Lecture Notes in Control and Information Sciences 460

Eric Feron *Editor*

Advances in Control System Technology for Aerospace Applications

LNCIS

Springer

Lecture Notes in Control and Information Sciences

Volume 460

About this Series

This series aims to report new developments in the fields of control and information sciences—quickly, informally and at a high level. The type of material considered for publication includes:

1. Preliminary drafts of monographs and advanced textbooks
2. Lectures on a new field, or presenting a new angle on a classical field
3. Research reports
4. Reports of meetings, provided they are
   (a) of exceptional interest and
   (b) devoted to a specific topic. The timeliness of subject material is very important.

More information about this series at http://www.springer.com/series/642

Eric Feron
Editor

Advances in Control System Technology for Aerospace Applications

Springer

Editor
Eric Feron
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA
USA

ISSN 0170-8643 ISSN 1610-7411 (electronic)
Lecture Notes in Control and Information Sciences
ISBN 978-3-662-47693-2 ISBN 978-3-662-47694-9 (eBook)
DOI 10.1007/978-3-662-47694-9

Library of Congress Control Number: 2015944442

Springer Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer-Verlag GmbH Berlin Heidelberg is part of Springer Science+Business Media (www.springer.com)

Preface

On June 11 and 12, 2012, several engineers and researchers from industry and academia met at the Georgia Institute of Technology to discuss the present and future of aerospace decision and control. This workshop was hosted by the School of Aerospace Engineering and the Decision and Control Laboratory. Featured in this workshop were aircraft and spacecraft control and autonomy, air traffic control and management, and embedded software verification and validation. From this workshop came the five essays printed thereafter.

Whether focusing on aeronautical or space applications, the decision and control sciences of today largely supersede the servomechanism theory that used to be, and still is, taught in all aerospace undergraduate curricula. Yet, the concern for mathematical rigor and safety present in even the most basic control course is the fertile ground upon which new disciplines, such as autonomy, can develop with a genuine concern for applicability to aerospace systems. In this volume, the reader will find a broad variety of topics that all share highly dynamical, real-time, and safety- or mission-critical decision-making as core elements.

When looking at the space adventure, the reader will see that autonomy is becoming, de facto, the prime mechanism through which humanity can project its mind and soul onto faraway, extraterrestrial destinations. In an increasingly technological world, the reader will, however, get some appreciation for the gap that separates the extremely high promise of autonomy technology for aerial applications from our ability to understand it well enough to let it take over part of our overhead traffic. Likewise, the reader will get an appreciation for the astonishing range of control issues raised by air transportation, including optimal control, queuing systems, and combinations of the above.

Professor Gary Balas understood, perhaps better than anyone else in the trade, the vastly expanded scope that the decision and control sciences need to cover to address the challenges that aerospace engineering faces today. He presided over the fast transformation of the aerospace decision sciences by fostering a climate of openness toward the new aerospace decision and control sciences, whether they are

named autonomy, software analysis, air traffic control, or human-centric systems, within his own University of Minnesota and the Department of Aerospace Engineering and Mechanics, which he led with enthusiasm and humor. This volume is dedicated to his memory.

Atlanta Eric Feron
March 2015

Contents

Contributors

Behçet Açıkmeşe Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California; University of Texas at Austin, Austin, TX, USA

Hamsa Balakrishnan Massachusetts Institute of Technology, Cambridge, MA, USA

David S. Bayard Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

Soon-Jo Chung University of Illinois at Urbana-Champaign, Urbana, IL, USA

Andrew Clare Massachusetts Institute of Technology, Cambridge, MA, USA

John-Paul Clarke Georgia Institute of Technology, Atlanta, GA, USA

Mary L. Cummings Duke University MEMS, Durham, USA

Arnaud Dieumegard ENSEEIHT, Toulouse, France

Eric M. Feron Georgia Institute of Technology, Atlanta, GA, USA

Pierre-Loïc Garoche ONERA—The French Aerospace Lab, Toulouse, France

Fred Y. Hadaegh Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

R. John Hansman Massachusetts Institute of Technology, Cambridge, MA, USA

Heber Herencia General Electric Global Research, Clifton Park, NY, USA

Hernando Jimenez Georgia Institute of Technology, Atlanta, GA, USA

Romain Jobredeaux Georgia Institute of Technology, Atlanta, GA, USA

Andrew E. Johnson Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California

Raman K. Mehra Scientific Systems Company, Inc., Woburn, MA, USA

Piero Miotto Draper Laboratory, Cambridge, MA, USA

Issa A. Nesnas Jet Propulsion Laboratory, Pasadena, CA, USA

James D. Paduano Aurora Flight Sciences, Cambridge, MA, USA

Marc Pantel ENSEEIHT, Toulouse, France

Marco Pavone Stanford University, Palo Alto, CA, USA

Leena Singh Draper Laboratory, Cambridge, MA, USA

Joseph A. Starek Stanford University, Palo Alto, CA, USA

Timothy Wang Georgia Institute of Technology, Atlanta, GA, USA

Lesley A. Weitz Mitre Corporation CAASD, New Jersey, USA

Chapter 1
Spacecraft Autonomy Challenges for Next-Generation Space Missions

Joseph A. Starek, Behçet Açıkmeşe, Issa A. Nesnas and Marco Pavone

1.1 Introduction

In early 2011, in an effort to streamline future resource allocation and refine its plans, NASA's Office of the Chief Technologist (OCT) released a set of technology roadmaps with the aim of fostering the development of concepts and cross-cutting technologies addressing NASA's needs for the 2011–2021 decade and beyond [101, 103]. This set was organized into 14 technology areas (TA01 through TA14), divided into a total of 64 technology subareas. In an attempt to engage the external technical community and enhance the development program in light of scarce resources, NASA reached out to the National Research Council (NRC) to review the program's objectives and prioritize its list of technologies. In January 2012, the NRC released its report entitled "Restoring NASA's Technological Edge and Paving the Way for a New Era in Space," which reviewed an initial 320 technologies [48]. The NRC report revolved around three technology objectives:

J.A. Starek · M. Pavone (✉)
Stanford University, Palo Alto, CA, USA
e-mail: pavone@stanford.edu

J.A. Starek
e-mail: jstarek@stanford.edu

B. Açıkmeşe
University of Texas at Austin, Austin, TX, USA
e-mail: behcet@austin.utexas.edu

I.A. Nesnas
Jet Propulsion Laboratory, Pasadena, CA, USA
e-mail: nesnas@jpl.nasa.gov

E. Feron (ed.), *Advances in Control System Technology for Aerospace Applications*, Lecture Notes in Control and Information Sciences 460, DOI 10.1007/978-3-662-47694-9_1

- **Technology Objective A**: *Extend and sustain human activities beyond low Earth orbit*. Invest in technologies to enable humans to travel throughout the solar system, including surviving longer space voyages, arriving and working effectively at specific extraterrestrial destinations, and finally returning to Earth safely;
- **Technology Objective B**: *Explore the evolution of the solar system and the potential for life elsewhere (in situ measurements)*. Investigate technologies that enable humans and robots to perform in situ measurements on other planetary bodies as well as on Earth analogues (i.e. astrobiology);
- **Technology Objective C**: *Expand understanding of Earth and the universe (remote measurements)*. Develop technologies for capturing remote measurements from platforms that orbit or fly-by Earth and other planetary bodies, and from other in-space and ground-based observatories.

In its study, the NRC defined evaluation criteria that included assessments of technological benefit, alignment with NASA, non-NASA aerospace, and non-aerospace national needs, technical risk and reasonableness, sequencing and timing (factoring in requisite technologies), and development time and effort required to achieve each goal. By the final ranking, the NRC had whittled the selection to a group of 16 top priorities for technology.

While the NRC report provides a systematic and thorough ranking of the future technology needs for NASA, it does not discuss in detail the *technical* aspects of the prioritized technologies (which is clearly beyond the scope of the report). This chapter, building upon the NRC report and an earlier assessment of NASA's needs in terms of guidance, navigation, and control technologies [14], aims at providing such technical details for a selected number of high-priority technologies in the autonomous systems area. Specifically, this chapter focuses on technology area TA04 "Robotics, Tele-Robotics, and Autonomous Systems" and discusses in some detail the technical aspects and challenges associated with three high-priority TA04 technologies: "Relative Guidance Algorithms," "Extreme Terrain Mobility," and "Small Body/Microgravity Mobility."

This chapter is structured as follows. The rest of this section provides a high-level description of the high-priority technologies for TA04. Then, Sects. 1.2–1.4 focus, respectively, on technical discussions of "Relative Guidance Algorithms," "Extreme Terrain Mobility," and "Small Body/Microgravity Mobility," each categorized as top priorities of TA04 and which represent the key areas of expertise of the authors. Finally, Sect. 1.5 draws conclusions with a summary of the technical challenges facing the engineering community and the unsolved technical areas that must be addressed to help NASA meet its vision. Each technology section follows the same structure: *Scope*, *Need*, *State of the Art*, and *Challenges and Future Directions*.

1.1.1 High-Level Challenges and High-Priority Technologies for Space Autonomous Systems

While the guidance, navigation and control of spacecraft has resulted in numerous successful space missions, its use in fully autonomous operations has thus far been limited, with mission planners often opting for ground-in-the-loop interventions for maneuver refinements and corrections, wherever possible. Where ground-in-the-loop control is not feasible, as in the cases of rendezvous about other planets or atmospheric entry, descent and landing for instance, autonomous operations are often restricted to minimal scope in order to minimize the impact of a very costly validation and verification process. In spite of numerous autonomous operation successes, a number of anomalies have occurred during shuttle operations [57] and other recent autonomous demonstration missions, e.g. [20, 38, 65, 77], that point to the need for development and maturation in this area. This serves to illustrate the degree of difficulty of autonomous navigation and control in space applications and on a broad scale the significant challenges that must be overcome in aerospace engineering.

NASA has repeatedly identified robotic, autonomous, and sensing systems as enabling technologies over its history, spanning as far back as the Gemini program in the 1960s [108]. For spaceflight, many valuable proposed technologies, including real-time autonomous decision-making, opportunistic science, and human-robotic cooperation, are being investigated but have not yet been flight-tested. Analogously, for roving applications, the capability does not yet exist for traversing extreme lunar, Martian, or dusty terrains, including polar cold traps, high-grade surfaces, and microgravity environments [9]. The advancement of robotics and autonomous systems will be central to the transition of space missions from current ground-in-the-loop (geocentric) architectures to self-sustainable, independent systems, a key step necessary for outer-planet exploration and for overcoming the many difficulties of interplanetary travel [123]. Drawing similar conclusions in their technological report, the NRC highlighted TA04 "Robotics, Tele-Robotics, and Autonomous Systems" specifically as a high-priority technology area, recognizing its importance in broadening access to space and expanding humanity's presence in the solar system.

The roadmap for TA04 was broken into seven technology subareas: sensing and perception; mobility; manipulation; human-systems integration; autonomy; autonomous rendezvous and docking (AR&D); and robotics, tele-robotics, and autonomous systems engineering. Within this context, the NRC identified the following six top challenges for robotics and autonomous systems (quoted from [48]):

- **Rendezvous**: develop the capability for highly reliable, autonomous rendezvous, proximity operations, and capture/attachment to (cooperative and non-cooperative) free-flying space objects;
- **Maneuvering**: enable robotic systems to maneuver in a wide range of NASA-relevant environmental, gravitational, and surface and subsurface conditions;

- **In Situ Analysis and Sample Return**: develop subsurface sampling and analysis exploration technologies to support in situ and sample return science missions;
- **Hazard Avoidance**: develop the capabilities to enable mobile robotic systems to autonomously and verifiably navigate and avoid hazards;
- **Time-Delayed Human-Robotic Interactions**: achieve more effective and safe human interaction with robotic systems (whether in proximity or remotely) that accommodates time-delay effects;
- **Object Recognition and Manipulation**: develop means for object recognition and dexterous manipulation that support engineering and science objectives.

This list is consistent with the recommendations of NASA's previous Vision for Space Exploration [92], the recommendations referenced for NASA Automated Rendezvous and Capture operations [108], the lessons learned from Apollo Guidance Navigation and Control (GN&C) [84], and the technology priorities described for the future of rovers [9].

In light of these six challenges, and of the general technology objectives presented at the beginning of this section, eight specific high-priority technologies were identified in the TA04 Roadmap:

- **Technology 4.2.1**, *Extreme Terrain Mobility*.
- **Technology 4.2.4**, *Small Body/Microgravity Mobility*.
- **Technology 4.3.2**, *Dexterous Manipulation*.
- **Technology 4.3.6**, *Robotic Drilling and Sample Processing*.
- **Technology 4.4.2**, *Supervisory Control*.
- **Technology 4.5.1**, *Vehicle Systems Management and Fault Detection Isolation and Recovery (FDIR)*.
- **Technology 4.6.2**, *Relative Guidance Algorithms*.
- **Technology 4.6.3**, *Docking and Capture Mechanisms/Interfaces*.

Technology advances in these areas will help towards accomplishing Technology Objectives A, B, and C by improving access to space, increasing available mass-to-surface, and enhancing robotic maneuvering capabilities, autonomous rendezvous and docking, and precision landing, all of which were labeled top engineering roadblocks that must be overcome to meet NASA's goals.

The remainder of this chapter is devoted to clarifying precisely what needs to be addressed for the three specific subcategories "Relative Guidance Algorithms," "Extreme Terrain Mobility," and "Small Body/Microgravity Mobility," according to the best knowledge and expert opinions of the authors. The benefits, current state of the art techniques, and technical aspects and challenges of each are discussed in detail to better prepare the technical community for delivering on these advancements and meeting the needs of next-generation space missions.

1.2 Relative Guidance Algorithmic Challenges for Autonomous Spacecraft

Relative guidance algorithms were categorized by the NRC as the top-ranked technology for robotics, tele-robotics, and autonomous systems; their improvement would mark a tremendous milestone for robustifying and augmenting current capabilities in autonomous guidance and control.

1.2.1 Scope

Guidance is the process of real-time planning of spacecraft state trajectories in both translational and rotational motion. This involves computing desired sets of translational and rotational states and corresponding control forces and torques as a function of time. Control, or more specifically feedback control, is responsible for following these trajectories based on real-time state updates in the presence of disturbances, measurement noise, and model uncertainties. Together they are referred to as Guidance, Navigation, and Control (GN&C, or just G&C). This section addresses the technical details and challenges for *relative guidance* of autonomous spacecraft in four key space-based areas:

- **Planetary Entry, Descent, and Landing (EDL)**
- **Proximity Operations for Primitive Bodies**
- **Autonomous Rendezvous and Docking (AR&D)**
- **Autonomous Inspection and Servicing (AIS)**

In each of these applications, the guidance problem can be posed as an optimal control problem with dynamics describing the motion of the spacecraft as well as constraints on the vehicle state and controls. This can be expressed generically as follows:

Problem G&C: *Generic Autonomous Spacecraft Guidance Optimal Control Problem*

$$\begin{aligned} \min_{t_f,u} \quad & J(x(t),u(t),t) = K(x(t_f),t_f) + \int_{t_0}^{t_f} L(x(t),u(t),t)\,\mathrm{d}t \\ \text{subject to} \quad & \dot{x}(t) = f(x(t),u(t),t) \\ & u(t) \in U(t) \\ & x(t) \in X(t), \quad \text{for all } t \in [t_0,t_f] \end{aligned}$$

where $x \in \mathbb{R}^n$ is the state of the spacecraft, $u \in \mathbb{R}^m$ is the control input, $t \in \mathbb{R}$ is time, $J : \mathbb{R}^{n+m+1} \to \mathbb{R}$ is the cost-functional (which combines terminal and incremental additive cost functions K and L), $f : \mathbb{R}^{n+m+1} \to \mathbb{R}^n$ defines the

dynamics, and $U : \mathbb{R} \to \mathbb{R}^m$ and $X : \mathbb{R} \to \mathbb{R}^n$ are set-valued maps defining spacecraft control and state constraints. Due to the existence of system dynamics and constraints, the resulting optimal control problem must be solved numerically [15, 45] via an optimization algorithm after a proper discretization [66, 128]. To meet the guidance challenges of next-generation space missions, onboard algorithms will need to meet the following specifications:

- **Real-time implementability**: Algorithms must be implemented and executed on real-time processors in a reasonable amount of time.
- **Optimality**: Given that feasible solutions exist, an optimal solution $x^*(t)$ that minimizes (at least approximately) the cost function J is desired.
- **Verifiability**: There must be design metrics that accurately describe the performance and robustness of GN&C algorithms, with accompanying methods for verifying these metrics.

1.2.2 Need

Autonomous spacecraft maneuvering, especially in *proximity* of artificial objects (e.g. satellites, debris, etc.) or solar system bodies (e.g. asteroids, comets, irregular satellites, etc.), is a key enabler for the majority of future NASA missions [48, 101]. In some cases this arises from obvious physical mission constraints, notably signal transmission delays to and from Earth. A very good example is Mars atmospheric Entry, Descent, and Landing (EDL), arguably one of the most tightly-constrained control sequences in modern spaceflight, which prohibits human intervention due to a nearly 26 minute two-way signal communication time that far exceeds the typical seven-minute descent duration. Similarly, close proximity operations around small bodies, many of which travel beyond the extent of Mars orbit, require autonomous guidance for the same reason. In other instances, the need for autonomy derives from a desire to increase mission frequency, robustness, and reliability. This includes Low Earth Orbit missions, such as Autonomous Rendezvous and Docking (AR&D) and Autonomous Inspection and Servicing (AIS). As space access improves through commercialization, the increased scheduling conflicts and labor overhead associated with ground-in-the-loop spacecraft guidance are expected to become prohibitively expensive. The risk of human error will increase as well. Spacecraft autonomy can circumvent these issues, as well as enable greater mission variety and improve the commercial and scientific return from space.

1.2.3 State of the Art

Current state-of-the-art techniques for autonomous spacecraft maneuvering include Apollo guidance (particularly phase-plane logic, glideslope, and sliding-mode controllers), Model Predictive Control (MPC) [2, 17, 27, 99, 104], and Artificial

Potential Functions (APFs) [7, 16, 87]. Unfortunately, such techniques, while valuable in static uncluttered settings, appear to fall short in scenarios where time-varying constraints (such as neighboring debris or other spacecraft), logical modes (e.g., safety modes), and complex maneuvering (e.g., terrain sampling or manipulation) become key features of the problem setup. In these cases, robotic motion planning techniques, though currently unproven in spaceflight, could provide a valuable alternative [81, 118]; they are hence discussed here as well. Brief synopses of each these methods are presented in Sects. 1.2.3.1–1.2.3.4 below, together with highlights of recent autonomous demonstration missions in Sect. 1.2.3.5.

1.2.3.1 Apollo Guidance

The COLOSSUS Program, developed by MIT for NASA's Apollo Program, called upon three Digital AutoPilot (DAP) systems to stabilize and control the Apollo Command Service Module (CSM) as part of its Primary Guidance Navigation and Control System (PGNCS) [132]. The techniques used, now considered part of *classical* control, formed one of the earliest successful deployments of spacecraft autonomy. Block-diagram schematics of the Apollo CSM control logic can be seen in Fig. 1.1. These Digital AutoPilot systems are each briefly described to provide context for more modern control techniques:

- **Orbital Re-entry Digital Autopilot (ENTRY DAP)**: assumed control of the Command Module (CM) after separation from the Service Module (SM) and handled all Command Module flight maneuvers beginning with reorientation into Entry

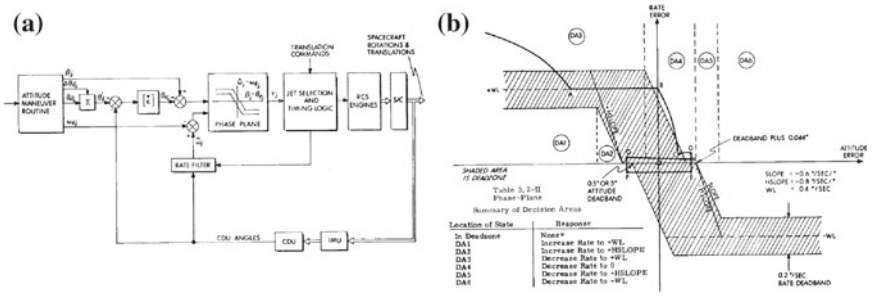

Fig. 1.1 Illustrations of one of the earliest successful spacecraft autonomous control systems for the NASA Apollo Command Service Module (CSM). **a** Block diagram logic used by Reaction Control System thrusters to control CSM attitude. Here θ_d represents the reference attitude angle, θ_e the attitude error, β an attitude bias, ω the attitude rate, and $\hat{\omega}$ the attitude rate estimate. **b** Phase-plane logic schematic. For double-integrator models, this design can be shown to drive the rate and attitude errors plotted on the x- and y-axes to the box-like area near the origin. The logic works by breaking the plane into disjoint zones, inside of which the spacecraft is pre-programmed to torque positively or negatively (*solid white areas*) or coast (*shaded region*); horizontal lines represent zero-acceleration trajectories or "coasting arcs," while parabolas represent lines of constant acceleration. *Images courtesy of* [132]

attitude up until drogue chute deployment. The autopilot called pairs of thrusters distributed along the rim of the base of the Command Module, as well as an additional pair near the tip for pitch-down control. The first phase of operation marked exoatmospheric mode, using various combinations of rate damping, attitude-hold, and attitude-control depending on the pitch angle value. Phase-plane logic controllers[1] (attitude rate versus attitude error) with biased deadzones drove the system to desired error tolerances. Once drag rose above 0.05 g, atmospheric mode was initiated. In this regime, roll control was maintained using a complex phase plane incorporating a straight control line, maximum velocity boundaries, and constant-acceleration switching lines, while yaw and pitch reverted to rate-damping using a yaw rate versus roll rate phase plane logic and a simple relay with deadband, respectively. The purpose of ENTRY DAP was to maintain the component of lift in the trajectory plane needed to target a desired landing site given the vehicle's current position and velocity.

- **Reaction-Control System Digital Autopilot (RCS DAP)**: Responsible for controlling the attitude and attitude rates of the Command Service Module during coasting flight, both with or without the Lunar Module (LM) stage attached. The Digital AutoPilot employed for pitch, yaw, and roll control four clusters, called quads, of four Reaction-Control System thrusters each, using a phase-plane logic controller with nonlinear switching lines, a central deadband, and built-in hysteresis. The timing and firing commands of individual thrusters were issued by a thruster-selection logic responsible for resolving Digital AutoPilot rotation commands with translation commands and executing them as economically as possible according to the distribution of functional thrusters available. A second-order angular-rate Kalman filter was used to compute estimates of angular velocity by weighted sum of (1) extrapolated values of previous estimates and (2) derivations from gimbal angle measurements.
- **Thrust-Vector-Control Digital Autopilot (TVC DAP)**: Used to control the Command Service Module during powered flight, both with or without the Lunar Module attached. Pitch and yaw were adjusted by actuating the gimbal servos of the main engine, while a separate autopilot called TVC ROLL DAP controlled the Command Service Module attitude and rate about the roll axis during powered flight via the Reaction-Control System thruster quads. TVC DAP fed estimates of attitude rate and angle errors to pitch and yaw compensation filters, with various combinations of attenuation and phase stabilization depending on the configuration of the Command Service Module due to the changes in overall center-of-mass position, bending modes, and fuel slosh instabilities. TVC ROLL DAP used an

[1] Phase-plane controllers are typically used to determine stabilizing on-off control inputs for one degree-of-freedom differential systems by defining a coordinate plane of two state variables (typically a state error and its corresponding state rate error) and a set of *switching curves* with accompanying "deadband," "hysteresis," etc. in such a way as to partition the space into disjoint control regions that drive the system to within certain limits of the coordinate plane origin. Figure 1.1 shows a schematic of the phase-planes used by the Apollo missions.

adaptation to the phase-plane switching logic of RCS DAP in free flight, modified with ideal parabolic switching curves for roll axis attitude-hold within a small tolerance. A number of logical constraints were additionally enacted in order to conserve fuel and minimize the risk of thruster failures.

1.2.3.2 Model Predictive Control

Model predictive control (MPC) is a feedback law based on the repeated solution of an optimal control problem (i.e. Problem G&C) that uses an assumed dynamics model f and the current state set as the initial condition. This problem is solved to yield a finite-horizon control trajectory that optimizes the predicted state response over the duration of a planning period or *time horizon*. Once solved, however, only the initial control segment is actually applied, after which the problem is reinitialized and the process repeats until convergence to the goal. This characteristic renewal procedure over a repeatedly updated horizon is what gives MPC its other common names: *receding horizon optimal control* or *moving horizon optimal control*. This concept allows one to design a feedback controller on the basis of nearly any open-loop optimal control approach, improving its robustness and imparting it the ability to handle disturbances and mitigate error growth. Even without prior disturbance modeling, one can demonstrate under appropriate assumptions that MPC can lead to closed-loop stability and state convergence to the target [86]. Other advantages of MPC include the ability to handle pointwise-in-time state and control constraints, the capability to withstand time delays, and reconfiguration in the presence of degradations and failure modes [25, 26]. As the robustness properties of MPC are contingent on fast resolvability, open-loop controllers for vehicle guidance for the most part must be restricted to convex optimization routines. Another common choice for use with MPC schemes in autonomous spacecraft guidance is Mixed-Integer Linear Programming (MILP) [22, 110], which are essentially solvers for linear optimization problems with embedded discrete variables to handle simple logical constraints such as mode switching and collision-avoidance.

1.2.3.3 Artificial Potential Functions

The artificial potential function (APF) method [7, 16, 87] transforms the guidance problem into particle motion within a potential field. Attractive potentials are used for goal regions, while repulsive potentials are used for obstacles—the value of occupying a particular state is then represented by the sum of individual terms. A gradient ascent/descent routine is often called to trace a feasible path from any initial state, which, when tuned appropriately, will safely circumnavigate neighboring obstacles and converge to a goal. Alternatively, an optimal control problem may be formed to plan a path that minimizes the path integral along the gradient force field (analogous to minimum-work in physical systems). The approach benefits greatly from the ability to react in real-time to environmental changes through adjustments in individual

potential functions. Some difficulty lies in adjusting each function such that the spacecraft behaves as desired (i.e. ensuring sufficient margin from obstacles, rapid convergence, etc.). However, the main drawback of APFs is their well-known susceptibility to converge to local minima, which cannot be avoided without additional heuristic techniques. This tendency can be mitigated by attempting random walks out of local wells, or instead relying on a global optimization routine for open-loop control, with an artificial potential function method called for closed-loop feedback (i.e. trajectory-following, bubble methods [109], or real-time path modification [23], for instance).

