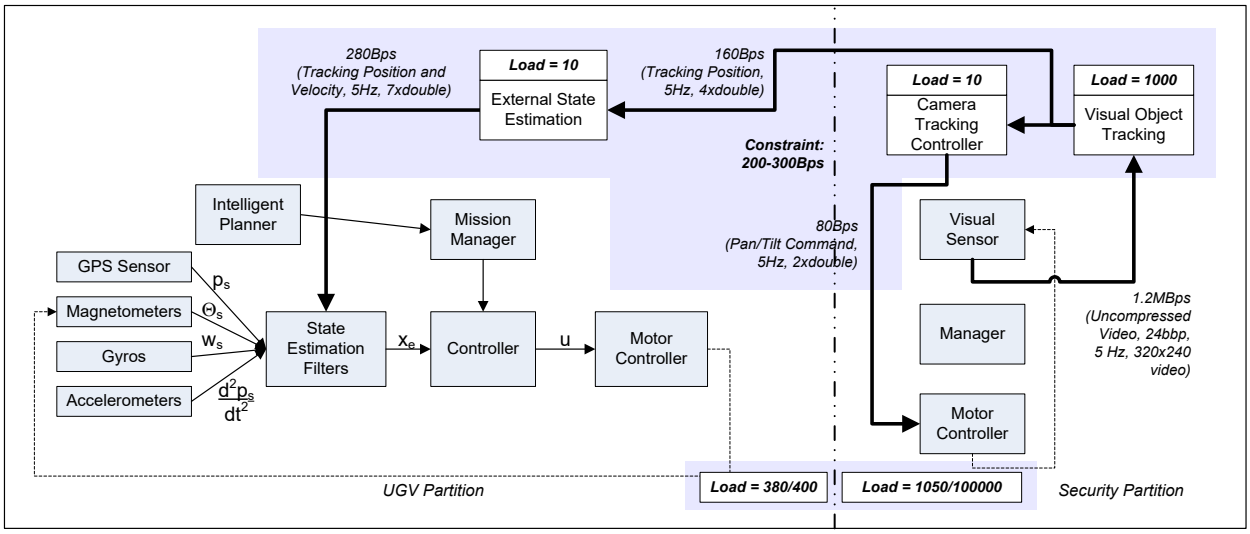


Figure 62. Candidate Reconfigured Control Structure (one of four possibilities)

Each candidate topology – except for the initial decentralized topology – results in a control structure that crosses over a single boundary between the UGV partition and a security camera partition. The PCS engine is required to determine a partition of the control structure graph that will assign a particular component to a particular processor. The partition for the reconfigured control structure is shown in Figure 62, where each component is assigned a

processing load estimate, and each edge is assigned a bandwidth estimate. The processing load estimates for each component are simple estimates, while the bandwith estimates on each edge were calculated based on the amount of data passed from one module to the other, in terms of bytes per second. For instance, the vision system produces an uncompressed 320x240 image at 24 bits per pixel, and will provide images at 5Hz, which results in around 1.2 MB of data per second. Note that this number does not include the overhead bandwidth. The load constraint on the UGV partition, set arbitrarily to 400 units, is much more restrictive than the load balance of the security system, set to an arbitrary large number. The constraint for communication between the UGV and security system was set to either 200Bps or 300Bps.

The PCS system is responsible for assigning components to each partition that balance the load on each processor along with the amount of communication required between partitions. At the minimum, the PCS system needs to establish feasible a feasible partition that meets the resource constraints. In general, load balancing is an NP hard problem, and many different approaches have been discussed in the literature (Hendrickson and Devine 2000). For this problem, the load balancing algorithm must solve the following problem: Find the partition which maximizes the estimated load margin on all partitions while maximizing the bandwidth margins across all partitions. The exact equation for calculating the graph partitioning cost function to be minimized is shown in equation (6.3-2). Here, $/p = [p_{ij}]$ is the partition matrix, $a = [a_{ik}]$ is the partition adjacency matrix, L_j is the processing cost estimate for component j, and B_{ik} is the sum of the edge bandwidth costs for all edges that cross from partition i to partition k. The values L_{m_i} and B_{m_i} represent the maximum allowable processing load and total outgoing bandwidth from component i, which represent constraints on the solution. The value α is a normalization factor, where $\alpha = 10$ was used in this simulation.

$$J(a,p) = \sum_i \left(\left(L_{m_i} - \sum_j p_{ij} L_j \right)^2 + \alpha \left(B_{m_i} - \sum_k a_{ik} B_{ik} \right)^2 \right) \qquad (6.3\text{-}2)$$

The partition matrix p is a binary matrix of size |**P**| by |**C**|, where |**P**| is number of partitions in the system, and |**C**| is the number of components. The elements of p are defined by $p_{ij} \equiv 1$ if component j is assigned to partition i, and $p_{ij} \equiv 0$ otherwise. Similarly, the partition adjacency matrix a is a |**P**| by |**P**| binary matrix defined by $a_{ik} \equiv 1$ if partitions j and k are adjacent, otherwise $a_{ik} \equiv 0$. See (Ippolito and Al-Ali, 2007) for more details.

The current algorithm performs a simple brute force search across all possible partitions to minimize, which is possible for this configuration given the small number of components. The "External State Estimation" component in Figure 62 will be assigned to different partitions depending on the bandwidth constraint selected. However, since the simulation is performed on a single machine, no performance difference was observed.

6.3.4 Control System Architecture of the UGV Agent

The UGV Agent is responsible for controlling the virtual UGV vehicle system, controlling interaction with the environment, and controlling interaction with other agents. The control architecture for the UGV Agent is a hierarchal control structure, shown in the graph in Figure 63. Higher-level (superior) components have responsibility over their direct subordinate modules, as indicated by solid arrows in the drawing. These responsibilities include assigning tasks, goals, control commands, and information requests to their subordinate systems. The dashed edges represent static data flow routes from one component to another

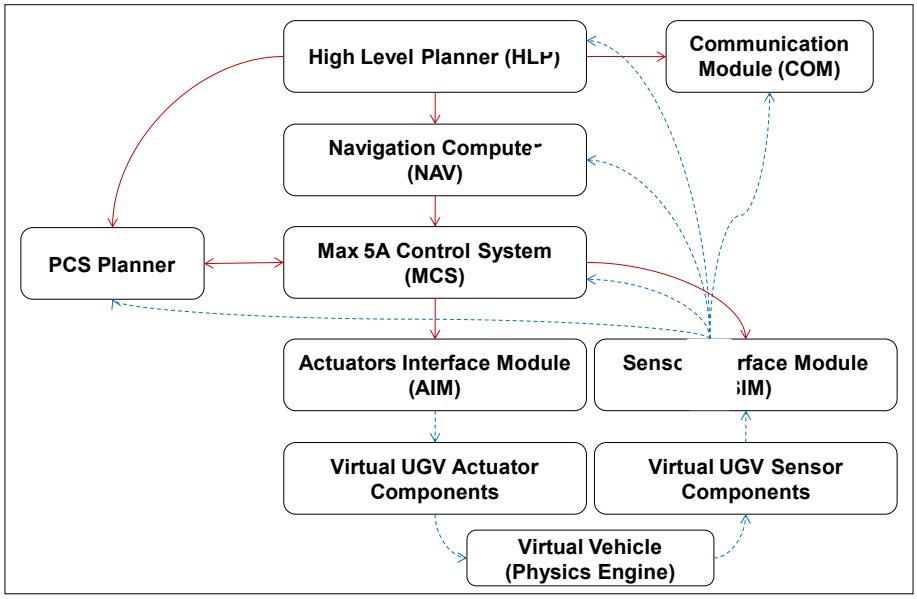


Figure 63. Intelligent Control System Architecture for the UGV Agent

6.3.5 High Level Planner

The High Level Planner (HLP) module has the highest level of responsibility in the UGV control architecture, and is responsible for instigating all actions. The HLP module produces sequences of commands (such as "Direct To" and "Track To") for the lower level Control System. Mission planning is outside the scope of this section, so a

sufficient algorithm was created to control execution through the experiment. This algorithm is recursive in nature, and shown in Figure 64.

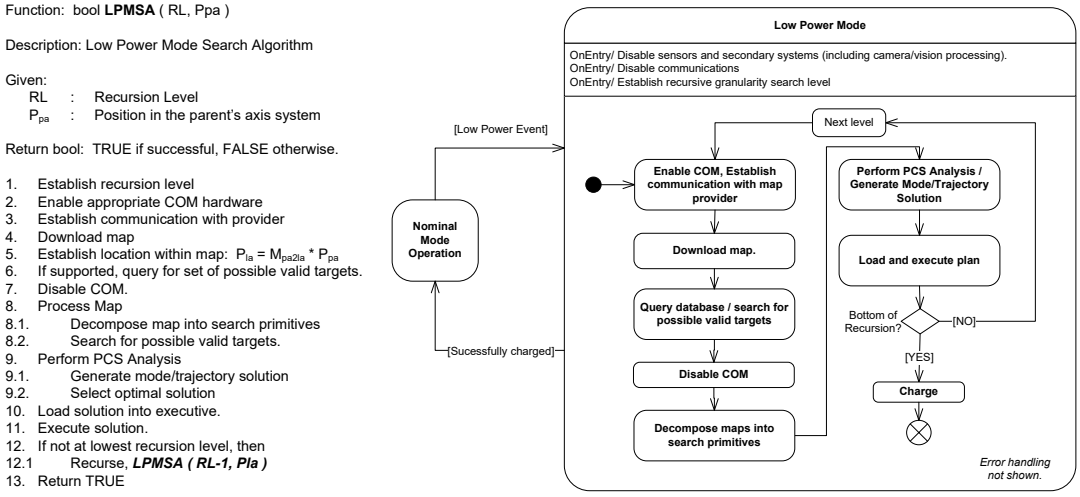


Figure 64. Higher Level Planner's "Low Power Mode" Algorithm

At each recursion level, the planner communicates with a central authority for that level. This authority provides the planner with the objective and constraints for that level, which is the navigation target location and the constraints. The HLP translates these goals and constraints to the PCS planner. In this experiment, two levels are implemented, campus level and building level. The authority provides the HLP with bitmap shown in Figure 65, which correlates to the maps from Figure 57 (a) and (b), where navigation targets are shown in red. The PCS planner utilizes direct pixel lookup to determine navigable locations in the map. As the UGV transitions to the building level, the authority also provides location and possible sensor locations of the cameras, and shown in Figure 65 (c-f).

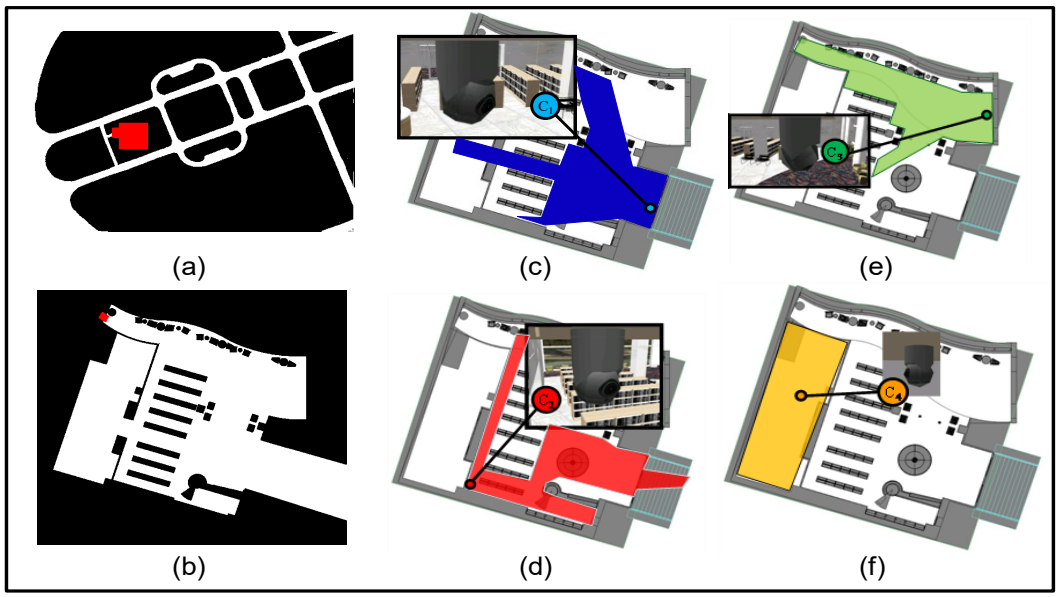


Figure 65. Navigation Objectives and Constraint Bitmaps.

The allowable navigation areas for the campus (a) and the building (b). The camera locations and the sensor coverage areas for each are shown in (c)-(f), monochromatic bitmap are shown overlaid on building map.

6.3.6 MAX Control System (MCS)

The Max Control System (MCS) (Figure 63) is responsible for engaging the specified mode commands from the NAV component, and implementing the feedback control laws that produce output to the vehicle actuators. The MCS implements a simple controller graph shown in Figure 66. Here, cross-track error (XTE) is used for lateral guidance through a simple PID control feedback law to compute the desired heading, and heading error is used to control the steering error through a PID inner loop controller. The along-track error (ATE) is ignored, and the speed controller (not shown) is a PID law from the wheel speed error to the throttle.

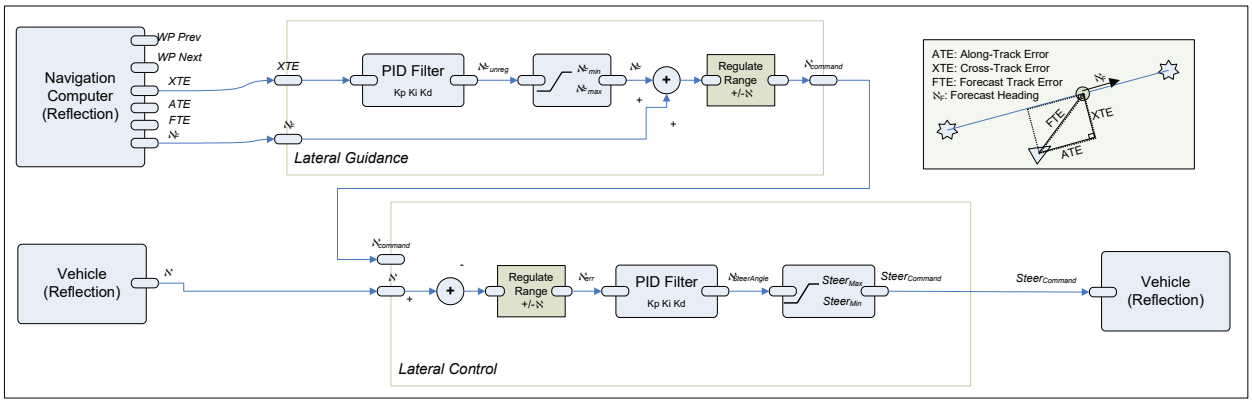


Figure 66. Max Control System (MCS) Controller Graph

6.3.7 Trajectory Planning through Suboptimal Search

The PCS planning module in Figure 63 must plan a trajectory to the target locations in Figure 65 while avoiding obstacles in (a) and (b), and navigating through observable locations in (c)-(f). The PCS Planner utilizes a random and pseudo-optimal trajectory optimization algorithm to solve the trajectory optimization function, adapted to include probabilities on the output. The algorithm is a generic search tree algorithm placed into the context of an optimal control formulation, and modified for trajectory pseudo-optimization over objective functions. This algorithm is a random tree search, similar in most respects to the rapidly exploring random trees algorithms (Ippolito and Yeh, 2009). It differs in that a heuristic distance estimate is not used to connect random locations back to closest points in the tree. The system relies on branch generation system through feedback control design that solves a subspace of the global problem. The branch generation system does not necessarily correlate to the vehicle systems and controllers used in implementation. The optimization does not necessarily correspond to the vehicle model space or the actual input space, and can represent the parameters over which any decision or control can be accomplished. In this problem, the input space is augmented with a discrete variable representing the decision to switch to a new camera.

Consider any general non-linear vehicle model of the following form, where $x \in X$ represents state, $y \in Y$ represents output, $u \in U$ represents costs, and $t \in \tau$ represents the planning time interval. Generally, $X \subseteq \mathbb{R}^n$, $Y \subseteq \mathbb{R}^l$, $U \in \mathbb{R}^m$, $\mathcal{T} \in [t_0, t_f]$, $t_0, t_f \in \mathcal{R}$, $t_0 < t_f$.

$$\begin{aligned} \dot{x} &= f_C(x, u, t) \\ y &= h(x, u, t) \end{aligned} \tag{6.3-3}$$

The dynamic model f_C is assumed to be in a form augmented to enforce general equality or inequality constraints applied to the system is placed in the set $C = \{C_E, C_I\}$, where $C_E(x, \dot{x}, u, t) = 0$ and $C_I(x, \dot{x}, u, t) \leq 0$. Let C_0 be inequality constraint for obstacles which will be handled explicitly by the tree search algorithm, $C_O(x, t) \leq 0$. Define the goal space $\tilde{X}$ to be a subset of the search space $\hat{X}$ which is a subset of the state space X, such that $\tilde{X} \subseteq \hat{X} \subseteq X$. Define a branch generation control system defined as $G_b: (\mathcal{T} \times \mathrm{X} \times \tilde{\mathrm{X}} \times \mathbb{R}) \rightarrow (X \times U \times \mathcal{T})$, which generates a trajectory given a time $t_0 \in \mathcal{T}$, state $x(t_0) \in X$, and destination goal $P_g \in \tilde{X}$.

$$\left(x'_{[t_0,t_1]}, u'_{[t_0,t_1]}, t_1, J'\right) = G_b(t_0, x(t_0), P_g) \tag{6.3-4}$$

Define final state cost $L(x_f, t_f)$, and define model sensors and objectives as a set $\Phi = \{\phi_i(x,u,t)\}$ of integral cost functions. The trajectory problem can be stated as follows. Find control input $u^*(t)_{[t_0,t_1]}$ and associated trajectory $x^*(t)_{[t_0,t_1]}$ leading from a given $x(t_0)$ to a given x_{goal} that minimizes J subject to constraints C.

$$u^* = \arg\min_{\mathrm{u}} \left(\sum_i J_i(x,u,t) \right)$$
$$subject\ to\ C_O, where$$
$$J_i(x,u,t) = L_i(x_f, t_f) + \sum_{\Phi} \left(\int_{t_0}^{t_f} \int \phi_i(x,u,t) dt \right) \tag{6.3-5}$$

Our extension to the previous algorithm presented in (Ippolito and Yeh, A Trajectory Generation Approach for Payload Directed Flight 2009) involves random tree branch node selection that is biased under a probability distribution applied to the tree nodes based on their cost function J, encouraging lower cost nodes to be expanded in the search. The probability function γ is given by the following, where σ and κ are constant factors for shaping the distribution. This function can be modified to include time, to encourage tree node selection down branches further in the tree.

$$\gamma(t) = \frac{1}{1 + \mathrm{e}^{\left(\frac{(t-\kappa)}{\sigma}\right)}} \tag{6.3-6}$$

The modified algorithm is shown in Figure 67. Branches are expanded by random selection over a probability mass function distribution. A priority queue is established in parallel with the tree, supporting linear time insertion and selection over a probability distribution.

1. FUNCTION GenerateBranch ($P_{goal} \in \tilde{x}$, Tree T, Priority Queue Q, Constraints C_O)
    1. Select a random point $(x_0, t_0) \in (X, \mathcal{T})$ on the tree T under the probability density function γ.
    2. Generate a random goal point $P_{goal} \in \tilde{X}$ in the goal space.
    3. Use G_b to generate a candidate trajectory branch $b' = \left(x'_{[t_0,t_1]}, u'_{[t_0,t_1]}, t_1, J'\right) = G_b(t_0, x(t_0), P_g)$ from x_0 to P_{goal} based on augmented plant.
    4. If b' violates obstacles in C_O, trim the trajectory.
    5. $T \leftarrow T + b'$ (add b' to tree T)
    6. Q.Enqueue (b', J')

Figure 67. Trajectory Algorithm

6.3.8 Vehicle Control System and Branch Generation

A custom branch generation system (BGS) facilitates rapid generation of branches and costs. The BGS must be efficient, as it will be repeatedly called during generation of the tree. This is conceptually shown in Figure 68. The branch generation system used for this experiment is a simplified closed-loop plant model, with sensor objectives model that calculates costs as shown below. An environment model was not required for this experiment.

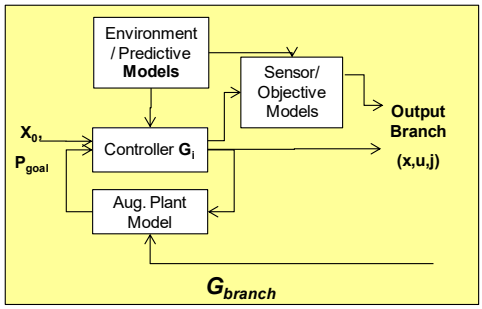
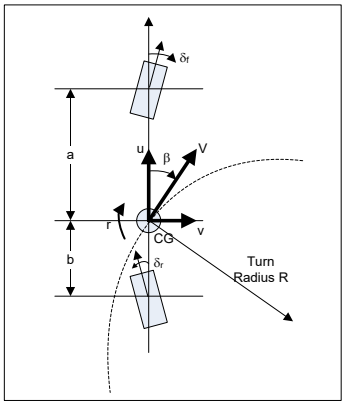
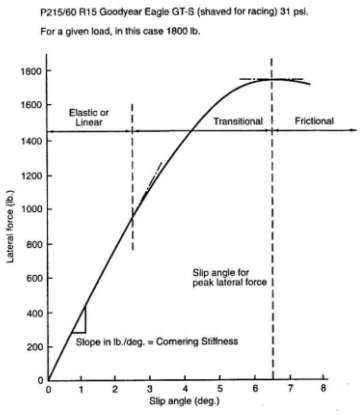


Figure 68. Branch Generation System.

The components of the BGS are a dynamic model, a control system, and the objective functions. This system was implemented in Matlab Simulink and autogenerated into C. The objective functions were described earlier for the rover and the security cameras. A simplified dynamic model of a rover was constructed for this control system. The model is shown in Figure 69.





$$\begin{bmatrix}\dot{x}_w\\ \dot{y}_w\end{bmatrix}=\begin{bmatrix}V\cos(\psi+\beta)\\ V\sin(\psi+\beta)\end{bmatrix}$$

$$\begin{bmatrix}\frac{(C_F+C_R)}{mV} & \frac{(aC_F-bC_R)}{mV^2}-1 & 0\\ \frac{(aC_F-bC_R)}{I_z} & \frac{(a^2C_F+b^2C_R)}{VI_z} & 0\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\beta\\ r\\ \psi\end{bmatrix}+\begin{bmatrix}-\frac{C_F}{mV} & \frac{C_R}{mV}\\ -\frac{aC_F}{I_z} & -\frac{bC_R}{I_z}\\ 0 & 0\end{bmatrix}\begin{bmatrix}\delta_F\\ \delta_R\end{bmatrix}$$

Figure 69. Planning Model in the Branch Generation System.
System plant is based on a two-wheel steering bicycle model.

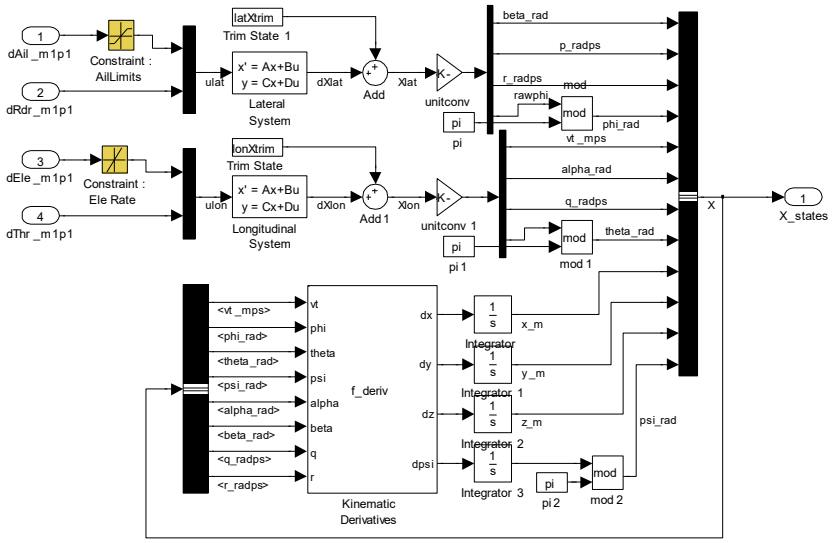


Figure 70. Implementation of the BGS plant in Matlab Simulink.

A track-to heading controller was designed for the BGS, taken from the UGV's embedded control system.