1.2.3.4 Spacecraft Motion Planning

Motion planning constitutes a class of algorithms used to generate sequences of decisions, called *plans*, that safely navigate robots from given initial states to a set of target states called *goals*. The framework is sufficiently general that it applies equally well to spacecraft and rovers as it does to traditional robots [80]. Motion planning techniques can be classified into two categories: *exact* (combinatorial) algorithms and *approximate* (sampling-based) algorithms. Exact approaches develop a strategy based on an explicit representation of the unsafe region of the state space, which allows them to guarantee a solution if one exists. Techniques typically involve the formation of roadmaps, which are topological graphs that efficiently capture the connectivity of points in the obstacle-free state space. Exact algorithms are often limited to problems of low-dimensionality, polygonally-shaped obstacles, and static environments. Sampling-based algorithms, on the other hand, forgo explicit construction of the unsafe state space and instead explore pathways via a sampling procedure, with safety verified by a "black-box" collision-detection routine. In many ways this idea is computationally advantageous; however, it has the obvious drawback that weaker notions of correctness and completeness must be tolerated—existence of solutions cannot be guaranteed in finite time without drawing an infinite set of samples. Prominent examples of sampling-based algorithms include Probabilistic Roadmaps (PRM) [76], the family of Rapidly-Exploring Random Tree (RRT) algorithms [80, 81], and Fast Marching Trees (FMT*) [68] together with its kinodynamic versions [115, 116]. Sampling-based motion planning algorithms such as these have been shown under mild conditions to quickly and uniformly explore the collision-free state space. Some of them (e.g., RRT* [75] and FMT* [68]) have the added benefit of *asymptotic optimality*; that is, they guarantee convergence to an optimal solution as the number of samples goes to infinity.

1.2.3.5 Recent Demonstration Missions

A handful of autonomous maneuvering missions have demonstrated at least a few of these state-of-the-art methods (combined with digital logic). Prominent examples include JAXA's ETS-VII [77, 102], AFRL's XSS-10 [38], DARPA's Orbital Express

[65], NASA's DART [111], and JAXA's Hayabusa [51, 135]. Sadly, notable guidance and control anomalies and mishaps occurred during the latter three missions, in some cases spelling their end [20, 77, 135]. The DART spacecraft, for instance, began using much more propellant than expected during proximity operations and initiated a series of maneuvers for departure and retirement, but eventually collided with the MULBCOM satellite [20]. This suggests that presently autonomous spacecraft navigation and maneuvering, even in static environments with well-understood dynamics, is still in its technological infancy [22, 48].

1.2.4 Challenges and Future Directions

Many technical hurdles remain to be solved before autonomous spacecraft relative guidance can become a mature technology. This section begins in Sect. 1.2.4.1 with a summary of the most important relative guidance challenges concerning general spaceflight, from which the discussion is specialized to two key areas: Planetary Entry, Descent, and Landing (EDL) in Sect. 1.2.4.2, and Proximity Operations in Sect. 1.2.4.3, a blanket term that encompasses Autonomous Rendezvous and Docking (AR&D), Autonomous Inspection and Servicing (AIS), and close-range operations about primitive bodies.

1.2.4.1 General Relative Guidance Challenges

The main guidance challenge for next-generation autonomous spacecraft is to solve the guidance and control problem (Problem G&C) with the appropriate dynamics and constraints onboard and in real-time. This onboard capability will enable the execution of missions with a much higher level of autonomy, ultimately prolonging mission times, increasing mission frequencies, decreasing costs, and returning more scientific data. Furthermore, it will allow the spacecraft designer to fully utilize the performance envelope, thereby maximizing achievable performance.

The most important technical challenges to meet this ambitious goal are:

- **Implementability**: Developing robust, real-time implementable, and verifiable onboard optimization algorithms for the solution of Problem G&C;
- **Verifiability**: Developing design metrics and verification and validation methods for real-time optimization-based guidance and control algorithms;
- **Formation Flight**: Extending guidance techniques to multiple collaborative vehicles;
- **Testing**: Demonstrating next-generation autonomous algorithms in representative flight testing.

Meeting these challenges will require development of new mathematical formulations and algorithms for *robust, real-time implementation* and for ground analysis. For example, if one can express Problem G&C as a convex optimization problem for

a given application, then one can employ Interior Point Method algorithms (IPMs) to achieve globally-optimal solutions [21, 98], as well as improve runtime execution speeds by 2–3 orders of magnitude [85]. This clearly motivates the use of real-time convex optimization for relative guidance whenever possible, either in the ideal case through lossless convexification (as in [58], for example) or through reasonable convex approximations, particularly for complex, difficult, or hazardous problems where the important need is a reasonably-good feasible solution obeying all mission constraints.

Verifiability of solution methods is also another interesting and important challenge. In classical linear feedback control, one has prescribed design metrics such as phase and gain margin specifications that serve as useful targets in the design of feedback controllers. It is relatively straightforward to check whether these requirements are satisfied at design time. In the case of more complex guidance algorithms, on the other hand, such general metrics do not exist. A good example can be given in the context of Mars precision landing, for which the trajectory designer must direct a vehicle from any initial state at the end of the parachute phase to a target on the Mars surface with zero velocity. Suppose the expected set of initial conditions at the start of powered descent is I_{pd}. Define I_c as the set of all initial conditions from which the lander can reach the target, assuming fixed control parameters such as propellant mass fraction, thrust-to-weight ratio, fuel consumption rate, etc. Then verification simply requires checking whether the following set inclusion relationship holds:

$$I_{pd} \subseteq I_c.$$

The next question is how to generate I_c for a given set of design parameters. Exact approaches devised for discrete systems conduct systematic searches through a finite state-space, collecting information about reachable sets and the properties of the states traversed [33, 136]. However, due to the exponential growth in state-space size with dimension, this is infeasible for continuous or high-dimensional systems. In such instances, one must resort to approximate techniques, collectively called *reachability analysis*, for computing the set I_c. Clearly one approach is exhaustive search of sample points in the set; however, this is very time consuming and not usable at design time. Many efficient alternatives have been developed, however, including (1) optimal control and Lyapunov-based theory [53], (2) state abstraction, in which state-space size is reduced by grouping states together through omission of less useful details [82], (3) propagation of conservative over-approximations to the true sets [120], and (4) convexification of Problem G&C through exploitation of the problem structure.

The next challenge is to extend guidance techniques to *multiple collaborative vehicles*. This complicates problem formulation and solution methods, rendering complex problems even more so when real-time solutions are demanded. The difficulty lies in the coupling between the safety of each vehicle to the future trajectories of all of its neighbors. This is often resolved in the literature by forming a hierarchy in planning, in which one vehicle neglects its neighbors and develops a plan, the second then develops a plan assuming the first's path is fixed, the third designs a path

under the consideration of the first and second, and so on. However, this technique makes the key assumption that all current and future state information of each vehicle is freely communicable to all other vehicles. As this illustrates, multiple vehicle collaboration and guidance entails the need for communication and scheduling. This generates the question of which control architecture, or rather communication architecture, is most suitable to the application. Control architectures vary from either fully individualized control called *distributed control*, or fully dependent control called *centralized control*, in which one vehicle or mothership determines the plans for all other vehicles. A number of methods have been developed to handle multiple spacecraft guidance, including specialized formation or coordination controllers (e.g. [13, 127]), passive/active relative orbit formulations (e.g. Clohessy-Wiltshire-Hill equations, halo orbits about libration points) [5, 55], optimal formation reconfigurations [113], rigid body or quasi-rigid body rotation planning [19], potential-based methods [31] and behavioral planning [67]. Much of the literature focuses on simple formation flight architectures, such as leader-follower formations. Formation flight and collaborative decision-making remain highly active areas of research.

In summary, the key for autonomous relative guidance is having robust solution techniques that can be made efficient for real-time implementation. Though some of these techniques may not be implementable on current space-qualified flight computers, the natural increase in onboard computational power and the use of multiple processors with algorithm parallelization could enable their use in the not-too-distant future. Therefore, priority in research must first be to develop robust solution methods for the right problems with appropriate constraints. Subsequently, these algorithms should be customized for flight implementation. Finally, a rigorous process (preferably combined with flight testing) should be established for solution verification and validation.

The following Sects. 1.2.4.2–1.2.4.3 specialize the general challenges of this subsection to planetary EDL and proximity operations, each illustrating the types of difficult, mission-critical control maneuvers that typically lie at the cutting edge of modern spacecraft autonomy research.

1.2.4.2 Challenges for Planetary Entry, Descent, and Landing

This section focuses on the GN&C challenges associated with planetary missions, first highlighting the difficulties of Mars and Moon landings before extending to other planetary bodies.

The main purpose of any planetary landing GN&C system is to execute a controlled deceleration from orbital or interplanetary velocities to near-rest conditions. For a typical Mars EDL mission, this begins with an entry phase (see Fig. 1.2) that cancels most of the planetary relative velocity. Once slowed to supersonic speeds, a parachute is deployed. Then at a prescribed altitude (e.g. approx. 2 km for the Mars Science Laboratory (MSL)), the parachute is discarded and the Powered Descent (PD) phase is initiated. At this point, passive descent during the parachute phase coupled with atmospheric density and weather uncertainties cause the predicted

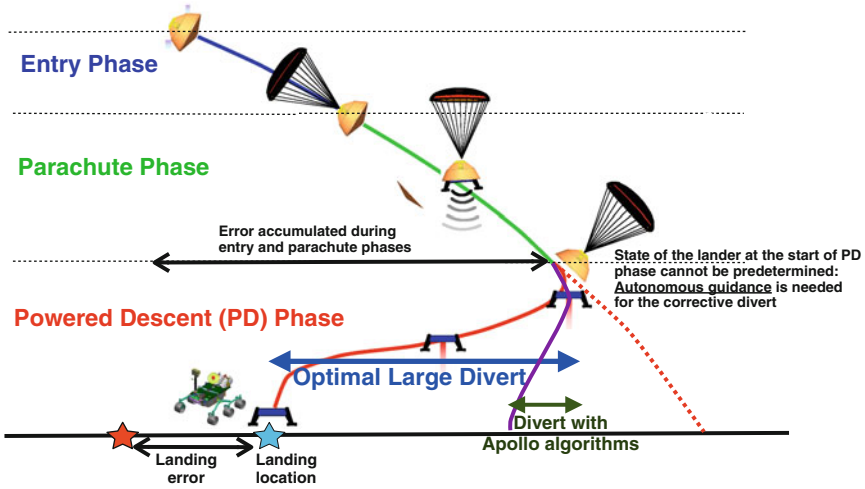

Fig. 1.2 Optimal Powered Descent Guidance (PDG) will enable planetary precision landing. These algorithms search over all physically possible diverts to find a fuel optimal one, significantly improving divert capability over current state-of-the-art onboard algorithms

positions and velocities relative to the target to disperse significantly (e.g. on the order of 8–10 km with a velocity trigger (used during the MSL mission) or 5–6 km with a range trigger at the start of the parachute phase [130]). To achieve precision landing (roughly 1 km of position error or less at touchdown), an autonomous, real-time Powered Descent Guidance (PDG) algorithm is required to continuously redirect the vehicle towards the surface target. In manned missions, the challenges are magnified and still largely unsolved. Though robotic landers can weigh as little as about 2 metric tons, they are expected to require as much as 50 metric tons in the manned case, which essentially precludes any passive means of deceleration. Successful planetary descent of such heavy landers will necessitate active control starting at supersonic speeds early in the EDL entry phase.

For planets or moons without an atmosphere, a solid rocket is typically used for lander deceleration in a Braking Burn phase, which is then followed by a Powered Descent controlled by liquid fuel propulsion for final landing. The process is complicated by the fact that solid rockets must burn all of their fuel to completion once initiated. Significant uncertainty generally exists in the associated burn-time, leading to uncertainty in the vehicle state relative to the target at the end of the burn phase. Analogous to atmospheric entry and descent, the Powered Descent phase is designed to correct for any error accumulated during the solid rocket phase; autonomous PD guidance algorithms must be called to guide the lander as close as possible to the given surface target in order to achieve optimal landing accuracies.

In all planetary or lunar landing missions, the associated autonomous guidance problems for translational motion can be expressed as highly-constrained optimal control problems [3, 18, 119]. Written in terms of Problem G&C, the guidance

equations can be represented as follows: Let $x = (x_1, x_2, x_3)$, where $x_1 \in \mathbb{R}^3$ and $x_2 \in \mathbb{R}^3$ are the position and velocity, respectively, relative to the target in the rotating frame of Mars, and $x_3 > 0$ is the lander mass. The guidance problem can be formulated as,

$$\dot{x} = f(t, x, u) = A(\omega)x + B\left(g(x_1) + \frac{u}{x_3}\right)$$

$$X(t) = \begin{cases} \{x \mid x(t) = x_0\} & \text{for } t = t_0 \\ \left\{x \;\middle|\; \gamma \hat{n}^{\mathrm{T}} x_1(t) \geq \|T x_1(t)\| \text{ and } \|x_2(t)\| \leq \bar{V}\right\} & \text{for } t \in (t_0, t_f) \\ \{x \mid Hx(t) = a\} & \text{for } t = t_f \end{cases}$$

$$U(t) = \left\{u \;\middle|\; \rho_1 \leq \|u\| \leq \rho_2 \text{ and } \hat{n}^{\mathrm{T}} u \geq \|u\| \cos\beta\right\}$$

where $A(\omega)$ defines the Newtonian motion in a rotating frame with fixed rotation rate ω and $g : \mathbb{R}^3 \to \mathbb{R}^3$ defines the gravitational field. Here $X(t)$ captures initial and final state constraints, along with constraints during the maneuver (known as "glide slope" constraints [18]). The control vector norm has both an upper bound ρ_2 and a nonzero lower bound ρ_1 due to the fact that the thrusters cannot be operated reliably below a prescribed value. The other constraint on the thrust vector is that it has to remain in a cone defined by the unit surface norm, $\hat{n} \in \mathbb{R}^3$, and half-angle β in order to avoid any possibility of rotating the lander excessively, which could interfere with sensors that must be directed towards the surface. Note that the vehicle is assumed to be a point mass with a thrust vector attached to it. This simplification is a valid one since the attitude control authority and bandwidth are much higher than those for translation, so that the vehicle can quickly adjust its orientation any time a thrust vector is commanded.

As one does not know with certainty the initial relative state x_0 into which the lander will inserted from interplanetary flight, this problem must be solved autonomously in real-time on-board the spacecraft. To accommodate this, some authors have developed simplified, approximate versions of this problem that lend themselves to analytical solutions [40, 78, 88, 91, 122]; however, to certify precision guidance across the entire landing envelope (the initial conditions from which it is physically possible to land), one must explicitly account for the full set of constraints. Unfortunately, the control constraints $U(t)$ define a non-convex set (due to $\rho_1 > 0$), which further emphasizes, as previously described, the benefits of lossless convexification techniques [1, 3, 18, 59] and convex relaxations that are solvable using Interior Point Methods (IPMs). The lossless convexification-based algorithm [3] has already been demonstrated successfully by NASA JPL. See [72–74] for flight videos. These test flights successively demonstrated increasingly aggressive optimal divert maneuvers, starting from 100 m for unoptimized flight and ending with the longest possible optimal divert of 750 m, showing strong evidence that performance boundaries can be pushed to the ultimate physical limits via onboard optimization.

1.2.4.3 Challenges for AR&D, AIS, and Proximity Operations About Primitive Bodies

In AR&D, AIS, and close proximity operations near space objects (such as spacecraft or primitive celestial bodies) that are cooperative or otherwise, the primary guidance objective is to compute a state trajectory that safely brings the spacecraft as close as needed (including docking) to its target object while consuming as little fuel as possible and avoiding any nearby hazards. In general, this introduces many difficult, non-convex trajectory constraints into the optimal control problem given by Problem G&C [110]. A detailed list of examples are included here to illustrate the point:

- **Constraining sensor field-of-view**: Often in proximity operations it is necessary to keep the target, spacecraft or primitive body in the field-of-view (FOV) of onboard sensors. This can be represented mathematically as:

$$\hat{n} \cdot (r - r_T) \geq \cos\alpha \|r - r_T\|$$

  where $\hat{n}$ is the unit vector describing the sensor boresight, r is the position vector of the spacecraft, r_T is the position vector of the target body, and α is the half-cone angle defining the FOV. This constraint *couples* the attitude and translational dynamics through $\hat{n}$, which is determined by the orientation of the spacecraft. To see this explicitly, if the position vectors are resolved in a rotating reference frame, e.g. LVLH (Local-Vertical-Local-Horizontal), and $\hat{n}$ is resolved in a spacecraft fixed frame, then the equation above can be re-expressed as,

$$(r - r_T)^{\mathrm{T}} C(q)\hat{n} \geq \cos\alpha \|r - r_T\|$$

  where q is the quaternion describing the attitude of the spacecraft, and $C(q)$ is the directional cosine matrix that takes a vector in the spacecraft body reference frame to the LVLH frame.

- **Avoiding plume impingement**: Impingement of thruster exhaust plumes on neighboring spacecraft poses a serious threat that can jeopardize sensitive optical devices, generate large force perturbations and disturbance torques, and disrupt thermal blankets and coatings [56]. Prevention requires restricting thrusters that are pointed towards neighboring vehicle(s) from firing below a prescribed relative distance. Unfortunately, this imposes a loss of control authority and necessitates special guidance or escape plans that never apply thrust forces directed away from the target when in close proximity. This constraint exists for primitive bodies as well due to scientific contamination concerns, i.e. during sample return.
  Represented mathematically, plume impingement constraints can be stated as, for $i = 1, \ldots, n_t$,

$$u_i = 0 \quad \text{when} \quad \begin{cases} (r - r_T)^{\mathrm{T}} C(q)\hat{t}_i \geq \cos\beta_p \|r - r_T\| \\ \|r - r_T\| \leq R_p \end{cases}$$

where n_t is the number of thrusters, u_i is the thruster force command, $\hat{t}_i$ is the unit vector for the thruster direction in a spacecraft fixed frame, β_p is the plume cone angle, and R_p is the maximum effective plume radius (plume is effective if the target is in this radius).

- **Handling thruster force upper and lower (impulse bit) bounds**: Due to fuel energy storage limitations and nozzle design constraints, it is evident that all thrusters have finite upper bounds on the amount of force that they can provide. There is also a minimum nonzero force or impulse (impulse bit) that imposes a lower bound on deliverable thrust; this means that arbitrarily small forces cannot be applied using thrusters. This limits the control precision that can be achieved, which can be critical during docking or proximity operations.
  These constraints, when using force commands, can be expressed as, for $j = 1, \ldots, n_t$,

  $u_j \in \{0\} \cup [u_{j,1},\ u_{j,2}]$ where $u_{j,1} > 0$ and $u_{2,j} > u_{j,1}$ are min. and max. thrusts

- **Avoiding collisions**: Nothing can be more catastrophic to a spacecraft mission than collisions, which damage or destroy participating vehicles and often mark an immediate mission failure. For AR&D and AIS, the collision avoidance constraint can be described as follows,

  $$r - r_T \notin \Omega_c$$

  where r_T is the position vector for the target and Ω_c is a set of relative positions that lead to collisions. For a two-spacecraft scenario as in AR&D and AIS, this can be simply a collision ball defined as $\Omega_c = \{z : \|z\| \leq R_c\}$ for some prescribed value of R_c. In proximity operations around primitive bodies, this region can be much more complicated due to their irregular and often ill-defined shapes.

- **Providing required thruster silence times**: As thrusters fire, large errors are introduced into the state estimation due to process noise at the instance of firings. Often after each burn there must be a prescribed period of thruster silence to allow the state estimator to filter this noise and re-converge to a prescribed level of accuracy.
  One approach to impose prescribed thruster silence is to have zero controls in prescribed time periods during a maneuver, i.e.,

  $$F_i(t) = 0, \ \forall i = 1, \ldots, n_t \quad \text{when} \quad t \in \bigcup_{j=1,\ldots,n_s} T_j, \tag{1.1}$$

  where $T_j, \ j = 1, \ldots, n_s$ form a disjoint set of zero-thrust time intervals.

- **Using minimal fuel**: Every spacecraft mission is constrained by a finite supply of fuel that must be transported with the scientific payload. The high cost of access to space currently inhibits the ability to refuel or resupply spacecraft, for the most part isolating them and imposing a mission lifetime synonymous with remaining fuel. This not only affects mission lifetime but also mission capability. For example, AIS

missions seek to maximize total inspection time, which has a direct correspondence with maximizing fuel efficiency. For primitive bodies, using fuel efficiently implies longer observation times and more attempts for surface contact.

- **Guaranteeing safety**: A trajectory solution is needed that can ensure spacecraft and mission safety at all times, for both the vehicle and its neighbors. Guarantees are typically classified into two forms: *passive safety*, in which coasting arcs emanating from points along the nominal guidance trajectory are certified as safe, or *active safety*, in which safe actuated abort sequences called collision avoidance maneuvers (CAMs) are enforced [46, Sect. 4.4]. In either case, hard (deterministic) safety constraints are required to guarantee viable escape options in the event of thruster allocation errors (misfirings, stuck-on or stuck-off valves, canted nozzles, etc.), unexpected environmental changes and disturbances, or even complete system shutdown. Often in practice this is achieved through ad-hoc open-loop trajectory design (guided by significant technical expertise). However, an automated approach, potentially using optimal control techniques [22], positively-invariant sets [26, 54, 131], motion planning with safe samples [49], or some combination of all three [118], will be needed in the future in order to achieve truly autonomous AR&D and AIS capability.
- **Handling uncertainties**: Thruster firings, aerodynamic drag in low Earth orbits, solar radiation pressure, and camera measurements can introduce uncertainties in relative state knowledge and control accuracy. As the spacecraft nears its target, these uncertainties can induce violations in any of the aforementioned mission constraints. Conversely, relative state accuracy typically improves as relative separation decreases. Hence one should embed in autonomous guidance and control algorithms the capability to handle any expected uncertainty directly, i.e. one should incorporate strategies to handle all "known unknowns."

Due to potential coupling between translational and attitude dynamics, one must consider both sets of dynamics in Problem G&C. This complicates the problem due to the inherent nonlinearity in the attitude dynamics, leading to nonlinear equality constraints after discretization. Having nonlinear equality constraints means having non-convex constraints, causing the resulting parameter optimization problem to be a non-convex optimization problem. This complicates the numerical solution of Problem G&C significantly. Another source of non-convexity is the collision avoidance constraint; its incorporation can also dramatically complicate the solution algorithm for the same reason.

As a consequence of the nature of these constraints, convexification approaches for AR&D, AIS, and proximity operations appear less suitable in this case than for Entry, Descent, and Landing problems due to the errors incurred through relaxation. Hence new tools will be needed.

It is in this context precisely that motion planning algorithms (Sect. 1.2.3.4) have the potential to shine. Numerous studies have already been conducted assessing their feasibility for realistic spacecraft proximity operation scenarios [49, 50, 81, 106, 118]. Though not yet flown on spacecraft hardware, their efficacy has already been proven in real-world systems with challenging dynamics, namely onboard real-time

guidance of urban vehicles during the 2007 DARPA Urban Challenge. Several winning entrants to the 60-mi autonomous urban driving race used motion planning as their primary guidance logic, including CMU's winning Boss car with Anytime-D^*, Stanford's 2nd-place Junior car with hybrid A^*, and MIT's 4th-place Talos car with RRTs [24, 79, 83, 90, 126]. The ability of these algorithms to handle such diverse constraints while providing robustness certificates in real-time applications is promising for autonomous spacecraft control.

1.3 Extreme Mobility

Among the top technical challenges of technology area TA04 is *maneuvering* in diverse NASA-relevant environments—a task that encompasses a wide range of environmental, gravitational, and topographical conditions. Space exploration in such environments is enabled by three types of maneuvering: surface mobility, above-surface mobility, and below-surface mobility. We focus here on the part of surface mobility called *extreme-terrain mobility*, which pertains to terrains with extreme topographies, large distributions of hazards, and/or unique regolith types. During the 2012 NRC review process, two different review boards ranked "extreme-terrain mobility" a high-priority[2] technology for NASA to develop within the next five years. This section discusses the technical aspects and challenges associated with meeting this goal.

1.3.1 Scope

Extreme-terrain mobility refers to surface mobility over a range of terrain topographies and regolith properties on bodies with substantial gravity fields. Examples of such topographies and regolith types include highly-sloped crater walls and floors, cold traps, gullies, canyons, very soft and friable terrains, and terrain with extreme rock densities. It is worth noting that other extreme environmental conditions may also be present at such sites, such as extreme temperatures or pressures. Extreme-terrain mobility covers capabilities that enable access and egress to such extreme terrains, safe traverses to designated targets, loitering for in situ measurements, and sample collection and extraction. Extreme-terrain mobility encompasses diverse platforms that may include wheeled, legged, snake, hopping, tracked, tethered and hybrid platforms. Surface guidance, navigation and control for such diverse platforms depend in part on the nature and constraints for the mobility approach. While access to and sampling from extreme terrains can also be accomplished through above-surface mobility, a key feature of extreme-terrain access is loitering at targets

[2]The National Research Council study panel ranked extreme-terrain mobility 6th, while its steering committee ranked it 8th [48, Table 3.7, p. 88].

of interest for in situ measurements. Since the NRC defined and prioritized above-surface mobility separately from extreme mobility, we only address the latter in this section.

1.3.2 Need

Extreme-terrain mobility would be an enabling technology for both science and human space exploration missions. For science missions, some of the most compelling targets for future exploration within our solar system lie in terrains that are inaccessible to state-of-the-art robotic platforms, including NASA's Mars Exploration Rovers [94] and the Mars Science Laboratory [93] rover.

For example, the recent discovery of recurring slope lineae (RSL), such as those observed in Newton crater on Mars, are on steep slopes (25°–40°) that are hundreds of meters down from the crater rim. *In situ* analysis and sample capture of these outflow deposits align with the science priorities that are described in both the Decadal Survey [103] and the goals of MEPAG [35]. Similarly, successive flybys by the Mars Global Surveyor revealed dynamic processes in the form of bright gully deposits on the walls of two separate unnamed Martian craters.[3] *In situ* samples of these flows would likely lead to new insights into Martian geology. Moreover, methane plumes that have been discovered over hazardous terrain on Mars are intriguing researchers who are now attempting to ascertain whether the source is geological or biological in nature[4]; this represents another question that extreme-terrain mobility could potentially answer.

Another compelling scientific site that lies within extreme terrain was discovered by NASA's Cassini spacecraft, which revealed what scientists believe to be a cryovolcano on the surface of Titan.[5] Direct sampling of cryovolcanic ejecta along its steep slopes would shed new light on the processes underlying cryovolcanism, as well as provide valuable access to material from Titan's interior.

A third example is from the LCROSS experiment. By impacting the lunar surface and analyzing the ejected debris, the LCROSS mission found evidence of water ice in the Moon's permanently shadowed Cabeus Crater[6] [34]. The shadowed regions lie at the bottom of a long, steep slope. These lunar cold traps, which have never received a single photon of sunlight, are believed to hold water ice within a few centimeters

[3]New Gully Deposit in a Crater in the Centauri Montes Region (2006). URL: http://www.nasa.gov/mission_pages/mars/images/pia09028.html. Retrieved January 14th, 2011.

[4]Martian Methane Reveals the Red Planet is Not a Dead Planet (2009). URL: http://www.nasa.gov/mission_pages/mars/news/marsmethane.html. Retrieved January 15th, 2011.

[5]Flyover of Sotra Facula, Titan (2011). URL: http://www.nasa.gov/mission_pages/cassini/multimedia/pia13695.html. Retrieved January 8th, 2011.

[6]Ten Cool Things Seen in the First Year of LRO (2010). URL: http://www.nasa.gov/mission_pages/LRO/news/first-year_prt.htm. Retrieved February 3, 2011.

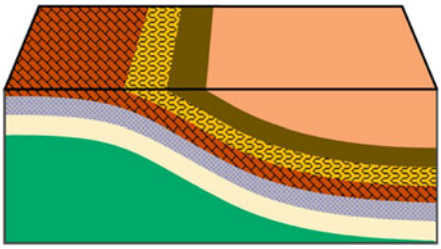

Fig. 1.3 Comparing horizontal and vertical access to stratigraphic layers. Some deeper layers may not be accessible via horizontal traverses

of the surface. At high-probability locales such as these, the assessment of in situ resources in terms of presence confirmation, abundance mapping, and extraction possibilities would be critical for precursor missions ahead of human exploration [129]. Other features such as lunar vents [60] and lava tubes are also potential sites for future exploration. Lava tubes, through observations of skylights on the Moon and on Mars, could potentially serve as future temporary habitats for astronauts, providing them with protection from space radiation [63]. The exploration of lava tubes could also be of scientific interest for similar reasons.

A new generation of robotic explorers is needed to explore these extreme terrains in order to access, probe, measure, extract and return samples. Traversing and loitering on steep, exposed substrate slopes reaching up to 90° would enable the examination of stratigraphic layers of exposed bedrock [35] and icy bodies. While current practice relies on long traverses across the surface to access these layers (Fig. 1.3), direct access of exposed strata enables close examination of the interface between stratigraphic layers, which, due to substantially less weathering, would offer more details compared to what may be obtained through horizontal traverse alone [36].

Traversing and loitering on slopes of granular and mixed media up to the angle of repose enables access to locales such as the sites of putative "water" seeps on Mars (Fig. 1.4). Traversing across and through alluvial fans for in situ examination would further our understanding of the underlying physical processes and composition of the ejected material [35]. As detailed topography of such fans may not be well-known *a priori*, robust and versatile mobility platforms are required for their exploration. Unfortunately, through the course of accessing such extreme terrain, hazards such as sinking into soft regolith or falling via landslides could be encountered. The ability to reliably avoid or survive such events in order to maintain an acceptable risk posture becomes a key feature of these platforms.

Extreme-terrain exploration could be embarked upon with remote robotic assets or could very well be part of human exploration missions. Extreme-terrain robots would extend astronaut surface access to regions deemed too risky for human access. They would also enable robotic precursor missions to explore hazardous sites likely to harbor needed resources for future habitation. Lunar robotic missions to extreme terrain could be operated from cis-Lunar orbit. Future human missions to Mars could tele-operate robotic assets from stations on Phobos, a body significantly more

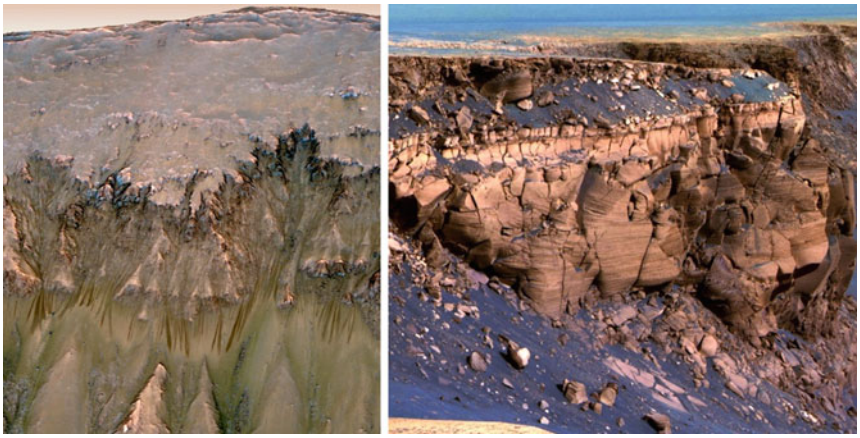

Fig. 1.4 Examples of extreme terrains on Mars: recurring slope lineae (RSL) in Newton crater hypothesized to be briny seeps (*left*, NASA/JPL-Caltech/University of Arizona—Mars Reconnaissance Orbiter HiRISE, 2011), and a false-color image of Mars' Victoria crater showing steep slopes, scattered rocks, bedrock, and tall cliffs (*right*, NASA/JPL/Cornell—Mars Exploration Rover: Opportunity 2007)

accessible than the Martian surface that would also provide astronauts relatively better protection from solar radiation.

In short, extreme mobility technologies enable access to otherwise denied areas. This provides NASA with the capability to maneuver its surface vehicles in extreme terrain in order to "follow the water"—a high-priority science focus for Martian and lunar science missions that generalizes to many extraterrestrial surface exploration missions, human or robotic [48].

While the primary motivation and focus here has been on planetary and lunar exploration, robotic vehicles that can traverse extreme terrain may have ample terrestrial applications as well, including in scientific research such as sampling of active volcanoes and Antarctic slopes, in civil applications such as search-and-rescue, or in commercial ventures such as mining.

1.3.3 State of the Art

Significant progress in terrestrial robot mobility has been made in recent years towards handling more challenging terrains. However, efforts have primarily focused on human-traversable terrains applicable to military purposes. For example, Boston Dynamics' BigDog and LS3 used dynamically-stable gaits to negotiate rough terrain and slopes of up to 35° grade under rough and slippery conditions [42, 43]. They also demonstrated robustness to external force disturbances sufficient to throw the platform off-balance.

For space robotics, the constraints on mass and power as well as the desire to traverse more extreme terrain have limited the adoption of such technologies.

Nevertheless, a number of developments have aimed at contributing to our current understanding of the potential strategies for extreme-terrain mobility on planetary bodies.

Both legged and wheeled robots, as well as tethered and untethered robots, have been proposed for exploring extreme terrestrial and planetary landscapes, several of which have been built and fielded. For example, the Dante II robot [10] was a tethered legged robot that was specifically engineered to descend into active volcanoes. Shigeo Hirose's group has explored self-anchoring tethers and tethered tracked vehicles for emergency response [62], as well as tethered leg vehicles for fieldwork [52]. The JPL legged ATHLETE robot, designed to handle cargo in support of a sustained human presence on the moon, has traversed rocky and sloped terrain at a number of analog sites in California and Arizona, including Black Point Lava Flow [134]. For slopes greater than 20°, the ATHLETE rover would also use a tether. The Axel rovers[7] [97], designed to explore very steep terrains, have demonstrated traversal of near-vertical slopes and sloped terrain littered with large boulders. Other robots that use leg-mounted active anchors in lieu of tethers have been proposed [8] and developed [105].

In addition to these legged robots, a number of wheeled designs have also been proposed, of which several prototypes have been built, fielded, and flown. One promising example is a recurring mechanism configuration used in either a six-wheeled rocker-bogie suspension (e.g. the MER and MSL rovers) or in a four-wheeled scissor-like active suspension that allows each wheel to be independently lifted off the ground. Such platforms were designed to lower the center-of-mass to provide greater stability. One such example is the Nanorover [70], a grapefruit-sized rover that was proposed for exploring an asteroid surface as part of the MUSES-C mission. This rover had a symmetric design and was capable of operating in an upside-down configuration. It actively controlled its center and was even capable of hopping on low-gravity planetary bodies. Follow-on concepts included tethering the Nanorover to a Sojourner class rover for future Mars missions. The architecturally-similar SCARAB rover demonstrated an inch-worming maneuver that synchronized wheel and suspension mechanism motion to traverse high-slip terrains [11]. Despite this ability, steeper slopes will likely require additional external stabilization, such as through a tether. A four-wheeled tethered rover was demonstrated with Cliffbot [107]. Unfortunately, this architecture required a minimum of three rovers. Two rovers would traverse the rim of a crater while a third rover, tethered to the other two, would descend into the crater. Lateral mobility with two tethers would generally be greater at closer distances to the rim, but this advantage diminishes as the rover descends deeper into the crater. The Cliffbot used the rim rovers to manage the tethers, which, unlike designs that pay out their own tether, risks higher abrasion from constant rock-scraping. Moreover, the Cliffbot cannot recover from tip-over, and the problem of planning the motions of two tethers adds extra complexity.

[7] Axel Videos (2011). URL:http://www.youtube.com/watch?v=Ijjo1nW94tY. Retrieved October 30th, 2014.

Outside of four-wheeled rovers, a number of previous efforts dating to the early 1970's have recognized the potential of two-wheeled rovers for steep terrains. Several efforts have converged on a robotic body morphology consisting of a simple axial body with two wheels and a caster, as recently exemplified by the Scout robots [121], designed for military applications. A similar tethered rover with three large inflatable wheels was proposed for future Mars missions [89]. Independently conceived, the family of Axel rovers was initially developed a decade ago to provide modularity and separation between the most failure-prone mobility elements and their respective science payloads [64, 95]. In 2006, the original Axel rover was retrofitted with a tether and adapted with grouser wheels for extreme-terrain mobility on slopes [96]. Such a configuration, with its symmetric design, has demonstrated potential for robust, flexible mobility and operations on challenging terrain. Its single tether was managed by the same mechanism that controls an articulated boom. This family of rovers has also included instrument bays housed inside the wheel hubs, which could be oriented in a turret-like fashion independent of wheel rotation.

The DuAxel concept included docking and undocking of the Axel rovers with a central module, enabling both untethered mobility for extreme-terrain access and tethered mobility on steep terrains [97].

While progress has been made with extreme-terrain mobility for terrestrial applications, at the date of this writing, there has been no planetary mission that has attempted access to extreme terrains. State-of-the-art surface exploration platforms, such as the highly successful Spirit and Opportunity rovers as well as the most recent Curiosity rover, were all designed to operate on relatively flat and shallow-sloped terrains with slopes of less than 20° and 30° grade, respectively.

1.3.4 Challenges and Future Directions

To date, planetary rovers have been designed to explore rocky but relatively flat regions and were not intended for terrains such as deep craters, canyons, fissures, gullies and cryovolcanoes. Such extreme terrains pose a unique set of challenges and requirements for a robotic explorer. Researchers developing extreme-terrain surface space robots have to contend with the system complexity that results from high degrees of articulation, tether management, and the challenges associated with limited power, communication, mass, volume, and computation, as well as with terrain variations that impact anchoring and other surface operations. Conventional, flat-topography rover designs must be completely re-evaluated in the context of high-risk terrain missions.

One of the most significant challenges associated with extreme-terrain exploration derives from having to land proximal to but outside of the target site, demanding an approach from afar over diverse topographies that may require unique mobility aids such as tethers, anchors, and higher traction devices. To illustrate, Fig. 1.4 shows a ground-level picture of Mars' Victoria Crater as imaged by the Opportunity Rover. Typical of Martian craters, Victoria consists of steep slopes, scattered rocks, exposed

bedrock, and tall cliffs. Rocker-bogie class rovers such as Spirit, Opportunity or Curiosity were not designed for such terrain, and would not likely be well-suited to navigate it. Such terrains would be very difficult to traverse since platform mobility decreases with slope grade, particularly in areas of loose regolith where traction forces can be severely diminished. Given that a sand trap on relatively smooth terrain was enough to ensnare the Spirit rover [6], even a small amount of loose soil on sloped terrain could prove insurmountable to traditional rovers trying to climb a crater wall against the forces of gravity. Extreme-terrain rovers must be able to operate robustly in such cases.

Another mobility hazard associated with traversing steep and rugged terrain is tip-over, a concern which must be taken into consideration when designing extreme terrain rovers. Tip-over can be caused by improper stabilization, or by other uncontrollable external factors such as wind, slippery ice, loose rocks, and many other environmental factors. In 1992, the eight-legged walking robot, Dante II, successfully descended into Alaska's Mt. Spurr volcano using a winch-cable system [10]. On the ascent trip, however, the rover fell on its side under the influence of large lateral tether forces and was unable to right itself. Extreme-terrain rovers can reduce the risk of tip-over by lowering their centers-of-mass and carefully planning safe routes around obstacles so as to avoid tether entanglement and potential tip-over conditions. Alternatively, such rovers can be designed to operate in both upright and upside-down configurations, thereby eliminating the end-of-mission dangers of tip-over altogether.

Another challenge for extreme terrain mobility is power and communication. Energy sources can be difficult to find in areas of extreme terrain. For example, the Cabeus Crater located near the Moon's south pole lies in a state of near-perpetual darkness, thus precluding the use of solar power. Even with consistent access to sunlight, cold-traps like caves and crevices along crater walls would be difficult to investigate for prolonged periods. Rough terrain consisting of tall peaks, deep craters, or canyons naturally restrict access to sunlight, and rovers charged with exploring these regions must be able to survive on a limited energy budget. Such terrains also present challenges for Earth-based communications with the rover, particularly in the absence of an orbiting communication satellite.

A problem that is unique to the robotic exploration of cold regions, such as the surface of Europa and other icy moons, is heat dissipation. In addition to traditional vehicle thermal engineering for ultra-cold climates, robotic explorers designed for these environments must minimize thermal pollution to nearby terrain so as to avoid disrupting the scientific analysis of volatile components. They must also be designed with sufficient exposed surface area to allow for adequate thermal regulation.

Due to the hazardous nature of the environments and the unique mechanical, thermal, and avionics designs likely required for extreme-terrain mobility, advanced control and autonomy strategies will be needed to operate extreme-terrain rovers safely. This will require more sophisticated onboard sensing, perception, planning and computational capabilities than for state-of-the-art flat-topography rovers due to the larger variations in terrain, more complex dynamics, and tighter operational constraints. Of all the avionics systems, flight-qualified processors typically represent

the bottleneck on computational capability and hence restrict the types of algorithms and approaches that may be considered. Unfortunately, the performance gap between current standard commercial processors and flight processors remains quite large. In the commercial sector, the trend is moving toward greater parallelism and multiple-core processing. Achieving comparable levels of computation, power consumption, robustness, and reliability with a similar form factor on space-rated processors in the face of increasing cost constraints remains an open problem.

In addition to these general challenges, each platform design would offer its own range of capabilities and introduce its own sets of constraints to be addressed and risks to be retired. A concerted and focused effort would be necessary to mature technology to readiness levels acceptable for future missions. Key areas of technology investments for extreme-terrain access include: traversal to designated targets in extreme terrains, retro traverse for captured samples, control of tethered or anchoring platforms including anchoring and deanchoring, avionics equipment built for hazardous terrain, traversability analysis and motion planning, and high-fidelity terrain modeling and simulation of extreme-terrain mobility. We now discuss in greater detail the major technical hurdles and challenges of each, below.

- **Traverse Technologies**: In the absence of higher precision and pinpoint landing capabilities that could deliver a payload to the vicinity of an extreme terrain site, it becomes necessary to traverse a distance of at least several kilometers to reach them by ground. In this case, technologies that would enable faster autonomous traverse for flight systems become critically important. State-of-the-art platforms currently navigate the surface at a rate of 20–30 m/sol using a computationally-demanding procedure. They first process stereo imagery, generate three-dimensional maps, and assess terrain traversability. If feasible, they then plan their motions and finally conduct their traverse. This sequential process can take up to several minutes for every half-meter step. This is primarily driven by the limited on-board power and computation on today's flight-qualified processors and by the lack of dedicated processors for computationally-demanding applications. Recent developments have made advances in migrating computationally-intensive vision processing and some navigation functions to flight-relevant field-programmable gate arrays (FPGAs). This also enables vision-based pose estimation (a.k.a. visual odometry) to run more frequently and consequently help build more accurate maps that enhance the quality of the navigation. Higher quality maps would enable rovers to handle more challenging terrain and execute tighter maneuvers in rock fields, such as thread-the-needle type maneuvers where the rover negotiates a path between two tightly-spaced obstacles. As terrain topographies become more uncompromising near extreme sites, algorithmic advances in surface navigation become more critical to reach targets of interest. One such example is driving upslope towards a crater's edge before deploying a tethered payload into the steeply-sloped interior of the crater wall. As mobility in extreme terrain is likely to become more dynamic, advances in computationally-efficient localization would be necessary to improve control and mapping. In the future, onboard sensing is likely to be fused with higher-resolution orbital imagery for assessing terrain traversability in more effective and automated ways.

- **Tethered/Anchored Mobility and Control**: This brings us to a second technology: tethered and anchored mobility. Highly-sloped terrains require strong and robust mechanical support to counteract the effects of gravity. One approach would be to use an external means of mechanical support. Research in tethered mobility has included the design and management of both single and multi-tethered platforms. Future studies would need to focus on strategies that preserve tether integrity, improve coordination, minimize damage, and reduce the risk of multiple-tether entanglement. Other technologies would include tether tension and shape sensing to assist in pose estimation and identify high stress (i.e. "pinch") points. Algorithms would have to become more sophisticated to incorporate this additional sensory information for control and motion planning. Anchoring, either alone or in combination with tethering, can be another means of providing mechanical support to climbing or rappelling platforms on highly-sloped terrains. This can be particularly challenging when terrain properties vary or are not known *a priori*, and would likely require onboard sensing and assessment of anchor bearing strengths. The development of technologies that enable multiple anchoring and de-anchoring across a wide range of terrain types would also be highly beneficial.
- **Avionics and Terrain Equipment**: Given the limited communication windows and bandwidths, some level of control and autonomy would be necessary during operations. While state-of-the-art rovers have demonstrated surface navigation (obstacle detection and avoidance) for hundreds of meters at a time across the Martian surface, such technology would have to be extended to extreme terrains where system dynamics from the challenging topographies and gravity vector direction become relevant. The unique design of extreme terrain mobility may impose additional challenges and constraints on sensor configurations, which would also require further development. Platforms that sport multiple appendages would likely require tool changes when transitioning from benign to extreme terrain. A hybrid legged platform on wheels would likely call for a transition between wheels and anchors when conducting an excursion across extreme terrain. In addition, given that extreme-terrain assets are more likely to be payloads rather than primary platforms due to the overall risk, their low mass constraints would drive a need for smaller and lighter sensors, cameras, inertial measurement units, and other instruments. Miniaturization of avionics equipment would increase payload capabilities. Mission-dependent objectives in extreme terrain such as sample acquisition, caching and handling present their own unique equipment challenges. Drilling and coring require stabilization of the platform or some form of grappling to impart necessary forces for percussion or coring, for instance.
- **Traversability Analysis and Motion Planning**: Control, traversability analysis and path planning for an extreme terrain mobility platform takes on a new meaning than for traditional flat-slope mobility. In extreme terrains, motion may be more constrained (especially for tethered systems), control may require more sophisticated dynamical models given the gravity field, and knowledge of regolith properties may be more critical. As compared with state-of-the-art motion planners that primarily consider terrain geometry and wheel characteristics for traversability, long-duration excursions in extreme terrain would demand more sophisticated

motion planning techniques that accurately account for gravitational forces and the effects of terrain properties. Model-predictive motion planners that incorporate dynamics may well play an important role for executing more predictable and controllable maneuvers in some of the most difficult terrains.

- **High-Fidelity Terrain Modeling and Mobility Simulation**: As a number of challenges need to be addressed to characterize extreme-terrain mobility in a relevant environment, some elements would likely benefit from advances in physics-based modeling and simulation tools. Recent and future advances in granular media simulations may prove quite effective in characterizing the interactions of the mobility platforms (or components) with regolith across a range of terrain types and under different gravity models. Given the hazardous environments and terrains, reliable fault protection and recovery systems would become essential parts of the hardware, software, or operational scenario design. For example, recovery from tip-overs could be addressed via a mechanical design that operates from all stable states or through an alternate operational strategy. With appropriate simulation tools to inform the design, such scenarios and strategies could be more readily investigated and evaluated.

In addition to mobility technologies themselves, there are a number of related technology areas complementary to and supportive of extreme terrain mobility whose advances would have direct impact to mobility research. Brief discussions of a few of the more important of these related technology areas are provided here.

Entry, Descent and Landing

One example is landing precision, which falls under the Entry, Descent and Landing technology area (TA-09); see Sect. 1.2.4.2 for a detailed description of relevant challenges. The key subcategories of relevance within *entry, descent and landing* are: (a) surface access to increase the ability to land at a variety of planetary locales and times; (b) precision landing that enables space vehicles to land with reduced error, and (c) surface hazard detection and avoidance to increase the robustness of landing systems to surface hazards. Since exploring extreme terrains would first require reaching extreme sites, technologies that would reduce the traverse distance by shrinking the size of the landing ellipse would not only increase the number of potential landing sites, they would also reduce the traverse distance requirement, and hence mission duration, to visit those sites. Further advances in the terminal descent phase, such as pin-point landing (within 100 m) could change the nature of extreme terrain exploration, enabling cheaper missions where the extreme-terrain platform could then be hoisted on a lander and its resources leveraged for power and communication.

Below-Surface Mobility

A second related area is *below-surface mobility*, which addresses vehicles that would transit under regolith, in caves, or immersed in bodies of liquid. For certain situations, the same technologies developed for extreme-terrain mobility could be

re-purposed for below-surface mobility applications. The exploration of collapsed lava tubes (caves) and lunar vents are two such potential scenarios. For example, tethered platforms originally designed for access to the interior of crater walls could also potentially be reapplied to lava tube exploration.

Microgravity Mobility

Technologies developed for *microgravity mobility* as discussed in Sect. 1.4, including anchoring, fixturing, and tethering, as well as articulated legged, tracked, wheeled and hybrid mechanisms, could additionally apply to extreme-terrain mobility applications and vice versa. Details on microgravity mobility systems will be given in the subsequent section.

1.4 Microgravity Mobility

The National Research Council recommended small-body/microgravity mobility as a high priority technology for NASA for the next five years. Initially, microgravity mobility was assigned a medium/low score due to the expensive nature of microgravity system development and testing and its limited applicability outside the aerospace community.

The panel later elevated the priority of this technology from medium to high because the NASA 2010 Authorization Act (P.L. 111–267) indicated that small body missions (to near-Earth asteroids) should be an objective for NASA human spaceflight beyond Earth orbit. If this goal is pursued as a high NASA priority, it would also likely require precursor robotic missions to small-body surfaces with applicable mobility capability. This section describes the benefit, technical aspects, and challenges facing the robotics community today in achieving microgravity mobility.

1.4.1 Scope

Small-body mobility concerns the spatial surface traversal of planetary bodies with substantially reduced gravitational fields for the purpose of science and human exploration. This includes mobility on Near-Earth Objects (NEOs), asteroids, comets, irregularly-shaped objects, and planetary moons, including Phobos, Deimos, Enceladus, and Phoebe, to name a few notable examples. Surface mobility platforms for small bodies differ from their planetary counterparts because the microgravity environment largely influences their design. Microgravity can be leveraged as an asset for mobility, as in the case for hopping platforms, or overcome as a challenge, as in the case for wheeled rovers and anchoring systems. Microgravity mobility includes hopping, wheeled, legged, hybrid and other novel types of mobility platforms. “Hoppers”—a term short for hopping mobility platforms—move via many diverse forms of actuation; examples include propulsive thrusters, spring-loaded

mechanisms, and internal actuation, which effects platform motion using internally moving parts that generate reactionary forces or changes in the platform center-of-gravity. Note that any impacts of hopping robots with the surface are unlikely to cause damage due to the very low gravitational acceleration associated with small-body objects. Broadly-speaking, revolutions in these hardware and mechanism designs, as well as improvements in multi-asset mission operations, low-power computing, and autonomous control algorithms, will be key to performing mobile missions in microgravitational environments.

1.4.2 Need

Weak gravitational fields (micro-g to milli-g), characteristic of celestial small bodies, hamper the adoption of traditional robotic mobility systems and call for the development of disruptively new technologies for both surface mobility and surface operations. The National Research Council has designated these mobility technologies for small-body and microgravity environments as a high-priority for NASA given their destination potential for human spaceflight beyond Earth orbit, an endeavor likely to require several precursor robotic missions. The relevance of enhancing small-body exploration in the context of future human exploration programs was highlighted in the exploration roadmap published by the Small Bodies Assessment Group [100] and in the objectives of the Strategic Knowledge Gaps for Human Exploration [129]. The need for these technologies is further emphasized by the fact that, to-date, no *mobility* system has ever been successfully deployed over the surface of a small body,[8] indicating that little is currently known about robotic operations in microgravity environments.

Surface investigation of small bodies with a low-mass platform for both large-scale coverage and fine-scale maneuvers (i.e. from kilometers to meters), as enabled by microgravity mobility, would be monumental to the advancement of space missions. Data obtained from recent missions to small bodies show that surface properties on most small bodies evolve over scales ranging from hundreds of meters to as little as a few meters (Fig. 1.5 highlights the diversity in surface properties at a variety of scales for two representative objects); this is in contrast to the long-held idea that the surfaces of small bodies are, in general, both chemically and physically homogeneous.

The benefit of microgravity mobility to expected scientific return can be seen explicitly in the recent decadal survey report for planetary science, which prioritized three main cross-cutting themes for planetary exploration: (1) the characterization

[8] Small-body soft landings of spacecraft orbiters and *static* landers have, however, been achieved; the first was NASA's NEAR Shoemaker on asteroid Eros in 2001 [41], followed by two touchdowns of JAXA's Hayabusa on asteroid Itokawa in 2005 [**?**]. ESA's Rosetta mission [125] achieved the first successful deployment of a static lander, named *Philae*, over the surface of comet 67P/Churyumov-Gerasimenko on November 12th, 2014 [44].