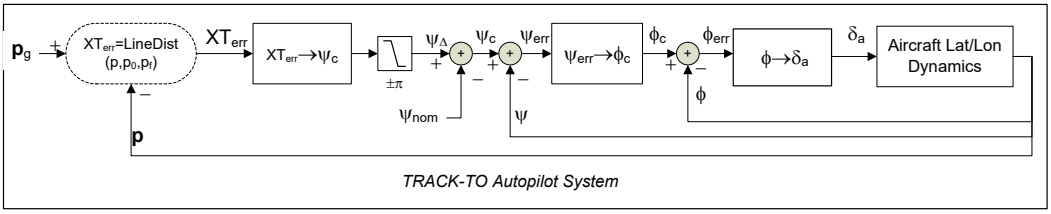


Figure 71. Track-To Control System

6.3.9 Multi Objective Cost Functions

Five cost functionals J_1 through J_5 were utilized in (6.3-6) to provide optimization objective for the trajectory planner, each with a gain factor that requires tuning and is the subject of sensitivity analysis for the results section. J1 cost function minimizes total time to complete the mission, J2 maintains a specified distance to the current camera, J3 minimizes the amount of time spent in each camera, J4 minimizes the number of times the camera switches, and J5 minimizes the total distance traveled. The cost functions are shown in (6.3-7) below.

$$
\begin{aligned}
J_1(x,u,t) &= K_1 t \\
J_2(x,u,t) &= K_2\, MIN\left(1, \frac{(\|p_{cam}(t,x) - p(x)\| - R_{des})^2}{R_{des}^2}\right) \\
J_3(x,u,t) &= K_3 \mathcal{T}_{cc} \\
J_4(x,u,t) &= K_4 N_{switch} \\
J_5(x,u,t) &= K_5 \int_{t_0}^{t} v(x)dt
\end{aligned}
\tag{6.3-7}
$$

Here $\mathcal{T}_{cc}$ is the amount of time spent in the current camera, N_{switch} is the number of topological switches that have occurred (number of times the UGV has moved from one camera to another), p(x) and v(x) are the position and velocity vectors respectively, $p_{cam}(t,x)$ is the position of the camera that would be utilized, R_{des} is the desired distance to maintain from the camera to the rover, and MIN(•,•) is a function that returns the minimum of two values.

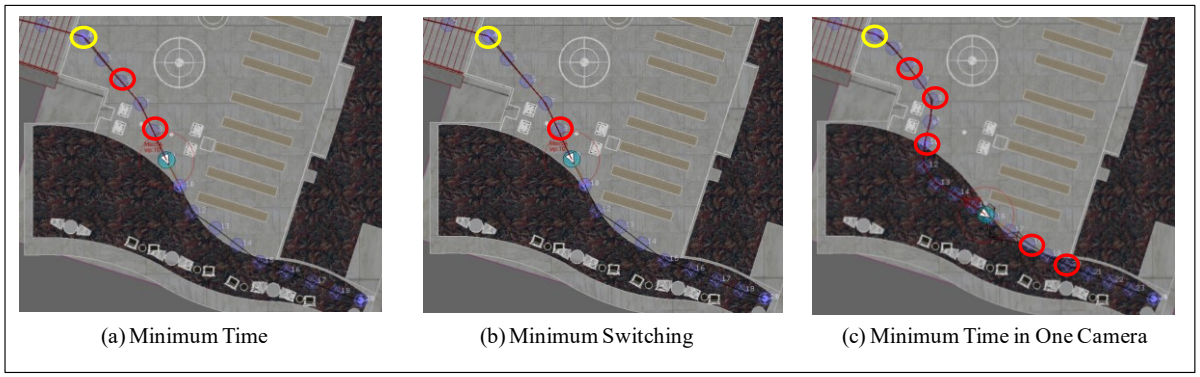
The objectives in (6.3-7) are competing and nonlinear, with a complex topology that is illustrated in the simulation results. For this reason, we utilize a random tree-search algorithm.

6.3.10 Simulation Results

The simulation tests were implemented in Reflection in the C++ programming language. Matlab Simulink graphs were converted into C++ through MathWorks Real-Time Workshop. Tests were run on an Intel Core 2 Duo X9650, 3 GHZ, 3GB RAM, Nvidia Quadro GPU. The system allowed the UGV planning times of 10 seconds, sharing processor with simulation. Search algorithm averaged 48,202 nodes (states) added to the tree, with 3435 branches, taking 15,002,030,502 CPU clock cycles. To simplify the rendering of the search tree results, results include up to 200 full length branches considering complete path distance and branch cost.

Trajectory generation may be able to utilize various trajectory optimization objectives to generate preferable trajectories and contingencies. To study this, the solutions were studied under varying gain proportions to represent differing agent priorities. The resulting trajectories generated by optimizing over various priorities are shown in Figure 72. In (a), only time was considered. The resulting path switches control topology around three cameras in the order (C_1,C_3,C_2), and the resulting trajectory is a direct path to the destination. In (b), the number of topology reconfigurations were minimized as well as time, and the resulting path is deflected towards the camera C2 to allow only two cameras (C_1,C_2) to be utilized (resulting in a single topology reconfiguration). In (c), the amount of time

spent in a single camera was minimized along with overall time. The resulting trajectory requires five switches, and the path diverges from previous solutions to allow for viewability from the different cameras.



(a) Minimum Time (b) Minimum Switching (c) Minimum Time in One Camera

Figure 72. Results of Optimizing Specific Objectives.

Screenshots of moving maps display showed. Starting position for the building level navigation is shown in yellow. The red circles show location for control switching.

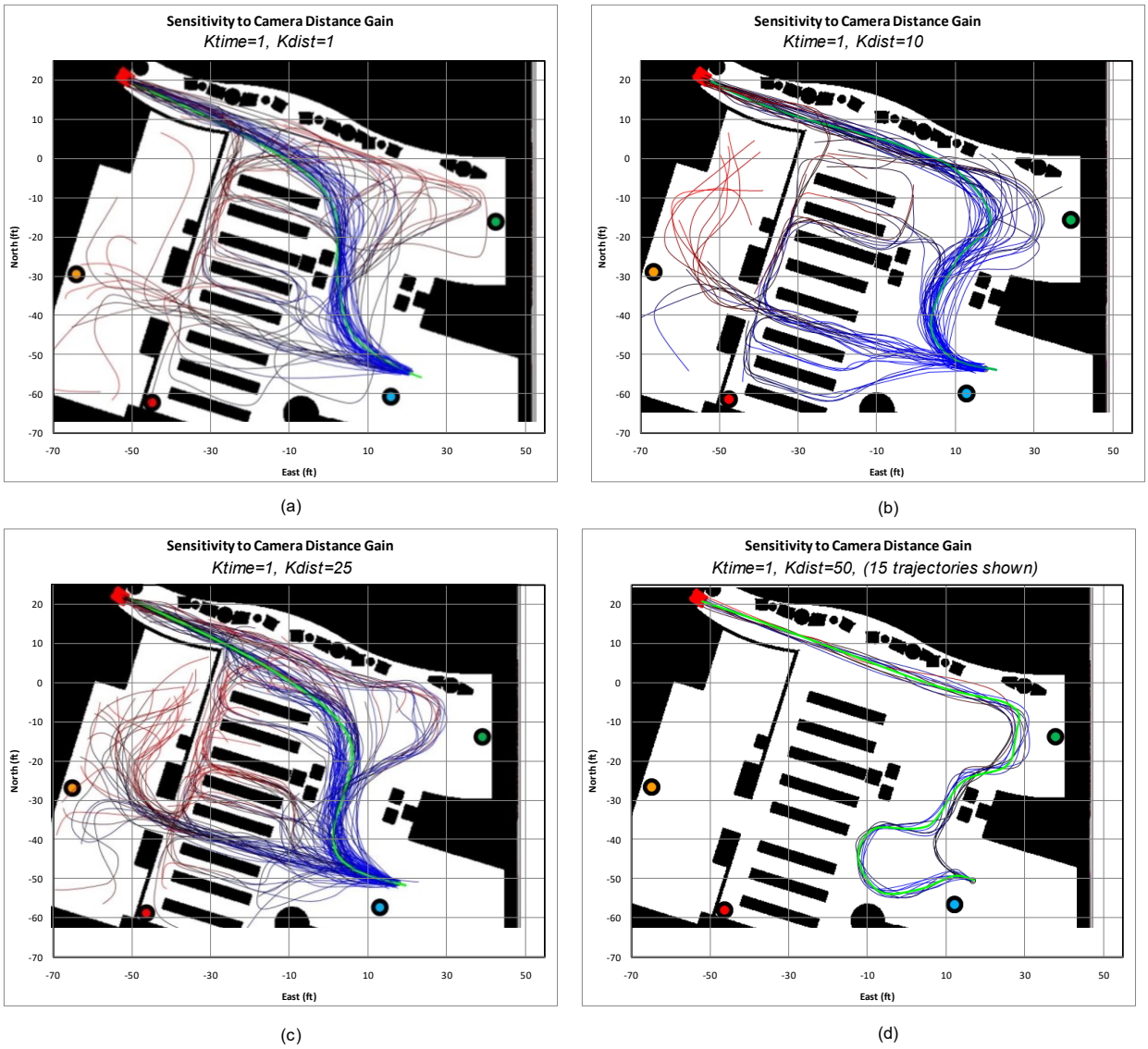


Figure 73. Sensitivity of Trajectory Search Trees to Time in Camera Gain

Optimizing over the time the UGV spends in a single camera results in the trajectory being diverted towards the distance from specific cameras. The sensitivity to this parameter was studied by varying the gain from K_{dist}=(1,10,25,50), and portions of the resulting trajectory trees are shown in Figure 73 (a)-(d). The path color indicates the cost of realizing that particular position, where blue represents lower costs, and red represents the highest state cost rendered in the plot.

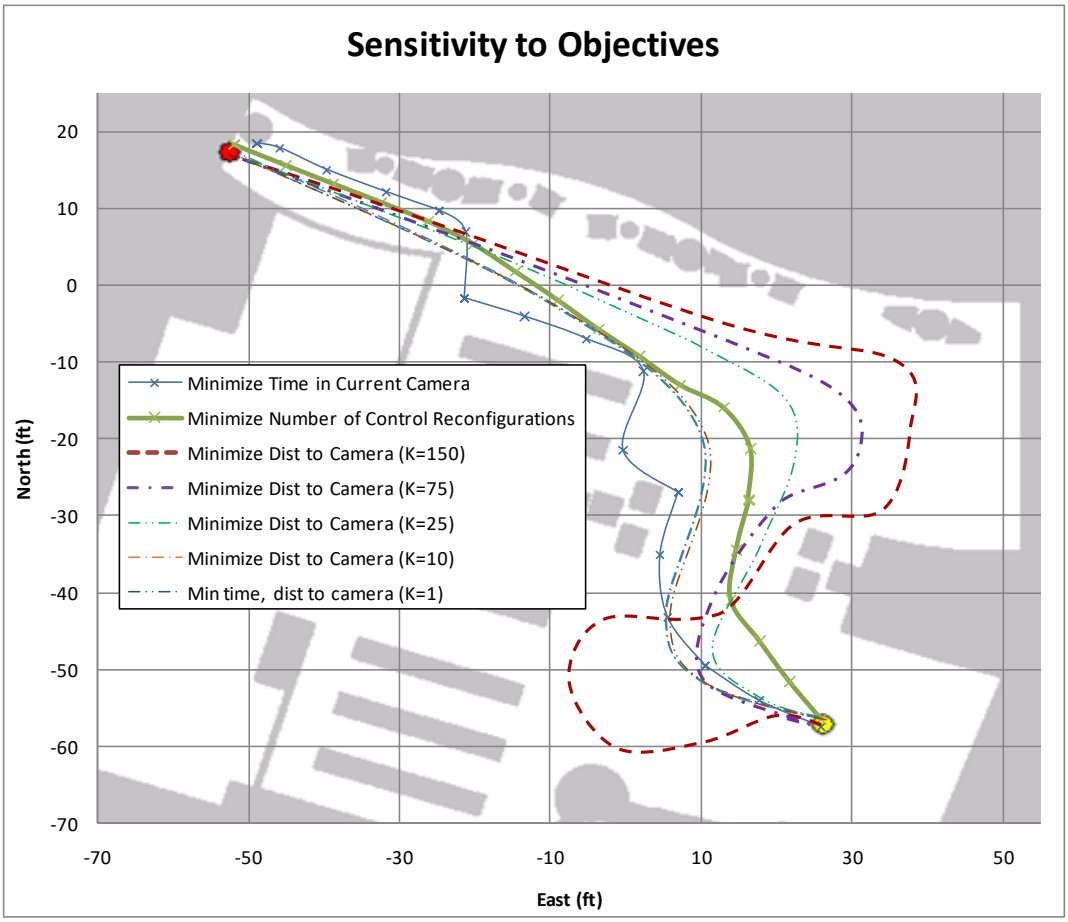


Figure 74. Comparison of Solution Objective Sensitivity

The final solution trajectories for various optimization objectives are summarized in Figure 74. Increasing or decreasing various objectives results in adjusted trajectories as shown, but the results are still viable.

6.4 Elastically Shaped Aircraft Control (ESAC) Application

In this section, we describe an initial optimization study of a Variable-Camber Continuous Trailing-Edge Flap (VCCTEF) system. The VCCTEF provides a light-weight control system for aircraft with long flexible wings, providing efficient high-lift capability for takeoff and landing, and greater efficiency with reduced drag at cruising flight by considering the effects of aeroelastic wing deformations in the control law. The VCCTEF system is comprised of a large number of distributed and individually-actuatable control surfaces that are constrained in movement relative to neighboring surfaces, and are non-trivially coupled through structural aeroelastic dynamics. Minimization of drag results in a constrained, coupled, non-linear optimization over a high-dimension search space. In this section, we describe the modeling, analysis, and optimization of the VCCTEF system control inputs for

minimum drag in cruise. The purpose of this initial study is to quantify the expected benefits of the system concept. The scope of this analysis is limited to consideration of a rigid wing without structural flexibility in a steady-state cruise condition at various fuel weights. For analysis, we developed an optimization engine that couples geometric synthesis with vortex-lattice analysis to automate the optimization procedure. In this section, we present and describe the VCCTEF system concept, optimization approach and tools, run-time performance, and results of the optimization at 20%, 50%, and 80% fuel load. This initial limited-scope study finds the VCCTEF system can potentially gain nearly 10% reduction in cruise drag, provides greater drag savings at lower operating weight, and efficiency is negatively impacted by the severity of relative constraints between control surfaces.

6.4.1 Introduction

Advanced light-weight materials show promise of increasing energy efficiency of modern aircraft in part through reduction of airframe weight. Lighter-weight wing structures constructed from these materials can provide sufficient load-carrying capacity for the aircraft with reduced structural rigidity. Increased structural flexibility may potentially degrade overall aerodynamic efficiency due to aeroelastic interactions between aerodynamic forces and wing-structure dynamics, which can have a significant impact on the aerodynamics of the aircraft. A recent NASA study[1] showed active control of wing aeroelasticity can be beneficial in achieving drag reduction. The study showed that highly flexible wing aerodynamic surfaces can be elastically shaped in-flight by active control of wing twist and vertical deflection in order to optimize the local angle of attack of wing sections to improve aerodynamic efficiency through drag reduction during cruise and enhanced lift performance during take-off and landing.

In this section, we present an initial assessement of a new numerical model of a variable continuous trailing-edge flap (VCCTEF) system on a commercial transport-class fixed-wing aircraft in subsonic cruise. The VCCTEF system offers potential pay-off for drag reduction by active aeroelastic shape control[2]. This system employs light-weight shape-memory alloy (SMA) technology for actuation and three separate chordwise segments shaped to provide a variable-camber to the flap. A VCCTEF conceptual schematic is shown below. The VCCTEF system consists of 2-foot spanwise flap segments that can be individually actuated, resulting in the ability to control the wing twist as a function of span to improve the lift-to-drag ratio (L/D) at any aircraft gross weight or flight condition. This provides an advantage over conventional flap systems, since wing-twist is permanently set for one cruise configuration on conventional aircraft, and is typically set for 50% loading or mid-point on the gross weight schedule. The VCCTEF system offers the ability to set wing-twist for any gross weight and flight conditions. Wing-twist may be independently

specified for climb, cruise and descent phases, achieving optimal L/D efficiency throughout all phases of flight. The individual 2-foot spanwise flap sections are connected with a flexible elastomeric covering with no gaps on the surface between flap segments, thus reducing drag by eliminating breaks in the flap continuity which otherwise would generate vorticity that results in a drag increase and also contributes to airframe noise[2,3,4].

Analysis is performed on a modified NASA Generic Transport Model (GTM) aircraft, which is based on a Boeing 757 airframe[5]. The GTM model was modified to incorporate a VCCTEF system consisting of 15 2-foot spanwise sections on each wing. Each segment contains three actuatable rotational joints. In this assessment we assume the deflection of each flap segment provides a third of the commanded angle to provide a smooth camber shape. Drag-reduction potential is calculated through direct numerical optimization of VCCTEF deflection angles on the modified GTM model to minimize drag at various fuel loads in cruise. The following study analyzes steady-state conditions only, assuming transient and unsteady aerodynamic effects can be neglected. Aerodynamic forces and moments are evaluated using the VORLAX software library[5]. VORLAX is a computational aerodynamic tool that utilizes vortex-lattice methods to evaluate the aerodynamics of arbitrary-shaped bodies. This tool utilizes potential flow technique to estimate induced drag in subsonic laminar flow regimes. The underlying assumptions in VORLAX are therefore extended to this analysis. This initial study was performed on a rigid aircraft assuming static wing geometry (e.g., invariant to aerodynamic loads). Aeroelastic effects and structural flexibility of the wing will be analyzed into follow-on studies based on models that were under development at the time of this initial study[6].

6.4.2 Vehicle Model

The model[4] utilized in this study describes the longitudinal dynamics of a generic transport aircraft at Mach 0.80 and Mach 0.88 at an altitude of 35,000 ft. The longitudinal dynamics are described by two aircraft states: angle of attack α and pitch rate q. The aircraft model contains 80 structural states capturing the 20 dominant flexible modes, and provides 23 control inputs: one elevator, 11 flaps, and 11 slats; as shown in Figure 75.

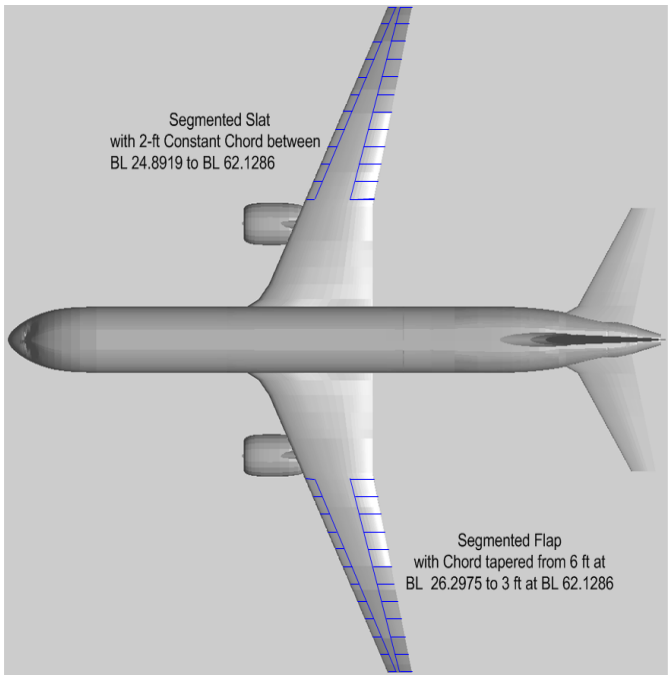


Figure 75. Distributed leading-edge (slats) and trailing-edges (flaps) actuator configuration.

The coupled equations of motion are given by

$$\begin{bmatrix}\dot{x}_r\\ \dot{x}_e\end{bmatrix} = \begin{bmatrix}A_{rr} & A_{re}\\ A_{er} & A_{ee}\end{bmatrix}\begin{bmatrix}x_r\\ x_e\end{bmatrix} + \begin{bmatrix}B_{rr} & B_{re}\\ B_{er} & B_{ee}\end{bmatrix}\begin{bmatrix}u_r\\ u_e\end{bmatrix} \tag{6.4-1}$$

where the states are given by $x_r = [\alpha \quad q]^T$, $x_e = [q_w \quad q_\theta \quad \dot{q}_w \quad \dot{q}_\theta]^T$, and q_w,$q_\theta \in \mathbb{R}^m$ are the generalized coordinate vectors for bending and torsion, respectively, along with their derivatives $\dot{q}_w, \dot{q}_\theta \in \mathbb{R}^m$. The elements of q_w and q_θ are indexed such that q_{w_i} and q_{θ_i} are the wing bending and torsional deflections of mode i = 1,…,m, where m=20 is the total number of mode shapes. The input $u_r = [\delta_e]$ is the elevator flap deflection angle, and $u_e = [\delta_f \quad \delta_s]^T$ where δ_f is the vector of 11 distributed flap inputs and δ_s is the vector of 11 distributed slat inputs. This results in a model with 82 states and 23 inputs. This initial study assumes all aeroelastic modes and vehicle states are observable.

The lift and drag characteristics of the aircraft are functions of the current vehicle states, flexible-mode states, and control inputs as described by the following equations

$$C_D = C_{D_0} + K\left(C_{L_0} + C_{L_x}x + C_{L_u}u\right)^2 + C_{D_u}u + u^T C_{D_{u^2}}u \tag{6.4-2}$$

where K is the drag polar parameter, C_D is the total drag coefficient, and C_L is the total lift coefficient.

The resulting geometric mesh is processed in the VORLAX engine over a range of angle of attacks, and the results are used to determine the trim state drag based on the given fuel load and total operating weight. The process for evaluation, given the input flap commands and fuel load, is summarized as follows.

1. Evaluate desired C_L as a function of fuel load and operating weight.
2. Generate the geometric mesh for the given flap deflection input vector u (assumes static/fixed geometry).
3. Evaluate C_L and C_D over a range of α's using VORLAX.
4. Fit a cubic hermite spline curve for interpolation of the C_L/C_D curve.
5. Interpolate to find C_D at the specified $C_L(W)$.

6.4.3 Graph-Space Model

Consider an aircraft with a flexible wing that employs a Variable Continuous Trailing Edge Flap (VCCTEF) system. The structure of the dynamic system is shown in Figure 76.

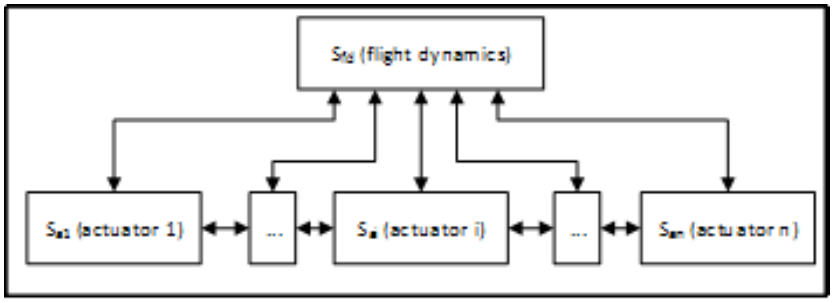


Figure 76. VCCTEF Dynamic System Model

The full derivation of the system is presented in (Ippolito et al, 2013). The longitudinal flight dynamics model for this system is given by the following $\mathcal{S}_{lon}$, derived from a standard four-state longitudinal mode dynamics model presented in (Etkin and Reed, pg 166).

$$\mathcal{S}_{lon}: \dot{x_{lon}} = A_{lon}\, x_{lon} + B_{lon}\, u_{lon}$$

$$A_{lon} = \begin{bmatrix} -0.006868 & 0.01395 & 0 & -32.2 \\ -0.09055 & -0.3151 & 773.98 & 0 \\ 0.0001187 & -0.001026 & -0.4285 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \; ; \; B_{lon} = \begin{bmatrix} 0.000187 & 9.66 \\ -17.85 & 0 \\ -1.158 & 0 \\ 0 & 0 \end{bmatrix} \tag{6.4-3}$$

$x = [\Delta u \quad w \quad q \quad \Delta\theta]^T, u = [\delta_{ele} \quad \delta_{rdr}]^T.$

The flexible dynamics for the aircraft wing augment this system with additional states to capture the flexible wing structure and modes. The complete system $\mathcal{S}_2$ is given by the following.

$$\mathcal{S}_2 : \dot{x} = Ax + Bu \tag{6.4-4}$$

where $x = [x_{lon} \quad x_{a1} \quad \dot{x}_{a1} \quad \dots \quad x_{a_{Na}} \quad \dot{x}_{a_{Na}}]^T$ and $u = [\delta_{thr} \; u_1 \dots u_{N_a}]$. The structure of the system is given by

$$A = \begin{bmatrix} A_{lon} & B_{a1} & B_{a2} & \dots & B_{ai} & \dots & B_{an} \\ 0 & A_i & A_{i+1} & & & & \\ \vdots & & & \ddots & & & \\ 0 & & & A_{i-1} & A_i & A_{i+1} & \\ \vdots & & & & & \ddots & \\ 0 & & & & & A_{i-1} & A_i \end{bmatrix}$$

$$B = \begin{bmatrix} B_{lon} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} & & & \bar{0}_{4\times N_a} & & \\ & \begin{bmatrix} 0 \\ b_f \end{bmatrix} & & & & \\ & & \ddots & & & \\ \bar{0}_{2N_a\times 1} & & & \begin{bmatrix} 0 \\ b_f \end{bmatrix} & & \\ & & & & \ddots & \\ & & & & & \begin{bmatrix} 0 \\ b_f \end{bmatrix} \end{bmatrix} \tag{6.4-5}$$

The block terms in the

$$A_i = \begin{bmatrix} 0 & 1 \\ -k_a - 2k_b & -b_a - 2b_b \end{bmatrix} \; ; \; B_{ai} = [B_{lon,i} \quad \bar{0}_{4\times 1}] \; ; \; A_{i+1} = A_{i-1} = \begin{bmatrix} 0 & 0 \\ k_b & b_b \end{bmatrix} \tag{6.4-6}$$

Each actuator's effectiveness is modeled with the following relationship.

$$B_{lon,i} = f(i) \cdot B_{lon} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{6.4-7}$$

The transformation of the system dynamics to the system graph through $\mathcal{T}_{\mathcal{S}\to\mathcal{H}}(\mathcal{S}_2)$ is represented in Figure 77.

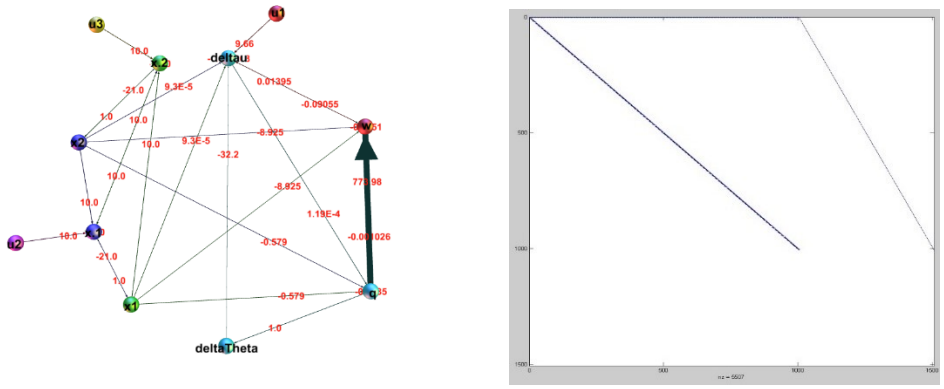


Figure 77. System Graph Transformation of $\mathcal{S}_2$.
Structure of the A matrix shown on the right.

6.4.4 Branch-generation controller

The control system is designed using full-state feedback with attack-angle tracking-error integrator feedback as developed in the literature [4] and summarized here. This formulation extends the model to include pitch-rate tracking as well as a reference signal filter. The controller tracks a commanded angle of attack signal, α_c. Two integrator states α_I, q_I integrate the reference error signals for control, given by

$$\begin{aligned} \dot{\alpha}_I &= \alpha_r - \alpha \\ \dot{q}_I &= q_r - q \end{aligned} \tag{6.4-8}$$

The reference model filter combined with the integrators above can be expressed as

$$\dot{x}_c = A_{cc}x_c + A_{cr}x_r + B_{cz}z$$

$$A_{cc} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\omega_r^2 & -2\omega_r \end{bmatrix} \quad ; \quad A_{cr} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad ; \quad B_{cz} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \omega_r^2 \end{bmatrix} \tag{6.4-9}$$

where $x_c = [\alpha_I \; q_I \; \alpha_r \; q_r]'$ and $z = [\alpha_c]$ is the input reference signal. The reference model in (6.4-9) is appropriate for the model given in (6.4-1) which has two states, α and q. Combining the systems in (6.4-9) and (6.4-1) forms the augmented system, given by

$$\begin{aligned} \dot{x} &= Ax + Bu + B_z z \\ y &= Cx + Du \end{aligned} \tag{6.4-10}$$

where $x \in \mathbb{R}^N$ is of size N=86 and $u \in \mathbb{R}^M$ is of size M=23, and

$$x = [x_r\ x_e\ x_a]^T\ ;\ u = [u_r\ u_e]^T\ \ ;\ \ z = [\alpha_{cmd}]$$
$$A = \begin{bmatrix} A_{rr} & A_{re} & 0 \\ A_{er} & A_{ee} & 0 \\ A_{cr} & 0 & A_{cc} \end{bmatrix};\ B = \begin{bmatrix} B_{rr} & B_{re} \\ B_{er} & B_{ee} \\ 0 & 0 \end{bmatrix};\ B_z = \begin{bmatrix} 0 \\ 0 \\ B_{cz} \end{bmatrix};\ C = I\ ;\ D = 0 \tag{6.4-11}$$

The optimal control formulation utilizes a cost function J with quadratic costs terms to address the multiple control objectives.

$$J(x,u,t) = \frac{1}{2}\int_0^\infty (x^T Q x + u^T R u + q_d C_D)dt \tag{6.4-12}$$

The weighting matrix $Q = diag\{Q_r, Q_e, Q_c\}$ and its submatrices are diagonal matrices which regulate vehicle states x_r, aeroelastic states x_e, and the control reference model x_c, respectively. Similarly, $R = diag\{R_f, R_s, R_e\}$ and its submatrices penalize control actuation effort of the flaps, slats, and elevator, respectively. The drag coefficient C_D term is included in the cost function with weight q_d. Substituting in J yields

$$J(x,u,t) = \frac{1}{2}\int_0^\infty \left(x^T Q x + u^T R u + q_d C_{D_0} + q_d K\left(C_{L_0} + C_{L_x} x + C_{L_u} u\right)^2 + q_d C_{D_u} u + u^T q_D C_{D_{u^2}} u\right) dt \tag{6.4-13}$$

The Hamiltonian equation is given by

$$H = \frac{1}{2}\left[x^T Q x + u^T R u + q_d C_{D_0} + q_d K\left(C_{L_0} + C_{L_x} x + C_{L_u} u\right)^2 + q_d C_{D_u} u + u^T q_d C_{D_{u^2}} u\right] + \lambda^T (Ax + Bu + B_z z) \tag{6.4-14}$$

The control solution for problems of this form[7] is given by

$$u(x,z) = u_x x + u_z z + u_c \tag{6.4-15}$$

where

$$\begin{gathered} u_x = -\bar{R}^{-1}\left(B^T P + q_d K C_{L_u}^T C_{L_x}\right) \\ u_z = -(\bar{R}^{-1} B^T S) \\ u_c = -\bar{R}^{-1}\left(q_d K C_{L_u}^T C_{L_0} + \frac{1}{2} q_d C_{D_u}^T + B^T \lambda_0\right) \end{gathered} \tag{6.4-16}$$

The value of P is found from the solution to the following algebraic Riccati equation

$$P\bar{A} + \bar{A}^T P - PB\bar{R}^{-1}B^T P + \bar{Q} = 0 \tag{6.4-17}$$

where

$$\begin{aligned}\bar{R} &= R + q_d K C_{L_u}^T C_{L_u} + q_d C_{D_{u^2}} \\ \bar{A} &= A - B\bar{R}^{-1} q_d K C_{L_u}^T C_{L_x} \\ \bar{Q} &= Q + q_d K C_{L_x}^T \left(C_{L_x} - C_{L_u} \bar{R}^{-1} q_d K C_{L_u}^T C_{L_x}\right)\end{aligned} \tag{6.4-18}$$

and

$$\begin{aligned}S &= (PB\bar{R}^{-1}B^T - \bar{A}^T)^{-1} PB_z \\ \lambda_0 &= (PB\bar{R}^{-1}B^T - \bar{A}^T)^{-1} \left(q_d K C_{L_x}^T C_{L_0} - \left(PB + q_d K C_{L_x}^T C_{L_u}\right)\bar{R}^{-1}\left(q_d K C_{L_u}^T C_{L_0} + \frac{1}{2} q_d C_{D_u}^T\right)\right)\end{aligned} \tag{6.4-19}$$

The sufficient condition for a solution to exist requires checking the following, from which $\bar{Q} > 0$ can be inferred[7].

$$q_d K C_{L_u}^T C_{L_x} < \bar{R} \tag{6.4-20}$$

6.4.4.1 Branch Generation Implementation

Forming the closed loop system of (6.4-10) under the feedback law (6.4-15) yields

$$\dot{x} = (A + Bu_x)x + (Bu_z + B_z)z + Bu_c \tag{6.4-21}$$

Note that the u_c term is constant in the derivative equation in (6.4-21), resulting in component x_s of the state vector given by

$$\begin{aligned}\tilde{x} &= x + x_s \\ x_s &= A^{-1}B\bar{R}^{-1}\left(q_d K C_{L_u}^T C_{L_0} + \frac{1}{2} q_d C_{D_u}^T + B^T \lambda_0\right)\end{aligned} \tag{6.4-22}$$

where $\tilde{x}$ is the complete solution state vector. Equation (6.4-22) must be solved using the original system matrix $\begin{bmatrix} A_{rr} & A_{re} \\ A_{er} & A_{ee} \end{bmatrix}$, which is invertible, and the effect of the constant u_c term in the integrator can be solved separately.

For evaluation of drag reduction, we define an average drag metric which integrates ΔC_D over the specified time interval.

$$C_{D_{avg}} = \frac{1}{t_f} \int_0^{t_f} \Delta C_D dt \tag{6.4-23}$$

The state and aeroelastic weighting matrices were initially selected based on Bryson's criteria, which set the initial weighting elements based on the maximum expected value of the state and input elements. The Q and R diagonal matrices are given by

$$
\begin{gathered}
Q_r = k_{Qr} \cdot diag\left[\frac{1}{\alpha_{max}^2}, \frac{1}{q_{max}^2}\right] \\
Q_e = k_{Qe} \cdot diag\left[1/q_{w_{max}}^2 \quad 1/q_{\theta_{max}}^2 \quad 1/\dot{q}_{w_{max}}^2 \quad 1/\dot{q}_{\theta_{max}}^2\right] \\
Q_c = k_{Qc} \\
R_r = k_{re} \cdot \left[\frac{1}{\delta_{e_{max}}^2}\right] \\
R_e = diag\left\{k_{rf} \cdot diag\left[\frac{1}{maxexp\left(u_{e_i}\right)^2}\right], k_{rs} \cdot diag\left[\frac{1}{maxexp\left(u_{e_i}\right)^2}\right]\right\}
\end{gathered}
\tag{6.4-24}
$$

The scalar parameter set $k = \{q_d, k_{Qr}, k_{Qe}, k_{Qa}, k_{Re}, k_{Rs}, k_{Rf}\}$ allows for tuning of relative weights in the cost function as permissible, subject to the conditions in (6.4-20).

The controller was implemented in a MATLAB Simulink simulation environment utilizing the model given in (6.4-10). The Simulink model is summarized in Figure 78.

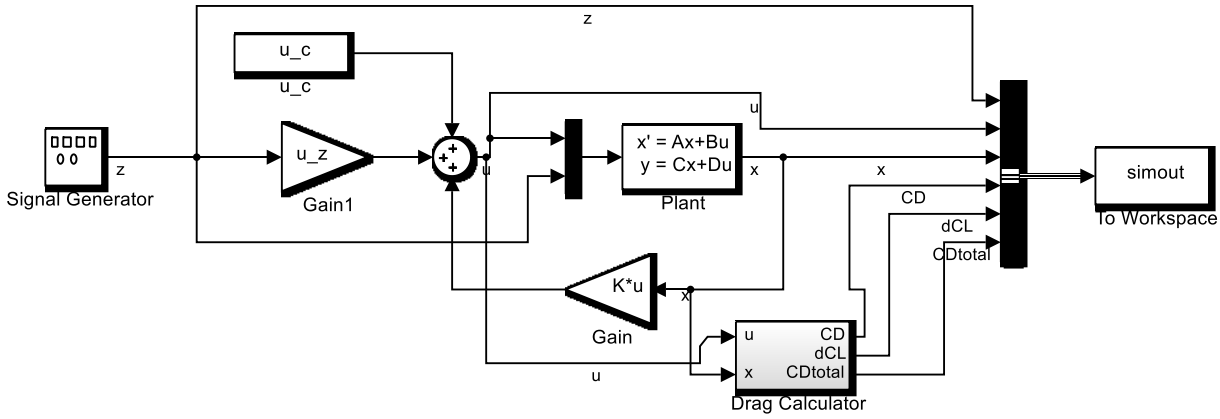


Figure 78. Branch-Generation System

6.4.5 Results

The CL vs CD curve for symmetric uniform flap deflections are shown in Figure 79. The minimum CL/CD point and trim CL/CD points given fuel loads of 20%, 50%, 80%, and 100% are highlighted. Uniformly adjusting the flap settings shifts the CL/CD curves and the resulting operating points as shown in Figure 80.

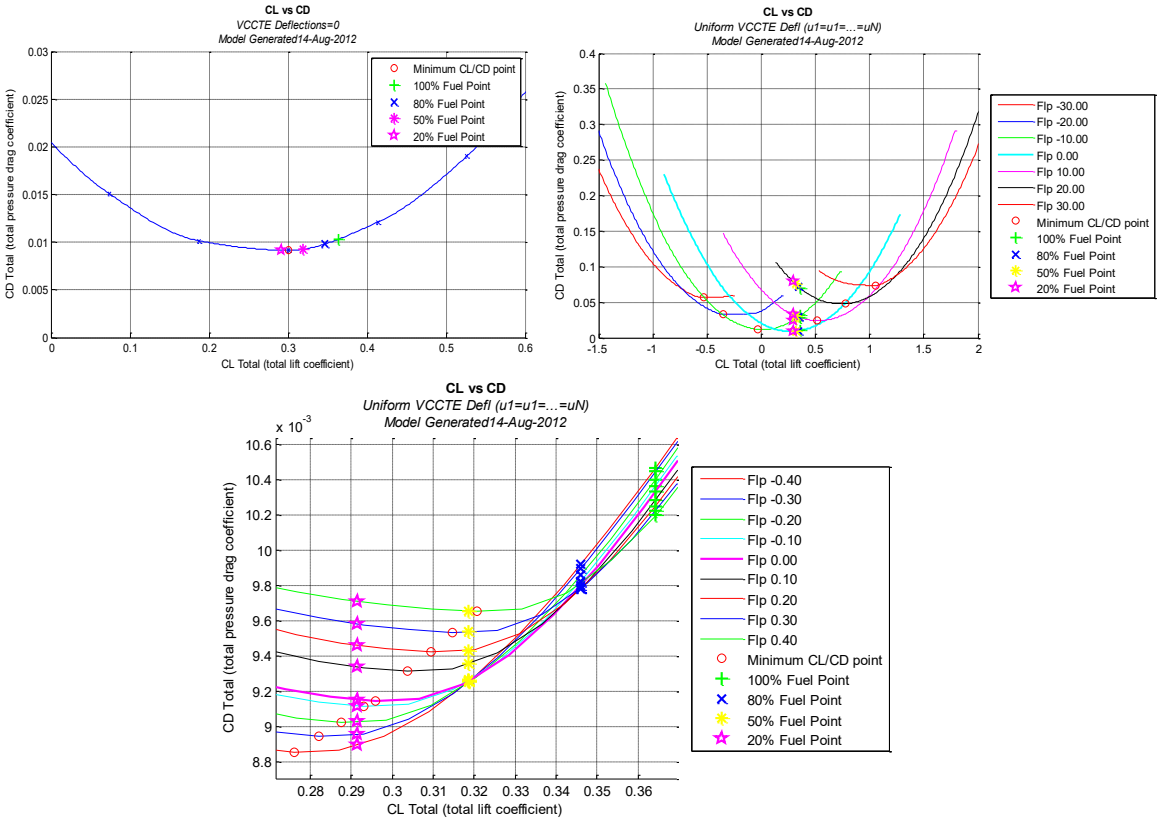


Figure 79. VCCTEF Drag Polars.

Drag polar at zero deflection (top-left). Uniform symmetric flap deployment (top-right). Symmetric deployment over small flap angles (bottom).

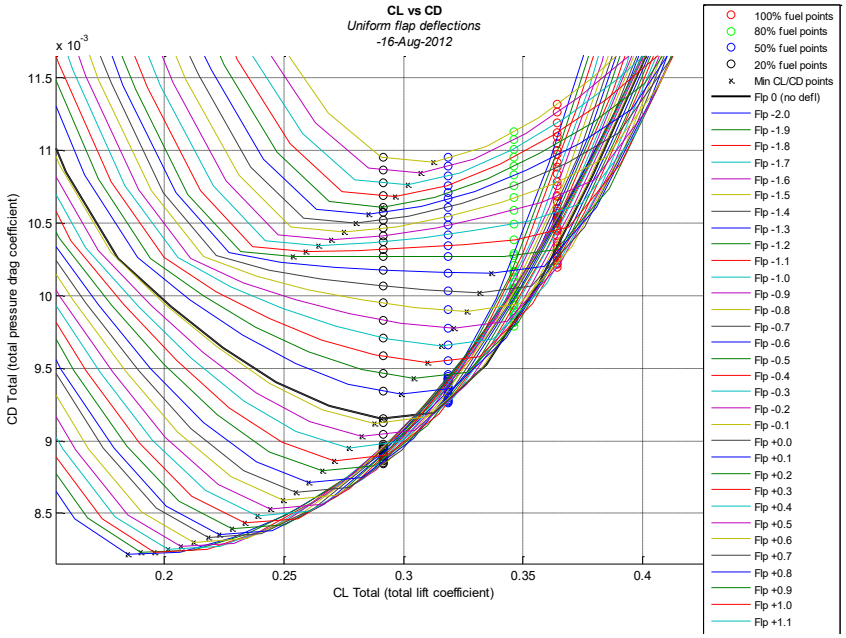


Figure 80. CL vs CD Effect of Small VCCTEF Deflections.
Uniform symmetric flap deflection from -2.0 to +2.0 degrees.

6.4.6 Effect of Uniform VCCTEF Deflections

The effect of symmetric uniform VCCTEF flap deflection on the total aircraft dynamics are shown in the following figures.

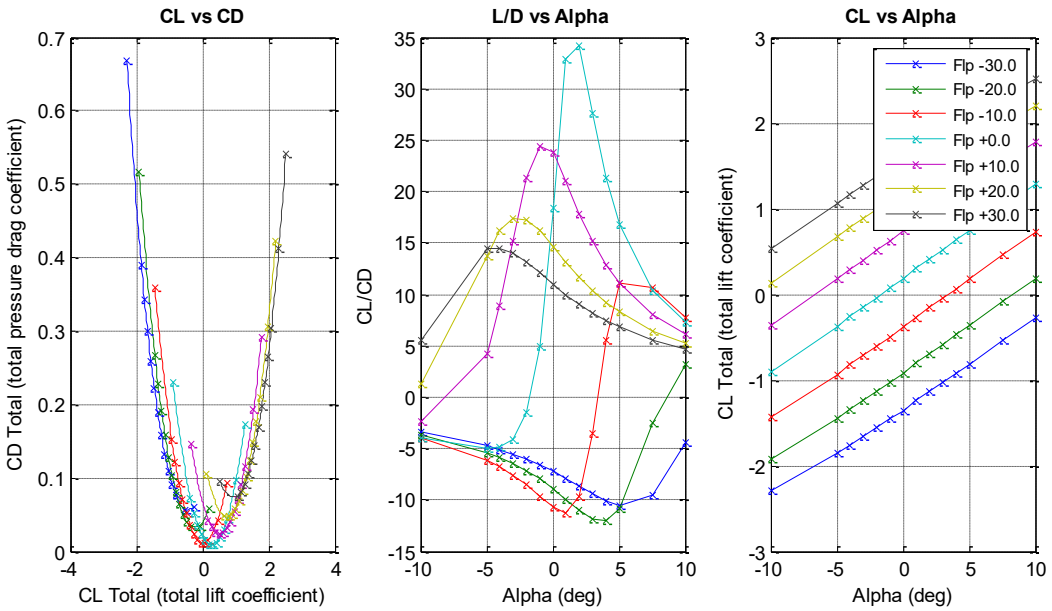


Figure 81. Effect of Uniform VCCTEF Deflection.

CL vs CD, L/D, and CL-alpha plots shown for flap deflections from -30 to 30 with steps of 10 deg.

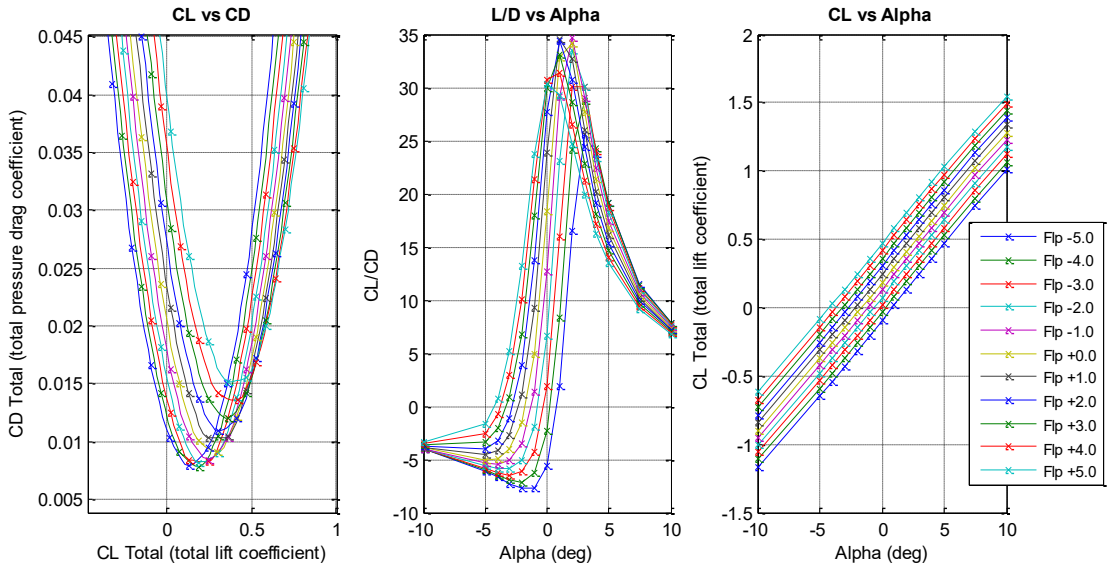


Figure 82. CL vs CD, L/D, and CL.

Flap deflections from -5 to 5 with steps of 1 deg.

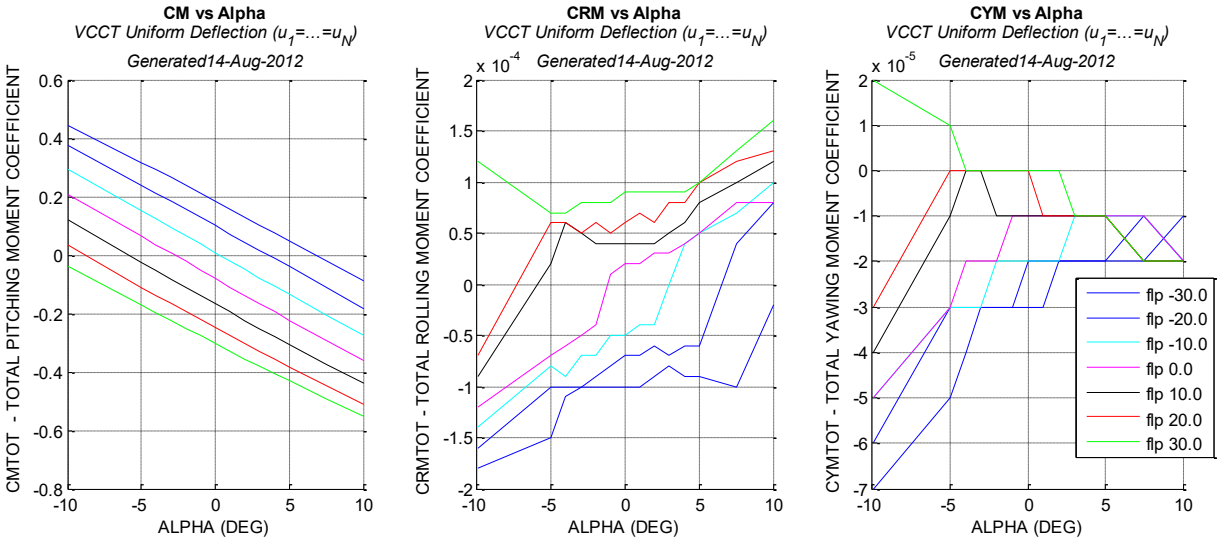


Figure 83. Pitching, Rolling, and Yawing Moment Coefficients.

Coefficients plotted versus alpha for -30 to 30 deg. flap deflections. All VCCTE flaps are at the same deflection angle, with positive being trailing edge down.

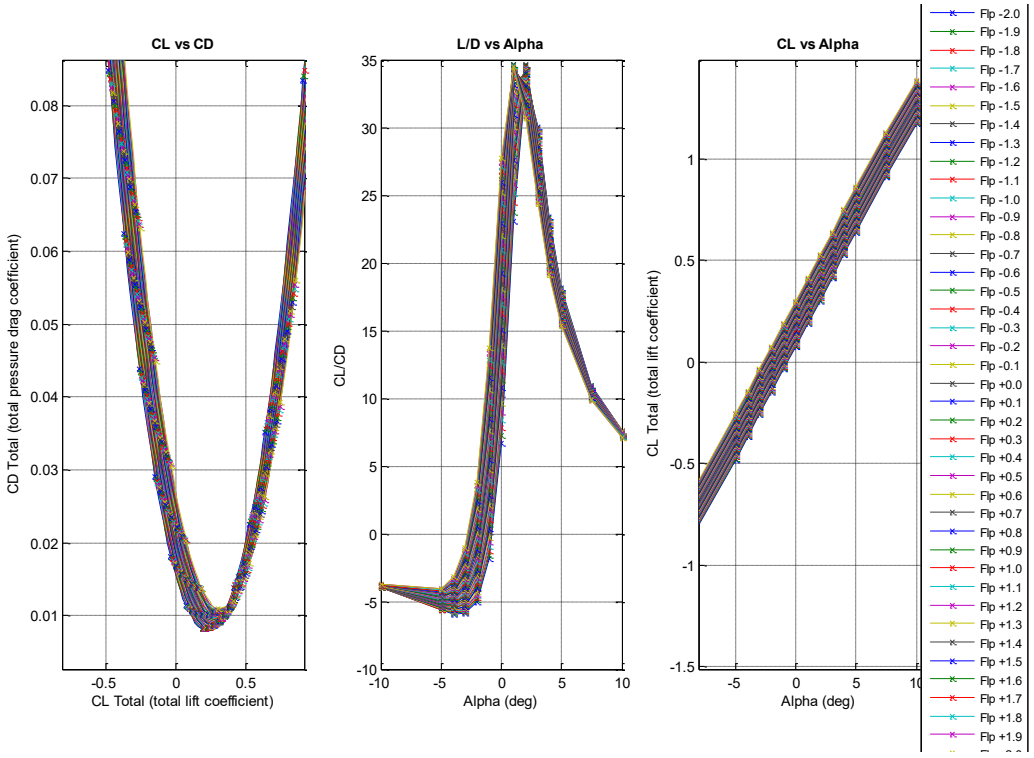


Figure 84. Lift Drag Summary.