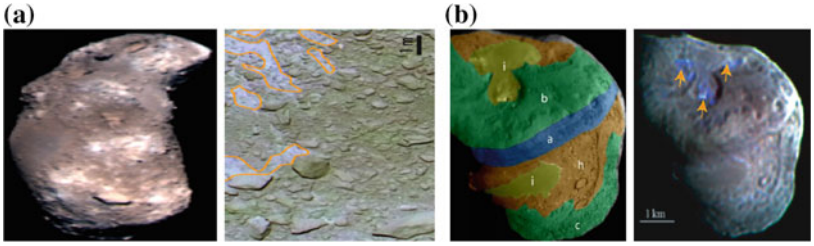

Fig. 1.5 Illustration of the diversity of landscapes and of physical and chemical properties encountered at small bodies. **a** Asteroid Itokawa (observed by *Hayabusa*) exhibits lateral variations in albedo at the regional scale due to the combination of space weathering and surface dynamics (*left*); high-resolution imaging of Itokawa reveals bright patches of "fresh" material excavated in discrete places with a spatial extent on the order of 1 m, distributed with a spatial wavelength of a few meters (*right*). **b** Observations of comet Tempel 1 by *Deep Impact* also indicate regional variations in geological properties (*left*), with the presence of volatiles confirmed in a few discrete places (indicated by *arrows*, *right*)

of the early solar system history, (2) the search for planetary habitats, and (3) an improved understanding about the nature of planetary processes [103]. A growing number of ground and space observations have recently shed new light on the astrobiological relevance of small bodies, indicating that the exploration of a selected subset of small solar system bodies would collectively address all three themes [28, 29]. The explorations of small bodies such as Near-Earth Objects and the moons of Mars are also key components of the flexible path for human exploration. In general, origins science and the search for habitats revolve around characterizing planetary material chemistry (elemental, isotopic, mineralogical, noble gas, organics, etc.). While some measurements could be obtained from remote platforms (such as space telescopes or orbiting spacecraft), most require direct contact with (or close proximity to) the surface, called *in situ measurement*, for an *extended* period of time at *multiple* locations [29]. This is also the case for the precursor science that enables human exploration, which first and foremost would require the detailed characterization of surface physics, including regolith mechanical properties, dust dynamics, and electrostatic charging [129]. Though in situ exploration of small bodies is currently in its "technological infancy," it is poised to become a major science enabler in the near future, as the following several paragraphs serve to illustrate.

Astronomical observations (such as seen in Fig. 1.6, made by ground-based and space observatories), though particularly suited to characterizing the orbital properties of large populations of objects, are insufficient for constraining the origins of single objects, as resonances can dramatically alter their orbital properties. As a result, in situ exploration plays a pivotal role in determining the density distributions and dynamical properties of small bodies, while allowing more accurate characterization of volatile composition and isotopic ratios. Though isotopic ratios could be determined in some cases through mass spectrometry of outgassing material, most small bodies neither out-gas nor present enough exospheric density to allow such measurements. Hence for a large class of small bodies, the measurement of isotopic

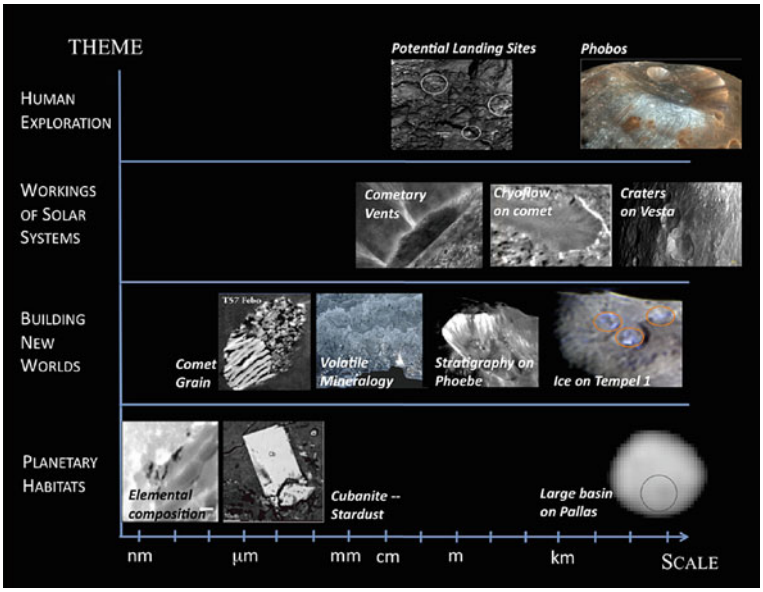

Fig. 1.6 Illustration of the type of observations to be achieved by space missions in order to successfully address the key science pertaining to the three cross-cutting themes highlighted in Vision and Voyages. Note that in general we lack high resolution observations at the millimeter to meter scale that can be best obtained by in situ exploration. *Image courtesy of* [29]

ratios requires in situ exploration. With appropriate instrumentation packages, this capability would enable physical and chemical characterization of surface properties relevant to both human and science exploration.

For a given science objective, in situ exploration at *designated* and *multiple locations* should be an integral component of future missions, and techniques for such operations will need to be developed. Two motivating scientific examples are presented here. First, the comet Hartley 2 exhibits two starkly different terrains: very granular areas with vents and smooth areas that have been interpreted as wasting areas. Full characterization of the comet's surface would require sampling at each location. Second, the comet Tempel 1 presents four distinct geological units; in particular, it exhibits cryoflow features (products of its geological evolution) near areas that appear to be less evolved and may be more representative of the original material (see Fig. 1.7). Spatially-extended exploration of Tempel 1 would be key to capturing information on the accretional environment of that object as well as on signatures of its long-term evolutionary processes.

In summary, in situ information enabled by surface mobility about the chemical and physical heterogeneity of small bodies has the potential to lead to a much improved understanding about their origins, evolution, and astrobiological relevance, yielding important ramifications for science and an expanded human presence in our solar system.

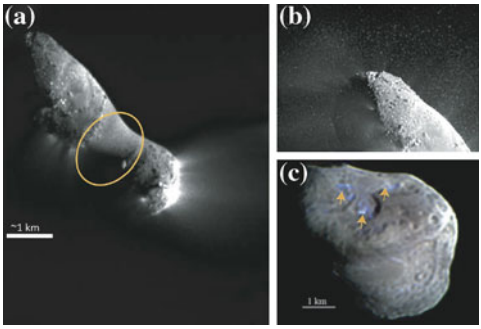

Fig. 1.7 Illustration of the variety of landscapes found at comets. **a** Picture of Hartley 2 obtained by EPOXI showing a contrast in surface roughness between active and waste areas. **b** This close up shows the variations of physical properties, especially roughness, at all scales. **c** In this close-up picture of Tempel 1 observed by Deep Impact lateral variations in chemistry (ice and dust) occurs on short spatial scales. *Image courtesy of* [29]

1.4.3 State of the Art

While there have been several attempts at small-body surface mobility, as of this writing no such system has successfully explored the surface of a small body. Traditional forms of robotic mobility, such as wheels and legs, present bevies of new challenges when operated in microgravity. As a result, a number of innovative designs have been attempted using unconventional means of locomotion; for instance, NASA, RKA, ESA, and JAXA have all attempted various forms of hopping strategies for traversing small bodies. In fact, three missions so far have included a robotic hopper as part of their payload: Phobos 2, Hayabusa, and Hayabusa 2. Their designs, as well as most attempts of hopping mobility, made use of two basic principles:

1. Hopping using a sticking mechanism (thus jumping away from the surface).
2. Hopping by moving an internal mass.

Phobos 2 was a Soviet RKA mission launched in 1988, aimed at studying Mars and its moons Phobos and Deimos. The plan was to deploy in close proximity to the surface of Phobos a 41-kg robotic hopper called PROP-F (see Fig. 1.8). Its actuation was based on a spring-loaded leg mechanism designed to adhere to the moon's surface. Unfortunately, when Phobos 2 was within 50 m of the Martian moon, communication with the spacecraft was lost before PROP-F was deployed [112]. Several years thereafter, the JAXA Hayabusa mission planned to carry JPL's Nanorover (see Fig. 1.8), a four-wheeled rover with articulated suspension that was capable of roving and hopping. Unfortunately, the rover was canceled due to budgetary concerns. Subsequently, JAXA/ISAS developed the MINERVA rover, a 591 g hopping rover that employed for locomotion a single internal flywheel mounted on a turntable, which imparted control over the direction of each hop. The MINERVA design was rated to surface traversal speeds as high as 0.1 m/s. Unfortunately, the MINERVA rover also

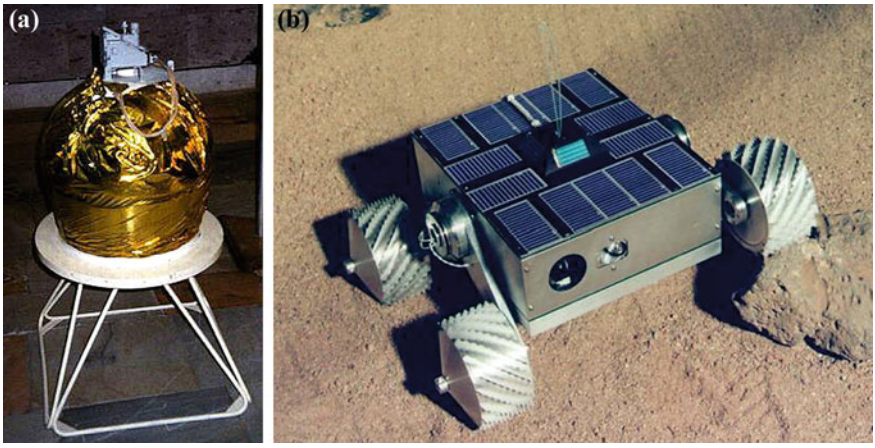

Fig. 1.8 **a** The PROP-F Phobos Hopper. *Image courtesy of* [112]. **b** The Nanorover. *Image courtesy of* [71]

failed upon deployment [69]. Both Nanorover and MINERVA were solar-powered systems and hence constrained to very limited power (on the order of a couple of Watts) and computation. Since then, a handful of other hopping designs have been attempted. NASA-JPL has prototyped several generations of robotic hoppers actuated by surface adhesion. ESA developed a small hopper rover, called MASCOT, actuated by spinning two eccentric masses. MASCOT is currently a part of the Hayabusa 2 spacecraft payload [39, 124].

All of these platforms were designed for exploring extended areas; however, both of NASA's hopper prototypes [47, 71] (that relied on a combination of wheels and sticking mechanisms), ESA's hopper prototype, RKA's unsuccessful landers for the exploration of Phobos, and JAXA's MINERVA lander did not allow for precision traverses to designated targets. Controlled mobility and precise positioning of instruments on the surfaces of small bodies are still active areas of current research. Researchers continue to examine several approaches to small-body mobility that include legged platforms with anchoring for traction [105, 133], as well as other forms of small-body legged mobility that allow drilling and surface sample collection [61]. In addition, a team from Stanford, JPL, and MIT is currently developing an internally actuated rover that encloses three mutually-orthogonal flywheels. Through controlled spinning of its internal flywheels, the rover can give rise to surface reaction forces that instigate rover tumbling (for fine mobility) or hopping (for large surface coverage) in a controllable direction (see Fig. 1.9) [4].

Other types of low gravity surface mobility have also been explored. Thrusters are the key actuation mechanism for the Comet Hopper (CHopper) mission concept, one of the three preselections for the NASA 2016 Discovery-class mission to comet 46P/Wirtanen [32]. The CHopper mission was designed to investigate changes in surface properties with heliocentric distance and land multiple times (4–5 times) on

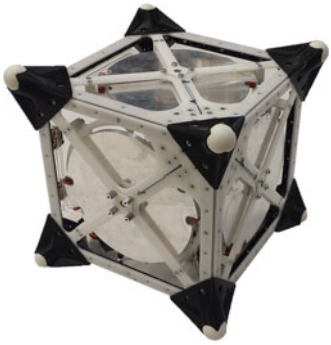

Fig. 1.9 A flywheel-actuated hopper designed for precise maneuverability. *Image courtesy of Stanford University*

the surface of the comet, hopping twice each time before coming to a stop; however, it did not make the final selection.

1.4.4 Challenges and Future Directions

Microgravity environments pose many challenges not only for mobility and manipulation at the surface of small bodies, but also for control, localization and navigation. Recent observations from both space mission and ground-based telescopes have revealed a more diverse landscape than previously thought. Small body surfaces can range from areas covered with a thick layer of fine regolith to ones with rocky and protruded regions. What may seem like simple operations on bodies with substantial gravity fields, such as drilling or coring, can be quite difficult for a robot in microgravity, unless some form of fixturing or anchoring is used to impart necessary stabilization forces. The use of tethers or other aids could enhance control and improve maneuvering precision, but they also yield the unfortunate side-effects of added mass and complexity.

Technologies relevant for small body mobility include advanced mobility and control techniques that would operate on a range of heterogeneous terrain types. They would also include specialized techniques for localization of surface assets, which are likely to require support from an orbiter given the number of significant line-of-sight occlusions that result from the large topographic changes characteristic of many small bodies. Localization is particularly complex for hopping and tumbling systems due to the discrete, impulsive changes in pose that result from actuation. The orbiter, hosting spacecraft, or "mothership," is also likely to be used for asset surface deployment; as a result, advances in control strategies exploiting synergistic operations between them and the mothership could also enhance asset mapping and motion planning, while simultaneously alleviating their computational load. To date, most of the proposed architectures involving in situ mobile platforms rely on *decoupled* mission operations, in the sense that the mothership is essentially used as a communication relay (a sort of "bent pipe"). This either requires sophisticated capabilities on-board the mobile assets for perception, localization and surface navigation, or

leads to platforms with limited maneuverability (when such onboard capabilities are not implemented). *Coupled*, hierarchical approaches, on the other hand, would allow end-to-end minimalistic design of mobile assets by redistributing their computational tasks. Here the functions that require wide-area information, such as perception and planning, are assigned to the mothership, while functions that rely solely on local information, such as obstacle avoidance, are assigned to the mobile platforms.

To facilitate the discussion of microgravity systems, classification of mobility platforms is divided into four groups according to their primary actuation mechanism.

- **Thruster Mobility**: Thruster actuation for small body exploration involves the use of thrusters for control of far operations, with occasional visitations by de-orbit onto the surface of the object. Once finished on the surface, sorties conclude when the spacecraft lifts off and resumes far operations. The premise is that landed operations allow an extended period of time for scientific data collection, while return to orbit can benefit the selection of and traversal to new scientifically-meaningful landing sites. Possible drawbacks of this architecture include the risk of damage to the lander during landing operations, the constrained number of visit locations due to a fixed fuel budget, and the limited surface mobility (which, combined with landing ellipse uncertainties, could limit the platform's ability to target specific sites of interest). Furthermore, for science missions, contamination of the landing site from thruster exhaust could potentially interfere with scientific measurements unless the lander had an alternate means of mobility or of reaching pristine terrain. To overcome these limitations, it has been suggested to use a thruster-actuated mother spacecraft that deploys hopping rovers for surface mobility [37]. The main drawbacks of this approach are its mechanical and operational complexity, and the fact that hovering at very low gravities can be extremely challenging.
- **Wheeled Mobility**: Wheeled vehicles have been quite successful on bodies with substantial gravity like the Moon and Mars, demonstrating as many as tens of kilometers in driving distance. However, gravitational accelerations in the milli-g to micro-g range limit their practicality for small body applications. Because of very low traction, wheeled vehicles are constrained to extremely low speeds of less than 1.5 mm/s [71], a major issue that prevents fast mobility in microgravity. Other concerns with wheeled vehicles are the complications in maintaining wheel surface contact (required for fine mobility and precision navigation to selected targets) and wheel mechanism sensitivity to dust contamination and external conditions that could cause the wheels to become "stuck." Furthermore, surface bumps that cause loss of contact can result in uncontrolled tumbling, a potentially catastrophic situation for roving in deep space.
- **Legged Mobility**: Legged mobility systems face many challenges in microgravitational environments. The primary drawbacks of legged systems are their mechanical and operational complexity, the need for some form of anchoring system, and a strong dependence of performance on regolith properties [30, 117]. Unfortunately, as surface characteristics and regolith physics are largely unknown before launch, designing legs with good grasping properties is challenging. On the positive side, legged systems would provide very precise mobility.

- **Hopping Mobility**: Hopping rovers, or "hoppers," are perhaps the most promising technology for future missions to microgravitational environments. Their key advantage is that, with a fairly simple actuation mechanism, they are capable of large surface coverage with relatively little control effort. Moreover, they are less sensitive to the regolith properties of small body objects. Indeed, unlike other types of actuation, hopper designs seek to exploit the low gravity to their advantage, rather than facing it as a constraint. A particularly useful bonus of internal actuation mechanisms on hopper platforms is self-containment of moving parts, which significantly reduces the problem of dust contamination and thermal control. One of the potential drawbacks to hopping mobility, however, is precision maneuvering for targeted instrument placement and sampling hard surfaces. In spite of this, if one is able to devise control strategies for fine mobility, hopping robots with internal actuation could represent a good trade-off between performance and complexity (see also an analogous conclusion in [114]).

Unlike typical rover developments targeted for larger bodies, development of microgravity technologies calls for specialized test beds (see Fig. 1.10), which are expensive and have operational constraints. As a result, a necessary task for microgravity technologies would be the development of high-fidelity simulations and cross-validation with results from experimental test beds and environments. High-fidelity physics-based simulations of the regolith and its interaction with the platforms, such as granular media microgravity simulations, would play a significant role in enhancing our understanding of small-body mobility.

Several subsidiary technologies would also be relevant to microgravity mobility. Robotic mobility advancements are strongly correlated with a number of fields, particularly power and energy regulation, thermal control, structural material development, planning and guidance algorithms, and telemetry and sensing. Each of these subcategories and their benefits to microgravity mobility is described below.

Power Supply

Mobility platforms, like all space-based applications, are tightly constrained by available power. This is particularly apt for operations in microgravity. For example, the

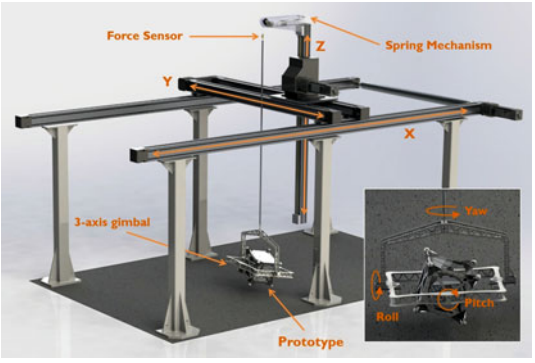

Fig. 1.10 A six degree-of-freedom gravity offload test bed for testing mobility platforms in emulated microgravity. *Image courtesy of Stanford University*

average power consumption for a Phobos-like environment is on the order of 15 W. For mobility systems functioning primarily off of batteries, with no recharging capability and assuming current state-of-the-art technology, lifetimes would be limited to a couple of days at the most. Future efforts should explore life-expanding power subsystem approaches, most likely including hybrid systems of multiple power sources. To increase microgravity assets' lifetimes beyond 48 hours, it may be necessary to consider a combination of solar panels and secondary batteries. The critical concerns for this system would be the available solar cell area and the possibility of solar cell regolith dust build-up. Contact with the surface or the use of thrusters that stir up dust may make solar cell/secondary battery choices unacceptably risky. Given the uncertainty of the dust environment, it may be that *miniaturized* Radioisotope Thermoelectric Generators (RTGs) would provide a lower-risk power alternative, despite the cost and regulatory issues. Recent breakthroughs in this field might make this option viable. Another alternative technology that appears promising are advanced regenerative fuel cell systems.

Thermal Control

Thermal requirements differ widely depending on the environment being explored. Continuing with the example of Phobos, the moon's rapid movement (7.66 h orbital period) helps to average out the hot and cold exposure experienced on its surface. First-order estimates show a thermal time constant on the order of the orbital period, with an average temperature slightly above freezing [29]. Hence, at least for Phobos and other short-period small bodies, passive thermal protection with additional coatings and multi-layer insulation could be acceptable. On the other hand, for the case of slowly-rotating NEOs, Radioisotope Heater Units (RHUs) may be required if worst-case temperatures fall below minimum values allowed for electrical heaters consistent with the planned electrical power system. An RTG or RHU would most likely require a heat switch designed to prevent overheating during the pre-launch and cruise phases. During surface operations, mobile assets would also need to be isolated against heat exchange with the ground.

Shielding Against Electrostatic Effects

Electrostatic effects arising from solar wind and plasma build-up in Debye sheaths on the dusty surfaces of celestial objects have the potential to wreak havoc on the electrical components of space vehicles. However, if the electrostatic field has a potential less than 100 V (as appears typical for most small bodies), electrostatic charging should not represent a significant problem for deployed rovers' operations, e.g. during telecommunications between mobile rovers and their mothership, arguably the most sensitive subsystem to static. For hoppers in continuous tumbles, any net accumulated charge should rapidly reach an equilibrium with the surface. The only phase that could represent a risk to such designs is the night-day transition; a possible solution would be to turn off all telecommunications and allocate an initial phase for the hopper to "shake" itself by tumbling. Other potential mitigation strategies for static electricity include: (1) encapsulating hoppers or thruster-actuated mobile assets in a wire cage that would prevent communications equipment from touching

the ground, or (2) automatic off-switches that activate when mobile assets are not in communication with the mothership.

Localization and Navigation

Localization and navigation are key challenges, particularly for unmapped environments such as small bodies, which have not yet been fully characterized. During local navigation across the terrain, existing localization approaches for rolling or walking robots may apply, such as the use of extended Kalman Filters to fuse celestial sensor data and optical-flow measurements [12]. Through dynamic sensors such as MEMS inertial measurement units, accelerometers, gyroscopes, and contact sensors, mobility platforms could also reconstruct their trajectory and hence determine their current position. One or more sun sensors or star trackers could be incorporated for attitude determination; thruster-actuated mobility platforms may be able to employ horizon sensors as well during far operations. However, dynamic sensing approaches may be subject to large position errors due to sensor drift. This motivates the use of imaging sensors, which can map the local environment to assess terrain hazards and identify nearby rocks and features to help with localization. Depending on the geometrical constraints of mobile assets, vision may not be feasible or ideal. Small and compact platforms would capture images from low vantage points, resulting in large occlusions and significant geometric variations. They also constrain the baseline for stereo vision (thus limiting depth perception). For hopping platforms, the continuously rotating fields of view would make mapping and localization particularly challenging and would call for new, less resource-intensive algorithms.

Multi-asset mission architectures, which employ a hosting spacecraft or mothership together with minimalistic mobile rovers, demand special attention. Given the low-mass, small-scale construction and the limited computational capabilities of such rovers, localization should rely on novel synergistic mission operations wherein the mothership and its daughter assets share the responsibility for localization and mapping. As this scenario is unprecedented, this presents some unique opportunities for technology development in the area of hierarchical synergistic operations. Within this architecture, localization of the rovers could be achieved through fusion of sensors onboard both the mothership and its daughter assets, with the mothership bearing the primary responsibility for rover localization. To keep the complexity, computation and power of the mobility assets to a minimum, the rovers should be responsible only for local perception and carry a minimal suite of navigational sensors. The major hurdle associated with this architecture is its sensitivity to reliable telecommunication.

On-Board Handling and Telemetry

Due to the largely uncertain environment on small-body objects, successful attempts at communication for control commands are likely to be sporadic and discontinuous. This poses a significant challenge, particularly for multi-asset operations. Irregular line-of-sight with the mothership would force each mobile platform to operate autonomously, collecting, compressing, and storing data in between available uplink opportunities. In low radiation environments, an FPGA, small micro-controller or

micro-processor solution would be a favorable choice with relatively high-density memory. The nature of the scientific payload would naturally allow for a high degree of sequential operation with the initial uplink of accelerometer data, followed by in situ data.

1.5 Conclusions

This chapter has addressed some of the engineering aspects and challenges associated with technology area TA04 "Robotics, Tele-Robotics, and Autonomous Systems," expanding the discussion of the 2011 NRC Report on top technology priorities for NASA's Office of the Chief Technologist to a more detailed, technical scope. Specifically, this chapter has discussed the "Relative Guidance Algorithms," "Extreme-Terrain Mobility," and "Small-Body/Microgravity Mobility" technologies within the autonomous systems area, motivating the importance of each, highlighting current state-of-the-art methods, and outlining the major technical hurdles facing the aerospace engineering and robotics communities.

Spacecraft guidance and control has attained a sufficient level of maturity that the majority of remaining technological advancement lies in on-board guidance capability and performance. Robust, real-time implementable, and verifiable optimization algorithms for "Relative Guidance," as discussed in the second section of this chapter, are necessary to address situations involving delayed communications, time-varying obstacles, elevated mission risk, and tight maneuver tolerances. Important applications on the forefront of today's capability include planetary entry, descent, and landing, autonomous rendezvous and docking, autonomous inspection and servicing, and proximity operations about small bodies. Enhanced autonomy in these difficult applications will require the extension of modern state-of-the-art techniques, including Mixed-Integer Linear Programming, Model Predictive Control, Artificial Potential Functions, and motion planning algorithms, as well as the invention of novel approaches. As described in the chapter, prospective approaches will need to be able to deal with logical modes, handle complex state-control constraints, and provide certificates of algorithm correctness and convergence rates, all while providing hard guarantees of mission safety

In addition to spacecraft, future science and human exploration missions will heavily rely on autonomous control of mobile systems operating on and in proximity of extreme, hazardous landscapes of extraterrestrial bodies, including deep craters, canyons, fissures, gullies and cryovolcanoes. The discussion in the third section of this chapter on "Extreme Terrain Mobility" prompts for further technology advancements toward the development of affordable and versatile mobility platforms that would enable access to otherwise inaccessible areas, capable of safely traversing to multiple and designated targets, loitering for in situ measurements, and harvesting samples from extreme terrains. Conventional, flat-topography rover designs must be re-evaluated in the context of such high-risk missions in order to avoid the dangers of tip-over, loose regolith, and other uncompromising terrain hazards. The advance-

ments described in this chapter revolved around novel traverse technologies, tethered mobility and control (including anchoring and fixturing deployment and management), avionics and terrain equipment, traversability analysis, motion planning techniques, and lastly high-fidelity terrain modeling and mobility simulation. Motion planning algorithms and control laws must be developed so that both fine mobility and instrument pointing can be reliably achieved over extreme terrains with narrower targets on motion accuracy.

The subject of mobility was extended further in the final section of the chapter to the specialized case of microgravity. Weak gravitational fields are characteristic of celestial small bodies, whose unique environments call for dramatically different modes of operation. "Small Body/Microgravity Mobility" constitutes mobile operations on Near-Earth Objects (NEOs), asteroids, comets, irregularly-shaped objects, and planetary moons, enabling the access to and study of entirely new and highly-prized scientific sites, including Phobos, Deimos, Enceladus, and Phoebe. Microgravity introduces a number of new and difficult challenges. Simple operations such as drilling or coring can be quite difficult unless some form of fixturing or anchoring is used to impart necessary stabilization forces. Rovers relying on traditional mobility concepts (such as wheels and legs) originally developed for high-gravity environments cannot be used without significant modifications. On the other hand, low gravity enables entirely new types of mobility, namely thruster-actuated locomotion and hopping by surface impact and/or internal actuation mechanisms. Concurrent technological maturation of key subsystems is needed to enable these extreme applications of engineering. Research must be done to identify power supply options to increase mobility platform lifetimes, further develop communication and localization strategies, improve thermal control and electrostatic shielding, and enable on-board handling and telemetry. Finally, trades between monolithic and multi-asset mission architectures will be needed to determine the most appropriate balance of computational load for localization, mapping and motion planning between mobile assets and potential host spacecraft; this paradigm-shifting approach for synergistic mission operations directly exploits small bodies' low gravity in the design process, rather than facing it as a constraint, a key design perspective that will need to be adopted in order to enable small-body missions.