Uniform deflections from -2.0 to +2.0 degrees.

6.4.7 Comparison of Parametric VCCTEF Model to Baseline GTM Model

The conversion of the baseline GTM geometry into a parametric model and replacing the conventional control surfaces with a smoothly varying VCCTEF geometry results in mesh variations between the original and resulting geometries. The differences between subsequent VORLAX analysis comparisons between the two models arise due to two sources: (1) model discrepancy errors introduced in the process of converting the static geometry to a parametric model, and (2) improved aerodynamic efficiency of VCCTEF geometry over fixed conventional control surfaces (e.g., smooth contours blending the wing and control surface geometries, removal of gaps). The variations are highlighted in Figure 85 below. The drag polar shows decent correlation, but the parametric model results in a noticeable lift improvement, as seen in the shifted CL-alpha curve, and improved L/D efficiency at small angles of attack.

Unfortunately, it is difficult to determine precisely what percentage of efficiency improvements shown in Figure 85 is due to the parametric conversion as opposed to model variations due to the parametric conversion process. Due to this difficulty, this assessment only compares drag improvements of a VCCTEF configuration against the undeformed parametric VCCTEF, rather than comparing against the original GTM geometry. Note that comparison against the original GTM would result in greater performance improvement.

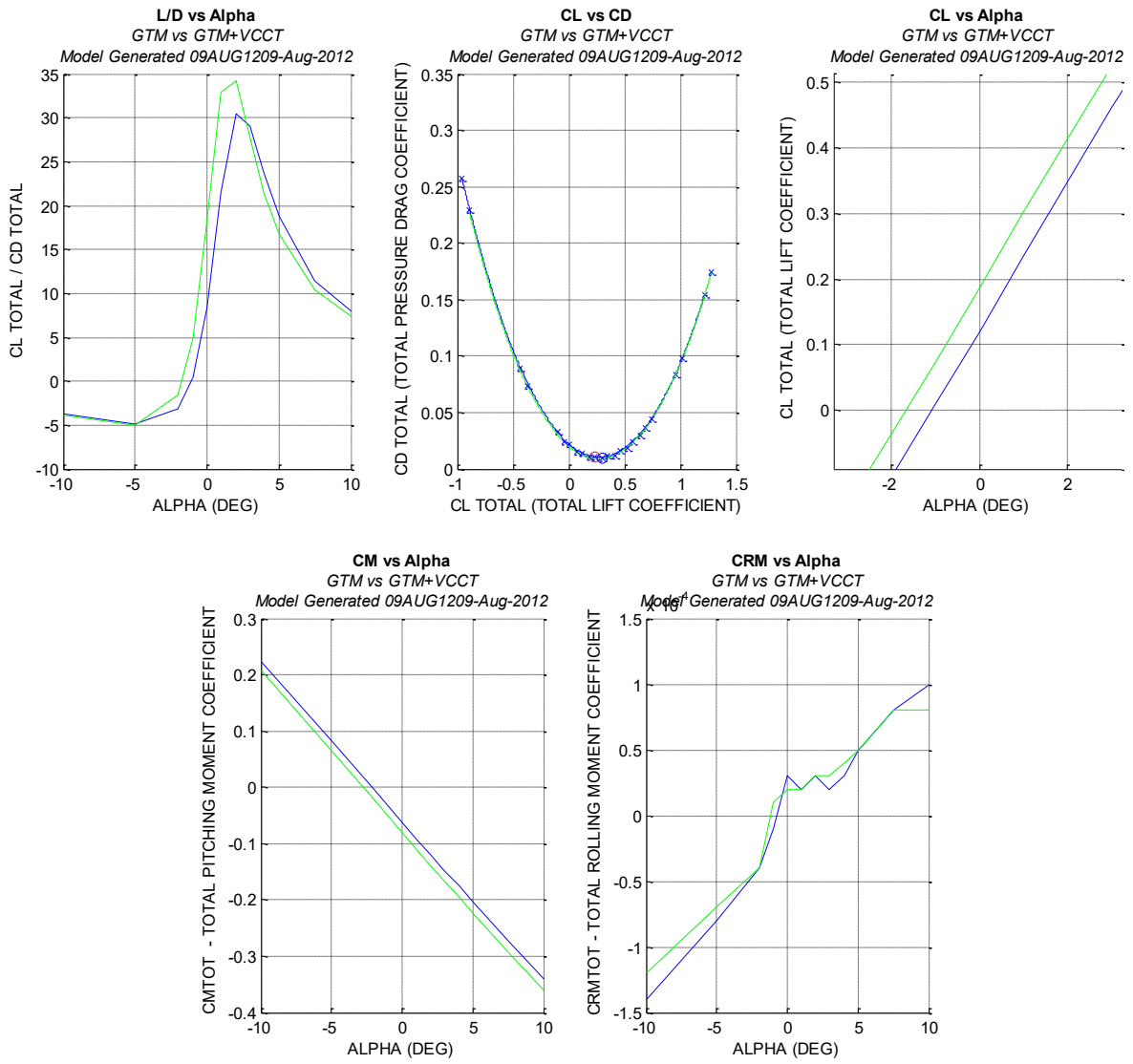


Figure 85. Comparison of Parametric VCCTEF Model with Conventional GTM Model.
Parametric model is shown in green, the original fixed geometry GTM model is shown in blue.

6.5 Intelligent Integrated Building Control System (IIBCS)

6.5.1 Introduction

Buildings are a large, pervasive and integral part of modern culture, with an estimated 128 million residential housing units and nearly 4.5 million office buildings in the United States alone. The building industry is a significant portion of the nation's economy, comprising the nation's largest manufacturing activity, more than 50% of the nation's wealth, and more than 13% of gross domestic product (United States Department of Energy 2009) (United States Department of Housing and Urban Development 2009). Unfortunately, building systems have also become a major consumer of natural resources and economic capital, and a major producer of waste, pollution, and harmful emissions. Buildings account for 39% of the nation's total energy consumption, 72% of total U.S. electricity demand, 12% of water use, and 39% of total U.S. CO2 emissions (United States Department of Housing and Urban Development 2009) (United States Environmental Protection Agency 2009). Given the large and distributed nature of buildings in our society, small improvements in efficiencies per building would yield large overall benefits given the large-scale distributed nature of buildings throughout the world.

Similar to next generation aerospace systems and spurred by advances in technology, building control systems are increasing in sophistication, functionality, and capability. Modern buildings rely heavily on information technology infrastructures, featuring wireless distributed networks of sensors and controllers, internet-enabled interfaces, dedicated database clusters, and dedicated networks linking multiple components of a building control system. Sensors are proliferating in modern buildings, providing greater numbers of sensor observations at higher levels of precision that previously available. Buildings actuators are increasingly being automated; actuated window panes automatically open and close for thermal comfort and air quality; window shades automatically deploy in response to ambient conditions; multiple lighting systems control lighting at the individual, room, zone, and building level; and ventilation systems, air exchangers, under-floor plenum systems, and ceiling fans are similarly trending towards greater automation. New technologies are being developed for building systems that provide alternative methods for control, including geothermal systems and radiant heating and cooling panels.

Despite the increasing sophistication in the state of the art, building control systems are not optimized for system efficiency or performance. Building control systems are composed of several independently designed and operated control systems that are dedicated to a particular building sub-system function. Control objectives, such as temperature set-points, are set from a preprogrammed schedule. Each controller is typically single-objective, designed for a low

number of inputs and outputs, does not intercommunicate, does not consider efficiency or performance in their feedback objective, implements simple control logic, and does not consider their effects on other building sub-systems. Each controller is not designed for performance or efficiency, and does not consider the effect these independent control systems have interacting in a large, coupled, distributed environment. Designing a control system for integrated building control of modern high performance buildings for optimal efficiency is a challenging endeavor. Office buildings are complex dynamic environments. Modeling heat and airflow alone results in large nonlinear models with significant coupling in dynamics and input actuation, a large number of inputs and outputs, and redundant overlapping control actuation.

6.5.2 Intelligent Integrated Building Control Systems Architecture

The Intelligent Integrated Building Control Systems (IIBCS), part of the NASA Sustainability Base project at NASA Ames Research Center, hypothesizes that high-performance building systems can increase operational efficiency and performance through intelligent, integrated, efficiency-seeking control that is coordinated across traditionally isolated building automation systems (Ippolito 2010). The IIBCS research applies NASA developed technology in multi-objective aerospace vehicle control optimization, but these model-based tools require a moderate fidelity model of the total building system. Savant-ML was developed to meet these needs, using first principles to accurately model the internal state of a building over time as influenced by building control actuators, external conditions, and internal building activity. Savant-ML was designed to service a multi-objective integrated building control optimization engine for optimal efficiency control planning while integrating with distributed networks of sensors to provide real-time building estimation.

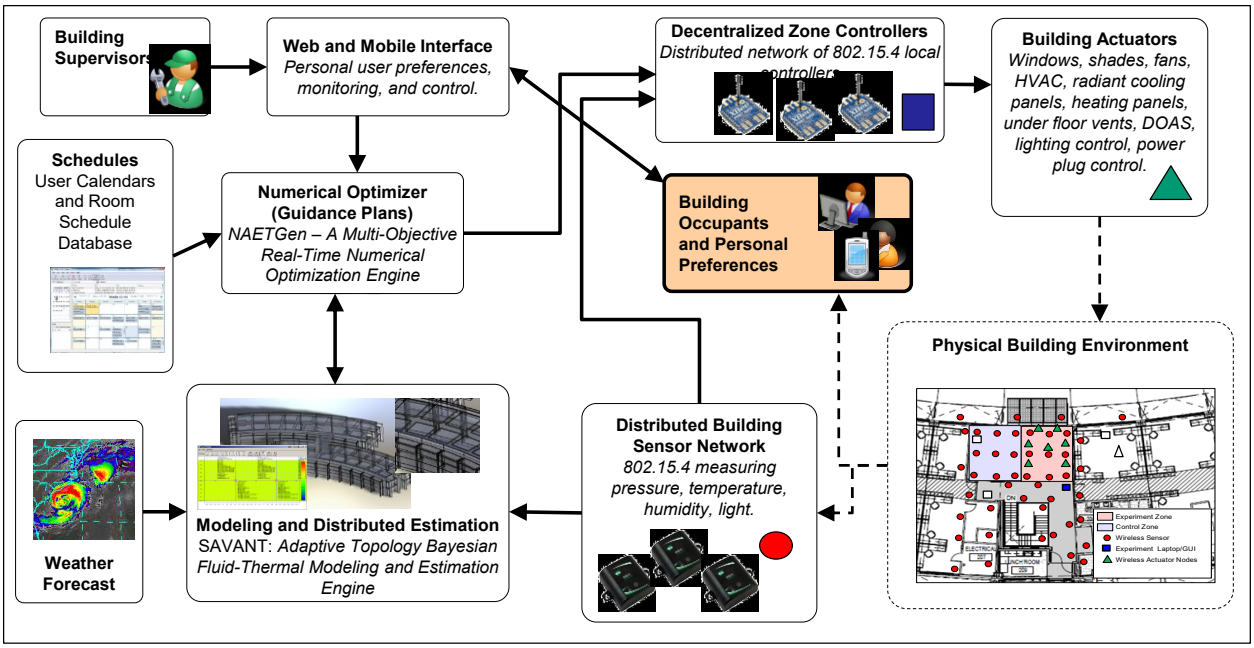


Figure 86. Intelligent Integrated Building Control System (IIBCS) Architecture

The proposed IIBCS control system architecture is shown in Figure 86. A distributed nonlinear thermal/fluid model of building will provide estimation and observation of system states derived from a distributed wireless sensor network. A numerical optimization engine utilizes the model along with weather forecasts to perform forward predictions of control effectiveness and develop trajectories that meets control requirements with optimized efficiency in terms of power usage. These trajectories are utilized by decentralized controllers who locally guide the system to the optimized trajectory plan.

6.5.3 Approach

The IIBCS control system's objective is to minimize overall energy use while satisfying the requirements for controlling the internal environment for occupants measured by temperature, humidity, and light levels. Building control objectives are received from human operators: building operations supervisors provide high level building control objectives, online calendaring database provides occupancy schedules for conference rooms (albeit with large uncertainties), and building occupants provide personal area control input commands (e.g., light levels, temperature adjustment, air flow requests). The control system takes real-time inputs from sensors in building sub-systems, wireless sensor network data, schedules of occupancy, weather data from external web-based sources, data from networked sensors exterior to the building, and input from the human occupants. The control system must generate and optimize plans for controlling the building that optimize objectives and meet constraints, must re-plan when

conditions change, and must provide real-time feedback control input to individual building actuators or higher level commands input into existing control sub-system controllers.

In order to create intelligent buildings that generate substantial energy savings compared to current practice, optimal or adaptive control may be required. Consequently, mathematical models of the complete building system describing air and thermal dynamic interactions are necessary to analyze control effectiveness of multiple heterogeneous actuation choices throughout the building. The model must receive actuation inputs, such as opening and closing individual windows, lowering or raising individual shades, opening or closing interior doors, control of individual light systems, operation of interior fans, effects of opening individual vents from an under floor plenums, cooling/heating thermal panels, time-varying thermal sources in the building (e.g., people and computers), control of dedicated outdoor air systems, outdoor ambient weather conditions, and control of a centralized HVAC system. The model must predict air flow in the building (e.g., effect of opening different combinations of windows in coordination with fans), predict thermal state, comfort level of occupants, predict lighting conditions, humidity and air quality. We require models appropriate for real-time estimation of building states from distributed networks of sensors, models for real-time control development, and models for planning and scheduling. Models must be amenable to automated interrogation and allow rapid evaluation in the algorithmic numerical solver described in (Ippolito and Yeh, 2009) (Ippolito, 2010); this solver searches the input space through guided stochastic search, requiring evaluation of thousands of evaluations.

There is a large body of literature on building modeling and simulation for a wide number of applications, a full survey is beyond the scope of this section. While the authors were unable to find an exact building modeling approach to meet the requirements for IIBCS and integrated full building control, our system builds on many different approaches. Building energy performance models have been developed and used for decades in the building industry and have achieved a high degree of maturation and sophistication (Crawleya, et al. 2008). These models often utilize high order thermal models for steady-state analysis over long time periods to evaluate building performance in the building design industry, and are useful in green-building certification (Newsham, Mancini and Birt 2009). Unfortunately, these models often lacking air flow analysis, evaluates only long-term steady-state, produce models that are very high order with unknown parameters, or cannot be integrated into real-time control involving multiple simultaneous system input (see survey in Crawleya, et al. 2008). Parameter estimation from sensors for model validation is of recent interest in this field (Newsham, Mancini and Birt 2009). Computational fluid dynamics (CFD)

is often utilized for analyzing smaller areas in a room or around sensitive hardware, such as studying temperature in room with rack-mounted servers (Perez-Lombard, et al. 2009). Unfortunately, CFD produces large-order models even in these limited cases, requiring intensive computational operations over fine grid tessellations to achieve accurate solutions, and do not easily handle dynamic environments well. Many approaches to advanced building thermal control have been proposed in the literature (Crawleya, et al. 2008). Mathematical analysis of the thermal behavior of a building generally results in nonlinear models that may widely differ from one building to another (Perez-Lombard, et al. 2009), and achieving better energy performance may require some form of optimal or adaptive control approach (Crawleya, et al. 2008). Many methods for thermal control have been proposed, include optimal, adaptive, neural network, and fuzzy logic (Dounis and Caraiscos 2008), (Freirea, Oliveirab and Mendesc 2008). Unfortunately, control models often focus on low order models of specific functionality required by HVAC systems, such the thermal response of a room to a small number of inputs (Xie, et al. 2011). Zhenhua presents a building model consisting of four essential components: a thermal power network, a DC electric power network, an AC power network and a smart energy management network (Jiang and Rahimi-Eichi 2009). Zhang propose an electrical circuit thermal model inspired by parsimonious lumped parameter models used to infer the properties of IC packages, and statistical techniques estimate the electrical system parameters in response to actual data (Perez-Lombard, et al. 2009). Unfortunately, these modeling approaches do not meet the requirements for integrated building control for various reasons, including the inability to easily add additional coupled dynamic phenomena, inability to accommodate changing environments, and inability to scale to simulate a complete and integrated building system.

6.5.4 Graph-Based Thermal and Fluid Dynamics Model

The following section presents a novel formulation of a discretized model of a building system for state observation and control optimization in the IIBCS architecture. Our approach utilizes the Navier-Stokes equations to model air flow coupled with a thermal formulation to model heat transfer. The governing equations are discretized using a finite volume approach that achieves a low-order model capable of incorporating into a real-time online optimization engine. The graphical model adaptively decomposes to the level of granularity needed for a given accuracy. To provide online state estimation, we propose a stochastic formulation extension thorough a probabilistic graphical model on top of the finite-volume implementation, allowing real-time sensor feedback from a wireless sensor network. We derive an analytical model from the derivation for model parameters identification, and present preliminary experimental results.

This system has been derived and implemented in a C/C++ API named the Savant-ML. Savant-ML allows users to build component-based models of full building systems for modeling, estimation, planning and control. Building systems are simulated through non-uniform volumetric spatial partitions which are assembled from primitive volumes. Spatial volumes can represent static objects (e.g., open air space and walls) and dynamic objects (e.g., machines and building occupants). Components are modeled as vertices and assembled in a variable-fidelity dynamic graph data structure. Edges represent fluid and thermal interfaces between vertices. Air flow dynamics are derived from an incompressible viscous finite-volume discretization scheme of the Navier-Stokes equations with thermal interactions.

Generally, model development begins with geometry or floor plan of the building system to be evaluated. For example, the floor plan of a NASA/CMU experimental test building currently being developed at NASA is shown in Figure 87. Walls and internal spaces have been modeled to provide a spatial partition of the environment, and the walls and air volumes have been dynamically sub-partitioned.

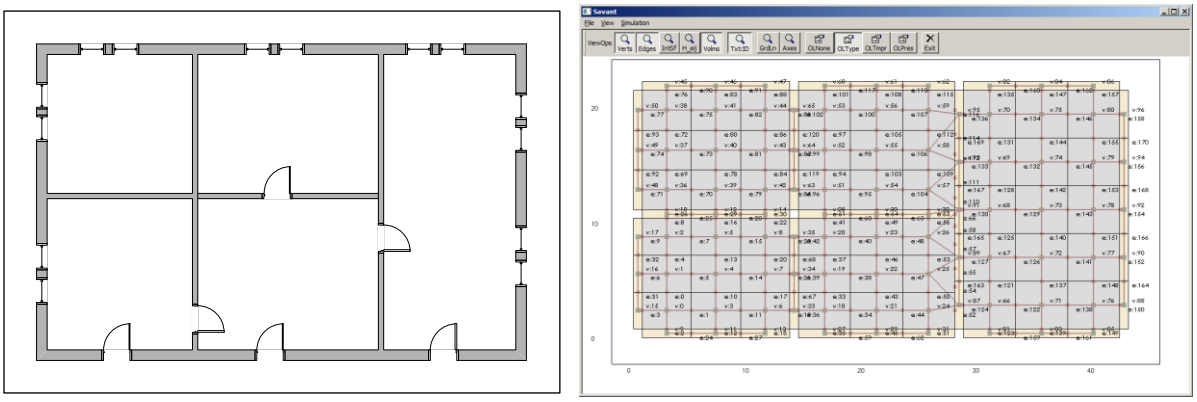


Figure 87. Floorplan of NASA/CMU Test Building (left) and the SAVANT-ML Graphical Model (right)

6.5.5 Arbitrary Axis-Aligned Control Volume (AACV) Graph

A spatial partition can be transformed into a undirected graph through a straightforward transformation process. Consider the following spatial partitioning illustrated in Figure 88, with a small room composed of 7 ‘air’ control volumes and a number of ‘wall’ control volumes. Each control volume is defined by a cube whose sides are aligned with the world coordinate axis system. The spatial decomposition can be represented by an equivalent graph representation, as illustrated in the example in Figure 88, and can be formally defined as follows.

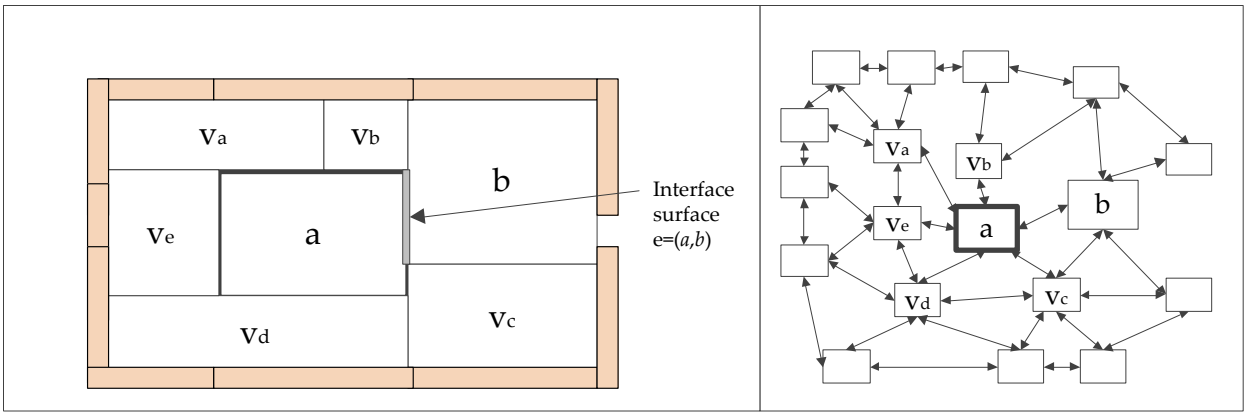


Figure 88. Spatial Partition and AACV Graph.
A spatial partition in 2D Cartesian coordinates (left) and the corresponding graph representation (right). Here, $\boldsymbol{a} \in \boldsymbol{V}(\boldsymbol{G}), \boldsymbol{N_e}(\boldsymbol{a}) = \{\boldsymbol{b}, \boldsymbol{v_a}, .., \boldsymbol{v_e}\}$

The following section defines the underlying graphical model used to represent a building system. For the spatial mapping, let $\boldsymbol{\Psi} \subset \mathbb{R}^3$ be a partition of a bounded compact continuous region in Euclidean space. Let $\boldsymbol{\mathcal{V}} = \{v_1 .. v_n\}$ be a set of connected finite bounded regions $v \subset \boldsymbol{\Psi}$ such that (1) $\boldsymbol{V}$ covers $\boldsymbol{\Psi}$, and (2) for any $u, v \in \boldsymbol{\mathcal{V}}$ if $u \cap v \neq \emptyset$ then $(u \cap v) \in \boldsymbol{\mathcal{E}}$ is the set of finite two-dimensional interface surfaces. Define a mapping $e: \boldsymbol{\mathcal{V}} \times \boldsymbol{\mathcal{V}} \to \boldsymbol{\mathcal{E}}$, such that $e(u, v) = u \cap v$, then the $\boldsymbol{\mathcal{E}}$ can be defined as $\boldsymbol{\mathcal{E}} = \{e(u, v,)\}$ for all $u, v \in \boldsymbol{\mathcal{V}}$ such that $e(u, v) \neq \emptyset$. We refer to $e \in \boldsymbol{\mathcal{E}}$ as interface surfaces, which are conduits for dynamic coupling between neighboring partitions. Let $\mathcal{V}(v): V \to \mathbb{R}$ be a mapping defined as $\mathcal{V}(v) \equiv \iiint_{\widetilde{R_v}} d\widetilde{\boldsymbol{R_v}}$ that returns the total volume inside the control volume, which is assumed to be constant over the length of integration.

The graph is defined based on this spatial definition. Let a set of vertices $\boldsymbol{V} = \{v_1 .. v_{|\boldsymbol{V}|}\}$ be defined where v_i has a one-to-one mapping $\mathcal{v}: \boldsymbol{V} \rightarrowtail \mathcal{V}$. Define a set of edges $\boldsymbol{E} = (\boldsymbol{V}, \boldsymbol{V}) = \{e_1 .. e_{|\boldsymbol{E}|}\}$ where $e_i \in \boldsymbol{E}$ has a one-to-one mapping $e: \boldsymbol{E} \rightarrowtail \boldsymbol{\mathcal{E}}$, such that for all $e = (u, v) \in E$, there exists $\tilde{e} = (\mathcal{v}(u), \mathcal{v}(v)) \in \boldsymbol{\mathcal{E}}$. Define the control volume graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$. Then G defines a standard undirected graph, where vertices $v \in \mathbf{V}$ are arbitrary axis-aligned control volumes and edges $e \in \mathbf{E}$ are interface surfaces between two adjacent vertices. Let $n_v \equiv |\mathbf{V}|$ be the number of control volumes in the system. For any $v \in \mathbf{V}$, let $\mathbf{N}_e(v) \subset \mathbf{V}$ be the set of all neighbors of vertex v, such that $w \in N_e(v) \Leftrightarrow \exists\, (w, v) \in \mathbf{E}$.

In Savant-ML's AACV scheme, control volumes that are part of the spatial partition graph may have several boundaries as shown in Figure 89. Edge boundaries exist between two nodes and represent an interface for transfer of conserved properties. Edge boundaries differ in behavior depending on the vertices they connect. 'Air to Air' edge

boundaries allow free movement of mass, momentum, and energy between vertices. 'Air to Wall' boundaries allow energy transfer including convective heat transfer as a function of flow rates. 'Wall to Wall' boundaries allow energy transfer between solid wall elements in the model. Boundaries can have variable parameters to simulate control actuation, such as windows, doors, and fans. In the absence of a bordering volume, nodes are assumed to have an implicit boundary. We impose adiabatic inviscid tangent flow conditions on the virtual boundaries, allowing no velocity component normal to the wall and no energy flux through the virtual wall.

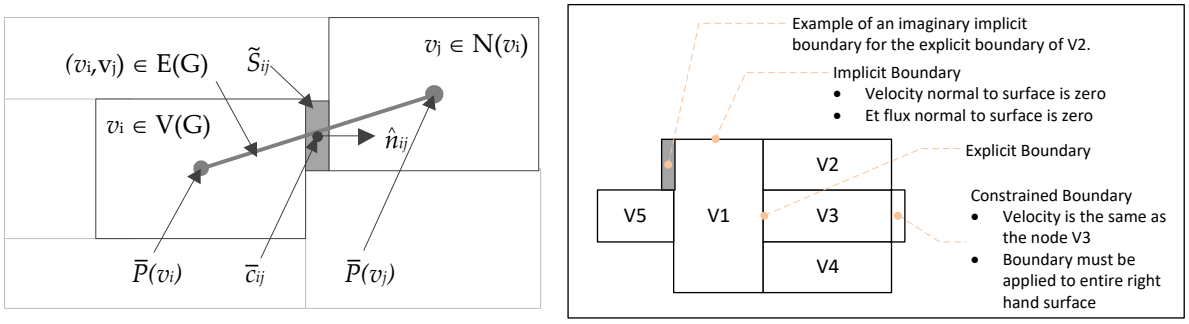


Figure 89. Control Volume Definition (left) and Control Volume Boundaries (right)

An 'air to air' boundary $b_{ij} \in B_i$ defines the interface between two vertices $v_i, v_j \in V(G)$, where $b_{ij} = e(i,j)$, $e(i,j) \in E(G)$. Let the exterior surface of vertex v_i be given by $\tilde{S}_i$. Let a boundary b_j on vertex v_i represent a surface $\tilde{S}_{ij}$ that is a contiguous portion of the exterior surface of control volume i, $\tilde{S}_{ij} \subset \tilde{S}_i$, as represented in Figure 90. Let B_i be the *disjoint covering set* of all boundaries on volume i such that the entire surface area around vertex i is covered by the disjoint boundaries in B_i. Specifically, B_i has the following properties.

1. $\tilde{S}_{ij} \in \tilde{S}_i$ for all $b_j \in B_i$ — Boundary set of v_i
2. $\hat{n}_{ij}$ is constant along any $b_j \in B_i$ — Boundaries are flat (no curvature)
3. $\tilde{S}_i = \bigcup_{j=1}^{|B_i|} \tilde{S}_{ij} = \tilde{S}_{i1} \cup \tilde{S}_{i2} \cup \ldots \cup \tilde{S}_{i|B_i|}$ — Coverage property of boundary set
4. $\tilde{S}_{ij} \cap \tilde{S}_{ik} = \emptyset$ for any $\tilde{S}_{ij} \neq \tilde{S}_{ik}$ where $\tilde{S}_{ij}, \tilde{S}_{ik} \in B_i$, — Disjoint property of boundary set

6.5.6 Dynamics Derivation

A representative control volume and graph geometry definitions are shown in Figure 90.

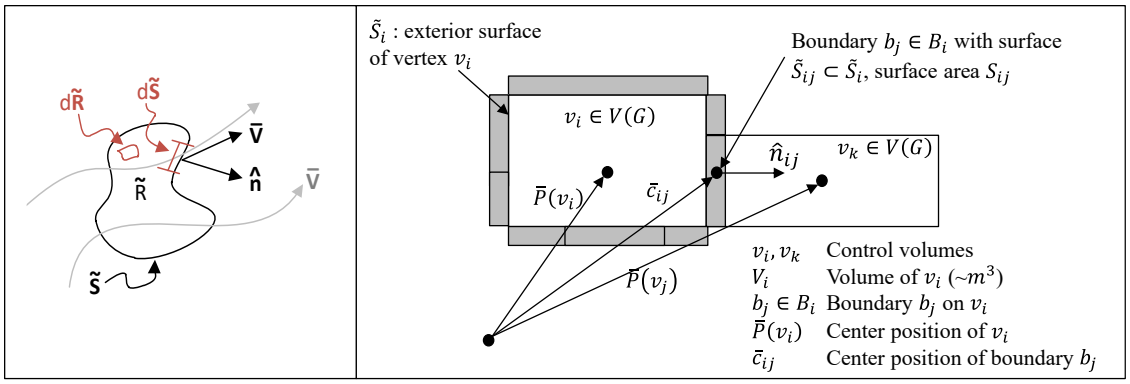


Figure 90. Conservation Law Control Volume (left) and Graph Geometry Definitions (right)

The Navier-Stokes equations given in conservation law are summarized as follows.

$$\frac{\partial}{\partial t}\mathbf{X} = -(\nabla \cdot \mathbf{H}) + \boldsymbol{Q}$$
$$\mathbf{H} = (\mathbf{E}_\mathrm{I} - \mathbf{E}_\mathrm{V})\hat{\imath} + (\mathbf{F}_\mathrm{I} - \mathbf{F}_\mathrm{V})\hat{\jmath} + (\mathbf{G}_\mathrm{I} - \mathbf{G}_\mathrm{V})\hat{k} \tag{6.5-8}$$

The flux vectors **E**, **F**, **G** are split into inviscid and viscous components, given by

$$\mathbf{X} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E_t \end{bmatrix} \; ; \; \mathbf{E}_\mathrm{I} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ (E_t + p)u + k\frac{\partial T}{\partial x} \end{bmatrix} ; \mathbf{F}_\mathrm{I} = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ (E_t + p)v + k\frac{\partial T}{\partial y} \end{bmatrix} ; \mathbf{G}_\mathrm{I} = \begin{bmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ (E_t + p)w + k\frac{\partial T}{\partial z} \end{bmatrix}$$
$$\mathbf{E}_V = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \end{bmatrix} ; \mathbf{F}_\mathrm{V} = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{xy} + v\tau_{yy} + w\tau_{yz} \end{bmatrix} \; ; \; \mathbf{G}_\mathrm{V} = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ u\tau_{xz} + v\tau_{yz} + w\tau_{zz} \end{bmatrix} \tag{6.5-9}$$

To establish relationships between our thermodynamic primitive variables (p, ρ, T, e, h) and the conserved variables, we assume a calorically perfect gas with constant specific heats.

$$\mathrm{P} = \rho\mathrm{RT} \; ; \; \mathrm{T} = \frac{\gamma - 1}{\mathrm{R}}\left(\frac{\mathrm{E_t}}{\rho} - \frac{1}{2}\mathrm{V}^2 - \bar{g} \cdot \bar{r}\right) \tag{6.5-10}$$

6.5.7 Derivation of Dynamics for Graph Volumes

We use an explicit finite-volume scheme for evaluating the dynamics equations. Integrating (6.5-8) over the control volume yields

$$\iiint_{\tilde{R}} \left(\frac{\partial}{\partial t}\mathbf{X}\right) dR = -\iiint_{\tilde{R}} (\nabla \cdot \mathbf{H} + \mathbf{Q}) dR \tag{6.5-11}$$

Considering Savant-ML's finite control volume spatial partition, for the i'th volume, let $\mathbf{X}_i(t)$ represent the state vector for that volume. Assuming that the state vector $\mathbf{X}_i$ is uniform over the control volume, $\boldsymbol{Q}_i$ is uniform and constant. Applying the divergence theorem yields

$$\frac{\partial}{\partial t}\mathbf{X}_i(t) = -\frac{1}{\mathrm{V_i}} \oiint_{\tilde{S}_i} (\mathbf{H} \cdot \hat{\mathbf{n}}) d\tilde{\boldsymbol{S}} - \frac{1}{\mathrm{V_i}} \boldsymbol{Q}_i(t) \tag{6.5-12}$$

Here $\tilde{S}_i$ is the closed surface surrounding vertex v_i, V_i is the volume of v_i, and $\hat{\mathbf{n}}$ is the surface normal that varies over $\tilde{S}_i$. We define $\tilde{S}_i$ to be composed of a finite set of piecewise continuous boundary surface segments $\tilde{S}_i = (\tilde{S}_{ij})$ with constant surface normal $\hat{\mathbf{n}}_{ij}$across the boundary segment.

6.5.8 Implicit Virtual Boundary Assumption

We propose without loss of generality that each piecewise continuous boundary surface segment $\tilde{\boldsymbol{S}}_{ij}$ can be modeled as being opposed by a virtual boundary $\tilde{\tilde{S}}_{ij}$ with equal area and opposite surface normal. We impose adiabatic invisicid tangent flow conditions on the virtual boundaries (no velocity component normal to the wall, no energy flux through the virtual wall). Let $\tilde{\tilde{\mathbf{H}}}_{ij}$ be the flux vector at the virtual boundary for a surface $\tilde{S}_{ij}$, then

$$\left(\tilde{\tilde{\mathbf{H}}}_{ij} \cdot \hat{\imath}\right) = \begin{bmatrix} 0 \\ p_i \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad ; \quad \left(\tilde{\tilde{\mathbf{H}}}_{ij} \cdot \hat{\jmath}\right) = \begin{bmatrix} 0 \\ 0 \\ p_i \\ 0 \\ 0 \end{bmatrix} \; ; \; \left(\tilde{\tilde{\mathbf{H}}}_{ij} \cdot \hat{k}\right) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ p_i \\ 0 \end{bmatrix} \tag{6.5-13}$$

Let $\mathrm{B_i^{cover}}$ be defined as the set of boundaries which allow ($\mathrm{B_i} \cup \mathrm{B_i^{cover}}$) to cover vertex $\mathrm{V_i}$. We assume implicit boundary conditions at the segments in $\mathrm{B_i^{cover}}$.

$$\mathrm{V_i} \frac{\partial}{\partial t}\mathbf{X}_i(t) = -\sum_{\mathrm{b_j} \in \left(\mathrm{B_i} \cup \mathrm{B_i^{cover}}\right)} \left(\iint_{\tilde{S}_{ij}} \left(\mathbf{H}(\xi, t) \cdot \hat{\mathbf{n}}_{ij}(\xi)\right) d\xi \right) + \boldsymbol{Q}_i(t) \tag{6.5-14}$$

Here $\xi \in \mathbb{R}^3$ is a spatial coordinate in $\mathbb{R}^3$ that is integrated over the surface $\tilde{S}_{ij}$. Over each boundary segment b_j, we assume $\mathbf{H}_{ij}$ is uniform across the segment, the normal $\hat{\mathbf{n}}_{ij}$ is fixed and points outwards from vertex i, and assume an implicit virtual counter-boundary to b_j with flux $\tilde{\tilde{\mathbf{H}}}_{ij}$.

$$\mathrm{V_i}\frac{\partial}{\partial t}\mathbf{X}_i(t) = -\sum_{\mathrm{b_j}\in\left(\mathrm{B_i}\cup\mathrm{B_i^{cover}}\right)}\left(\iint_{\tilde{\boldsymbol{S}}_{ij}}\left(\mathbf{H}_{\mathrm{ij}}\cdot\hat{\mathbf{n}}_{ij}\right)d\xi - \iint_{\tilde{\boldsymbol{S}}_{ij}}\left(\tilde{\tilde{\mathbf{H}}}_{\mathrm{ij}}\cdot\hat{\mathbf{n}}_{ij}\right)d\xi\right) + \boldsymbol{Q_i}(t)$$

$$\mathrm{V_i}\frac{\partial}{\partial t}\mathbf{X}_i(t) = -\sum_{\mathrm{b_j}\in\mathrm{B_i}}\left(\iint_{\tilde{\boldsymbol{S}}_{ij}}\left(\mathbf{H}_{\mathrm{ij}} - \tilde{\tilde{\mathbf{H}}}_{\mathrm{ij}}\right)\cdot\hat{\mathbf{n}}_{ij}d\xi\right) - \sum_{\mathrm{b_j}\in\mathrm{B_i^{cover}}}\left(\iint_{\tilde{\boldsymbol{S}}_{ij}}\left(\mathbf{H}_{\mathrm{ij}} - \tilde{\tilde{\mathbf{H}}}_{\mathrm{ij}}\right)\cdot\hat{\mathbf{n}}_{ij}d\xi\right) + \boldsymbol{Q_i}(t)$$

$$\mathrm{V_i}\frac{\partial}{\partial t}\mathbf{X}_i(t) = -\sum_{\mathrm{b_j}\in\mathrm{B_i}}\left[\left(\mathbf{H}_{\mathrm{ij}} - \tilde{\tilde{\mathbf{H}}}_{\mathrm{ij}}\right)\cdot S_{ij}\hat{\mathbf{n}}_{ij}\right] - \sum_{\mathrm{b_k}\in\mathrm{B_k^{cover}}}\left[\left(\mathbf{H}_{\mathrm{ik}} - \tilde{\tilde{\mathbf{H}}}_{\mathrm{ik}}\right)\cdot S_{ik}\hat{\mathbf{n}}_{ik}\right] + \boldsymbol{Q_i}(t) \tag{6.5-15}$$

Since flux over boundaries in $\mathrm{B_i^{cover}}$ has the same conditions as their virtual counter boundaries, then $\left[\mathbf{H}_{\mathrm{ik}} = \tilde{\tilde{\mathbf{H}}}_{\mathrm{ik}}\right]_{\mathrm{b_j}\in\mathrm{B_i^{cover}}}$ and their contribution drops out of the equation, leaving

$$\mathrm{V_i}\frac{\partial}{\partial t}\mathbf{X}_i(t) = -\sum_{\mathrm{b_j}\in\mathrm{B_i}}\left[\left(\mathbf{H}_{\mathrm{ij}} - \tilde{\tilde{\mathbf{H}}}_{\mathrm{ij}}\right)\cdot S_{ij}\hat{\mathbf{n}}_{ij}\right] + \boldsymbol{Q_i}(t) \tag{6.5-16}$$

For implementation, the model equations can now be written in the form

$$\frac{d}{dt}\mathbf{X}_i(t) = \sum_{\mathrm{j}\in\mathrm{B_i}}\dot{\boldsymbol{X}}_{ij} + \dot{\mathbf{X}}_{ii} + \dot{\mathbf{X}}_{ui} \tag{6.5-17}$$

where

$$\dot{\boldsymbol{X}}_{ij} = -\frac{1}{\mathrm{V_i}}\left[\left(\mathbf{H}_{\mathrm{ij}} - \tilde{\tilde{\mathbf{H}}}_{\mathrm{ij}}\right)\cdot S_{ij}\hat{\mathbf{n}}_{ij}\right] \quad ; \quad \dot{\mathbf{X}}_{ii} = 0 \quad ; \quad \dot{\mathbf{X}}_{u_i} = \frac{1}{\mathrm{V_i}}\boldsymbol{Q}_i(t) \tag{6.5-18}$$

Each edge j∈ $\mathrm{B_i}$ contributes $\dot{\boldsymbol{X}}_{ij}$ to the derivative of vertex i's state. Let $\Delta p_{ij} \equiv p_{ij} - p_i$ where $\Delta p_{ij} \neq \Delta p_{ji}$, and define $\sigma_{ij} \in \{-1, +1\}$ to be the sign of the normal $\hat{\boldsymbol{n}}$ along the coordinate direction. Then $\dot{\boldsymbol{X}}_{ij}$ can be written as follows.

$$\dot{\boldsymbol{X}}_{ij} = -\frac{\sigma_{\bar{\imath}j}S_{ij}}{\mathrm{V_{\bar{\imath}}}}\left(\begin{bmatrix}\rho_{ij}u_{ij}\\ \rho_{ij}u_{ij}u_{ij}\\ \rho_{ij}v_{ij}u_{ij}\\ \rho_{ij}w_{ij}u_{ij}\\ \left(E_{t_{ij}} + p_{ij}\right)u_{ij}\end{bmatrix} + \begin{bmatrix}0\\ \Delta p_{\bar{\imath}j}\\ 0\\ 0\\ k_{ij}\left(\frac{\partial T}{\partial x}\right)_{\bar{\imath}j}\end{bmatrix} - \mathbf{E}_{v_{ij}}\right) \quad \ldots\ when\ \hat{\mathbf{n}}_{ij} = \sigma_{ij}\hat{\imath}$$

$$\dot{\boldsymbol{X}}_{ij} = -\frac{\sigma_{\bar{\imath}j}S_{ij}}{\mathrm{V_{\bar{\imath}}}}\left(\begin{bmatrix}\rho_{ij}v_{ij}\\ \rho_{ij}u_{ij}v_{ij}\\ \rho_{ij}v_{ij}v_{ij}\\ \rho_{ij}w_{ij}v_{ij}\\ \left(E_{t_{ij}} + p_{ij}\right)v_{ij}\end{bmatrix} + \begin{bmatrix}0\\ 0\\ \Delta p_{\bar{\imath}j}\\ 0\\ k_{ij}\left(\frac{\partial T}{\partial y}\right)_{\bar{\imath}j}\end{bmatrix} - \mathbf{F}_{v_{ij}}\right) \quad \ldots\ when\ \hat{\mathbf{n}}_{ij} = \sigma_{ij}\hat{\jmath} \tag{6.5-19}$$

$$\dot{\boldsymbol{X}}_{ij} = -\frac{\sigma_{\bar{\imath}j} S_{ij}}{\mathrm{V}_{\bar{\imath}}} \left(\begin{bmatrix} \rho_{ij} w_{ij} \\ \rho_{ij} u_{ij} w_{ij} \\ \rho_{ij} v_{ij} w_{ij} \\ \rho_{ij} w_{ij} w_{ij} \\ \left(E_{t_{ij}} + p_{ij}\right) w_{ij} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \Delta p_{\bar{\imath}j} \\ k_{ij} \left(\frac{\partial T}{\partial z}\right)_{\bar{\imath}j} \end{bmatrix} - \mathbf{G}_{v_{ij}} \right) \quad \ldots \textit{ when } \hat{\mathbf{n}}_{ij} = \sigma_{ij} \hat{k}$$

For illustrative purposes, an overbar on the vertex index subscripts was added to highlight parameters that are not symmetric with respect to the vertex index order (e.g., $\sigma_{\bar{\imath}j} \neq \sigma_{\bar{\jmath}i}$ is not symmetric, but $S_{ij} = S_{ji}$ is symmetric). Assuming the second coefficient of viscosity is negligible, the viscous stress flux vectors are given by

$$\mathbf{E}_V = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \end{bmatrix}; \mathbf{F}_\mathrm{V} = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{xx} \\ \tau_{yz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \end{bmatrix} \; ; \; \mathbf{G}_\mathrm{V} = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{xx} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} \end{bmatrix}$$

$$\tau_{xx_{ij}} = 2\mu_{ij} \frac{\Delta \mathrm{u}_{\bar{\imath}j}}{\Delta x_{\bar{\imath}j}} \quad ; \quad \tau_{yy_{ij}} = 2\mu_{ij} \frac{\Delta v_{\bar{\imath}j}}{\Delta y_{\bar{\imath}j}} \quad ; \quad \tau_{zz_{ij}} = 2\mu_{ij} \frac{\Delta w_{\bar{\imath}j}}{\Delta z_{\bar{\imath}j}} \quad ; \quad \tau_{xy_{ij}} = \mu_{ij} \left(\frac{\Delta v_{\bar{\imath}j}}{\Delta x_{\bar{\imath}j}} + \frac{\Delta u_{\bar{\imath}j}}{\Delta y_{\bar{\imath}j}} \right) \tag{6.5-20}$$

$$\tau_{xz_{ij}} = \mu_{ij} \left(\frac{\Delta w_{\bar{\imath}j}}{\Delta x_{\bar{\imath}j}} + \frac{\Delta u_{\bar{\imath}j}}{\Delta z_{\bar{\imath}j}} \right) \quad ; \quad \tau_{yz_{ij}} = \mu_{ij} \left(\frac{\Delta w_{\bar{\imath}j}}{\Delta y_{\bar{\imath}j}} + \frac{\Delta v_{\bar{\imath}j}}{\Delta z_{\bar{\imath}j}} \right)$$

6.5.9 Flux Evaluation at Interface Boundaries

The flux at the centroid of the boundary $\mathbf{H}_{ij} = \mathbf{H}\left(\bar{c}_{ij}, t\right)$ must be computed, but the fluid states $\mathbf{X}_i = \mathbf{X}(P(i), t)$ are only maintained at the vertex centroids. Several different methods are currently implemented to evaluate $\mathbf{H}_{ij}$. Let the primitive fluid variables of density, velocity, and pressure be represented by the vector $\boldsymbol{x}_p = [\rho, u, v, w, p]^T$. Define a scaling vector $\alpha \in [0,1]$ (typically $\alpha = 0.5$). The first method averages the primitive values at the centroid of each vertex, and the flux vector is calculated based on the average state

$$\mathbf{H}_{ij} = \mathbf{H}_{ij}\left(\boldsymbol{x}_{\mathrm{p}_{ij}}\right) \textit{ where } \boldsymbol{x}_{p_{ij}} = \boldsymbol{x}_{pi} + \alpha\left(\boldsymbol{x}_{p_j} - \boldsymbol{x}_{pi}\right) \tag{6.5-21}$$

The second method computes the flux from the average of the conserved variables at each vertex center

$$\mathbf{H}_{ij} = \mathbf{H}_{ij}\left(\boldsymbol{X}_{ij}\right) \textit{ where } \boldsymbol{X}_{ij} = \boldsymbol{X}_i + \alpha\left(\boldsymbol{X}_j - \boldsymbol{X}_i\right) \tag{6.5-22}$$

The third method averages the flux computed at each vertex center

$$\mathbf{H}_{ij} = \mathbf{H}_i(\boldsymbol{X}_i) + \alpha\left(\mathbf{H}_j\left(\boldsymbol{X}_j\right) - \mathbf{H}_i(\boldsymbol{X}_i)\right) \tag{6.5-23}$$

The final method combines estimates on the left and right of the interface boundary.

$$\mathbf{H}_{ij} = \mathbf{H}_{ij}^{i} + \alpha\left(\mathbf{H}_{ij}^{j} - \mathbf{H}_{ij}^{i}\right) \; where \; \mathbf{H}_{ij}^{i} = \mathbf{H}\left(\boldsymbol{X}_i + \alpha_i\left(\boldsymbol{X}_{ij} - \boldsymbol{X}_i\right)\right) \; ; \; \mathbf{H}_{ij}^{j} = \mathbf{H}\left(\boldsymbol{X}_{ij} + \alpha_j\left(\boldsymbol{X}_j - \boldsymbol{X}_{ik}\right)\right) \tag{6.5-24}$$

Note that constant valued limiters are applied to the state derivative vector and the state vector after each integration process.

6.5.10 Simulation Update Algorithm and RK4 Integration

The dynamics are given in the following form.

$$\frac{d}{dt}\mathbf{X}_i(t) = \sum_{\mathrm{j}\in\mathrm{B_i}} \dot{\boldsymbol{X}}_{ij} + \dot{\mathbf{X}}_{ii} + \dot{\mathbf{X}}_{ui} \tag{6.5-25}$$

Here, $\mathbf{X}_i$ is the state of node i, $\dot{\boldsymbol{X}}_{ij}$ is the influence of surface e(i,j) on vertex i (from vertex j to vertex i), $\dot{\mathbf{X}}_{ii}$ is internal effects, and $\dot{\mathbf{X}}_{ui}$ are the external inputs.