Acknowledgments The development of this chapter was partially supported by an Early Career Faculty grant from NASA's Space Technology Research Grants Program (Grant NNX12AQ43G), by the NASA Innovative Advanced Concepts program (Grant NNX14AT49G), and by JPL under the R&TD, CIF, and CAP programs. The authors wish to acknowledge insightful discussions with Dr. Cinzia Zuffada (JPL), Dr. Tom Cwik (JPL), and Dr. Jonas Zmuidzinas (JPL). Government sponsorship acknowledged.

References

1. Açıkmeşe, B., Blackmore, L.: Lossless Convexification of a Class of Optimal Control Problems with Non-convex Control Constraints. Automatica **47**(2), 341–347 (2011)

2. Açikmeşe, B., Carson, J.M., Bayard, D.S.: A Robust Model Predictive Control Algorithm for Incrementally Conic Uncertain/Nonlinear Systems. International Journal on Robust and Nonlinear Control **21**(5), 563–590 (2011)
3. Açıkmese, B., Ploen, S.R.: Convex Programming Approach to Powered Descent Guidance for Mars Landing. AIAA Journal of Guidance, Control, and Dynamics **30**(5), 1353–1366 (2007)
4. Allen, R., Pavone, M., McQuin, C., Nesnas, I. A. D., Castillo-Rogez, J. C., Nguyen, T.-N., and Hoffman, J. A. Internally-Actuated Rovers for All-Access Surface Mobility: Theory and Experimentation. Proc. IEEE Conf. on Robotics and Automation, pp. 5481–5488. Karlsruhe, Germany, 2013
5. Ardaens, J.-S., D'Amico, S.: Spaceborne Autonomous Relative Control System for Dual Satellite Formations. AIAA Journal of Guidance, Control, and Dynamics **32**(6), 1859–1870 (2009)
6. Arvidson, R. E., Bell, J. F., Bellutta, P., Cabrol, N. A., Catalano, J. G., Cohen, J., Crumpler, L. S., Des Marais, D. J., Estlin, T. A., and Farrand, W. H. e. a. Spirit Mars Rover Mission: Overview and Selected Results from the Northern Home Plate Winter Haven to the Side of Scamander Crater. Journal of Geophysical Research, vol. 115 (E7), 2010
7. Badawy, A., McInnes, C.R.: On-Orbit Assembly Using Superquadric Potential Fields. AIAA Journal of Guidance, Control, and Dynamics **31**(1), 30–43 (2008)
8. Badescu, M., Bao, X., Bar-Cohen, Y., Chang, Z., Dabiri, B. E., Kennedy, B., and Sherrit, S. Adapting the Ultrasonic/Sonic Driller/Corer for Walking/Climbing Robotic Applications. SPIE Smart Structures and Materials, pp. 160–168. San Diego, CA, 2005
9. Bajracharya, M., Maimone, M.W., Helmick, D.: Autonomy for Mars Rovers: Past, Present, and Future. IEEE Computer **41**(12), 44–50 (2008)
10. Bares, J.E., Wettergreen, D.S.: Dante II: Technical Description, Results, and Lessons Learned. International Journal of Robotics Research **18**(7), 621–649 (1999)
11. Bartlett, P., Wettergreen, D. S., and Whittaker, W. Design of the SCARAB Rover for Mobility and Drilling in the Lunar Cold Traps. i-SAIRAS, pp. 3–6. ESA, Hollywood, CA, 2008
12. Baumgartner, E. T., Schenker, P. S., Leger, C., and Huntsberger, T. L. Sensor-fused Navigation and Manipulation from a Planetary Rover. SPIE Symposium on Sensor Fusion and Decentralized Control in Robotic Systems, vol. 3523, pp. 125–134. Boston, MA, 1998
13. Beard, R.W., Lawton, J., Hadaegh, F.Y.: A Coordination Architecture for Spacecraft Formation Control. IEEE Transactions on Control Systems Technology **9**(6), 777–790 (2001)
14. Beauchamp, P. M., Cutts, J. A., Quadrelli, M., Wood, L., Riedel, J. E., McHenry, M. C., Aung, M., Volpe, R., and Cangahuala, L. Guidance Navigation and Control Technology Assessment for Future Planetary Science Missions. AIAA SPACE Conferences & Exposition. San Diego, CA, 2013
15. Betts, J.T.: Survey of Numerical Methods for Trajectory Optimization. AIAA Journal of Guidance, Control, and Dynamics **21**(2), 193–207 (1998)
16. Bevilacqua, R., Lehmann, T., Romano, M.: Development and Experimentation of LQR/APF Guidance and Control for Autonomous Proximity Maneuvers of Multiple Spacecraft. Acta Astronautica **68**(7–8), 1260–1275 (2011a)
17. Bevilacqua, R., Romano, M., Curti, F., Caprari, A. P., and Pellegrini, V. Guidance Navigation and Control for Autonomous Multiple Spacecraft Assembly: Analysis and Experimentation. Int. Journal of Aerospace Engineering, pp. 1–18, 2011b
18. Blackmore, L., Açıkmeşe, B., Scharf, D.P.: Minimum Landing-Error Powered-Descent Guidance for Mars Landing using Convex Optimization. AIAA Journal of Guidance, Control, and Dynamics **33**(4), 1161–1171 (2010)
19. Blake, C. Dynamics and Control of Satellite Formations Using a Quasi-rigid Body Formulation. Ph.D. thesis, McGill University, Montreal, Quebec, CA, 2008
20. Board, D. M. I. Overview of the DART Mishap Investigation Results. Tech. rep., NASA, 2006. Available at http://www.nasa.gov/pdf/148072main_DART_mishap_overview.pdf
21. Boyd, S. and Vandenberghe, L. Convex Optimization. Cambridge University Press, 2004
22. Breger, L., How, J.P.: Safe Trajectories for Autonomous Rendezvous of Spacecraft. AIAA Journal of Guidance, Control, and Dynamics **31**(5), 1478–1489 (2008)

23. Brock, O. and Khatib, O. Real-time Re-planning in High-Dimensional Configuration Spaces Using Sets of Homotopic Paths. Proc. IEEE Conf. on Robotics and Automation, vol. 1, pp. 550–555. San Francisco, CA, 2000
24. Buehler, M., Iagnemma, K., and Singh, S. (eds.). The DARPA Urban Challenge: Autonomous Vehicles in City Traffic, vol. 56 of Tracts in Advanced Robotics. Springer, 1st edn., 2010
25. Camacho, E. F. and Bordons, C. Nonlinear Model Predictive Control: An Introductory Review. Findeisen, R., Allgöwer, F., and B., L. T. (eds.), Assessment and Future Directions of Nonlinear Model Predictive Control, vol. 358 of Lecture Notes in Control and Information Sciences, pp. 1–16. Springer, 2007
26. Carson, J. M., Açikmeşe, B., Murray, R. M., and MacMynowski, D. G. A Robust Model Predictive Control Algorithm with a Reactive Safety Mode. Chung, M. J. and Misra, P. (eds.), IFAC World Congress, vol. 17, pp. 13175–13181. Gangnam-gu Seoul, South Korea, 2008
27. Carson III, J. M., Açıkmeşe, B., Blackmore, L., and Wolf, A. A. Capabilities of Convex Powered-Descent Guidance Algorithms for Pinpoint and Precision Landing. IEEE Aerospace Conference, pp. 1–8. Big Sky, MT, 2011
28. Castillo-Rogez, J. C. and Lunine, J. I. Small Habitable Worlds. Impey, C., Lunine, J. I., and Funes, J. (eds.), Frontiers of Astrobiology, chap. 10, p. 331. Cambridge University Press, Cambridge, UK, 2012
29. Castillo-Rogez, J. C., Pavone, M., Nesnas, I. A. D., and Hoffman, J. A. Expected Science Return of Spatially-Extended In-situ Exploration at Small Solar System Bodies. IEEE Aerospace Conference, pp. 1–15. Big Sky, MT, 2012
30. Chacin, M., Mora, A., and Yoshida, K. Motion Control of Multi-Limbed Robots for Asteroid Exploration Missions. Proc. IEEE Conf. on Robotics and Automation, pp. 3037–3042. Kobe, Japan, 2009
31. Chang, D. E., Shadden, S. C., Marsden, J. E., and Olfati-Saber, R. Collision Avoidance for Multiple Agent Systems. Proc. IEEE Conf. on Decision and Control, vol. 1, pp. 539–543. Maui, HI, 2003
32. Clark, B. C., Sunshine, J. M., A'Hearn, M. F., Cochran, A. L., Farnham, T. L., Harris, W. M., McCoy, T. J., and Veverka, J. NASA Comet Hopper Mission. LPI Asteroids, Comets, Meteors, vol. 1405, p. 8131. Baltimore, MD, 2008
33. Clarke, E., Grumberg, O., and Long, D. Verification Tools for Finite-State Concurrent Systems. de Bakker, J. W., de Roever, W.-P., and Rozenberg, G. (eds.), A Decade of Concurrency Reflections and Perspectives, Lecture Notes in Computer Science, chap. 19, pp. 124–175. Springer, 1994
34. Colaprete, A., Schultz, P., Heldmann, J., Wooden, D., Shirley, M., Ennico, K., Hermalyn, B., Marshall, W., Ricco, A., and Elphic, R. C. e. a. Detection of Water in the LCROSS Ejecta Plume. Science, vol. 330 (6003): pp. 463–468, 2010
35. Committee, M.E.P.A.G.M.G.: Mars Science Goals, Objectives, Investigations, and Priorities: 2010. Tech. rep, NASA (2010)
36. Conrad, P. G. Steep Terrain and the Evolution of Martian Surface Environments: Implications for Habitability. Workshop on Mission Concepts for Accessing and Sampling High-Risk Terrain. Keck Institute for Space Studies, Pasadena, CA, 2009
37. Cunio, P. M., Alibay, F., Meira, P., Sheerin, T., Lanford, E., Krupczak, E., and Hoffman, J. A. Options in the Solar System for Planetary Surface Exploration via Hopping. IEEE Aerospace Conference, pp. 1–10. Big Sky, MT, 2011
38. Davis, T. M. and Melanson, D. XSS-10 Microsatellite Flight Demonstration Program Results. Jr., P. T. and Wright, M. (eds.), Proc. of SPIE, vol. 5419 of SPIE Spacecraft Platforms and Infrastructure, pp. 16–25. Orlando, FL, 2004
39. Dietze, C., Herrmann, F., Kuß, S., Lange, C., Scharringhausen, M., Witte, L., van Zoest, T., Yano, H.: Landing and Mobility Concept for the Small Asteroid Lander MASCOT on Asteroid 1999 JU3. Czech Republic, Int. Astronautical Congress. Prague (2010)
40. D'Souza, C. An Optimal Guidance Law for Planetary Landing. AIAA Conf. on Guidance, Navigation and Control. New Orleans, LA, 1997

41. Dunham, D. W., Farquhar, R. W., McAdams, J. V., Holdridge, M., Nelson, R., Whittenburg, K., Antreasian, P., Chesley, S., Helfrich, C., and Owen, W. M. e. a. Implementation of the First Asteroid Landing. Icarus, vol. 159 (2): pp. 433–438, 2002
42. Dynamics, B. BigDog - The Most Advanced Rough-Terrain Robot on Earth. http://www.bostondynamics.com/robot_bigdog.html, 2012a. Accessed 01 September 2012
43. Dynamics, B. LS3 - Legged Squad Support Systems. http://www.bostondynamics.com/robot_ls3.html, 2012b. Accessed 01 September 2012
44. ESA. Touchdown! Rosetta's Philae Probe Lands on Comet. http://www.esa.int/Our_Activities/Space_Science/Rosetta/Touchdown!_Rosetta_s_Philae_probe_lands_on_comet, 2014
45. Fahroo, F., Ross, I.M.: Direct Trajectory Optimization by a Chebyshev Pseudospectral Method. AIAA Journal of Guidance, Control, and Dynamics **25**(1), 160–166 (2002)
46. Fehse, W. Automated Rendezvous and Docking of Spacecraft, vol. 16. Cambridge University Press, 2003
47. Fiorini, P., Burdick, J.: The Development of Hopping Capabilities for Small Robots. Autonomous Robots **14**(2), 239–254 (2003)
48. for NASA Technology Roadmaps, S. C. NASA Space Technology Roadmaps and Priorities: Restoring NASA's Technological Edge and Paving the Way for a New Era in Space. Tech. rep., NRC, Washington, D.C., 2012. Available at http://www.nap.edu/openbook.php?record_id=13354
49. Frazzoli, E.: Quasi-Random Algorithms for Real-Time Spacecraft Motion Planning and Coordination. Acta Astronautica **53**(4–10), 485–495 (2003)
50. Frazzoli, E., Dahleh, M. A., Feron, E., and Kornfeld, R. A Randomized Attitude Slew Planning Algorithm for Autonomous Spacecraft. AIAA Conf. on Guidance, Navigation and Control, pp. 1–8. Montreal, Quebec, Canada, 2001
51. Fujiwara, A., Kawaguchi, J., Yeomans, D.K., Abe, M., Mukai, T., Okada, T., Saito, J., Yano, H., Yoshikawa, M., Scheeres, D.J., Barnouin-Jha, O., Cheng, A.F., Demura, H., Gaskell, R.W., Hirata, N., Ikeda, H., Kominato, T., Miyamoto, H., Nakamura, A.M., Nakamura, R., Sasaki, S., Uesugi, K.: The Rubble-Pile Asteroid Itokawa as Observed by Hayabusa. Science **312**(5778), 1330–1334 (2006)
52. Fukushima, E.F., Kitamura, N., Hirose, S.: Development of Tethered Autonomous Mobile Robot Systems for Field Works. Advanced Robotics **15**(4), 481–496 (2001)
53. Gayek, J. E. A Survey of Techniques for Approximating Reachable and Controllable Sets. Proc. IEEE Conf. on Decision and Control, pp. 1724–1729. Brighton, England, 1991
54. Gaylor, D. E. and Barbee, B. W. Algorithms for Safe Spacecraft Proximity Operations. AAS Meeting, vol. 127 of Advances in the Astronautical Sciences, pp. 1–20. Seattle, WA, 2007
55. Gill, E., Montenbruck, O., D'Amico, S.: Autonomous Formation Flying for the PRISMA Mission. AIAA Journal of Spacecraft and Rockets **44**(3), 671–681 (2007)
56. Goodman, J.L.: History of Space Shuttle Rendezvous and Proximity Operations. AIAA Journal of Spacecraft and Rockets **43**(5), 944–959 (2006)
57. Goodman, J. L. Lessons Learned From Seven Space Shuttle Missions. Tech. Rep. NASA/CR-2007-213697, United Space Alliance, Houston, TX, 2007
58. Harris, M.W., Açıkmeşe, B.: Lossless convexification of non-convex optimal control problems for state constrained linear systems. Automatica **50**(9), 2304–2311 (2014a)
59. Harris, M.W., Açıkmeşe, B.: Maximum Divert for Planetary Landing Using Convex Optimization. Journal of Optimization Theory & Applications **162**(3), 975–995 (2014b)
60. Hawke, B. R., Giguere, T. A., Gaddis, L. R., Gustafson, O., Lawrence, S. J., Stopar, J. D., Peterson, C. A., Bell, J. F., and Robinson, M. S. e. a. Localized Pyroclastic Deposits in the Grimaldi Region of the Moon. LPI Science Conference Abstracts, vol. 43, p. 1749. The Woodlands, TX, 2012
61. Helmick, D., Douillard, B., Bajracharya, M.: Small Body Surface Mobility with a Limbed Robot. IEEE/RSJ Int. Conf. on Intelligent Robots & Systems. Chicago, IL (2014)
62. Hirose, S., Fukushima, E.F.: Snakes and Strings: New Robotic Components for Rescue Operations. International Journal of Robotics Research **23**(4–5), 341–349 (2004)

63. Horz, F. Lava Tubes - Potential Shelters for Habitats. Mendell, W. W. (ed.), Lunar Bases and Space Activities of the 21st Century, vol. 6, chap. 6, pp. 405–412. Lunar and Planetary Institute, Houston, TX, 1985
64. Howard, A., Nesnas, I. A. D., Werger, B., and Helmick, D. A Novel Reconfigurable Robotic Exploratory Vehicle for Navigation on Rough Terrain. Proc. of the Int. Symposium on Robotics and Applications, pp. 1–6. Seville, Spain, 2004
65. Howard, R. T., Heaton, A. F., Pinson, R. M., and Carrington, C. K. Orbital Express Advanced Video Guidance Sensor. IEEE Aerospace Conference, pp. 1–10. Big Sky, MT, 2008
66. Hull, D.G.: Conversion of Optimal Control Problems into Parameter Optimization Problems. AIAA Journal of Guidance, Control, and Dynamics **20**(1), 57–60 (1997)
67. Izzo, D., Pettazzi, L.: Autonomous and Distributed Motion Planning for Satellite Swarm. AIAA Journal of Guidance, Control, and Dynamics **30**(2), 449–459 (2007)
68. Janson, L. and Pavone, M. Fast Marching Trees: A Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions. International Symposium on Robotics Research, 2013
69. JAXA. Hayabusa Mission. http://hayabusa.jaxa.jp/e/index.html, 2000
70. Jones, R., Wilcox, B.: Nanorover Technology and the MUSES-CN Mission. Tech. rep, JPL (1997)
71. Jones, R., Wilcox, B.: The MUSES CN Rover and Asteroid Exploration Mission. Tech. rep, JPL (2000)
72. JPL and Masten Space Systems. 500 meter Xombie Divert Test Flight for G-FOLD (Guidance for Fuel Optimal Large Divert) Validation. http://www.youtube.com/watch?v=1GRwimo1AwY, 2012a
73. JPL and Masten Space Systems. 650 meter Xombie Divert Test Flight for G-FOLD (Guidance for Fuel Optimal Large Divert) Validation. http://www.youtube.com/watch?v=WU4TZlA3jsg, 2012b
74. JPL and Masten Space Systems. 750 meter Xombie Divert Test Flight for G-FOLD (Guidance for Fuel Optimal Large Divert) Validation. http://www.youtube.com/watch?v=jl6pw2ossU, 2012c
75. Karaman, S. and Frazzoli, E. Optimal Kinodynamic Motion Planning using Incremental Sampling-based Methods. Proc. IEEE Conf. on Decision and Control, pp. 7681–7687, 2010
76. Kavraki, L.E., Švestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic Roadmaps for Path Planning in High-Dimensional Spaces. IEEE Transactions on Robotics and Automation **12**(4), 566–580 (1996)
77. Kawano, I., Mokuno, M., Kasai, T., Suzuki, T.: Result of Autonomous Rendezvous Docking Experiment of Engineering Test Satellite-VII. AIAA Journal of Spacecraft and Rockets **38**(1), 105–111 (2001)
78. Klumpp, A.R.: Apollo Lunar Descent Guidance. Automatica **10**(2), 133–146 (1974)
79. Kuwata, Y., Teo, J., Fiore, G., Karaman, S., Frazzoli, E., How, J.P.: Real-Time Motion Planning With Applications to Autonomous Urban Driving. IEEE Transactions on Control Systems Technology **17**(5), 1105–1118 (2009)
80. LaValle, S. M. Planning Algorithms. Cambridge University Press, 2006
81. LaValle, S.M., Kuffner, J.J.: Randomized Kinodynamic Planning. International Journal of Robotics Research **20**(5), 378–400 (2001)
82. Lefebvre, M.-A., Guéguen, H.: Hybrid Abstractions of Affine Systems. Nonlinear Analysis: Theory, Methods & Applications **65**(6), 1150–1167 (2006)
83. Leonard, J., How, J.P., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., Koch, O., Kuwata, Y., Moore, D., Olson, E., Peters, S., Teo, J., Truax, R., Walter, M., Barrett, D., Epstein, A., Maheloni, K., Moyer, K., Jones, T., Buckley, R., Antone, M., Galejs, R., Krishnamurthy, S., Williams, J.: A Perception-Driven Autonomous Urban Vehicle. Journal of Field Robotics **25**(10), 727–774 (2008)
84. Major, L. M., Brady, T. M., and Paschall, S. C. Apollo Looking Forward: Crew Task Challenges. IEEE Aerospace Conference, pp. 1–8. Big Sky, MT, 2009

85. Mattingley, J., Boyd, S.: Real-time Convex Optimization in Signal Processing. IEEE Signal Processing Magazine **27**(3), 50–61 (2010)
86. Mayne, D., Rawlings, J., Rao, C., Scokaert, P.: Constrained Model Predictive Control: Stability and Optimality. Automatica **36**(6), 789–814 (2000)
87. McInnes, C. R. Potential Function Methods for Autonomous Spacecraft Guidance and Control. AAS Astrodynamics Specialist Conference, pp. 2093–2109. Halifax, Nova Scotia, Canada, 1995
88. Meditch, J.S.: On the Problem of Optimal Thrust Programming for a Lunar Soft Landing. IEEE Transactions on Automatic Control **9**(4), 477–484 (1964)
89. Miller, S. L., Bell, J. L., Graf, J. E., and Matousek, S. E. Potential Future Mars Missions. AIAA SPACE Conferences & Exposition. Long Beach, CA, 2000
90. Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., et al.: Junior: The Stanford Entry in the Urban Challenge. Journal of Field Robotics **25**(9), 569–597 (2008)
91. Najson, F. and Mease, K. D. A Computationally Inexpensive Guidance Algorithm for Fuel-Efficient Terminal Descent. AIAA Journal of Guidance, Control, and Dynamics, vol. 29 (4), 2006
92. NASA. The Vision for Space Exploration. http://www.nasa.gov/pdf/55583main_vision_space_exploration2.pdf, 2004
93. NASA. Mars Science Laboratory Curiosity Rover. http://marsprogram.jpl.nasa.gov/msl/, 2011a. Retrieved January 8th, 2011
94. NASA. Spirit and Opportunity, Mars Exploration Rovers. http://www.nasa.gov/mission_pages/mer/, 2011b. Retrieved January 8th, 2011
95. Nesnas, I.A.D.: Reconfigurable Exploratory Robotic Vehicles. NASA Tech Briefs **25**(7), 56 (2001)
96. Nesnas, I. A. D., Abad-Manterola, P., Edlund, J. A., and Burdick, J. W. Axel Mobility Platform for Steep Terrain Excursions and Sampling on Planetary Surfaces. IEEE Aerospace Conference, pp. 1–11. Big Sky, MT, 2008
97. Nesnas, I.A.D., Matthews, J.B., Abad-Manterola, P., Burdick, J.W., Edlund, J.A., Morrison, J.C., Peters, R.D., Tanner, M.M., Miyake, R.N., Solish, B.S., et al.: Axel and DuAxel Rovers for the Sustainable Exploration of Extreme Terrains. Journal of Field Robotics **29**(4), 663–685 (2012)
98. Nesterov, Y., Nemirovsky, A.: Interior-point Polynomial Methods in Convex Programming. SIAM, Philadelphia, PA (1994)
99. Nolet, S., Kong, E., and Miller, D. W. Design of an Algorithm for Autonomous Docking with a Freely Tumbling Target. Motaghedi, P. (ed.), Proc. of SPIE, vol. 5799 of SPIE Modeling, Simulation, and Verification of Space-based Systems, pp. 123–134. Orlando, FL, 2005
100. Nuth, J., Fernandez, Y., Britt, D., Zolensky, M., Moore, M., Nesvomy, D., Abell, P., Dankanich, J., Sykes, M., et al. Roadmap for Small Bodies Exploration. Tech. rep., Small Bodies Assessment Group, 2011. Available at http://www.lpi.usra.edu/sbag/roadmap/
101. OCT, N. NASA Space Technology Roadmaps. Tech. Rep. TA04 - Robotics, Tele-Robotics and Autonomous Systems, NASA, 2013
102. Oda, M. ETS-VII: Achievements, Troubles and Future. i-SAIRAS, pp. 1–7. ESA, Montreal, Quebec, Canada, 2001
103. on the Planetary Science Decadal Survey, C. Vision and Voyages For Planetary Science in the Decade 2013–2022. Tech. rep., NRC, Washington, D.C., 2011. Available at http://solarsystem.nasa.gov/2013decadal/
104. Park, H., Di Cairano, S., and Kolmanovsky, I. V. Model Predictive Control for Spacecraft Rendezvous and Docking with a Rotating/Tumbling Platform and for Debris Avoidance. American Control Conference, pp. 1922–1927. San Francisco, CA, 2011
105. Parness, A., Frost, M., Thatte, N., King, J.P., Witkoe, K., Nevarez, M., Garrett, M., Aghazarian, H., Kennedy, B.: Gravity-Independent Rock-Climbing Robot and a Sample Acquisition Tool with Microspine Grippers. Journal of Field Robotics **30**(6), 897–915 (2013)