Savant-ML implements the forward simulation using a two-step update procedure that passes over each edge in the graph then each vertex using a fourth-order Runge-Kutta integration scheme by default. Each vertex contains a set of arrays to store state values and derivatives during the multi-stage integrator. The DERIV_INTEGRATE routine calculates the derivative and integrates the solution, receiving an integrator index that indicates which integration stage the integrator is on and which array location to use to store the results. The INTEGRATE_RK4 routine controls the Runge-Kutta integration process, making repeated calls to the DERIV_INTEGRATE routine.

```
INTEGRATE_RK4 (
        Graph   G=(V,E),    // graph G
        double  t0,         // time at start of derivative
        double  dt,         // step time
        )
DERIV_INTEGRATE ( d=0, G, t0, dt/2)
DERIV_INTEGRATE ( d=1, G, t0+dt, dt/2 )
DERIV_INTEGRATE ( d=2, G, t0+dt, dt )
DERIV_INTEGRATE ( d=3, G, t0+dt )
For each control volume v ∈ V do
        X[v]_{d=0} ← X[v]_{d=0} + (dt/6)(Ẋ[v]_{d=0} + 2Ẋ[v]_{d=1} + 2Ẋ[v]_{d=2} + Ẋ[v]_{d=3})
        COMPUTE_H (H[v]_{d=0}, X[v]_{d=0})
        COMPUTE_AUXVARS(v)
        CLEAR_DXDT_ACCUMS(v)
Next v
```

Figure 91. INTEGRATE_RK4 Algorithm

```
Function DERIV_INTEGRATE (
        Graph   G=(V,E),        // graph G
        int     d               // integrator step count (integrator array index)
        double  t0,             // time at start of derivative
        double  dt              // integration time to step forward
        )
For each edge e = (i, j) ∈ E(G) do
        Compute symmetric ij variables Hij
        Compute non-symmetric ij variables, ΔPij, ΔPji, Ẋij and Ẋji
        Accum e's contribution to vertex i derivative,  Ẋ[i]d ← Ẋ[i]d + Ẋji
        Accum e's contribution to vertex j derivative,  Ẋ[j]d ← Ẋ[j]d + Ẋji
Next e
For each vertex v ∈ V(G) do
        Accum control input contribution, Ẋ[v]d ← Ẋ[v]d + Qv(t)
        Step forward in time by dt, X[v]d+1 ← X[v]d=0 + dt · Ẋ[v]d
        COMPUTE_H(H[v]d+1, X[v]d+1)
Next v
```

Figure 92. DERIVE_INTEGRATE Algorithm

6.5.11 Analysis Model

A derived model was created for analysis, control synthesis, and online parameter identification. The model presented above can be reorganized to separate the internal dynamics from the externally coupled dynamics.

$$\frac{d}{dt}\mathbf{X}_i(t) = \sum_{\mathrm{j}\in\mathrm{N_e}} \frac{S_{ij}}{\mathrm{V_i}}\left[\{(\alpha_{ij}-1)\mathbf{H}_i\}\cdot\hat{\mathbf{n}}_{\boldsymbol{ij}}\right] + \sum_{\mathrm{j}\in\mathrm{N_e}} \frac{-S_{ij}}{\mathrm{V_i}}\left[(\alpha_{ij}\mathbf{H}_j)\cdot\hat{\mathbf{n}}_{\boldsymbol{ij}}\right] - \boldsymbol{Q_i}(t) \tag{6.5-26}$$

This is of the form

$$\frac{d}{dt}\mathbf{X}_i(t) = \sum_{\mathrm{j}\in\mathrm{N_e}} \boldsymbol{f}_{\mathrm{ii}}(\mathbf{X}_\mathrm{i}, t) + \sum_{\mathrm{j}\in\mathrm{N_e}} \boldsymbol{f}_{\mathrm{ij}}(\mathbf{X}_\mathrm{j}, t) + \boldsymbol{g}_\mathrm{i}(\mathbf{U}_\mathrm{i}, t) \tag{6.5-27}$$

where

$$\begin{aligned} \boldsymbol{f}_{ii}(\mathbf{X}_\mathrm{i}, t) &= \sum_{\mathrm{j}\in\mathrm{N_e}} \frac{S_{ij}}{\mathrm{V_i}}\left[\{(\alpha_{ij}-1)\mathbf{H}_i\}\cdot\hat{\mathbf{n}}_{\boldsymbol{ij}}\right] \\ \boldsymbol{f}_{ij}(\boldsymbol{X}_j, t) &= \sum_{\mathrm{j}\in\mathrm{N_e}} \frac{-S_{ij}}{\mathrm{V_i}}\left[(\alpha_{ij}\mathbf{H}_j)\cdot\hat{\mathbf{n}}_{\boldsymbol{ij}}\right] \end{aligned} \tag{6.5-28}$$

In general, $\boldsymbol{f}_{ii}(X_i, t)$, $\boldsymbol{f}_{ij}(X_j, t)$ and $\boldsymbol{g}_i(\mathbf{X}_i, t)$ are nonlinear vector valued functions. For instance, for Euler flows (compressible, inviscid), the functions for $\boldsymbol{f}_{ii}(X_i, t)$ and $\boldsymbol{f}_{ij}(X_j, t)$ become

$$\boldsymbol{f}_{ii}(\mathbf{X}_\mathrm{i}, t) = \frac{S_{ij}}{\mathrm{V_i}}\left[\left(\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ (E_t+p)u+kT_x\end{bmatrix}_i \hat{\imath} + \begin{bmatrix}\rho v\\ \rho vu\\ \rho v^2+p\\ \rho vw\\ (E_t+p)v+kT_y\end{bmatrix}_i \hat{\jmath} + \begin{bmatrix}\rho w\\ \rho wu\\ \rho wv\\ \rho w^2+p\\ (E_t+p)w+kT_z\end{bmatrix}_i \hat{k}\right)\cdot(\alpha_{ij}-1)\hat{\mathbf{n}}_{\boldsymbol{ij}}\right] \tag{6.5-29}$$

$$\boldsymbol{f}_{\mathrm{ij}}(\mathbf{U}_{\mathrm{j}},t)=-\frac{S_{ij}}{\mathrm{V_i}}\left[\left(\left(\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ (E_t+p)u+kT_x\end{bmatrix}_j\hat{\imath}+\begin{bmatrix}\rho v\\ \rho vu\\ \rho v^2+p\\ \rho vw\\ (E_t+p)v+kT_y\end{bmatrix}_j\hat{\jmath}+\begin{bmatrix}\rho w\\ \rho wu\\ \rho wv\\ \rho w^2+p\\ (E_t+p)w+kT_z\end{bmatrix}_j\hat{k}\right)\right)\cdot\left(\alpha_{ij}\hat{\mathbf{n}}_{\boldsymbol{ij}}\right)\right] \tag{6.5-30}$$

Note that the term $\left(\bar{a}_{ij}-\mathbf{1}\right)\circ\hat{\mathbf{n}}_{\boldsymbol{ij}}$ is a function of the spatial partition. The number of terms in the summations over the neighbor sets is a function of the graph topology. The values for $\boldsymbol{g}_i(\mathbf{X}_i,t)$ are also nonlinear and are a function of the particular control input into a vertex.

$$\begin{aligned}\frac{d}{dt}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ E_t\end{bmatrix}&=\sum_{\mathrm{j}\in\mathrm{N_e}}\frac{S_{ij}}{\mathrm{V_i}}\left(\alpha_{ij}-1\right)\begin{bmatrix}0&n_x&n_y&n_z&0\\ 0&un_x&wn_y&wn_z&0\\ 0&vn_x&vn_y&vn_z&0\\ 0&wn_x&wn_y&wn_z&0\end{bmatrix}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ E_t\end{bmatrix}\\ &+\begin{bmatrix}0\\ \left(\tilde{\gamma}E_t-\frac{\tilde{\gamma}\rho uu}{2}-\frac{\tilde{\gamma}\rho uv}{2}-\frac{\tilde{\gamma}\rho uw}{2}-\tilde{\gamma}\rho\bar{g}\cdot\bar{r}\right)n_x\\ \left(\tilde{\gamma}E_t-\frac{\tilde{\gamma}\rho uu}{2}-\frac{\tilde{\gamma}\rho uv}{2}-\frac{\tilde{\gamma}\rho uw}{2}-\tilde{\gamma}\rho\bar{g}\cdot\bar{r}\right)n_y\\ \left(\tilde{\gamma}E_t-\frac{\tilde{\gamma}\rho uu}{2}-\frac{\tilde{\gamma}\rho uv}{2}-\frac{\tilde{\gamma}\rho uw}{2}-\tilde{\gamma}\rho\bar{g}\cdot\bar{r}\right)n_z\\ 0\end{bmatrix}\\ &=\cdots+\begin{bmatrix}0\\ \left(\tilde{\gamma}n_xE_t-\frac{\tilde{\gamma}un_x}{2}\rho u-\frac{\tilde{\gamma}un_x}{2}\rho v-\frac{\tilde{\gamma}un_x}{2}\rho w-\tilde{\gamma}n_x\rho\bar{g}\cdot\bar{r}\right)\\ \left(\tilde{\gamma}n_yE_t-\frac{\tilde{\gamma}n_yu}{2}\rho u-\frac{\tilde{\gamma}n_yu}{2}\rho v-\frac{\tilde{\gamma}n_yu}{2}\rho w-\tilde{\gamma}n_y\rho\bar{g}\cdot\bar{r}\right)\\ \left(\tilde{\gamma}n_zE_t-\frac{\tilde{\gamma}n_zu}{2}\rho u-\frac{\tilde{\gamma}n_zu}{2}\rho v-\frac{\tilde{\gamma}n_zu}{2}\rho w-\tilde{\gamma}\rho\bar{g}\cdot\bar{r}\right)\\ 0\end{bmatrix}\end{aligned} \tag{6.5-31}$$

$$\frac{d}{dt}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ E_t\end{bmatrix}=\sum_{\mathrm{j}\in\mathrm{N_e}}\frac{S_{ij}}{\mathrm{V_i}}\left(\alpha_{ij}-1\right)\begin{bmatrix}0&n_x&n_y&n_z&0\\ 0&un_x-\frac{\tilde{\gamma}un_x}{2}&wn_y-\frac{\tilde{\gamma}un_x}{2}&wn_z-\frac{\tilde{\gamma}un_x}{2}&\tilde{\gamma}n_x\\ 0&vn_x-\frac{\tilde{\gamma}n_yu}{2}&vn_y-\frac{\tilde{\gamma}n_yu}{2}&vn_z-\frac{\tilde{\gamma}n_yu}{2}&\tilde{\gamma}n_y\\ 0&wn_x-\frac{\tilde{\gamma}n_zu}{2}&wn_y-\frac{\tilde{\gamma}n_zu}{2}&wn_z-\frac{\tilde{\gamma}n_zu}{2}&\tilde{\gamma}n_z\end{bmatrix}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ E_t\end{bmatrix}+\begin{bmatrix}0\\ \left(-\tilde{\gamma}n_x\rho\bar{g}\cdot\bar{r}\right)\\ \left(-\tilde{\gamma}n_y\rho\bar{g}\cdot\bar{r}\right)\\ \left(-\tilde{\gamma}\rho\bar{g}\cdot\bar{r}\right)\\ 0\end{bmatrix}$$

6.5.12 API Class Structure

The major for the Savant-ML classes are shown below.

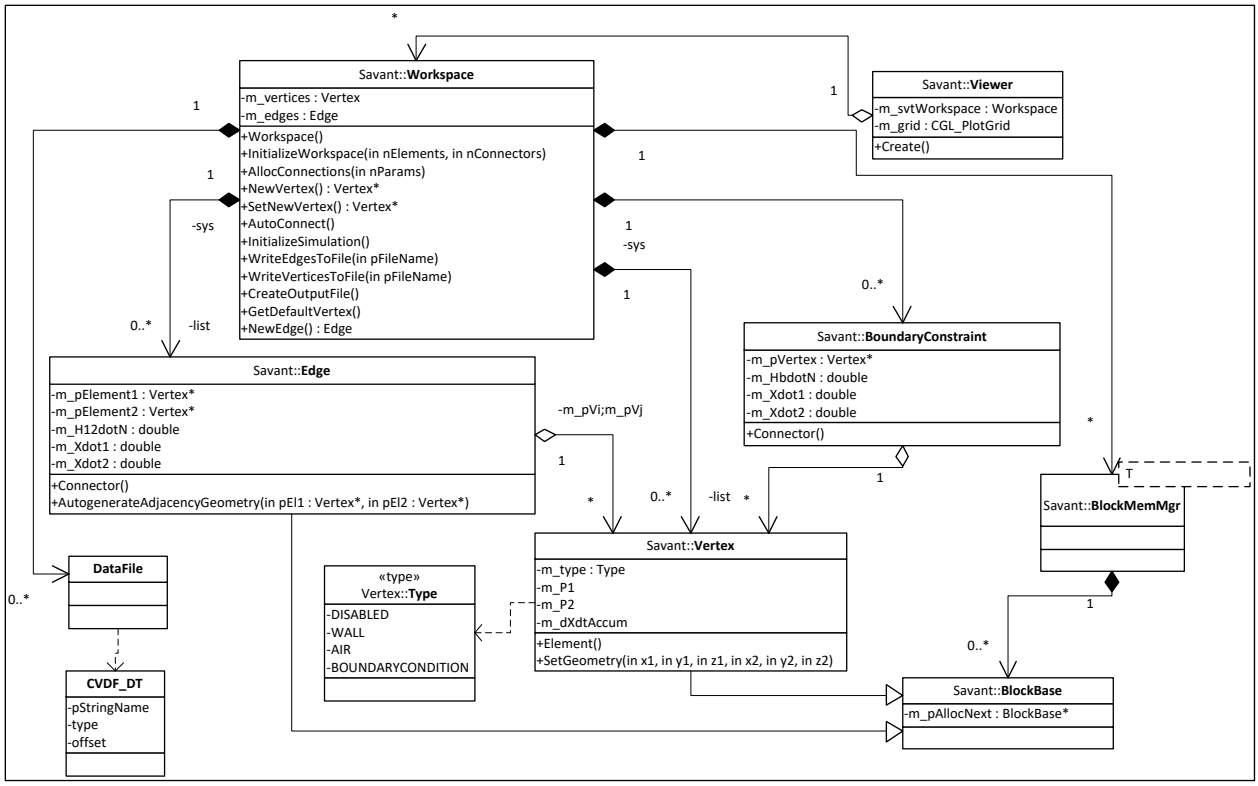


Figure 93. Major Classes in SAVANT-ML

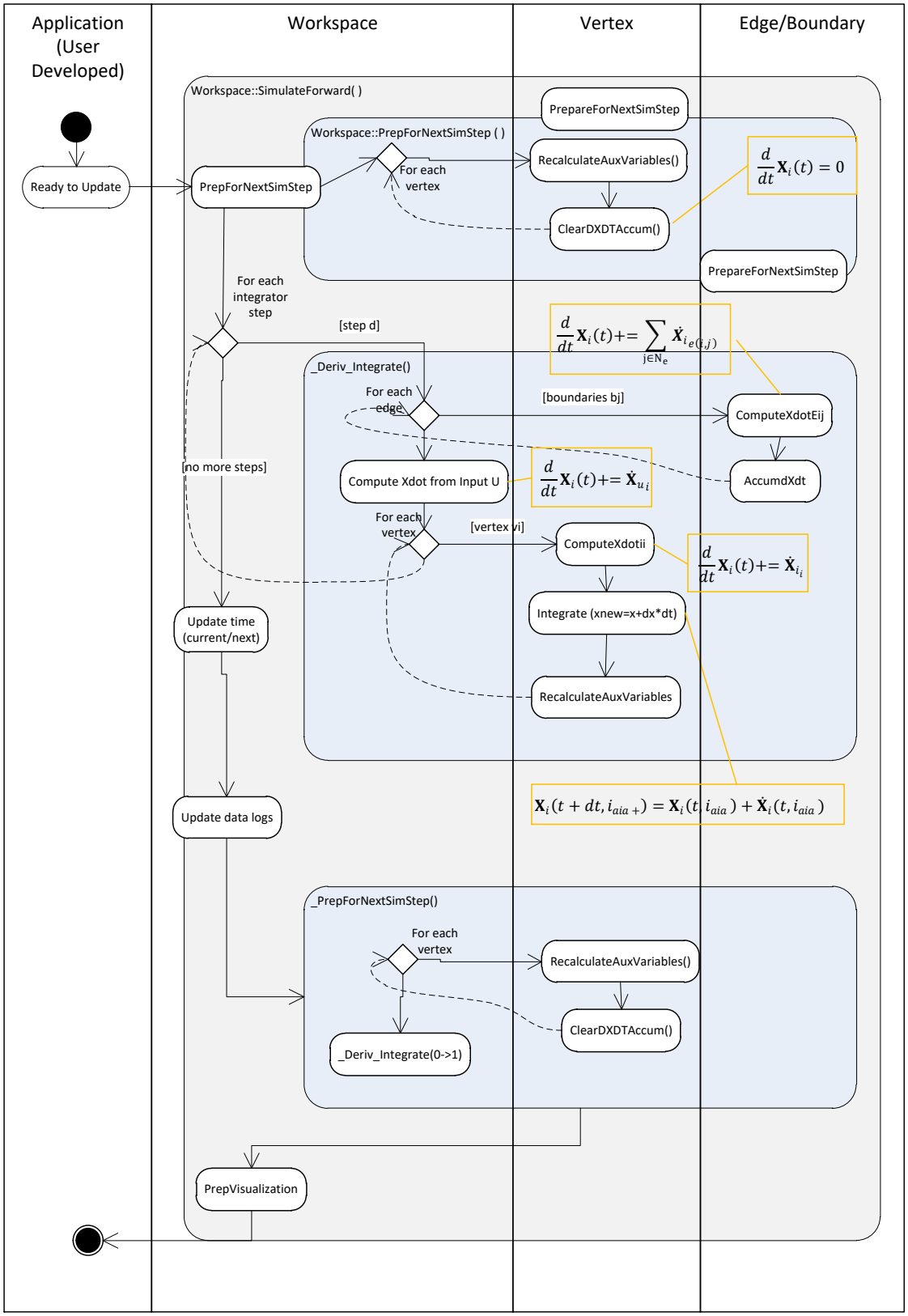


Figure 94. State-Chart for SAVANT-ML Simulation Update

6.5.13 Simulation Experiments

The following building test cases were developed in evaluation of the algorithms and approach, as shown in Figure 95, Figure 96 and Figure 97. The first configuration in Figure 95 utilized a linear configuration of rooms, followed by a T configuration in Figure 96, and a more complex configuration in Figure 97

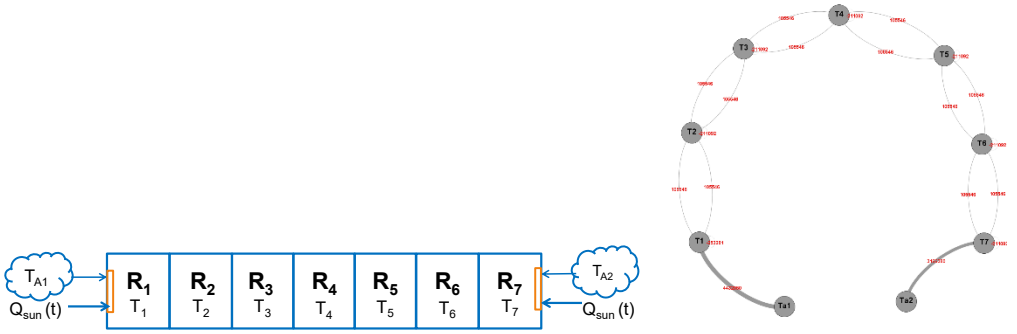


Figure 95. Configuration 1: Linear Room Building Configuration.

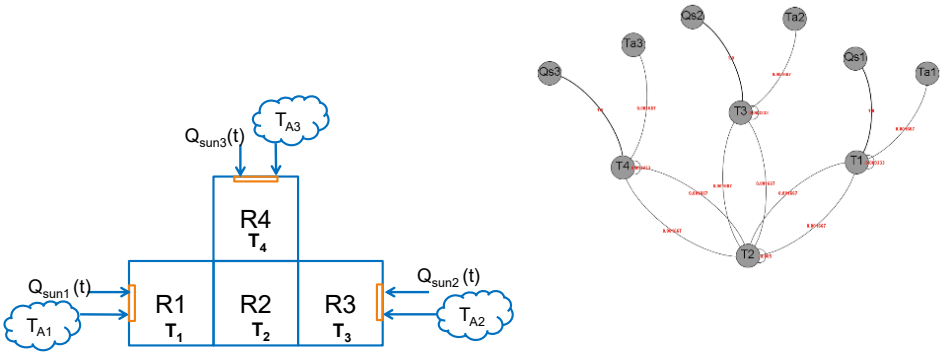


Figure 96. Configuration 2: T-Shaped Building Configuration (left), System Graph (right).

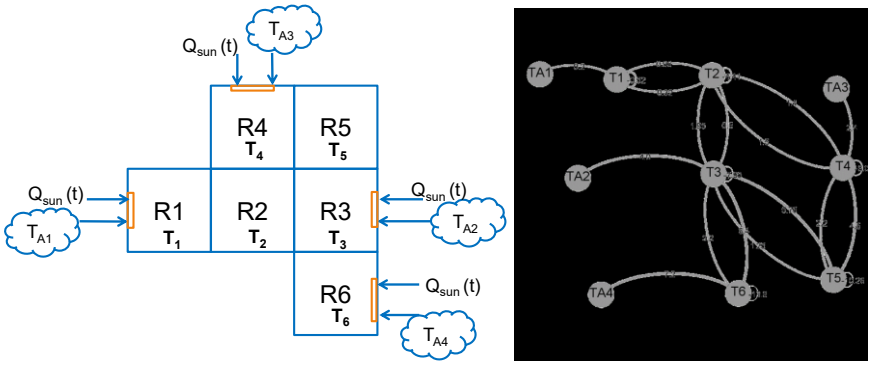


Figure 97. Configuration 3: Generic Building Configuration (left), System Graph (right).

6.5.14 Model Validation – Experimental Data Collection and Parameter Estimation

An experiment was conducted in development of Savant-ML to validate the model and derivation. In this experiment, we developed and deployed a wireless sensor network in a section of a test building to record pressure and temperature over the course of a week. The rooms were connected with a door, and the rooms had a window to the outside environment. Two sensors, sensor node1 and node 3, were placed next to window openings so that they could provide the ambient environmental conditions surrounding the building. Sensor node 2 was placed in the center of the room, within a control volume whose temperature and pressure we were aiming to predict. Its main purpose is to act as a comparison to the predicted values of the model. The building section's HVAC control was disabled during this experiment. The experiment configuration and estimation model is shown in Figure 98. A gradient descent optimization algorithm was created to optimize the model parameters based on a set of training data and set of testing data.

The goals of the preliminary experiment were to (1) model a building section using Savant-ML's finite-volume scheme, (2) evaluate Savant-ML as a modeling and estimation scheme for PDE's over wireless distributed sensor networks, and (3) estimate the hidden building parameters of the adaptive model through online data regression and parameter estimation.

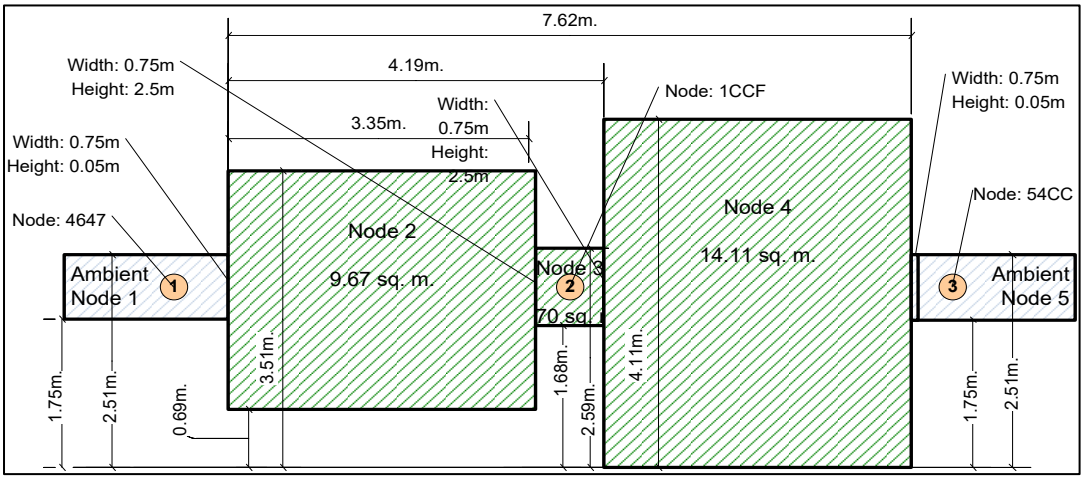


Figure 98. Experiment Configuration Model – Room and Sensor Layout

6.5.14.1 Experiment Hardware

The wireless sensor network hardware configuration is shown in Figure 99. Data from the three building sensors were transmitted over a wireless sensor network to a base station computer. We utilized Java Sun SPOT development

boards to implement the wireless network. Each SPOT contains a 180 MHz 32-bit ARM920T processor, a light sensor, and 5 general-purpose digital I/O pins. Two of these pins (D2 and D3) were used to communicate through I2C with a Bosch Sensortec BMP085 barometric pressure and temperature sensor. The BMP085 pressure sensor is based on piezo-resistive technology and offers a measuring range of 300 to 1100 hPa with an absolute accuracy of down to 0.03 hPa and 0.625°C. Sensor data recorded from the sensor was sent from the SPOT to the base station over wireless IEEE 802.15.4 and relayed to the laptop computer. The laptop processed and stored the data, then fed the data into the modeling and estimation program.

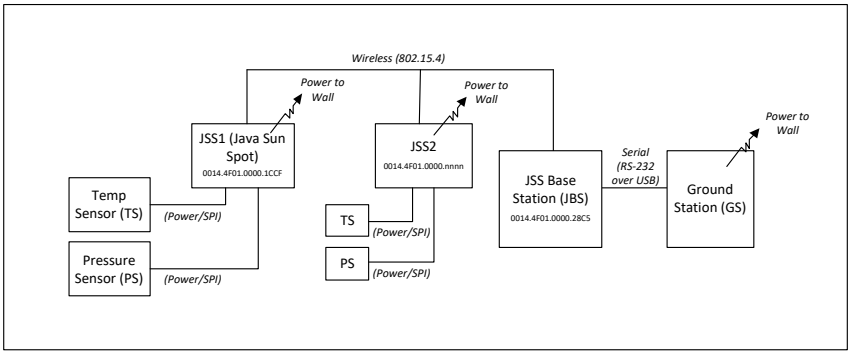


Figure 99. Hardware Architecture

6.5.14.2 Results

Data was gathered over a one week period at the experiment location. The temperature and pressure plots for this time are shown in Figure 100

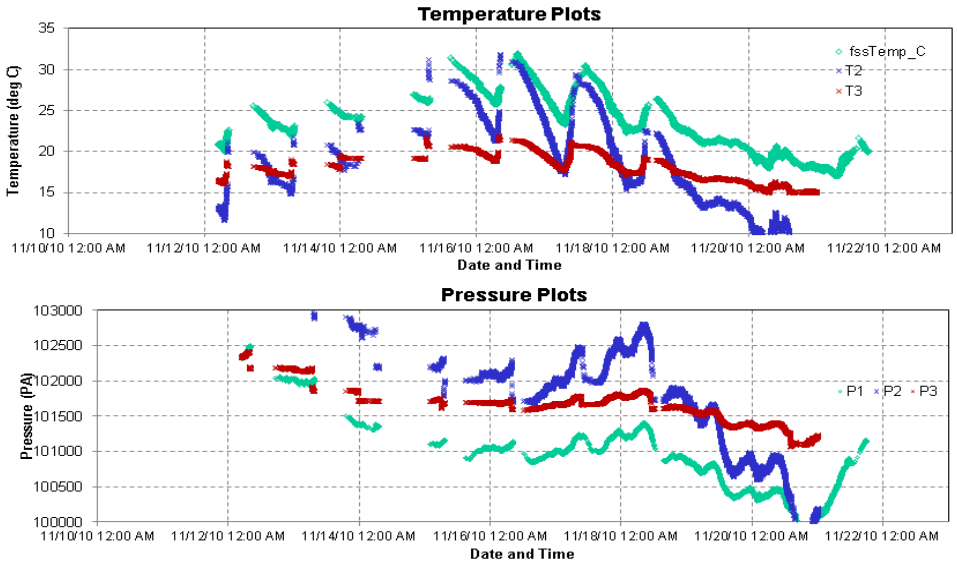


Figure 100. Temperature and Pressure - Recorded Experiment Data

The optimizer was tested on a set of data recorded between 11/15/2010 at 7:30PM and 11/16/2010 at 7:00AM (data set 20101115) and trained on the remaining data sets. The resulting optimization improved the accuracy of the simulation by 4 orders of magnitude, from 4.8E4 at iteration i=1 to 1.59E1 at iteration i=250. The optimizer stabilized to within 5% of its final value after 193 iterations. The residual errors after each iteration are shown in Figure 101 (also shown in log scale).

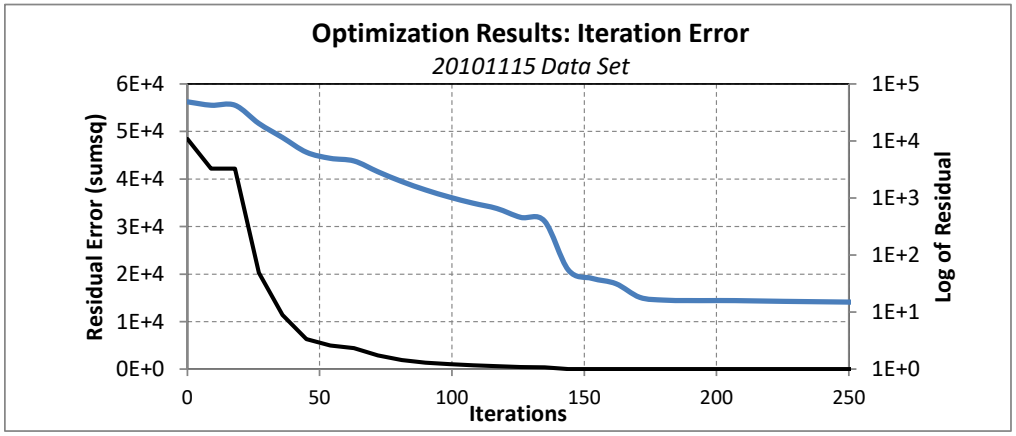


Figure 101. Optimizer Results

Figure 102 shows the results of the optimizer, comparing node 3 with the sensed temperature at node 3. The simulations were conducted after various iterations (corresponding to the error shown in Figure 101).

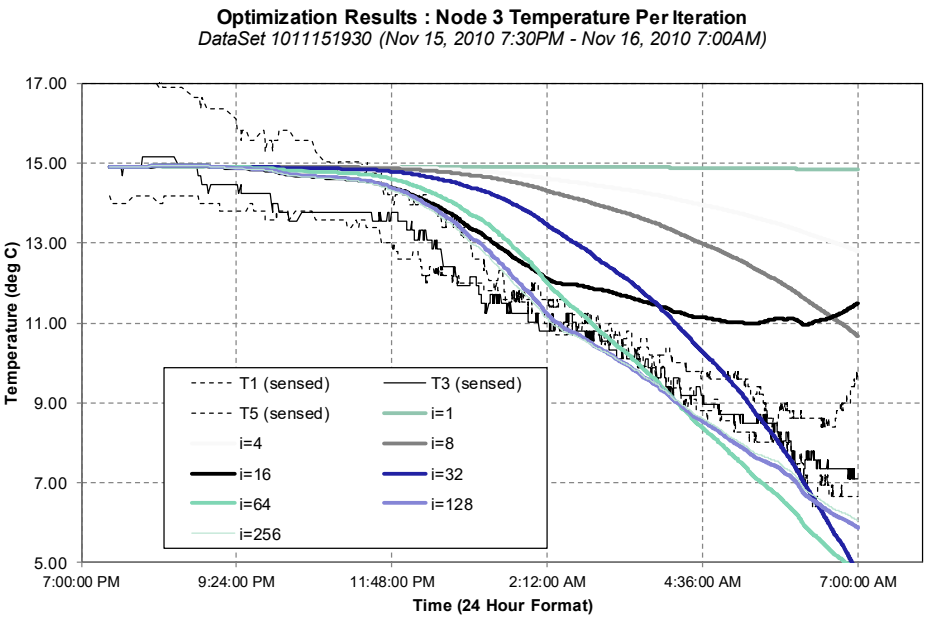


Figure 102. Optimizer Progress for Node 3 Against Collected Data.

Simulation results from the test Node 3, compared against the data set after iterations 1, 4, 8, 16, 32, 64, 128, and 256.

Chapter 7 Conclusion

The necessity for control systems theory to address larger more complex systems continues to drive research into interdisciplinary approaches for control of large-scale systems. In this dissertation, we have presented techniques to address several challenges facing the current state of the art in this field. In developing the solutions for a real-time self-assembling decentralized control system for large-scale complex dynamic system, the larger problem was broken down into smaller problems that addressed different areas and different disciplinary fields of study, and the challenges facing the larger system was distributed accordingly to appropriate sub-problems. We presented generalized formulations for each sub-problem, and developed solutions and techniques in each domain that can generally be applied beyond this framework, each of which extends current research approaches to address limitations documented in the literature. To demonstrate applicability and utility of the novel contributions, we presented several applications on various problems with a relatively high degree of fidelity and detail.

In particular, we presented a new formulation for control of large-scale distributed dynamic systems through the System Component Graph (SCG) method. This method extends the formulation of directed B-hypergraphs and applies it to the general problem of modeling, analysis, and synthesis of decentralized control structures over these systems. The SCG formulation allows for expansion and contraction to arbitrary levels of detail, which was utilized to demonstrate analysis techniques and algorithms that developed automated decentralized control structures for systems in real-time without prior understanding of the structure of the system. An optimal control formulation for components was introduced to solve the problem of control over each decentralized sub-system component. This method was demonstrated on applications for lateral and longitudinal mode flight control of a fixed-wing aircraft with a continuously actuated control surface on a flexible wing structure. Next, a new method for optimal guidance trajectory planning for large-scale decentrally controlled systems was introduced through the method of Random Subcomplement Trees (RST). This method was demonstrated on a fixed-wing unmanned aircraft control planning problem for automated wildfire smoke plume monitoring. The SGC method and RST method were then demonstrated on an indoor robotic navigation problem with cooperation between sub-systems in a smart building environment. Finally, we demonstrated the proposed framework on environmental control for Intelligent Integrated Building Control (IIBC). We presented a new formulation for large-scale building dynamics modeling and simulation in the

SAVANT-ML modeling library. We derived dynamics of fluid-thermal building systems through first-principles derivation of the Navier-Stokes equation using a finite-volume discretization method. We presented results from experiments which evaluated the model through offline parameter identification. The resulting system and parameters were utilized in the analysis of several different building configurations to perform automated decentralized decomposition and control synthesis.

7.1 Summary of Contributions

The following summarizes the novel research contributions presented in this dissertation.

1. A new concept and framework for self-assembling self-configuring and structurally-adaptive control systems
2. A new mathematical formalism for modeling, analysis, and synthesis of large-scale distributed control problems
3. A new clustering algorithm and metric for real-time decentralization of large systems
4. The Random Subcomplement Trees (RST) method for pseudo-optimal real-time trajectory planning and optimization
5. A novel fluid/thermal model derived from the Navier-Stokes equation that uses graph-based constructs and constant-volume methods, which was implemented in the Savant-ML C/C++ library and API for building control analysis and simulation
6. New dynamic models and simulation software for fixed-wing aircraft, UAS, and UGV were derived, implemented, and presented

To further elaborate on item 6 above, the following list summarizes the application problems presented in this dissertation. With each problem, we derived dynamic models from first principles and implemented these models in a simulation environment. Each example tested one or more of the novel contributions listed above.

1. Three-mass system example problem
2. Pseudo-Optimal Real-Time Path Planning for Automated Wildfire Monitoring from an Unmanned Aerial Vehicle
3. Decentralized Swarm UAS Control for Volcanic Plume Monitoring
4. Real-time Decentralized Reconfigurable Control for Mobile Robotics in Smart Environments

5. Multi-Objective Flight Control and Structural Control of a Variable Continuous Trailing Edge Control System on a Thin Flexible Aircraft Wing
6. Intelligent Integrated Environmental Control for a Smart Building

7.1.1 Self-assembling Self-configuring Structurally-adaptive Control Systems

The concept for self-assembly and self-configuration addresses challenges faced in control of modern large-scale cyber-physical systems that exhibit substantial variability in system structure. This dissertation presented a concept for control system constructs that can morph by adjusting the control structure in real-time to match the current structure of the large-scale interconnected system. In outlining this approach, we presented a process and framework for realization of this concept, then outlined the requirements and challenges in each step of this process. We formulated and developed approaches to solve each of these steps, which involved deriving new tools and approaches in these areas.

7.1.2 Mathematical Framework for Modeling, Analysis, and Synthesis of Large-Scale Control Problems

In this dissertation, we formalized a novel formal mathematical framework for addressing large-scale control problems. There are many unique aspects and desirable properties of this formulation. This formulization allows the problem to be abstracted once, then passed through each step of the process all the way through to implementation. The mathematical framework provides a unification of the various representations that would otherwise be needed, providing the foundation for algorithmic development, dynamic systems, control system formulation, system structure, and the implementation. To the knowledge of the author, a unified mathematical basis has never been presented. All the approaches reviewed in the background section, for instance, require one-way transformation of the problem into a temporary intermediate representation that is domain specific with limited utility beyond a particular set of operations. The formulation presented provides a method for transforming the problem between various system representations - e.g., between state space and system graph space - without any loss of information in a reversible manner, and we showed the transformation definitions are isomorphic (linear and reversible). The recursive nature of the formulation allows the system to be transformed to any selected level of granularity through contraction and expansion of sub-graph clusters. The formulation is amenable to extensions describing a wide range of constraints, such as wireless network bandwidth limits as demonstrated in the UGV indoor navigation application.

Further, graphical formulations in the literature, such as structure graphs, only address linear relationships, whereas the formulation presented can be applied to a large class of linear and non-linear systems.

7.1.3 Clustering Algorithm and Metric for Online Partitioning

In this dissertation, we presented a new clustering algorithm and associated metric that operate on the system graph representation to quickly identify system partitions. The clusters resulting from this algorithm define the structural topology of the control constructs for the local and global systems. The algorithm and metrics extend recent methods presented in the literature for partitioning of structure graphs for dynamic structure analysis, but extend the optimization criteria beyond system dynamics to include a wider range of constraints and objectives that are introduced for these problems, some of which arise from multi-disciplinary considerations, for instance communication network bandwidth limits, as demonstrated in the UGV indoor navigation application.

7.1.4 Random Subcomplement Trees (RST) Method for Pseudo-Optimal Real-Time Trajectory Planning and Optimization

We presented a new approach for online optimization and trajectory generation that extends the concept of Rapidly-Exploring Random Trees in the literature. The RST method provides a formalism to effectively reduce the order of the problem presented through transformation into an intermediate controller space where optimal transitions between states can be derived. This intermediate controller space formulation also addresses the issue of low-quality trajectory solutions in the original concept, which is a documented limitation of the original approach.

7.1.5 Savant-ML Fluid/Thermal Building Control Model and C++ Programming Library

For this dissertation, we derived and implemented a new model for simulation and environmental control system development of building systems. This system models fluid and thermal flow through fixed finite constant volumes from first principles through the Navier-Stokes equations. The modeling library is a novel graph-based formulation derived using the mathematical system graph formulation presented. The modelling derivation was developed into an API and implemented in the Savant-ML C++ programming library. Simulation utilizes a high-order Runge-Kutta integration scheme.

7.1.6 Dynamic Models and Simulation Software for Fixed-wing Aircraft, UAS, and UGV

Each application presented in this dissertation required mathematical dynamic models for analysis and simulation. In the applications chapter, we derived and presented dynamic models for the various application systems. The models were implemented in stand-alone simulation tools for each application, then integrated with the control system concepts being tested. The simulation results were used to evaluate and verify proper functionality of the novel contributions presented in this dissertation.

7.2 Future Research

In this dissertation, we presented the concept of a self-assembling self-configuring control system framework for control of large-scale variably-structured systems. The framework presented is foundational in nature, presenting the concept and developing the mathematical tools and algorithms necessary for the system to function. The presented framework is multidisciplinary in nature, extending concepts in a number of fields that includes graph and set theory, decentralized and optimal control, fluid dynamics, algorithmic complexity, numerical computation, and pervasive computing. There is tremendous room for extending the concept for self-assembling self-configuring control through continued research.

The method for system partition presented utilized a simple metric to establish variable 'closeness' which was incorporated into a graph clustering algorithm. Many alternative and more sophisticated methods could be conceived for the metric. A list of suggested metrics was provided in the background chapter. Selection of a particular metric would allow the controller to form based on different aspects of the system. Different resulting system partitions would result in different behavior in the final system response. The implications of adjusting the metric could be explored, but could be extended to handle additional constraints on the graph, such as network capacity, latency expectations across wireless links, and bandwidth limitations. Note that the basis for graph partitioning presented was limited to analysis of the dynamics over the graph structure. Additional types of constraints should be amenable to this system, but the efficacy of the framework for handling these types of constraints could be explored.

Alternative methods to the algorithmic graph clustering method presented could also be explored. Many methods have been proposed in the literature for offline analysis in the decentralized control literature, and the method for online implementation and the trade-offs could further be explored.

The basis for decentralized control generation for sub-systems partitioning utilized a popular method in the decentralization literature to address overlapping systems utilizing optimal LQ control. Several modifications and extensions have been presented in the literature to this approach as described in the background chapter, including extension for robust and adaptive control formulations to additional resilience within a clustered partition and graph sub-system. In addition, many alternative approaches exist to the overlapping systems method utilized. We presented several approaches in the background section, many of which could be explored as alternatives.

The examples provided largely evaluated results in simulation studies. A logical next step is to implement these systems and evaluate them in real-world applications. In evaluating the building fluid-thermal model, for instance, we built a wireless network for data collection and evaluation. These wireless nodes could be networked into the control actuators of the building system. Likewise, additional tests could be performed for the other applications, such as swarm control for UAS.

Additionally, the mathematical formulation presented could be extended in future research directions. For instance, application of this framework to problems in other fields may find it necessary to extend the framework or adjust the formulation. The formulation could, for instance, be extended to handle certain classes of nonlinear control, which would allow this framework to be applied to a wider class of problems. While the formulation can handle nonlinearity in terms of atomic components, these components cannot be further decomposed in the analysis, and therefore the structure of the system cannot be determined through the clustering metric and algorithm presented. Future research extension to nonlinear systems would need to elaborate and address these challenges.

Bibliography

Adami, Tony, J. M., Jim Zhu, Abraham Ishihara, Yoo-Hsiu Yeh, and Corey Ippolito. 2009. "Six-DOF trajectory tracking for payload directed flight using trajectory linearization control." *In Proc. of AIAA Guidance, Navigation and Control Conference, Seattle, WA, USA* 1897-1916.

Afanasiev, K., and M. Hinze. 2001. "Adaptive Control of a Wake Flow Using Proper Orthogonal Decomposition." In *Lecture Notes in Pure and Applied Mathematics: Shape optimization and optimal design*, edited by M. Dekker, 317–332.

2011. *American Housing Survey for the United States: 2009.* United States Department of Housing and Urban Development.

Arian, E., M. Fahl, and E. Sachs. 2000. "Trust-region proper orthogonal decomposition for flow control." Technical Report, Universitt Trier .

Bachmayer, R., and N. E. Leonard. 2002. "Vehicle Networks for gradient Descent in a Sampled Environment." *Proc. IEEE Conf. on Decision and Control, Las Vegas, NV* 112–117.

Bakule, L., J. Rodellar, and J. M. Rossell. 2001. "Controllability-observability of expanded composite systems." *Linear Algebra and its Applications* 381–400.

Bakule, L., J. Rodellar, J.M. Rossell, and P. Rubio. 2001. "Preservation of controllability-observability in expanded systems." *IEEE Transactions on Automatic Control* 46(7):1155–1162.

Bamieh, B, and P. G. Voulgaris. 2005. "A convex characterization of classes of problems in control with specific interaction and communication structures." *Systems & Control Letters* 54 (6): 575-583.

Bamieh, B, and P. G. Voulgaris. 2002. "Optimal distributed control with distributed delayed measurements." *IFAC 15th Triennial World Congress* Barcelona, Spain.

Benner, P., E. S. Quintana-Ort, and G. and Quintana-Ort. 2000. "Balanced Truncation Model Reduction of Large-Scale Dense Systems on Parallel Computers." *Mathematical and Computer Modelling of Dynamical Systems* 6 (4): 383-405.

Bieniawski, S, D Wolpert, and I Kroo. 2004. " Discrete, Continuous, and Constrained Optimization Using Collectives." *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.* Albany, NY.

Bieniawski, S, I. Kroo, and D. Wolpert. 2005. "Flight Control with Distributed Effectors." *AIAA Guidance, Navigation, and Control Conference, , August 15-18, 2005.* San Francisco, CA.

Blanchard, Benjamin S., and Wolter J. Fabrycky. 2010. *Systems Engineering and Analysis, Prentice Hall International Series in Industrial & Systems Engineering.* Fifth Edition. Prentice Hall.

Blondel, V. D., and J. N. Tsitsiklis. 2000. "A survey of computational complexity results in systems and control." *Automatica* 36 (9): 1249-1274.

Bui-Thanh, T., K. E. Willcox, O. Ghattas, and B. van Bloemen Waanders. 2007. "Goal-Oriented,Model-Constrained Optimization for Reduction of Large-Scale Systems." *Journal of Computational Physics* 224: 880–896.

Caltabiano, Daniele, Muscato Giovanni, Orlo Angelo, Federico Cinzia, Giudice Gaetano, and Guerrieri Sergio. 2005. "Architecture of a UAV for volcanic gas sampling." *In Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on, vol. 1* 6.

Chatterjee, A. 2000. "An introduction to the proper orthogonal decomposition." *CURRENT SCIENCE* 78 (7).

Chen, X, and S Stankovic. 2005. "Decomposition and decentralized control of systems with multi-overlapping structure." *Automatica* 41 (10): 1765-1772.

Corrales, J., Yetty Madrigal, David Pieri, Geoff Bl, Ted Miles, and Matthew Fladel. 2012. "Volcano monitoring with small unmanned aerial systems." *American Institute of Aeronautics and Astronautics Infotech Aerospace Conference* 2522.

Crawleya, D. B., J. W. Handb, M. Kummertc, and B. T. Griffithd. 2008. "Contrasting the capabilities of building energy performance simulation programs." *Building and Environment* 43 (4): 661-673.

Culley, D, and A Behbahani. 2008. "Communication Needs Assessment for Distributed Turbine Engine Control." *44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit, 21 - 23 July* . Hartford, CT.

Davison, E. J., and N. Tripathi. 1978. "The optimal decentralized control of a large power system: load and frequency control." *IEEE Transactions on Automatic Control* AC-23 (2): 312-325.

Desai, J. P., J. Ostrowski, and V. Kumar. 1998. "Controlling formations of multiple mobile robots ." *Proc. Conf. Robotics and Automation.* Leuven, Belgium. 2864–2869.

Desai, J. P., V. Kumar, and J. P. Ostrowski. 2001. "Modeling and control of formations of nonholonomic mobile robots." *IEEE Trans. Robot. Automat.* 17: 905–908.

Diankov, Rosen, and James Kuffner. 2007. "Randomized statistical path planning." *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems* 1-6.

Dounis, A.I, and C. Caraiscos. 2008. "Advanced control systems engineering for energy and comfort management in a building environment- A review." *Elsevier.*

Duan, Z, J Wang, and L. Huang. 2006. "Special decentralized control problems in discrete-time interconnected systems composed of two subsystems." *2006 Chinese Control Conference.* 1080-1085.

Evangelatos, D. S. 1995. "Decomposition for augmented forms of large-scale systems." *International Journal of Systems Science* 26 (2): 387-412.

Fan, C., J. L. Speyer, and C. R. Jaensch. 1994. "Centralized and decentralized solutions of the linear-exponential-gaussian problem." *IEEE Trans. Autom. Control* 39 (10): 1986-2003.

Feddema, J., C. Lewis, and D. Schoenwald. 2002. " Decentralized control of cooperative robotic vehicles: theory and application." *IEEE Transactions on Robotics and Automation* 18 (5): 852-864 .

Filipovic, P, and A Meystel. 1993. "Hierarchical feedback control system." *Proceedings of the 32nd IEEE Conference on Decision and Control.* San Antonio, TX.

Findeisen, W. 1980. *Control and coordination in hierarchical systems.* New York, NY: J. Wiley.

Freirea, R. Z., G. H. C. Oliveirab, and N. Mendesc. 2008. "Predictive controllers for thermal comfort optimization and energy savings." *Energy and Buildings* 40 (7): 1353-1365.

Gardonio, P, E Bianchi, and S. J. Elliot. 2004. "Smart panel with multiple decentralized units for the control of sound transmission." *Journal of Sound and Vibration* 274 (1-2): 163-192.

Georgiou, A, and C A Floudas. 1990. "Structural properties of large scale systems." *International Journal of Control* 51 (1): 169-187.

Geraerts, Roland, and Mark H. Overmars. 2007. "Creating high-quality paths for motion planning." *The International Journal of Robotics Research 26, no. 8* 845-863.

Gilboa, I., and E. Zemel. 1989. "Nash and Correlated Equilibrium: Some Complexity Considerations." *Games and Economic Behavior* 1: 80-93.

Goebel, R., R. Sanfelice, and A. Teel. 2009. "Hybrid Dynamical Systems." *IEEE Control Systems Magazine* 29 (2): 28-93.

Gugercin, S., and A. Antoulas. 2004. "A survey of model reduction by balanced truncation and some new results." *International Journal of Control* 748-766.

Halpern, J.Y. 2008. "Computer science and game theory: A brief survey." *The New Palgrave Dictionary of Economics.*

Hendrickson, B, and K Devine. 2000. "Dynamic Load Balancing in Computational Mechanics." *Computational Methods in Applied Mechanics and Engineering* 184 (2-4): 485-500.

Ho, Y. C., and K. C. Chu. 1972. "Team decision theory and information structures in optimal control problems - Part I." *IEEE Trans. Autom. Control* 17 (1): 15-22.

Hodzic, M., and DD Siljak. 1986. "A Parallel Estimation Algorithm." In *Parallel Processing Techniques for Simulation*. New York, NY: Plenum Publishing Corporation.

Holmes, P., J. L. Lumley, and G. Berkooz. 1996. "Turbulence, Coherent Structures, Dynamical Systems and Symmetry." *Cambridge Monogr. Mech., Cambridge University Press.*

Hovland, S., J. Gravdahl, and K. Willcox. 2007. "Explicit Model predictive Control for Large-Scale Systems via Model Reduction." *AIAA Journal for Guidance, Control, and Dynamics.*

Hozdic, M., and D. D. Siljak. 1985. "Estimation and control of large sparse systems." *Automatica* 21 (3): 277-292.

Huisman, L., and S. Weiland. 2003. "Identification and model predictive control of an industrial glass feeder." *Preprints of the 13th IFAC Symposium on System Identification.* Rotterdam, The Netherlands. 1685–1689.

Hwang, C.L., L.J. Chang, and S.Y. Han. 2006. "Path Tracking and Obstacle Avoidance of Car-Like Mobile Robots in an Intelligent Space Using Mixed H 2/H∞ Decentralized Control." *IEEE International Conference on Systems, Man and Cybernetics, 2006. SMC'06.*

I.F. Akyildiz, W. Su, Y. Sankarasubramaniam,E. Cayirci. 2002. "A Survey on Sensor Networks." *IEEE Communications Magazine.*

Iamratanakul, D, G Meyer, G Chatterji, and S. Devasia. 2004. "Quantification of Airspace Sector Capacity using Decentralized Conflict Resolution Procedures." *43rd IEEE Conference on Decision and Control.* Atlantis, Paradise Island, Bahamas.