106. Phillips, J. M., Kavraki, L. E., and Bedrossian, N. Spacecraft Rendezvous and Docking with Real-Time, Randomized Optimization. AIAA Conf. on Guidance, Navigation and Control, pp. 1–11. Austin, TX, 2003
107. Pirjanian, P., Leger, C., Mumm, E., Kennedy, B., Garrett, M., Aghazarian, H., Farritor, S., and Schenker, P. Distributed Control for a Modular, Reconfigurable Cliff Robot. Proc. IEEE Conf. on Robotics and Automation, vol. 4, pp. 4083–4088. Washington D.C., 2002
108. Polites, M. E. An Assessment of the Technology of Automated Rendezvous and Capture in Space. Tech. Rep. NASA/TP-1998-208528, NASA, 1998
109. Quinlan, S. and Khatib, O. Elastic Bands: Connecting Path Planning and Control. Proc. IEEE Conf. on Robotics and Automation, vol. 2, pp. 802–807. Atlanta, GA, 1993
110. Richards, A., Schouwenaars, T., How, J.P., Feron, E.: Spacecraft Trajectory Planning With Avoidance Constraints Using Mixed-Integer Linear Programming. AIAA Journal of Guidance, Control, and Dynamics **25**(4), 755–765 (2002)
111. Rumford, T. E. Demonstration of Autonomous Rendezvous Technology (DART) Project Summary. Jr., P. T. and Shoemaker, J. (eds.), Proc. of SPIE, vol. 5088 of SPIE Space Systems Technology and Operations, pp. 10–19. Orlando, FL, 2003
112. Sagdeev, R.Z., Zakharov, A.V.: Brief History of the Phobos Mission. Nature **341**(6243), 581–585 (1989)
113. Scharf, D. P., Hadaegh, F. Y., and Ploen, S. R. A Survey of Spacecraft Formation Flying Guidance and Control, Part II: Control. American Control Conference, vol. 4, pp. 2976–2985. Boston, MA, 2004
114. Scheeres, D. J. Close Proximity Operations for Implementing Mitigation Strategies. Planetary Defense Conference, pp. 1–11. AIAA, Orange County, CA, 2004
115. Schmerling, E., Janson, L., Pavone, M.: Optimal sampling-based motion planning under differential constraints: the drift case with linear affine dynamics (2014). http://arxiv.org/pdf/1405.7421.pdf
116. Schmerling, E., Janson, L., Pavone, M.: Optimal sampling-based motion planning under differential constraints: the driftless case. In: Proceedings of the IEEE Conference on Robotics and Automation (2015)
117. Seeni, A., Schafer, B., Rebele, B., and Tolyarenko, N. Robot Mobility Concepts for Extraterrestrial Surface Exploration. IEEE Aerospace Conference, pp. 1–14. Big Sky, MT, 2008
118. Starek, J. A., Barbee, B. W., and Pavone, M. A Sampling-Based Approach to Spacecraft Autonomous Maneuvering with Safety Specifications. AAS GN&C Conference. Breckenridge, CO, 2015
119. Steinfeld, B.A., Grant, M.J., Matz, D.A., Braun, R.D., Barton, G.H.: Guidance, Navigation, and Control System Performance Trades for Mars Pinpoint Landing. AIAA Journal of Spacecraft and Rockets **47**(1), 188–198 (2010)
120. Stipanovic, D.M., Hwang, I., Tomlin, C.J.: Computation of an Over-Approximation of the Backward Reachable Set Using Subsystem Level Set Functions. Dynamics of Continuous Discrete and Impulsive Systems **11**, 397–412 (2004)
121. Stoeter, S.A., Papanikolopoulos, N.: Kinematic Motion Model for Jumping Scout Robots. IEEE Transactions on Robotics and Automation **22**(2), 397–402 (2006)
122. Topcu, U., Casoliva, J., Mease, K.D.: Minimum-Fuel Powered Descent for Mars Pinpoint Landing. AIAA Journal of Spacecraft and Rockets **44**(2), 324–331 (2007)
123. Truszkowski, W.F., Hinchey, M.G., Rash, J.L., Rouff, C.A.: Autonomous and Autonomic Systems: A Paradigm for Future Space Exploration Missions. IEEE Transactions on Systems, Man, & Cybernetics. Part C: Applications & Reviews **36**(3), 279–291 (2006)
124. Tsuda, Y., Yoshikawa, M., Abe, M., Minamino, H., Nakazawa, S.: System Design of the Hayabusa 2 – Asteroid Sample Return Mission to 1999 JU3. Acta Astronautica **91**, 356–362 (2013)
125. Ulamec, S., Biele, J.: Surface Elements and Landing Strategies for Small Bodies Missions - Philae and Beyond. Advances in Space Research **44**(7), 847–858 (2009)
126. Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M.N., Dolan, J., Duggins, D., Galatali, T., Geyer, C., et al.: Autonomous Driving in Urban Environments: Boss and the Urban Challenge. Journal of Field Robotics **25**(8), 425–466 (2008)

127. VanDyke, M.C., Hall, C.D.: Decentralized Coordinated Attitude Control Within a Formation of Spacecraft. AIAA Journal of Guidance, Control, and Dynamics **29**(5), 1101–1109 (2006)
128. Vlassenbroeck, J., Dooren, R.V.: A Chebyshev Technique for Solving Nonlinear Optimal Control Problems. IEEE Transactions on Automatic Control **33**(4), 333–340 (1988)
129. Wargo, M. J. HEOMD Strategic Knowledge Gaps: Planning for Safe, Effective, and Efficient Human Exploration of the Solar System. http://science.nasa.gov/media/medialibrary/2012/05/04/HEOMD_Strategic_Knowledge_Gaps_Mike_Wargo.pdf, 2012
130. Way, D. On the Use of a Range Trigger for the Mars Science Laboratory Entry, Descent, and Landing. IEEE Aerospace Conference, pp. 1–8. Big Sky, MT, 2011
131. Weiss, A., Baldwin, M., Petersen, C., Erwin, R. S., and Kolmanovsky, I. V. Spacecraft Constrained Maneuver Planning for Moving Debris Avoidance Using Positively Invariant Constraint Admissible Sets. American Control Conference, pp. 4802–4807. Washington, DC, 2013
132. Widnall, W. S. Apollo Guidance Navigation and Control: Guidance System Operations Plan for Manned CM Earth Orbital and Lunar Missions Using Program COLOSSUS I and Program COLOSSUS IA. Tech. Rep. R-577 Section 3, MIT Instrumentation Laboratory, Cambridge, MA, 1968
133. Wilcox, B. H. ATHLETE: A Cargo-Handling Vehicle for Solar System Exploration. IEEE Aerospace Conference, pp. 1–8. Big Sky, MT, 2011
134. Wilcox, B.H., Litwin, T., Biesiadecki, J., Matthews, J., Heverly, M., Morrison, J., Townsend, J., Ahmad, N., Sirota, A., Cooper, B.: ATHLETE: A Cargo Handling and Manipulation Robot for the Moon. Journal of Field Robotics **24**(5), 421–434 (2007)
135. Yano, H., Kubota, T., Miyamoto, H., Okada, T., Scheeres, D., Takagi, Y., Yoshida, K., Abe, M., Abe, S., Barnouin-Jha, O., Fujiwara, A., Hasegawa, S., Hashimoto, T., Ishiguro, M., Kato, M., Kawaguchi, J., Mukai, T., Saito, J., Sasaki, S., Yoshikawa, M.: Touchdown of the Hayabusa Spacecraft at the Muses Sea on Itokawa. Science **312**(5778), 1350–1353 (2006)
136. Yu, J. X. and Cheng, J. Graph Reachability Queries: A Survey. Managing and Mining Graph Data, pp. 181–215, 2010

Chapter 2
New Guidance, Navigation, and Control Technologies for Formation Flying Spacecraft and Planetary Landing

Fred Y. Hadaegh, Andrew E. Johnson, David S. Bayard, Behçet Açıkmeşe, Soon-Jo Chung and Raman K. Mehra

2.1 Introduction

The recent landing of the massive Mars Science Laboratory (MSL) rover Curiosity was made possible by the sky-crane touch down system. The sky-crane used a high rate, six degree-of-freedom guidance, navigation, and control (GN&C) system to slowly place the rover on the surface, detect touchdown, and fly away. MSL clearly showed the advantages of on-board closed loop GN&C and has opened the door for infusion of new GN&C technologies into the next Mars lander missions as well as other future spacecraft missions. However, MSL was not equipped with the capabilities of pin-point landing and local hazard avoidance. This chapter begins with a review of recent advances in perception technologies for on-board Hazard Detection (HD) and Terrain Relative Navigation (TRN) in Sect. 2.2. The HD and TRN perception technologies will enable the next Mars lander missions to recognize landmarks for pin-point landing or detect landing hazards on the fly for local hazard avoidance.

F.Y. Hadaegh (✉) · A.E. Johnson · D.S. Bayard · B. Açıkmeşe
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California
e-mail: fred.y.hadaegh@jpl.nasa.gov

A.E. Johnson
e-mail: aej@jpl.nasa.gov

D.S. Bayard
e-mail: david.s.bayard@jpl.nasa.gov

B. Açıkmeşe
e-mail: behcet@austin.utexas.edu

S.-J. Chung
University of Illinois at Urbana-Champaign, Urbana, IL, USA
e-mail: sjchung@illinois.edu

R.K. Mehra
Scientific Systems Company, Inc., Woburn, MA, USA
e-mail: rkm@ssci.com

E. Feron (ed.), *Advances in Control System Technology for Aerospace Applications*, Lecture Notes in Control and Information Sciences 460, DOI 10.1007/978-3-662-47694-9_2

This chapter also considers the problem of flying a swarm of spacecraft in Earth orbit. Distributed spacecraft systems can collectively match or exceed the capability of a more complex monolithic space system. Spacecraft swarms [11, 27] will push the envelope of the existing formation flying spacecraft concepts by increasing the number of spacecraft comprising the swarm by one or two orders of magnitude (100–1000s), hence maximizing the benefit of distributed spacecraft systems.

The swarm approach opens up the possibility for enabling many new mission concepts like creating massively distributed large space apertures, distributed antennas, decentralized sensing networks, anti-satellite disruptors, and geometric arrangements optimized for decoy/camouflage. Moreover, spacecraft swarms can be controlled to exhibit desired complex behaviors, which cannot be achieved by a single spacecraft. Spacecraft swarms are motivated by nature and are examples of bio-inspired engineering systems. Examples from nature include a colony of ants searching for food, or a swarm of bees protecting a bee hive from intruders. Swarming behaviors observed in nature inspire and guide the development of efficient coordination and control algorithms for spacecraft swarms.

In order to construct desired formations of spacecraft swarms and permit coordinated maneuvers of spacecraft, the distributed controller needs to efficiently handle a large of number of spacecraft in the network. Two methods are presented to deal with the added complexity of large number of spacecraft. First, if the aim is to control the relative motions of the spacecraft to generate desired formations of spacecraft swarms, the new method of automatically generating evolving network topologies, presented in Sect. 2.3, can be used to determine the flow of control information among the spacecraft. Second, Sect. 2.4 presents the novel approach of driving the swarm to a desired density distribution in a prescribed region of the configuration space. Instead of controlling individual spacecraft, the probabilistic guidance approach controls the average number of spacecraft per unit volume, ensuring that spatial averages converge to the desired density distribution. The swarm guidance and control methods described in Sects. 2.3 and 2.4 are predicated on an effective solution to (a) detect potential changes in existing orbital trajectories that may lead to damaging collisions; and (b) localize, track, and assess trajectories headed towards collisions. Hence, Sect. 2.5 expands on necessary models, simulations, and methods for deriving, evaluating, and comparing such optimal constellations that satisfy the stated objectives (a) and (b) for various swarm collision scenarios.

2.2 GN&C Technologies for Planetary Landing in Hazardous Terrain

2.2.1 Introduction

All robotic landers to date, including MSL, have landed blindly. They have measured altitude and surface relative velocity and used these measurements for soft landing, but they have not had the ability to recognize landmarks for pin-point landing or

detect landing hazards on the fly for local hazard avoidance. Blind landing has forced missions to select landing ellipses that are free of hazards. This greatly constrains the possible landing locations and either limits the science to what is possible in benign terrain or requires the addition of a long traverse roving capability. For example, MSL landed on the flat and smooth Gale Crater floor and will have to drive out of its 10 kilometer landing ellipse to reach the most interesting science locations.

Hazard Detection and Avoidance (HDA) [19, 21] is an on-board function that collects sensor data to map the landing site, applies an algorithm to determine the safest place to land and then guides the vehicle to the safe landing site. HDA enables the selection of landing ellipses that have a large number of know hazards that can be avoided with a small divert (10–100 m divert). It also enables landing at possibly hazardous locations with limited reconnaissance. HDA requires perception technology for on-board Hazard Detection (HD) and GN&C technology to land the vehicle precisely at the safe landing site.

In contrast, Pin-Point Landing (PPL) is an on-board function that collects sensor data and matches this data to a map of the landing ellipse generated before landing. This match is then used to obtain a map-relative position fix. From this position fix, the lander can compute a trajectory that guides the vehicle to the landing site (1–10 km divert). There is no active detection of hazards on-board, but PPL can be used to avoid large hazards in the landing ellipse. It can also be used for precision deployment of surface assets (e.g., the multi-mission Mars Sample Return architecture or building up a lunar outpost). PPL requires Terrain Relative Navigation (TRN) perception technology [3, 8, 24, 28] and GN&C technology for fuel optimal powered descent guidance.

Figure 2.1 shows a variety of planetary landing sites that can be made accessible by pin-point landing or hazard detection and avoidance. For planetary science, PPL

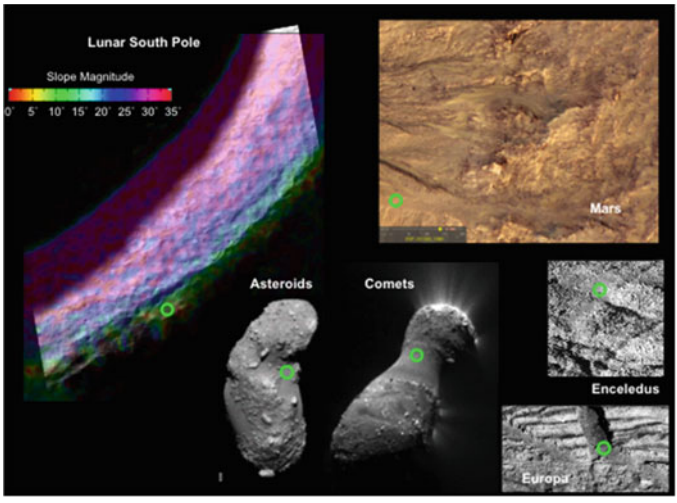

Fig. 2.1 Planetary landing sites that require pin-point landing or hazard detection and avoidance

and HDA will enable access to dust flows on comets and asteroids, seepage features on the side of a crater wall on Mars, the boulder strewn source of water plumes on Enceladus and the cracked and fissured icy terrain of Europa. For manned exploration, these technologies will enable access to the peaks of permanent light near the rugged lunar south pole which are ideal locations for a lunar outpost due to the constant illumination for solar power generation and thermal management and possible near sources of volatiles for in-situ resource utilization. It should be noted that the Apollo lunar program recognized the need for pin-point landing and hazard detection to avoid small and large hazards, but the capability was implemented manually.

The focus of this section is on the TRN and HD perception technologies; whereas GN&C technologies for powered descent are described in another chapter.

2.2.2 Design Considerations

The lander's flight system and descent and landing approach greatly influence TRN and HD design. The trajectory has a major impact on TRN and HD system design because it dictates the time available for data collection and processing, the ranges and off nadir angles for sensor operation, and the attitude rates and velocities for tracking. The mechanical design of the lander sets thresholds on the hazard detection capability by dictating the tolerable surface slope and roughness at touchdown. The performance of the lander's GN&C and propulsion system provide constraints on divert sizes and accuracy, which affect the overall pin-point landing accuracy and safe landing site size. Since TRN and HD algorithms require extensive processing in a short amount of time, it is necessary to have a dedicated, possibly high performance, processor. The size of the lander will also influence the mass, power and volume available for the TRN and HD system.

The environment plays an important role as well. The transmission properties of the atmosphere and the reflectivity of the terrain will influence sensor range and image contrast. If present, dust can reduce sensor range or add noise to sensor measurements. The size and distribution of terrain features (rocks, craters, scarps, hills, etc.) will determine the density of safe landing areas and influence the area needed for HD imaging and PPL divert distances. During passive approaches, the terrain as well as the illumination will influence the appearance of the imaged scene. PPL requires a map made prior to landing; while the performance of TRN depends on the pixel size and quality of the map data. Passive TRN approaches could require rendering of a digital elevation map to generate an image for matching, which makes them more sensitive to map quality than active sensor based approaches [8].

Because planetary landings occur only once, it is difficult if not impossible to test system performance prior to the actual landing. TRN and HD systems must undergo extensive validation and must be designed from the bottom up to be robust. Bolton TRN and HD systems, composed of sensors, processors and algorithms with a low bandwidth interface to the spacecraft, facilitate validation because the entire system can be tested in the field under relevant conditions without the rest of the spacecraft.

Bolt on systems also localize the timing of sensor data and alignment of sensors, which greatly simplifies integration with the spacecraft.

The mission specific design constraints flow down into requirements on the sensors (field of view, number of pixels, maximum range, measurement errors, etc.), algorithms (position accuracy, detection rates, map size, etc.) and processing (clock speed, memory, etc.). As the design space is quite large, there is no generic system for TRN and HD. Instead TRN and HD need to be tailored to each specific application. Since algorithms and sensors have already been developed that meet TRN and HD requirements, the focus has now moved to the development of complete systems. Below, we describe two systems under development: one for Mars robotic landing and the other for crewed lunar landing.

2.2.3 Case Study 1: Mars Robotic System

Mars landers have an entry phase which is used to reduce most of the surface velocity and, in the case of MSL, reduce the landing ellipse size down to 10 Km radius. After entry, a parachute is deployed and the heat-shield is ejected. At this point sensors can image the ground and the TRN function can start. The parachute descent phase lasts from 10 km altitude to 2 Km altitude, with vertical velocities near 100 m/s and horizontal velocities less than 30 m/s. Toward the end of the parachute descent, the off nadir angle is less than 20° and attitude rates are less than 20°/s. TRN processing completes when powered descent starts around 2 km. Powered descent performs a TRN commanded large divert and then goes into a vertical descent phase around 250 m with a 30 m/s vertical velocity. HD occurs quickly at the start of vertical descent, when the off-nadir angle is low and the nominal landing site is directly below. Once the safe site is identified, the lander targets it with a small divert. The entire descent from heat-shield separation to landing, takes on the order of 100 s.

Mars landing typically occurs during the day and at low latitudes, to have direct communication with Earth during landing, so that more accurate and mature camera based TRN approaches can be employed. As was done for MSL, a 1 m digital elevation map with co-registered images can be generated using Mars Reconnaissance Orbiter images. This high resolution map enabled TRN accuracy in the order of 10 m.

Robotic landers are small with touchdown areas in the order of 10 m^2, but Mars is also hazardous with plenty of rocks, scarps and craters. Assuming the MSL hazard tolerance of 55 cm rocks and 22° slopes, studies of the MSL landing sites have shown that a 12 × 12 m hazard map generated with a single flash LIDAR image followed by a 6 m divert greatly reduced the chances of landing on a hazard.

Based on these constraints, the Lander Vision System (LVS) was conceived within NASA's Science Mission Directorate as a tightly integrated bolt-on smart sensor system that has well defined path to flight implementation [20]. The LVS measures terrain relative position, velocity, attitude and altitude while also detecting landing hazards. The LVS sensor suite includes an imaging camera for the landmark recognition required for terrain relative position estimation and image-to-image feature

tracking for horizontal velocity estimation [28]. A dual use flash LIDAR is used for near surface hazard detection and long distance ranging, for measuring altitude through the entire descent. An inertial measurement unit (IMU) propagates vehicle motion between image and LIDAR measurements, so that high rate state information can be provided to the spacecraft. The sensors are tightly integrated with a high performance computer that performs all processing required for TRN, HD, altimetry and velocimetry. The current best estimates for the LVS mass and power are 8 kg and 65 W respectively.

The LVS is optimized to generate robust and accurate measurements from a minimal suite of sensors. Each sensor serves multiple purposes, which reduces mass, volume and development costs. The flash LIDAR is the ideal sensor for hazard detection because it can generate all the data required with a single, low noise, range image taken at long distances [23, 29]. By decreasing the width of the laser illumination, the flash LIDAR can also be used to measure range at high altitude, thereby removing the need to add a separate altimeter [22]. The camera provides images of landmarks for position estimation [8]. It also provides image-to-image feature tracks for velocity estimation, which eliminates the need for a separate velocimeter. Finally, the IMU provides the attitude propagated from the spacecraft's cruise phase that is needed to start TRN and it also allows for high rate updates of the entire navigation state, which is required for closed loop powered descent guidance.

The LVS sensors have low development risks. Cameras and IMUs have already flown on many missions and are not a concern for development. Advanced Scientific Corporation (ASC) has developed a flash LIDAR, and, under NASA funded Small Business Innovative Research Grants, ASC has also built a prototype of a flash LIDAR that satisfies the LVS requirements and also uses parts with flight equivalents [29]. Through extensive field testing, the ASC flash LIDAR has demonstrated that it can detect hazards at low altitude (¡ 500 m) [23] and measure accurate ranges at high altitude (up to 8 km) [22]. Given all these advances toward flight qualification, the flash LIDAR is also a fairly low risk sensor.

The computational tasks are done using a flight qualified processor and a Field Programmable Gate Array (FPGA). The FPGA interfaces with the sensors for fast data acquisition and accurate timing. The FPGA also stores the computationally intensive software vision modules for image normalization, homography-based image warping, image interest operator, frequency domain image correlation and spatial domain image correlation. These high speed modules are used for terrain relative position and horizontal velocity estimation. The processor coordinates the flow of data from the sensors and the processing of modules on the FPGA. It also runs the navigation filter [28] that fuses inertial, imaging and range measurements and interfaces with the spacecraft to receive LVS commands and sends back navigation states and safe landing site locations. The interface between the compute element and the sensors has high bandwidth with tight constraints on timing and data latency. In contrast, the interface with the spacecraft is simpler, with a low data rate and relaxed timetag requirements.

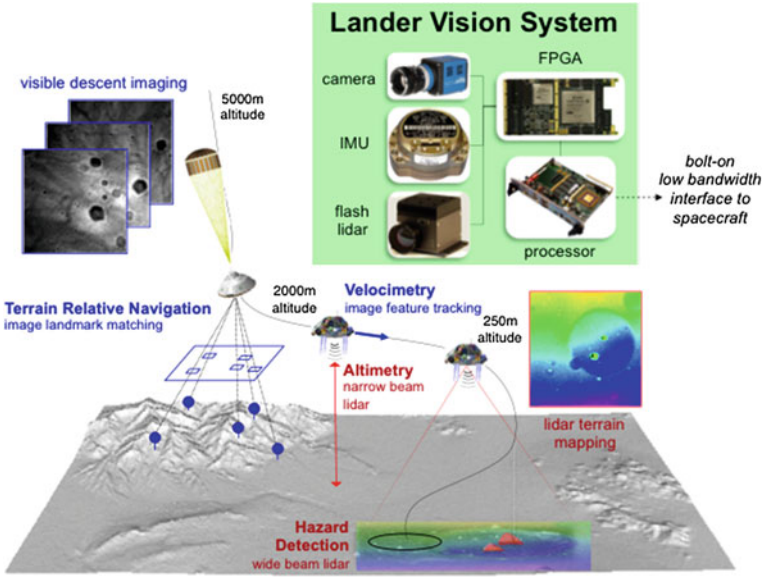

Fig. 2.2 Mars robotic lander vision system concept of operation and components

2.2.4 Case Study 2: Crewed Lunar System

The Autonomous Landing and Hazard Avoidance Technology (ALHAT) project under NASA's Human Exploration and Operations Mission Directorate is developing sensors and algorithms to increase safety during crewed and un-crewed lunar landings [15]. ALHAT's charter is to develop a system that can land anywhere, under any lighting conditions and the lunar south pole is a challenging case that has focused the ALHAT development. Operation under any lighting conditions has resulted in a system that employs active sensors for TRN and HD (Fig. 2.2).

Following the approach used for Apollo, crewed lunar landing starts with a de-orbit maneuver that places the lander on a long shallow trajectory to the surface. During descent, attitude rates are very low and velocities start at 2000 m/s near orbit. When the lander reaches 15 Km altitude, the braking burn begins and the LIDAR is activated to take range measurements. For TRN, the ranges are combined into an elevation contour and this contour is matched with a digital elevation map to obtain a position fix [22, 24]. Based on the TRN measurement, a trajectory is computed on board to clean up the trajectory dispersions (<1 km). This process repeats until around 1–2 km range and 30 m/s velocity, at which point the lander pitches up for landing and crew viewing of the landing site.

The hazard detection phase begins at 1 Km slant range and 30° angle from horizontal. As shown in Fig. 2.3, the ALHAT Hazard Detection System (HDS) raster scans a gimbaled flash LIDAR across a 100×100 m area; the LIDAR combines the flash LIDAR detector from ASC with large collection optics and a high power laser to

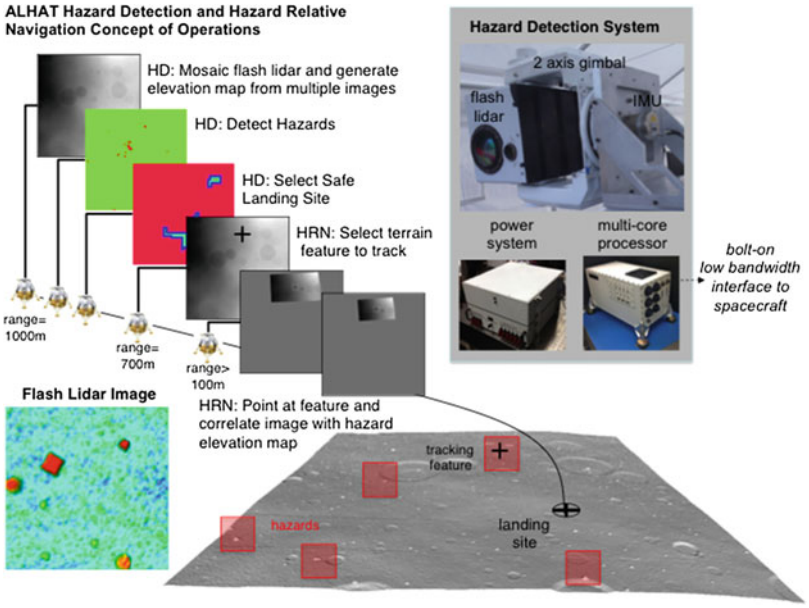

Fig. 2.3 The ALHAT hazard detection system concept of operation and hardware components

obtain the 1 Km imaging range required for ALHAT [3]. The multiple flash LIDAR images are stitched together into a single digital elevation map (DEM) using the onboard navigation solution for coarse placement and a LIDAR data-driven alignment process for fine alignment. The DEM is then passed to an HD algorithm that detects multiple safe landing sites for a lander with a touchdown area around 200 m^2 and sensitive to rocks greater than 30 cm high and slopes greater than 12°.

Once a single safe site is selected by the crew, a trajectory is generated to bring the vehicle to a point 30 m above the target. The HDS then performs Hazard Relative Navigation (HRN) [23] to keep the lander on trajectory to the safe site. During HRN, the LIDAR is pointed at a prominent elevation feature near the landing site, and the feature position is tracked during descent to provide safe site relative navigation updates. The tracked feature actually changes during descent to keep the tracked feature in front of the lander and deal with the large change in sensor footprint. Due to dust kicked up by the propulsion system, the vehicle descends the final 30 m using inertial sensors only. Fortunately, the HRN measurements and velocity from a Doppler LIDAR velocimeter, also developed in ALHAT [3], provide navigation state that is accurate enough to seed this inertial propagation and bring the lander to within 1 m of the selected safe landing site.

There are boulder fields on the Moon, but the more prevalent hazards are small craters and steep slopes. The best lunar terrain maps have been generated from Lunar Reconnaissance Orbiter (LRO) data. LRO is in a polar orbit and has provided a polar

DEM with a ground resolution of 25 m. Near the equator, stereo imaging from the LRO Narrow Angle Camera must be used to generate DEMs of sufficient resolution for TRN.

The hazard detection data collection and processing must be completed in 10 s to leave time for safe site selection by the pilot. The HDS uses a hybrid processing approach to meet this requirement. An FPGA collects and times the data from the sensors. The FPGA then passes the data to a multi-core general purpose processor which runs all of the algorithms for DEM generation, HD and HRN [32]. The multicore processor allows parallelization of time consuming processes and is straight forward to program.