Iftar, A. 1993. "Overlapping decentralized dynamic optimal control." *International Journal of Control* 58(1): 187–209.

Iftar, A., and U. Ozguner. 1998. "Overlapping decompositions, expansions, contractions, and stability of hybrid systems." *IEEE Transactions on Automatic Control* 43: 1040–1055.

Ikeda, M, D.D. Siljak, and D.E. White. 1981. "Decentralized control with overlapping information sets." *Journal of Optimization Theory and Applications* 34(2): 279–310.

Ikeda, M., D. D. Siljak, and D. E. White. 1984. "An inclusion principle for dynamic systems." *IEEE Transactions on Automatic Control* 29: 244–249.

Ippolito, C., and K. Al-Ali. 2007. "Topological Constructs for Automatic Reconfiguration of Polymorphic Control Systems." *AIAA Infotech@Aerospace Conference and Exhibit, AIAA-2007-2832.*

Ippolito, C., and Y. Yeh. 2009. "A Trajectory Generation Approach for Payload Directed Flight." *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition.* Orlando, Florida.

Ippolito, C., and Y. Yeh. 2009. "A Trajectory Generation Approach for Payload Directed Flight." *In 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition* 1351.

Ippolito, C., G. Pisanich, and K. Al-Ali. 2005. "Component-Based Plug-and-Play Methodologies for Rapid Embedded Technology Development." *AIAA Infotech@Aerospoace Conference.* Arlington, VA.

Ippolito, Corey, Al-Ali Khalid, and John Dolan. 2005. "Polymorphic Control and Trajectory Optimization of an Autonomous Ground Vehicle over Wireless Mobile Networks." *In AIAA Infotech@ Aerospace 2005 Conference.*

Ippolito, Corey, Matt Fladeland, and Yoo Hsiu Yeh. 2009. "Applications of payload directed flight." *IEEE 2009 Aerospace Conference* 1-15.

Ito, K., and S. S. Ravindran. 1998. "A Reduced-Basis Method for Control Problems Governed by PDEs, Control and Estimation of Distributed Parameter Systems." *International Series of Numerical Mathematics* 126: 153-168.

Jaillet, Léonard, Juan Cortés, and Thierry Siméon. 2010. "Sampling-based path planning on configuration-space costmaps." *IEEE Transactions on Robotics 26, no. 4* 635-646.

Ji, Z., H. Lin, and T. H. Lee. 2008. "A graph theory based characterization of controllability for nearest neighbor interconnections with fixed topology." *Proc. of the 46th IEEE Conference on Decision and Control.*

Jiang, J., and D. Li. 2010. "Decentralized guaranteed cost static output feedback vibration control for piezoelectric smart structures." *Smart Materials and Structures* (Institute of Physics Publishing) 19: 015018.

Jiang, Z., and H. Rahimi-Eichi. 2009. "Design, modeling and simulation of a green building energy system." *IEEE PES 09 Power and Energy Society General Meeting.*

Kalman, R. E. 1964. "When is a linear system optimal." *Transactions of the ASME, Journal of Basic Engineering, Series D.*

Khatib, O, K Yokoi, K Chang, D Ruspini, and R Holmberg. 1998. "Coordination and decentralized cooperation of multiple mobile manipulators." *Journal of Robotic Systems* 13 (11): 755 - 764.

Kho, J., A. Rogers, and N.R. Jennings. 2009. "Decentralized control of adaptive sampling in wireless sensor networks." *ACM Transactions on Sensor Networks (TOSN)* 5 (3): 1-35.

Kim, J., K.D. Kim, V. Natarajan, S.D. Kelly, and J. Bentsman. 2008. "PdE-based model reference adaptive control of uncertain heterogeneous multiagent networks." *Nonlinear Analysis: Hybrid Systems* 2 (4): 115-1167.

Kim, J.-H., S. Lall, W. Merrill, and A. Behbahani. 2010. "Quadratic Invariance for Distributed Jet Engine Control Optimization." *American Control Conference.*

Kim, J.Y., V. Natarajan, S.D. Kelly, and J. Bentsman. 2009. "Partial difference equation based model reference control of a multiagent network of underactuated aquatic vehicles with strongly nonlinear dynamics." *Nonlinear Analysis: Hybrid Systems.*

Korolevayz, O. I, M Krsticx, and G Schmid-Schonbein. 2004. "Decentralized and adaptive control of nonlinear fluid flow networks." *International Journal of Control.*

Kovacina, Michael, Palmer Daniel, Yang Guang, and Ravi Vaidyanathan. 2002. "Multi-agent control algorithms for chemical cloud detection and mapping using unmanned air vehicles." *In Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, vol. 3* 2782-2788.

Krozel, J, and M Peters. 2000. "DECENTRALIZED CONTROL TECHNIQUES FOR DISTRIBUTED AIR/GROUND SEPARATION." RTO-36, 1266-140, SEAGULL TECHNOLOGY, INC.

Krtolica, R, and D. D. Siljak. 1980. "Suboptimality of decentralized stochastic control and estimation." *IEEE Transaction on Automatic Control* 25 (1): 76-83.

Krtolica, R., M. Hodzic, and DD Siljak. 1991. "A stochastic inclusion principle." In *Differential Equations: Stability and Control*, edited by S. Elaydi, 295-320. CRC Press.

Kunisch, K., and S. Volkwein. 1999. "Control of the Burgers Equation by a Reduced-Order Approach Using Proper Orthogonal Decomposition." *Journal of Optimization Theory and Applications* 102 (2): 345–371.

Kunisch, K., and S. Volkwein. 2006. "Proper orthogonal decomposition for optimality systems." Tech. Rep. 10, Institute for Mathematics and Scientific Computing, University of Graz.

Lavaei, J, and A. G. Aghdam. 2006. "Decentralized Control Design for Interconnected Systems Based on A Centralized Reference Controller." *Proceedings of the 45th IEEE Conference on Decision and Control* 1189-1195.

Lee, C, and D Wolpert. 2004. "Product distribution theory for control of multi-agent systems." *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems.* New York, NY.

Lee, R, C Ippolito, Y Yeh, J. Spritzer, and G. Phelps. 2010. "Payload- Directed Control of Geophysical Magnetic Surveys." *AIAA Infotech@Aerospace.* Atlanta, GA.

Lee, Ritchie, and Corey Ippolito. 2009. "A Perception and Mapping Approach for Plume Detection in Payload Directed Flight." *AIAA Infotech @ Aerospace.*

Levine, J., and M. Athans. 1966. "On the optimal error regulation of a string of moving vehicles." *IEEE Transactions of Automatic Control* 11 (Nov): 355-361.

Lewis, F. 2004. "Wireless Sensor Networks." In *Smart Environments: Technologies, Protocols, and Applications*, edited by D. J. Cook and S.K. Das. New York: John Wiley.

Li, J., and J. White. 2002. "Low rank solution of Lyapunov equations." *Journal on Matrix Analysis and Applications* 24 (1): 260-280.

Li, K, E. Kosmatopoulos, P. Ioannou, and H. Boussalis. 1999. "Centralized, Decentralized, and Overlapping Control Designs for a Segmented Telescope." *International Symposium on Intelligent COntrol/Intelligent Systems and Semiotics.* Cambridge, MA.

Liberzon, D., and R. Tempo. 2004. "Common Lyapunov functions and gradient algorithms." *IEEE Transactions on Automatic Control* 49 (6): 990-994.

Lin, C. T. 1974. "Structural Controllability." *IEEE Transactions* AC-19: 201-208.

Lin, H., and P. J. Antsaklis. 2007. "Switching stabilizability for continuoustime uncertain switched linear systems." *IEEE Transactions on Automatic Control* 52 (4): 633-646.

Lin, H., and PJ Antsaklis. 2009. "Stability and stabilizability of switched linear systems: a survey of recent results." *IEEE Transactions on Automatic Control* 54 (2): 308-322.

Linnermann, A. 1984. "Decentralized Control of Dynamically Interconnected Systems." *IEEE Transactions on Automatic Control* 29.

Liu, Y., K. Passino, and M. Polycarpou. 2001. "Stability analysis of one-dimensional asynchronous swarms." *Proc. American Control Conference.* Arlington, VA. 716–721.

Loh, Chin-Hsiung, and Chia-Ming Chang. 2008. "Application of Centralized and Decentralized Control to Building Structure: Analytical Study." *Journal of Engineering Mechanics* 134 (11): 970-982 .

Ly, H. V., and H. T. Tran. 1998. " Proper Orthogonal Decomposition for Flow Calculations and Optimal Control in a Horizontal CVD Reactor." CRSC-TR98-12, Center for Research in Scientific Computation, North Carolina State University.

Lygeros, J. 1996. "Hierarchical, Hybrid Control of Large Scale Systems." PhD Dissertation, Engineering-Electrical Engineering and Computer Science, University of California, Berkeley, Berkeley, CA.

Lynch, J, and K Law. 2002. "DECENTRALIZED CONTROL TECHNIQUES FOR LARGE-SCALE CIVIL STRUCTURAL SYSTEMS." *Proceedings of the 20th International Modal Analysis Conference (IMAC).* Los Angeles, CA.

Lynch, J.P., and K.H. Law. 2002. "Decentralized control techniques for large-scale civil structural systems." *Proc. of the 20th Int. Modal Analysis Conference (IMAC XX).*

Mahmoud, M, M Hassan, and M Darwish. 1985. *Loarge-Scale Control Systems, Theories and Techniques.* New York: Marcel Dekker, Inc.

Malinowsk, K., and M. Singh. 1985. "Controllability and observability of expanded systems with overlapping decompositions." *Automatica* 21: 203-208.

Malinowski, k, and M Singh. 1985. "Controllability and observability of expanded systems with overlapping decompositions." *Automatica* 21(2):203–208.

Mayeda, H., and T. Yamada. 1979. "Strong Structural Controllability." *SIAM Journal on Control and Optimization* 17: 123-138.

Melzer, S. M., and B. C. Kuo. 1971. "Optimal Regulations of Systems Described by a Countably Infinite Number of Objects." *Journal Automatica* 7: 359-366.

Menon, R., A MacKenzie, R. Buehrer, and J Reed. 2004. "Game Theory and Interference Avoidance in Decentralized Networks." *SDR 04 Technical Conference and Product Exposition.*

Meystel, A. 1993. "Multiresolutional System: Complexity and Reliability." In *Intelligent Systems: Safety, Reliability, and Maintainability Issues*, edited by O. Kaynak, G. Honderd and E. Grant, 11-22. Proceedings of the NATO Advanced Research Workshop on Intelligent Systems.

Molnar, P., and J. Starke. 2000. "Communication fault tolerance in distributed robotic systems." *Distributed Autonomous Robotic Systems 4 2000, pp. 99–108.* 4: 99–108.

Narendra, R. Shorten and K. 2002. "Necessary and sufficient conditions for the existence of a CQLF for a finite number of stable LTI systems." *International Journal on Adaptive Control Signal Processing* 16 (10): 709-728.

Newsham, G, S. Mancini, and B. Birt. 2009. "Do LEED-certified buildings save energy? Yes, but ..." *Energy and Buildings* (Elsevier) 41 (8): 897-905.

Neyman, A. 1985. "Bounded complexity justifies cooperation in finitely repated prisoner's dilemma." *Economic Letters* 19: 227–229.

Nise, S. N. 2004. *Control Systems Engineering.* Hoboken, New Jersey: John Wiley & Sons Inc.

Nour-Eldin, H. A. 1987. "Linear Multivariable Systems Controllability and Observability: Numerical Aspects." In *Systems and Control Encyclopedia*, edited by M. G. Singh, 2816-2827. Oxford, UK: Pergamon Press.

Ohtsuka, K, and Y. Morioka. 1997. "A decentralized control system for stabilizing a longitudinal power system using tieline power flow measurements." *IEEE Transactions on Power Systems* 12 (3): 1202-1209.

Palacios, Francisco, Gisela Pujol, Jose Rodellar, and Josep Rossell. 2006. "Optimal complementary matrices in systems with overlapping decomposition: A computational approach." *Proceedings of the 45th IEEE Conference on Decision & Control.* San Diego: IEEE.

Papadimitriou, C. H. and M. Yannakakis. 1994. "On complexity as bounded rationality." *In Proc. 26th ACM Symposium on Theory of Computing* 726–733.

Peng, Liqian, Lipinski Doug, and Kamran Mohseni. 2014. "Dynamic data driven application system for plume estimation using uavs." *Journal of Intelligent & Robotic Systems 74, no. 1-2* 421-436.

Perez-Lombard, L., J. Ortiz, R. Gonzalez, and J. R. Maestre. 2009. "A review of benchmarking, rating and labeling concepts within the framework of building energy certification schemes." *Energy and Buildings* 40 (3): 272-278.

Pichai, V., ME Sezer, and DD Siljak. 1983. "Graph-theoretic algorithm for hierarchical decomposition of dynamic systems with applications to estimation and control." *IEEE Transactions on Systems, Man, and Cybernetics* 13 (2): 197-207.

Pieri, D., and M. Abrams. 2004. "ASTER watches the world's volcanoes: a new paradigm for volcanological observations from orbit." *Journal of Volcanology and Geothermal Research 135, no. 1* 13-28.

Pieri, David, Jorge A Diaz, Bland Geoffrey, Fladeland Matthew, Yetty Madrigal, Ernesto Corrales, Oscar Alegria, et al. 2013. "In situ observations and sampling of volcanic emissions with NASA and UCR unmanned aircraft, including a case study at Turrialba Volcano, Costa Rica." *Geological Society, London, Special Publications 380, no. 1* 321-352.

Pisano, William J., Dale A. Lawrence, and Kamran Mohseni. 2006. "Concentration gradient and information energy for decentralized UAV control." *In AIAA Guidance, Navigation, and Control Conference and Exhibit, pp.* 21-24.

Pressman, Roger S. 2009. *Software Engineering: A Practitioner's Approach.* Seventh Edition. McGraw-Hill Science.

Qi, X,, M. Salapaka, P. Voulgaris, and M. Khammash. 2004. "Structured optimal and robust control with multiple criteria: A convex solution." *IEEE Trans. Autom. Control* 49 (10): 1623-1640.

Rahman, A, M Ji, M Mesbahi, and M. Egersted. 2009. "Controllability of Multi-Agent Systems from a Graph-Theoretic Perspective." *SIAM J. Control Optimization* 48 (1): 162-186.

Ravindran, S. S. 2000. "A Reduced-Order Approach for Optimal Control of Fluids Using Proper Orthogonal Decomposition." *International Journal for Numerical Methods in Fluids* 34: 425–448.

Rehák, M, M Pěchouček, and J Tožička. 2005. *Adversarial Behavior in Multi-agent Systems.* Berlin: Springer Berlin / Heidelberg.

Rose, C, S Ulukus, and R. D. Yates. 2002. "Wireless Systems and Interference Avoidance." *IEEE Transactions on Wireless Communications* 1 (3): 415-427.

Rotkowitz, M., and S. Lall. 2002. "Decentralized control information structures preserved under feedback." *Proc. IEEE Conf. Decision and Control.* 569-575.

Rotkowitz, Michael, and Sanjay Lall. 2005. "A Characterization of Convex Problems in Decentralized Control." *IEEE Transactions On Automatic Control* 50 (12): 1984-1996.

Safonov, MG. 1980. *Stability and robustness of multivariable feedback systems.* Cambridge, MA: MIT Press.

Saggiani, G., F. Persiani, Ceruti A, Tortora P, Troiani E, Giuletti F, Amici S, et al. 2007. "A UAV system for observing volcanoes and natural hazards." *AGU Fall Meeting Abstracts. Vol. 1.*

Schneider, F. E., D. Wildermuth, and H.-L. Wolf. 2000. "Motion coordination in formations of multiple robots using a potential field approach." *Distributed Autonomous Robotic Systems 4* 305-314.

Sesak, J. R., and T. Coradetti. 1979. "Decentralized Control of Large Space Structures via Forced Singular Perturbation." *17th Aerospace Sciences Meeting.* New Orleans, LA.

Sezer, M. E., and D. D. Siljak. 1983. "Minimal Essential Feedback Patterns for Pole Assignment using Dynamic Compensation." *Proceedings of the 22tn IEEE Conference on Deceision and Control.* San Antonio, TX.

Sezer, M. E., and D. D. Šiljak. 1991. "Nested epsilon decompositions of linear systems: weakly coupled and overlapping blocks." *SIAM Journal on Matrix Analysis and Applications 12, no. 3* 521-533.

Sezer, M. E., and K. Unyelioglu. 1988. "On optimum selection of stabilizing feedback structures for multivariable control systems." *Proceedings of the IFAC Symposium on Distributed Intelligent Systems.* Varna, Bulgaria. 145-154.

Sezer, ME, and DD Siljak. 1981. "Structurally Fixed Modes." *Systems and Control Letters* 1 (1): 60-64.

Shigemasa, T, and K Ratnesh. 2008. "Synthesis of Inference-Based Decentralized Control for Discrete Event Systems." *IEEE Transactions on Automatic Control* 53 (2): 522-534.

Shorten, R., and K Narendra. 2002. "Necessary and sufficient conditions for the existence of a CQLF for a finite number of stable LTI systems." *Journal of Adaptive Control and Signal Processing* 16 (10): 709-728.

Shorten, R., KS Narendra, and O. Mason. 2003. "A result on common quadratic Lyapunov functions." *IEEE Transactions on Automatic Control* 48 (1): 110-113.

Siljak, D. D. 1991. *Decentralized Control of Complex Systems.* Boston, MA: Academic Press.

Siljak, D. D. 1977. "On reachability of dynamic systems." *Internation Journal of Systems Science* 8: 321-338.

Siljak, D. 1978. *Large-Scale Dynamic Systems: Stability and Structure.* New York : North Holland.

Siljak, D.D. 2011. *Decentralized Control of Complex Systems.* Dover Publications.

Simpson, James J., Gary Hufford, David Pieri, and Jared Berg. 2000. "Failures in detecting volcanic ash from a satellite-based technique." *Remote Sensing of Environment 72, no. 2* 191-217.

Singh, M. 1981. *Decentralized Control.* Elsevier Science & Technology Books.

Singh, S.N., J.H. Myatt, G.A. Addington, S.S. Banda, and J.K. Hall. 2002. "Adaptive feedback linearizing control of proper orthogonal decomposition nonlinear flow models." *Nonlinear Dynamics* 28 (1): 71-81.

Sirovich, L. 1987. "Turbulence and the dynamics of coherent structures, parts I–III." *Quarterly of Applied Mathematics* XLV (3): 561–590.

Sorensen, D., and A. Antoulas. 2002. "The sylvester equation and approximate balanced reduction." *Linear Algebra and Its Applications, Fourth Special Issue on Linear Systems and Control* 260–445.

Srivastava, V., J. Neel, A.B. MacKenzie, R. Menon, L.A. DaSilva, J.E. Hicks, J.H. Reed, and R.P Gilles. 2005. "Using Game Theory to Analyze Wireless Ad Hoc." *IEEE Communications Surveys and Tutorials* 7 (4): 46-56.

Stankovic, S. S., M. J. Stanojevic, and D. D. Siljak. 2000. "Decentralized overlapping control of a platoon of vehicles." *IEEE Transactions on Control System Technology* 8: 816-831.

Stevens, B, and F Lewis. 1992. *Aircraft Control and Simulation.* Wiley-Interscience.

Stipanovic, Dusan M., Gokhan Inalhan, Rodney Teo, and Claire J. Tomlin. 2004. "Decentralized overlapping control of a formation of unmanned aerial vehicles." *Automatica* (40): 1285 – 1296.

Swaroop, D., and J. Hedrick. 1995. "String Stability of Interconnected Systems." Seattle, WA. Proceedings of the American Control Conference.

Tang, K. Y., W. R. Graham, and J. Peraire. 1996. "Active Flow Control Using a Reduced-Order Model and Optimum Control." Department of Aeronautics and Astronautics, Computational Aerospace Sciences Laboratory, MIT.

Tanner, H. 2004. "On the controllability of nearest neighbor interconnections." *Proceedings of the 43rd IEEE Conference on Decision and Control.*

Tavakoli, S., I. Griffin, and P.J. Fleming. 2005. "Decentralized PI control of a rolls-royce jet engine." *Proceedings of 2005 IEEE Conference on Control Applications.*

Tomlin, C, G Pappas, J Kosecka, J Lygeros, and S Sastry. 2006. "Advanced air traffic automation: A case study in distributed decentralized control." In *Control Problems in Robotics and Automation*. New York: Springer.

Trave, L., AM Tarras, and A. Titli. 1987. "Minimal feedback structure avoiding structurally fixed modes." *International Journal of Control* 46 (1): 313-325.

Tsitsiklis, C. H. Papadimitriou and J. N. 1986. "Intractable problems in control theory." *SIAM Journal of Optimal Control* 24: 639-654.

Tucsnaka, M, and M Vanninathanb. 2009. "Locally distributed control for a model of fluid–structure interaction." *Systems & Control Letters* 58 (8): 547-552.

Ulukus, S., and R. D. Yates. 2001. "Iterative Construction of Optimum Signature Sequence Sets in Synchronous CDMA systems." *IEEE Transactions on Information Theory* 47 (5): 1989-1998.

United States Department of Energy. 2009. "U.S. Buildings Energy Data Book 2009." Accessed April 2011. http://buildingsdatabook.eren.doe.gov/.

United States Department of Housing and Urban Development. 2009. "American Housing Survey for the United States: 2009."

United States Environmental Protection Agency. 2009. "Buildings and their Impact on the Environment: A Statistical Summary." Accessed April 2011. http:// www.epa.gov/greenbuilding.

Vincent, Patrick, and Izhak Rubin. 2004. "A framework and analysis for cooperative search using UAV swarms." *In Proceedings of the 2004 ACM Symposium on Applied Computing* 79-86.

Voulgaris, P. G. 1999. "Control under structural constraints: An input-output approach." In *Lecture Notes in Control and Information Sciences: Robustness in identification and control*, 287-305. London: Springer London.

Wang, Y, R Swartz, J Lynch, K Law, K Lu, and C Loh. 2007. "Decentralized Civil Structural Control using Real-time Wireless Sensing and Embedded Computing." *Smart Structures and Systems.* Techno Press. 3(3):321-340.

Wang, Y., R.A. Swartz, J.P. Lynch, K.H. Law, K.C. Lu, and C.H. Loh. 2007. "Decentralized civil structural control using real-time wireless sensing and embedded computing." *Smart Structures and Systems* 3 (3): 321-340.

Wertz, James R., and Wiley J. Larson. 1999. *Space Mission Analysis and Design.* Third Edition. Microcosm Press.

Winfield, A. 2000. "Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots." In *Distributed Autonomous Robotic Systems 4*. New York: Springer-Verlag.

Witsenhausen, H. S. 1968. "A counterexample in stochastic optimum control." *SIAM Journal of Control* 9 (1): 131-147.

Wolfe, J. D., D. F. Chichka, and J. L. Speyer. 1996. "Decentralized controllers for unmanned aerial vehicle formation Fight." *AIAA Guidance Navigation and Control Conference.* San Diego, CA.

Wollkind, S, J Valasek, and T. R. Ioerger. 2004. "Automated Conflict Resolution for Air Traffic Management Using Cooperative Multiagent Negotiation." *AIAA Guidance, Navigation, and Control Conference.*

Wolpert, D. 2006. "Information Theory — The Bridge Connecting Bounded Rational Game Theory and Statistical Physics." In *Complex Engineering Systems*, edited by Y Bar-Yam and D Braha. Perseus Books.

Wortelboer, P.M., M., Steinbuch, and O. H. Bosgra. 1999. "“Iterativemodel and controller reduction using closed-loop balancing, with application to a compact disc mechanism." *Int. J. Robust Nonlinear Control* 9 (3): 123-142.

Xi, X., M. S. Johnson, M Fladel, D Pieri, J Diaz, S Jeong, and G Bland. 2014. "Top-Down Estimates of SO2 Degassing Emissions from the Turrialba Volcano Using in Situ Measurements from Unmanned Aerial Systems and the WRF-Stilt Model." *In AGU Fall Meeting Abstracts, vol. 1* 3284.

Xie, D., Z. Liu, J. Xiong, and X. Peng. 2011. "Investigation on indoor thermal environment in an office room in summer." *2011 International Conference on Computer Distributed Control and Intelligent Environmental Monitoring.* 1568-1571.

Yamada, T, and L Foulds. 2006. "A graph-theoretic approach to investigate structural and qualitative properties of systems: A survey." *Networks, An Internationl Journal* 20 (4): 427-452.

Yamaguchi, Yasushi, Anne B. Kahle, Hiroji Tsu, Toru Kawakami, and Moshe Pniel. 1998. "Overview of advanced spaceborne thermal emission and reflection radiometer (ASTER)." *Geoscience and Remote Sensing, IEEE Transactions on 36, no. 4* 1062-1071.

Young, KD. 1990. "Distributed finite-element modeling and control approach for large flexible structures." *Journal of Guidance, Control, and Dynamics* 13 (4): 703-713.

Yousef, H, Ey El-Madboul, Eteim D, and Hamdy M. 2006. "Adaptive Fuzzy Decentralized Control for a Class of Large-Scale Nonlinear Systems with MIMO Subsystems." *Journal of Computer Engineering and Systems.*

Zamani, M., and H. Lin. 2009. "Structural controllability of multi-agent systems." *Proceedings of the 2009 conference on American Control Conference.* St. Louis, Missouri.

Zhou, K, JC Doyle, and K Glover. 1996. *Robust and optimal control.* Englewood Cliffs, NJ: Prentice Hall .