2.2.5 System Comparison

Figure 2.4 shows a side-by-side comparison the LVS and ALHAT systems. The LVS has less challenging requirements because of the daytime landing, small size of the robotic lander and the significant hazard tolerance of the lander. These enabled a system that can perform TRN with the mature computer vision algorithms and sensors derived from the Descent Image Motion Estimation System on Mars Exploration Rover [25] and HD with a single flash LIDAR image, which mitigates the need for a gimbal or stitching of flash LIDAR images. Since the LVS is being developed for the

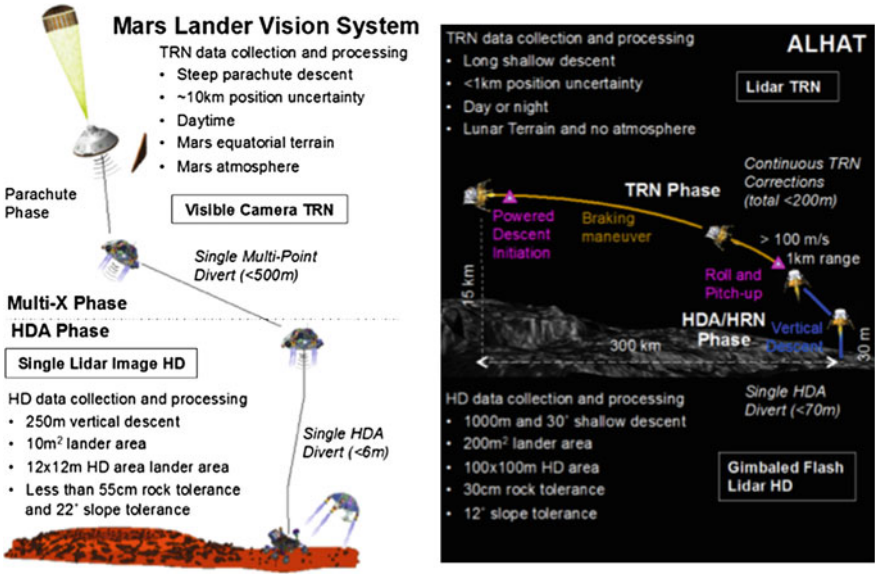

Fig. 2.4 A comparison of TRN and HD systems for Mars robotic missions and Lunar crewed missions

next Mars lander (2016–2022), the design focused on using components that could be flight qualified in the short term. A major early design choice to maintain the short duration path to flight was the selection of an FPGA based processor architecture over one utilizing multi-core processing.

The requirements on the ALHAT system are more challenging than what is needed for robotic landing and the infusion period is farther in the future (2020–2030). Consequently, the ALHAT development has focused more on meeting performance requirements and less on detailed path to flight designs. The ALHAT system must detect hazards from far away, with high resolution over a large area to safely land the 'not very hazard tolerant' crewed lunar lander. This drove the design to a gimbaled flash LIDAR and the associated increase in complexity, mass and power. The 'any lighting condition' requirement drives the TRN approach to use the less mature LIDAR sensors and contour matching algorithm. The multi-core processing approach is better from flexibility and ease of programming stand point, but its path to flight requires flight qualification of rad-hard processors.

Both the LVS and ALHAT developments are needed to meet short and long term goals of NASA. Moreover, TRN and HD remain fertile technology development areas for applications beyond planetary landing. Proximity operations around comets and near earth asteroids will use very similar algorithms and sensors. Satellite servicing and orbital debris mitigation could probably use similar algorithms and sensors. On the Earth, autonomous helicopters are being developed for cargo delivery and ship board landing, that will rely on TRN and HD functions to deal with unknown and unprepared landing sites. For all of these application domains, TRN and HD can be expanded to include additional sensing modalities like high frequency radar, thermal imaging and multi-return scanning LIDAR.

2.3 Phase Synchronization Control of Spacecraft Swarms

The objective of this section is to present an effective method for automatically generating evolving network topologies that determine how control information flows among the agents, thereby reducing the complexity of controlling a large number of spacecraft in the swarm. Directed graphs are used instead of undirected graphs to account for heterogeneous sensing or communication capabilities of spacecraft in the swarm network.

We review the recent results from our prior work [7, 9–11, 13, 27] in this section. The proposed framework of adaptive graphs is useful especially when we deal with a large number of spacecraft that perform arbitrary reconfiguration maneuvers. Another benefit of the proposed method is that the required gain for stabilization is smaller than the gain of an uncoupled control law by employing an adaptive graph Laplacian. Also, the error bound of the proposed synchronization control law is shown to be smaller than that of an uncoupled tracking control law. This justifies the use of a synchronization framework that can help to maintain a desired shape, even if individual spacecraft are shifted from their desired locations, as illustrated in

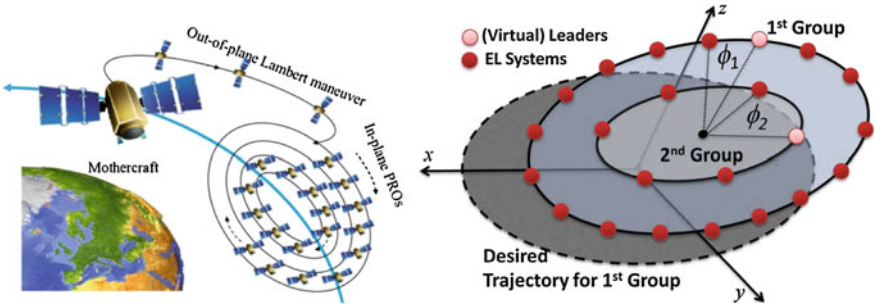

Fig. 2.5 Swarm deployment to relative elliptical orbits called Passive Relative Orbits (PROs) (*left*); Phase synchronization control of multiple spacecraft following elliptical orbits in the LVLH frame (*right*)

Fig. 2.5. The high-fidelity nonlinear dynamic model of swarm spacecraft motions, that include the effects of both Earth's oblateness and air drag in LEO [27], is used to derive the nonlinear stability proof of robust synchronization of coupled Euler-Lagrange equations, first derived in [9, 13].

2.3.1 Problem Statement—Controlling the Phase Differences in Periodic Orbits

Consider multiple spacecraft following some relative elliptical orbits as shown in Fig. 2.5. We use the term *relative*, since the elliptical motions are generated in the local-vertical local-horizontal (LVLH) frame attached to the chief orbital motion. Hence, it should not be confused with an elliptical orbit around the Earth. The relative orbital motions ($q_{tr,j} \in \mathbb{R}^3, 1 \leq j \leq p$) in the LVLH frame, of each (possibly heterogeneous) spacecraft comprising the swarm network, is governed by

$$m_j \mathbf{I}_3 \ddot{\mathbf{q}}_{tr,j} + \mathbf{C}_{tr,j}(\mathfrak{oe}(t))\dot{\mathbf{q}}_{tr,j} + \mathbf{G}_{tr,j}(\mathbf{q}_{tr,j}, \mathfrak{oe}(t)) + \mathbf{D}_{tr,j}(\mathbf{q}_{tr,j}, \dot{\mathbf{q}}_{tr,j}, \mathfrak{oe}(t)) = \tau_{tr,j} \tag{2.1}$$

where m_j is the mass of each spacecraft and the nonlinear terms, including the graviation orces with J_2 effects and the dissipative forces due to air drag, are given in [27]. In particular, the vector of six orbital parameters, $\mathfrak{oe}(t)$, which defines the origin of the LVLH frame, as shown in Fig. 2.5, is governed by $\dot{\mathfrak{oe}} = \mathbf{f}_{\mathfrak{oe}}(\mathfrak{oe})$ [27]. Furthermore, the attitude dynamics of each spacecraft can be represented by the EL form as

$$\mathbf{M}_{rot,j}(\mathbf{q}_{rot,j})\ddot{\mathbf{q}}_{rot,j} + \mathbf{C}_{rot,j}(\mathbf{q}_{rot,j}, \dot{\mathbf{q}}_{rot,j})\dot{\mathbf{q}}_{rot,j} + \mathbf{G}_{rot,j}(\mathbf{q}_{rot,j}, \mathfrak{oe}(t)) = \tau_{rot,j} \tag{2.2}$$

where the attitude vector $\mathbf{q}_{rot,j}$ can be represented by the first three values of quaternions or the modified Rodrigues parameters [9]. note that we can combine

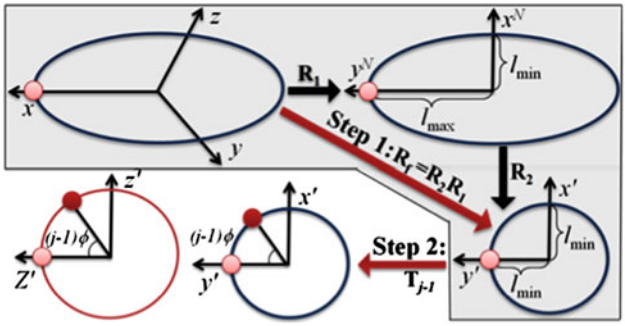

Fig. 2.6 Transformation of elliptical orbit in 3D to two *circles* for phase rotation

the dynamics of (2.1) and (2.2) to derive the dynamics of $\mathbf{q}_j = (\mathbf{q}_{rot,j}^{\mathrm{T}}, \mathbf{q}_{tr,j}^{\mathrm{T}})^{\mathrm{T}}$. Since $\mathbf{C}_{tr,j}(\text{œ}(t))$ is skew-symmetric, $\dot{\mathbf{M}}_{rot,j}(\mathbf{q}_{rot,j}) - 2\mathbf{C}_{rot,j}(\mathbf{q}_{rot,j}, \dot{\mathbf{q}}_{rot,j})$ is also skew-symmetric. This property is essential for the stability proof used in this section.

Fuel efficient, J_2-invariant elliptical orbits in the LVLH frame can be written as $\mathbf{q}_{d,tr}(t) = (x_e \sin(nt + \psi_{e_0}), y_e \cos(nt + \psi_{e_0}), z_e \sin(nt + \psi_{z_0}))^{\mathrm{T}}$ and they are used as the desired elliptical trajectory [27].

In this section, the complexity of controlling multiple spacecraft is reduced by setting a common phase angle for each desired elliptical orbit (ϕ_1 and ϕ_2, as shown in Fig. 2.5). This common phase difference in each ellipse also sets some safe collision-free distance between each pair of spacecraft. For spacecraft swarm applications, such as sparse aperture arrays, it is more important to maintain a formation shape,by ensuring constant phase differences between the spacecraft, than exactly following a desired elliptical trajectory. Such a phase control of oscillators, called engineered central pattern generators, has been successfully applied to control multi-joint locomotive systems [12, 13, 26]. Hence we define a common phase angle in an elliptical orbit in 3D. It turns out that $\mathbf{q}_{d,tr}(t)$ can be transformed into $\mathbf{q}'_{d,tr}(t)$in the new x'-y'-z' frame, comprised of a circular motion in the x'-y' plane and a sinusoidal function in the z' axis (see (Fig. 2.6) and [10] for details).

We can now perform a phase rotation by applying a rotation matrix $\mathbf{T}_{j-1} = \mathbf{I}_{n-3} \oplus \mathbf{T}((j-1)\phi) \oplus \mathbf{R}((j-1)\phi)$ for both the x'-y' frame and the z' and Z', where Z' is introduced to perform this phase rotation on the z' coordinate [10]. For the attitude dynamics of $\mathbf{q}_{rot,j}$, we do not apply phase synchronization, although it is also a straightforward extension (see [9]).

The control objective is to drive the tracking control error for each j, $\mathbf{v}''_j = \mathbf{T}_{j-1}^{\mathrm{T}}\mathbf{s}''_j = \begin{bmatrix}\mathbf{v}'_j \; v'_{jZ'}\end{bmatrix} = \mathbf{T}_{j-1}^{\mathrm{T}}\dot{\mathbf{q}}''_j - \dot{\mathbf{q}}''_d + \Lambda''(\mathbf{T}_{j-1}^{\mathrm{T}}\mathbf{q}''_j - \mathbf{q}''_d)$ or $\mathbf{v}'_j$ exponentially to zero. In other words, in the presence of modeling errors, we should show $\lim_{t\to\infty} \left\|\mathbf{v}'_j(t)\right\| \leq \Delta_T$. The phase synchronization control should yield a smaller synchronization error than the trackin gerror, such that for each connected pair j and k, $\lim_{t\to\infty} \left\|\mathbf{v}'_j(t) - \mathbf{v}'_k(t)\right\| \leq \Delta_S < \Delta_T$.

A conventional consensus controller without tracking control, results in undesirable drifting of the synchronized states. Then, the synchronization control ensures a smaller synchronization error that helps to maintain a formation shape as shown in Fig. 2.5. In contrast with prior work, this section uses an adaptive control scheme to automatically compute a time-varying network topology. In other words, the adaptive Laplacian matrix determines not only which neighbors each member should communicate with, but also the actual values of the time-varying gains.

2.3.2 Phase Synchronization Control Law with Adaptive Graphs

The matrices and functions in (2.1) and (2.2) are converted to the new x'-y'-z' frame by the matrix $\mathbf{R}_f$ that defines $\mathbf{q}'_j = \mathbf{R}_f \mathbf{q}_j$ and $\mathbf{q}'_d = \mathbf{R}_f \mathbf{q}_d$. Then, the dynamics in the new frame and the controllers are given as

$$M''_j(\mathbf{q}_j)\ddot{\mathbf{q}}''_j + C''_j(\mathbf{q}_j, \dot{\mathbf{q}}_j)\dot{\mathbf{q}}''_j + G''_j(\mathbf{q}_j) + D''_j(\mathbf{q}_j, \dot{\mathbf{q}}_j) = \begin{bmatrix} \mathbf{R}_f \tau_j(t) \\ \tau_{jZ'} \end{bmatrix} \tag{2.3}$$

$$\begin{aligned} \begin{bmatrix} \mathbf{R}_f \tau_j(t) \\ \tau_{jZ'} \end{bmatrix} &= M''_j(\mathbf{q}_j)\ddot{\mathbf{q}}''_{j,r} + C''_j(\mathbf{q}_j, \dot{\mathbf{q}}_j)\dot{\mathbf{q}}''_{j,r} + G''_j(\mathbf{q}_j) \\ &\quad + D''_j(\mathbf{q}_j, \dot{\mathbf{q}}_j) - (k_1 + c_{jj}(t))\mathbf{s}''_j - \mathbf{T}_{j-1} W''_j(\{\mathbf{v}''\}, \tilde{\mathbf{n}}_j)\mathbf{c}_j \end{aligned} \tag{2.4}$$

where $W''_j(\{\mathbf{v}''\}, \tilde{\mathbf{n}}_j) = \left[-\tilde{n}_{j1}\mathbf{v}'_1 \cdots -\tilde{n}_{j(j-1)}\mathbf{v}'_{j-1} \; -\tilde{n}_{j(j+1)}\mathbf{v}'_{j+1} \cdots -\tilde{n}_{jp}\mathbf{v}'_p\right]$ and $c_{jj}(t) = \sum_{k=1, k\neq j}^{p} \tilde{n}_{jk}(t)\left|c_{jk}(t)\right|$. Also, $\mathbf{c}_j = \left[c_{j1}\; c_{j2} \cdots c_{j(j-1)}\; c_{j(j+1)} \cdots c_{jp}\right]^{\mathrm{T}}$ is adapted by $\dot{\mathbf{c}}_j = \sum_j \mathrm{Proj}\left(\mathbf{c}_j, W_j^{\mathrm{T}}(\{\mathbf{v}'\}, \tilde{\mathbf{n}}_j)\mathbf{v}'_j\right) - \sum_j \mathbf{S}_{\mathbf{c}_j}(\mathbf{c}_j)\mathbf{c}_j$.

The nonnegative function $\tilde{n}_j = \left[\tilde{n}_{j1} \cdots \tilde{n}_{jp}\right]^{\mathrm{T}}$ sets the adaptation rate based on the relative distance with its neighbors and the synchronization errors. Note that $\tilde{n}_{jp}$ is a nonzero scalar only if the relative distance is within the maximum communication or sensing distance ($d_{jk} \leq d_{\mathrm{limit},j}$): for $j \neq k$,

$$\tilde{n}_{jp}\left(\left\|\mathbf{v}'_j - \mathbf{v}'_k\right\|, d_{jk}\right) = \frac{\tanh\left(\alpha_j\left\|\mathbf{v}'_j - \mathbf{v}'_k\right\|\right) + \tilde{n}_0}{1 + \tilde{n}_0} \frac{1 - \tanh\left(\beta_j(d_{jk}^2 - r_{c,j}^2)\right)}{1 + \tanh(\beta_j r_{c,j}^2)} \tag{2.5}$$

Furthermore, each positive element of the diagonal matrix $\mathbf{S}_{\mathbf{c}_j}$ switches to zero if the distance is outside the communication/sensing boundary or the corresponding c_{jk} exceeds the maximum allowable value. This means that outside the communication boundary, the coupling gain exponentially tends to zero.

The closed-loop systems from (2.3) and (2.4) are coupled through a diffusive term in each controller, whose coupling gains are computed by an adaptive control law. Then, the information flow in the network is epitomized by the adaptive graph Laplacian matrix $[\mathbf{L}(t)]_a$ defined $[\mathbf{L}(t)]_a = diag(\begin{bmatrix} c_{11} & \cdots & c_{pp} \end{bmatrix}) \otimes \mathbf{I}_n + [\mathbf{c}(t)]$ where

$$[\mathbf{c}(t)] = \begin{bmatrix} 0 & -\tilde{n}_{12}(t)c_{12}(t) & \cdots & -\tilde{n}_{1p}(t)c_{1p}(t) \\ -\tilde{n}_{21}(t)c_{21}(t) & 0 & \cdots & -\tilde{n}_{2p}(t)c_{2p}(t) \\ \vdots & \vdots & \ddots & \vdots \\ -\tilde{n}_{p1}(t)c_{p1}(t) & -\tilde{n}_{p2}(t)c_{p2}(t) & \cdots & 0 \end{bmatrix} \otimes \mathbf{I}_n \tag{2.6}$$

Note that many elements of $[\mathbf{c}(t)]$ are zero, since many pairs of the agents have no directed communication link. Hence, $[\mathbf{L}(t)]_a$ of a large network will inevitably be a sparse matrix.

2.3.3 Main Stability Theorems and Simulation Results

Theorem 2.1 ([10]) *The network EL systems from (2.3) and (2.4) globally exponentially converge to their desired trajectories with bounded errors such that the distance* $R_2(t) = \int_{\mathbf{q}'_d}^{(\mathbf{T}^{\mathrm{T}}_{j-1}\mathbf{q}''_j)'} \|\delta z\|$ *between each* $(\mathbf{T}^{\mathrm{T}}_{j-1}\mathbf{q}''_j)' = [\mathbf{I}_n \ \mathbf{0}_{n\times 1}]\mathbf{T}^{\mathrm{T}}_{j-1}\mathbf{q}''_j$ *and the desired trajectory* $\mathbf{q}'_d(t)$ *exponentially tends to the ball of radius* $R_2(t) \leq \frac{\lambda_{\max}([\mathbf{H}(\mathbf{q})])}{\lambda'' k_0 \lambda_{\min}([\mathbf{H}(\mathbf{q})])} (\|\{\Delta_d\}\|)$ *with a contraction rate of* $k_0/\lambda_{\max}([\mathbf{H}(\mathbf{q})])$ *for* $\mathbf{s}''_j$ *and* λ'' *for* $\mathbf{q}''_j$*, if each diagonal element* ℓ_k *of* $\mathbf{S}_{\mathbf{c}_j}$ *satisfies* $\ell_k > k_0$ *for the design parameter* $k_0 > 0$ *chosen, and if*

$$k_1(t) \geq k_0 + \min\left(\deg_o \tilde{n}_m c_{\max}/2, \sqrt{2m_n(m_n + \deg_o)}\tilde{n}_m c_{\max}\right) \tag{2.7}$$

Note that $c_{\max}$ denotes the known boundary value of c_{ij}, used for the adaptive control law, and $\deg_o \leq p-1$ the maximum out-degree of each member (vertex). The out-degree for each vertex defines how many other agents are receiving information from that member. Also, m_n is the maximum in-degree of each system, that is, the maximum number of nonzero elements in each row of $[\mathbf{c}(t)]$.

The results of Theorem 2.1 can also be applied when the adaptive graph Laplacian $[\mathbf{L}(t)]_a$ is augmenting a certain (fixed) baseline digraph constructed by a graph Laplacian $[L(k_1, k_2)]$ in the closed-loop system. For this purpose, the main control law (2.4) can be modified as

$$\begin{aligned} \begin{bmatrix} \mathbf{R}_f \tau_j(t) \\ \tau_{jZ'} \end{bmatrix} =& M''_j(\mathbf{q}_j)\ddot{\mathbf{q}}''_{j,r} + C''_j(\mathbf{q}_j, \dot{\mathbf{q}}_j)\dot{\mathbf{q}}''_{j,r} + G''_j(\mathbf{q}_j) \\ &+ D''_j(\mathbf{q}_j, \dot{\mathbf{q}}_j) - (k_1 + c_{jj}(t))\mathbf{s}''_j + \sum_{k\in N_j} k_2 \mathbf{s}''_k - \mathbf{T}_{j-1} W''_j \mathbf{c}_j \end{aligned} \tag{2.8}$$

where $k_1 > 0, k_2 > 0$. The fixed baseline topology is determined by the set of (incoming) neighbors N_j.

Theorem 2.2 *Adaptive coupling gain augmentation. For the closed-loop system obtained by (2.3) and (2.8), Theorem 2.1 holds by replacing k_1 in (2.7) by $\lambda_{\min}([L(k_1, k_2)]_{\text{sym}})$ by using Weyl's theorem [18].*

We now show that the proposed control law guarantees both faster convergence and smaller error bounds of synchronization between the coupled variables than those of tracking control.

Theorem 2.3 ([10]) *Faster synchronization. A balanced graph with the symmetric Laplacian matrix $[\mathbf{L}(t)]_b$ can be constructed for each $[\mathbf{L}(t)]_a$ as*

$$[\mathbf{L}(t)]_b = \left([\mathbf{L}(t)]_a + [\mathbf{L}(t)]_a^{\mathrm{T}}\right)/2 - \text{diag}\left([\mathbf{L}(t)]_a^{\mathrm{T}}[1, 1, \ldots, 1]^{\mathrm{T}}/2)\right)$$

If Theorem 2.1 holds, there exists a subset $\left(\mathbf{V}_{ss}^{\mathrm{T}}\{\mathbf{v}'_{tr}\} = \mathbf{0}\right)$ of the original synchronization manifold, $\left(\mathbf{V}_s^{\mathrm{T}}\{\mathbf{v}'_{tr}\} = \mathbf{0}\right)$, with $\mathbf{V}_{ss}$ being a subset of orthonormal eigenvectors $(\mathbf{V}_s)$ of $[\mathbf{L}(t)]_b$ such that

1. *synchronization occurs faster than tracking:*

$$\lambda_s = \frac{k_1 + \lambda_{\min}(\mathbf{V}_s^{\mathrm{T}}[\mathbf{L}]_{a,\text{sym}}\mathbf{V}_s)}{\lambda_{\max}([\mathbf{1}]^{\mathrm{T}}[M'][\mathbf{1}])} > \lambda_t = \frac{k_1}{\lambda_{\max}(\mathbf{V}_s^{\mathrm{T}}[M']\mathbf{V}_s)} \tag{2.9}$$

2. *the synchronization error is smaller than the tracking error if the disturbance field is more co-directional (i.e., $\left\|[\mathbf{1}]^{\mathrm{T}}\mathbf{d}(t)\right\| > \left\|\mathbf{V}_{ss}^{\mathrm{T}}\mathbf{d}(t)\right\|$):*

$$\int_{\mathbf{0}}^{\mathbf{V}_s^{\mathrm{T}}\{\mathbf{v}'_{tr}\}} \|\delta\mathbf{y}_s\| \geq \frac{\lambda_{\max}(\mathbf{V}_s^{\mathrm{T}}[M']\mathbf{V}_s)\left\|\mathbf{V}_s^{\mathrm{T}}\mathbf{d}(t)\right\|}{\lambda_{\min}(\mathbf{V}_s^{\mathrm{T}}[M']\mathbf{V}_s)(k_1 + \lambda_{\min}(\mathbf{V}_s^{\mathrm{T}}[\mathbf{L}]_{a,\text{sym}}\mathbf{V}_s))} \tag{2.10}$$

Hence, the control objective in Sect. 2.3.1 is met despite disturbances and modeling errors. Again, the benefit of Theorem 2.3 is that a formation shape on relative elliptical orbits can be maintained more precisely than tracking some desired positions. Moreover, the adaptive graphs of communication or relative sensing connections are automatically determined by synchronization errors and relative distances. Figure 2.7 shows a result of simulation of reconfiguring 275 spacecraft moving in the LVLH relative frame by using (2.1) and the proposed adaptive phase-synchronization control (2.4). Also, the figure shows the changes in the network topology during the reconfiguration maneuver, due to changes in the adaptive graph Laplacian matrix. These results demonstrate that spacecraft can automatically determine in a distributed manner, an evolving digraph topology of a large network of spacecraft swarms based on the synchronization errors and relative distances. The adaptive graph Laplician matrix represents a time-varying digraph that considers the heterogeneous capabilities of communication or relative sensing among the spacecraft members in the

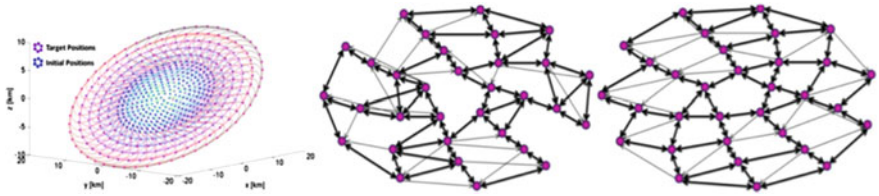

Fig. 2.7 Reconfiguration of 275 spacecraft using the proposed control law (*left*); Change of the network topology when we reconfigure 32 spacecraft (Images taken from [10])

swarm network. The objective of phase synchronization control is to maintain a desired phase difference on a periodic orbit. This implies that the complexity of controlling such a large number of spacecraft moving in relative elliptical orbits reduces to setting a proper phase difference. Interestingly, this method of phase synchronization is conceptually similar to phase synchronization of CPG oscillators that generate phase synchronized rhythms in a self-sustained fashion on spinal cords of vertebrates [12, 30].

2.4 Application of Probabilistic Guidance to Swarms of Spacecraft Operating in Earth Orbit

2.4.1 Introduction

This section reviews the theory behind a probabilistic guidance approach to guiding an arbitrary swarm of agents [1], and reviews its application to coordinating a spacecraft swarm operating in Earth orbit [2]. The main idea is to drive the swarm to a desired density distribution in a prescribed region of the configuration space. Rather than control individual spacecraft, the probabilistic guidance approach controls the average number of spacecraft per unit volume, ensuring that spatial averages converge to the desired targeted density distribution (see Fig. 2.8 for an example scenario). The targeted density distribution is achieved as an emergent behavior of the swarm, and is associated mathematically with achieving a statistical steady-state condition, i.e. a fixed point of a spatial Markov process.

In its simplest form, the probabilistic guidance approach is completely decentralized and does not require communication or collaboration between spacecraft. Specifically, spacecraft make statistically independent probabilistic decisions based solely on their own state, which ultimately guides the swarm toward the desired targeted density distribution. In addition to being completely decentralized, the probabilistic guidance approach has a novel autonomous self-repair property: Once the desired swarm density distribution is attained, the spacecraft automatically repair damage to the swarm distribution without collaborating and without explicit knowl-

Fig. 2.8 In this example scenario, thousands of spacecraft are deployed in Earth orbit. First they match their periods to ensure that they do not quickly drift apart. Then they configure themselves in order to achieve a desired prescribed pattern. The motion of this pattern is determined by the Earth orbital dynamics

edge that damage has occurred. First, the probabilistic guidance method is reviewed for swarms operating in deep space. Then an adaptation of the probabilistic guidance concept relevant to Earth orbiting swarms is reviewed, where orbital dynamics are explicitly considered based on Hill's equations. An illustrative example is given showing theoretical swarm convergence and emergent behaviors.

2.4.2 Probabilistic Guidance Problem

This section describes the swarm distribution guidance problem. The physical domain over which the swarm agents are distributed is denoted as R. It is assumed that region R is partitioned as the union of m disjoint sub-regions, which are referred to as bins:

$$R_i,\ i = 1, \ldots, m, \text{ such that } R = \bigcup_{i=1}^{m} R_i$$

Let an agent have position $r(t)$ at time index t. Let $x(t)$ be a vector of probabilities such that the sum of its entries is one and the i'th element $x[i](t)$ is the probability of the event that this agent will be in the i'th bin R_i at time t,

$$x[i](t) := \text{Prob}(r(t) \in R_i) \tag{2.11}$$

The time index t will also be referred to as the "stage" in the remainder of the section. Consider a swarm comprised of N agents. Each agent is assumed to act independently

of the other agents, so that (2.11) holds for N separate events,

$$x[i](t) := \text{Prob}(r_k(t) \in R_i),\ k = 1, \ldots, N$$

where $r_k(t)$ denotes the position of the k'th agent at time index t, and the probabilities of these N events are jointly statistically independent. We refer to $x(t)$ as the swarm distribution. This is to be distinguished from the ensemble of agent positions $\{r_k(t)\}_{k=1:N}$ which, by the law of large numbers, has a distribution that approaches $x(t)$ as the number of agents N is increased.

The distribution guidance problem is defined as follows: Given any initial distribution $x(0)$, it is desired to guide the agents toward a specified steady-state distribution described by a probability vector v,

$$\lim_{t\to\infty} x[i](t) = v[i] \text{ for } i = 1, \ldots, m$$

$$\text{where } v[i] \geq 0, \qquad \sum_{i=1}^{m} v[i] = 1$$

subject to motion constraints specified in terms of an adjacency matrix A_a as follows:

$$A_a^{\mathrm{T}}[i, j] = 0 \Rightarrow r(t+1) \notin R_i \text{ when } r(t) \in R_j, \forall t.$$

Here, the adjacency matrix A_a of the edges of a directed graph specifies the allowable transitions between bins.

The desired distribution v has the following interpretation: We have m bins in the physical space corresponding to where agents can be located, and the element $v[i]$ is the desired probability of finding an agent in the i'th bin. If there are N agents, then $Nv[i]$ describes the expected number of agents to be found in the i'th bin. Let $n = [n[1], \ldots, n[m]]^{\mathrm{T}}$ denote the actual number of agents in each bin. Then the number of agents $n[i]$ found in the i'th bin will generally be different from $Nv[i]$, although it follows from the independent and identically distributed (iid) agent realizations that $v = \mathbb{E}[n]/N$, and from the law of large numbers that $n/N \longrightarrow v$ as N becomes large. Hence the vector v is a discrete probability distribution specifying the desired average fraction of agents in each bin of the physical domain, which, in practice, will only be approximated by the histogram n/N of agents. However, the nature of the approximation is that v is equal to the mean $\mathbb{E}[n/N]$ of the agent histogram, and by the law of large numbers, $n/N \longrightarrow v$ as N becomes large.

2.4.3 Probabilistic Guidance Algorithm (PGA)

The key idea of the probabilistic guidance law is to synthesize a column stochastic matrix [6, 18] M, which we call Markov matrix for PGA, with$v$ as its eigenvector

corresponding to its largest eigenvalue 1 [16, 18], that is, M must satisfy

$$M \geq 0, \qquad \mathbf{1}^{\mathrm{T}} M = \mathbf{1}^{\mathrm{T}}$$

where $\mathbf{1}$ is a vector of ones. The entries of matrix M are defined as transition probabilities. Specifically, for any time instance,

$$M[i, j] = \mathrm{Prob}(r(t+1) \in R_i | r(t) \in R_j), \ i, j = 1, \ldots m$$

i.e., an agent in bin j transitions to bin i between two consecutive stages with probability $M[i, j]$. The matrix M determines the time evolution of the probability vector x as

$$x(t+1) = Mx(t), \ t = 0, 1, 2, \ldots \tag{2.12}$$

Note that the probability vector $x(t)$ stays normalized in the sense that the sum of its entries is one for all time instances. The probabilistic guidance problem becomes one of designing a specific Markov process (2.12) for x that converges to a desired distribution v.

It is desired for $x(t)$ to asymptotically converge to v, i.e., for v to be a globally attractive stationary distribution for M. The main result of this section shows that asymptotic convergence to v is ensured by imposing an additional constraint on the design of matrix M, denoted as the spectral radius condition,

$$\rho(M - v\mathbf{1}^{\mathrm{T}}) < 1 \tag{2.13}$$

The synthesis of the Markov matrix for PGA can be achieved by using a variety of methods. Methods to synthesize PGA using convex optimization and the "Metropolis-Hastings Algorithm" have been developed in [1]. Once the Markov matrix is synthesized, the following PGA can be used to implement it by providing a copy of the matrix M to each of the agents, and then having each agent propagate its position as an independent realization of the Markov chain as follows:

Probabilistic Guidance Algorithm (PGA)

1. Each agent determines its current bin, e.g., $r_k(t) \in R_i$.
2. Each agent generates a random number z uniformly distributed in [0, 1].
3. Each agent goes to bin j, i.e., $r_k(t+1) \in R_j$, if $\sum_{l=1}^{j-1} M[l, i] \leq z \leq \sum_{l=1}^{j} M[l, i]$.

The first step determines the agent's current bin number. The last two steps sample from the discrete distribution defined by the column of M corresponding to the agent's current bin number.

The following theorem (see [1, 2] for a proof) gives a necessary and sufficient condition for asymptotic convergence of x to v for the generic PGA.

Theorem 2.4 *Consider the PGA below with column stochastic matrix M such that $Mv = v$. Then for any at probability vector $x(0)$, it follows that $x(t) \longrightarrow v$ as $t \to \infty$ for the system in (2.11) if and only if Eq. (2.13) is satisfied.*

Theorem 2.4 is important because it indicates that a probabilistic guidance law for a swarm is asymptotically convergent if and only if the spectral radius condition (2.13) is satisfied. This condition has been interpreted in the context of Perron-Frobenius theory and can be used as a basis for applying modern control theory to synthesizing asymptotically convergent swarm guidance laws in [2].

2.4.4 Adaptation of PGA to Earth Orbiting Swarms

We consider spacecraft swarms in circular orbits around Earth. The dynamics of each spacecraft relative to the circular orbit are given by Hill's equations [31], also referred to as the Clohessy Wiltshire equations,

$$\begin{aligned}\ddot{x} &= 2\omega\dot{y} + 3\omega^2 x + f_x \\ \ddot{y} &= -2\omega\dot{x} + f_y \\ \ddot{z} &= -\omega^2 z + f_z\end{aligned}$$

Here x, y and z are the spacecraft's local-vertical local-horizontal (LVLH) cartesian coordinates associated with an orbital frame, which is defined at a point on the orbit, and oriented such that the x-axis points in the zenith direction, the z-axis is normal to the orbital plane, and the y-axis completes the right-hand system; f_x, f_y and f_z are the specific forces applied to each axis; and ω is the orbital frequency. From the analytic solution to the Hill's equations it can be shown that the swarm remains bounded if all spacecraft initial states satisfy the following condition (in the absence of any other external forces than the central gravitational field),

$$-6\omega x(0) - 3\dot{y}(0) = 0$$

In this case, all the spacecraft are period matched. From this point on we will only consider spacecraft swarms that are period matched. Furthermore, all spacecraft are assumed to be in-plane, i.e., $z(t) = 0$ for all t. This latter constraint is not necessary but imposed to keep the discussion contained according to the space limitations. The complete generalization of all subsequent results to out-of-plane motion is given in [2]. We now define useful notion of a swarm that is Orbit Type Matched (i.e., an OTM swarm).

Definition 2.1 A swarm is referred to as a planar OTM swarm if it satisfies the following conditions: For all $i = 1, \ldots, N$,

1. Period Matched with parameter ω, $\dot{y}_i = -2\omega x_i(0)$

2. Centroid Matched with parameter y_o, $y_{o,i} := y_i(0) - 2\dot{x}_i(0)/\omega = y_o$
3. $z_i(t) = 0$ for all t

A new set of position coordinates is introduced and denoted as Scaled Rotated (SR) coordinates that are instrumental in defining the motion of a planar OTM swarm:

$$\begin{bmatrix} \bar{x}(0) \\ \bar{y}(0) \end{bmatrix} = R^{\mathrm{T}}(\omega t) S^{-1} \begin{bmatrix} x(t) \\ y(t) - y_o \end{bmatrix} \text{ where } R(\omega t) = \begin{bmatrix} \cos(\omega t) & \sin(\omega t) \\ -\sin(\omega t) & \cos(\omega t) \end{bmatrix}, \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

Note that the coordinate transformation is time dependent while the SR coordinates of any spacecraft in a planar OTM swarm are time-invariant. In a planar OTM swarm, there are only two degrees of freedom that determine the motion of each spacecraft. Out SR coordinates naturally capture this fact. Next we can describe the PGA adaptation for in-plane OTM swarms, where each agent follows the following steps:

- Step 1: At t, map current position $(x(t), y(t))$ in LVLH to SR coordinates $r(t) = \left[\bar{x}(0), \bar{y}(0)\right]_t^{\mathrm{T}}$ as,
$$r(t) = R^{\mathrm{T}}(\omega t) S^{-1} \begin{bmatrix} x(t) \\ y(t) - y_o \end{bmatrix}$$
- Step 2: Determine current region R_j s.t. $r(t) \in R_j$, and propagate the Markov Chain one step to calculate the next desired state $r(t+1) \in R_i$ using,
$$M[i, j](t) = \mathrm{Prob}(r(t+1) \in R_i | r(t) \in R_j)$$
- Step 3: Map $r(t+1)$ back to LVLH according to,
$$\begin{bmatrix} x(t+1) \\ y(t+1) - y_o \end{bmatrix} = SR(\omega(t+1)) r(t+1)$$
- Step 4: Complete the desired state in LVLH by OTM,
$$\begin{aligned} z(t+1) &= 0 \\ \dot{x}(t+1) &= \frac{\omega}{2}(y(t+1) - y_o) \\ \dot{y}(t+1) &= -2\omega x(t+1) \end{aligned}$$
- Step 5: Command agent to new state $[x, y, z, \dot{x}, \dot{y}, \dot{z}](t+1)$.
- Step 6: $t \leftarrow t+1$, and go back to Step 1.

The idea behind this sequence of steps is to have each agent move according to a Markov chain in SR coordinates. Physically, this corresponds to each agent moving from one Hill's trajectory to another in the plane of motion. Since the statistics of the swarm propagate as the Markov chain, the swarm will converge asymptotically to the desired distribution v in SR coordinates, regardless of the initial distribution. The converged asymptotic distribution in LVLH, in turn, will be a rotated and stretched version of v, corresponding to the time-varying mapping from SR coordinates to

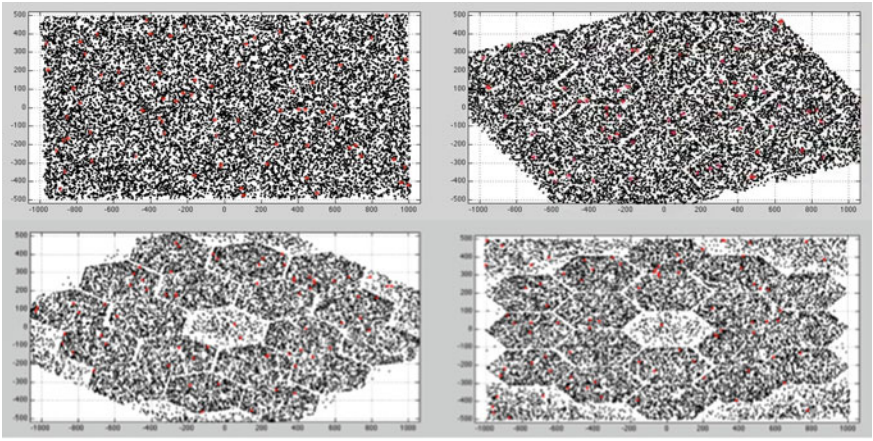

Fig. 2.9 An in-plane OTM swarm of 16,000 spacecraft in a near-circular Earth orbit starts as a uniform distribution and evolves over time into a representative (18 *hexagon*) large aperture pattern

LVLH coordinates. Within this parameterization, PGA can guide the swarm to attain a wide variety of shapes and distributions useful for a diverse range of potential applications. A simulation is performed that demonstrates 16,000 spacecraft in a near-circular Earth orbit. The spacecraft start at a random initial distribution and converge out to a prescribed swarm distribution. Figure 2.9 shows four instances of the swarm density evolution. A concluding remark is given in Sect. 2.6.

2.5 Nonlinear State Estimation And Sensor Optimization Problems for Detection of Space Collision Events

The objective of the collision event detection and tracking is to prevent damaging collisions of currently active LEO satellites with other space objects. This includes efficient detection and tracking of changes in trajectories of Resident Space Objects (RSOs) that might cause the collisions. The collision concept considered here is presented in Fig. 2.10. An important challenge in achieving the collision avoidance objective is the possibility of a short warning time, that is, the time between the change to a collision trajectory and the collision itself may be as short as a few minutes. In this case, we have very stringent requirements on timing of collision RSO's detection and estimation of its new dynamic state values that would allow possible effective avoidance actions.

An effective solution to the collision and avoidance problem is to design and deploy constellations of LEO satellites equipped with EO/IR sensors that can: (1) detect potential changes in existing orbital trajectories that may lead to damaging collisions; and (2) localize, track, and assess trajectories headed towards collisions.

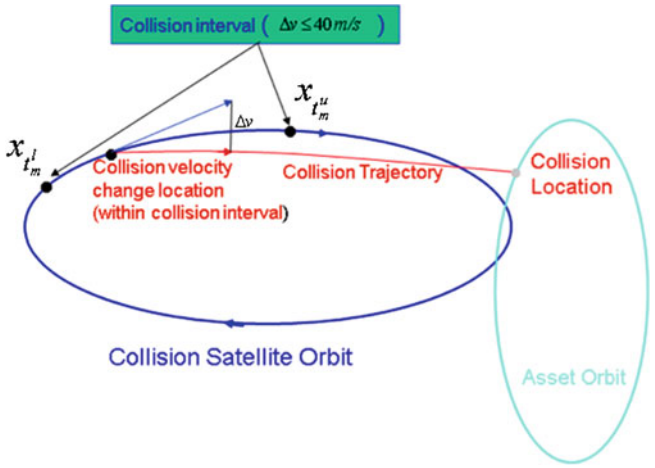

Fig. 2.10 Collision event concept: An event is a collision interval $[t_m^l, t_m^u]$ during which a RSO can change from its current orbit (*dark blue*) to a collision trajectory (*red*) that leads to a collision location on the asset orbit (*cyan*). The collision interval is determined from a limit on the collision satellite's change in velocity (Δv m/s)

We have developed necessary models, simulations, and methods for deriving, evaluating, and comparing such constellations and also derived optimal constellation designs that satisfy the stated objectives for given collision scenarios. In the following sections, we give short summary of our results. In Sect. 2.5.1, we summarize the design of collision event testbed and candidate LEO sensor constellation designs. Satellite collision modeling and algorithms for tracking, collision detection, and sensor management are presented in Sect. 2.5.2.

2.5.1 LEO Sensor Constellation Design and Collision Event Testbed

The scenario that we consider assumes that there are three asset satellites that can possibly collide with 23 other satellites during the day of April 1st of 2006. The three satellites are AQUA, PARASOL, and AURA with the Two Line Elements (TLEs) given from the NORAD catalog (http://www.space-track.org). We derive LEO trajectories by using "standard" Simplified General Perturbations-4 (SGP4) propagator [17] that may include higher fidelity orbital perturbations caused by environmental factors such as gravity, atmosphere, and solar pressure and also satellite physical properties such as ballistic coefficient. In a similar way,

EO/IR sensor constellation design: We considered a large number of constellation designs for sensor orbits in order to obtain the best coverage of collision events with minimal number of sensors [34, 35]. We considered sensor constellation designs

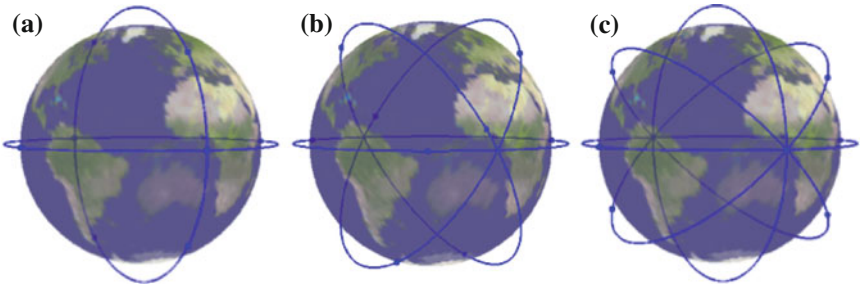

Fig. 2.11 Three EO/IR constellations for collisions event comparisons: design in Fig. 2.11a is shown to be optimal for detecting collisions. **a** $N_p = 2, N_{\frac{s}{p}} = 4$. **b** $N_p = 3, N_{\frac{s}{p}} = 2$. **c** $N_p = 3, N_{\frac{s}{p}} = 3$

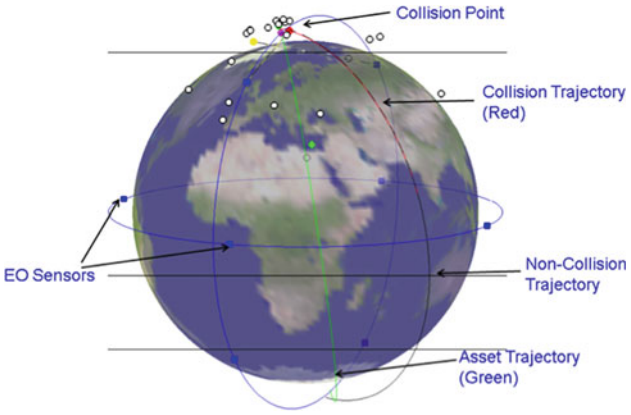

Fig. 2.12 Testbed snapshot showing collision satellite OPS 0203 colliding with satellite PARASOL at 11:26:06 (Hms) with collision trajectory length 21.4 min and collision interval 28.42 min

using symmetry requirement, number of orbital planes N_p, and number of satellites per orbital plane $N_{\frac{s}{p}}$. An example of candidate constellations designs considered in [35] is shown in Fig. 2.11. By considering Space Based Visible (SBV) observations, our analysis reported in [14] demonstrated that it is enough to have four observations of the collision satellite locations after the end of the event interval to determine if the collision related to that event is going to happen or not. This result was derived by assuming that the time between observations is $\Delta t = 50$ s. Testbed simulation snapshot for one of collision events is shown in Fig. 2.12.

2.5.2 Satellite Collision Modeling and Estimation

From a given TLE set, we derive six Kepler parameters (orbital elements) $\xi_t = (\alpha_t, e_t, i_t, \Omega_t, \omega_t, M_t)$ that slowly change with time $t \geq 0$, where the *semi-major axis* a_t and the *eccentricity* e_t, describe the form of the orbit, the mean anomaly M_t defines the position along the orbit, and the three other elements that include the *right ascension of the ascending node* Ω_t, the *inclination* i_t, and the *argument of perigee* ω_t, define the orientation of the orbit in space [26]. The derived state variable is $\xi_t = \left[\dot{\xi}_t, \xi_t\right]$ where the first six coordinates are derivatives of the Kepler parameters $\bar{\xi}_t$. More sophisticated stochastic modeling of the satellite states were derived and applied in [36] as extension of simplified state model in [35]. The measurement model of azimuth angle $\alpha_t^{i,j}$ and elevation angle $\beta_t^{i,j}$ looking from j'th sensor platform toward i'th collision satellite at time $t \geq 0$, for $\gamma_t^{i,j} = \left[\alpha_t^{i,j}, \beta_t^{i,j}\right]$, is given by

$$\tilde{\gamma}_t^{i,j} = \gamma_t^{i,j} + \sigma_a \nu_t^{i,j}, t \geq 0,$$

where the error standard deviation is $\sigma_a = 1$ arcsec, and $\nu_t^{i,j} \tilde{N}(0, I_2)$ are normalized Gaussian variables.

A measurement is collected by a sensor only if the following three conditions are satisfied: (a) the sensor points its CCD array toward the predicted location of the target; (b) the true location of the target is within the sensor field of view (FoV); and (c) the target has "star background" looking from the sensor. Condition (c) is necessary for accurate sensor attitude determination.

For each event interval, we collect observations to determine if the collision satellite has changed its trajectory to any one of the candidate collision trajectories. Since for realistic bound on change in velocity (usually denoted by and called "delta v"), the collision can happen only on a small part of the asset's orbit and therefore, as an realistic approximation, we choose one point on the asset orbit as a location of the collision that then corresponds to a unique intercept time. Then for any given time inside the event interval, by using the Generalized Parametrized Battin (GPB) method presented in [35] and derived from [5], we can determine a unique collision trajectory that is parametrized by ξ (e.g. Kepler elements). By discretizing the event interval at every second, we have a finite number of possible collision trajectories that we denote by ξ_l^q and its corresponding hypothesis by H_l^q, where superscript q denotes event (event index) and subscript l is an index of hypothesis for each event. As time progresses (in our case at each second), we are creating hypotheses corresponding to candidate collision trajectories. The zero hypothesis ($l = 0$) corresponds to the non-collision trajectory. Each hypothesized trajectory is initialized with an appropriate uncertainty matrix to account for time discretization approximation of the event interval. Also, each hypothesis is tracked by applying a given tracking algorithm, e.g. see a version described in Sect. 2.5.2.2. As observations are collected, we calculate posterior probability of each hypothesis by applying the collision detection algorithm described in Sect. 2.5.2.2. The hypotheses with very small

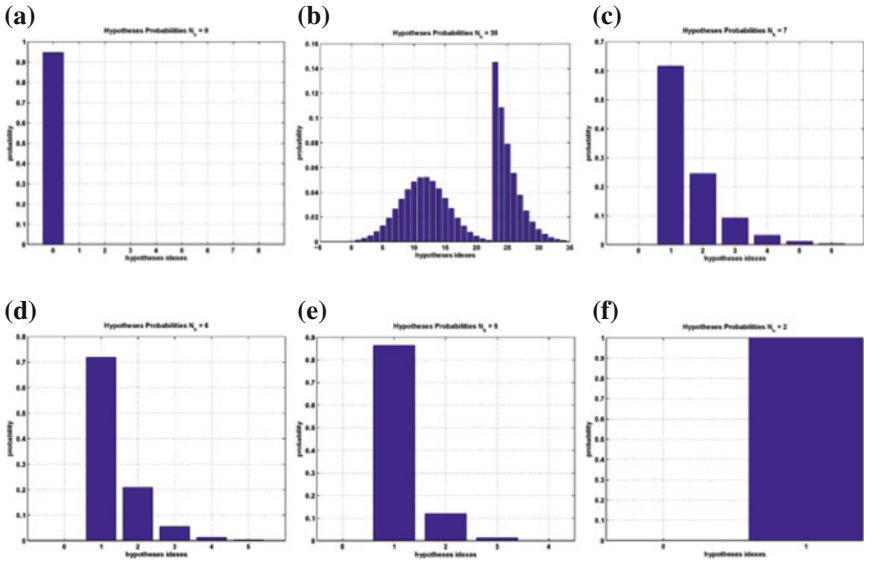

Fig. 2.13 Hypotheses posteriors at each scan (50 s between scans) after the collision satellite moved from its non-collision trajectory (hypothesis 0) to a collision trajectory (hypothesis 1): (**a**) after $t = 8$ s, collision probability is 0.055, i.e. collision threat is not detected; (**b**) after $t = 58$ s, collision probability is one, i.e. collision threat is detected; (**c**) after $t = 108$ s, collision trajectory is singled out (probability ≥ 0.6); (**d**) after $t = 158$ s, number of hypotheses decreases and true collision trajectory probability is ≥ 0.7; (**e**) after $t = 208$ s, true collision trajectory probability is > 0.8; (**f**) after $t = 258$ s, true trajectory probability is one and all other hypotheses disappeared, i.e. collision trajectory is uniquely determined

probabilities are discarded and ones with higher probabilities are kept till they are fully resolved as shown in Fig. 2.13. The sensors management algorithm is discussed in Sect. 2.5.2.3.

2.5.2.1 Tracking Algorithm

A simplified version of state perturbation model and its corresponding filter are derived and demonstrated in [35]. The more advanced stochastic model of state perturbations, i.e. stochastic target state model, with nonlinear particle filter based tracking and estimation algorithm is derived and demonstrated in [36].

Tracking algorithms developed and demonstrated in [14, 33–36] consists of the following five steps:

Step 1: Initialization
Step 2: Derivation of prediction state estimate
Step 3: Estimation of optimal sensor parameters (Sensor Management and Tracking)
Step 4: Collection of measurements
Step 5: Updating of state estimates

Nonlinear state and measurement equations are used in posterior propagation and update using Particle Filtering algorithms involving propagation of particle states and update of particle weights, respectively. The selected tracking results are summarized in Fig. 2.15.

2.5.2.2 Collision Detection Algorithm

The collision detection algorithm is based on multi-hypothesis testing, using likelihood derivation and initialization [14] as follows:

1. At time step k, we create $N_k^{h,q}$ hypotheses for each current event q.
2. For event q at time step k, we have hypothesis H_l^q that is associated with element set $\xi_k^{q,l}$.
3. From the measurement model, we derive hypotheses likelihoods:

$$\log p(t, \tilde{\gamma}_t^{i,j} | H_l^q) \tilde{-} \frac{1}{2\sigma_a^2} \| \tilde{\gamma}_t^{i,j} - \gamma_t^j \left((\xi_t^{q,l}) \right) \|^2$$

   where $\gamma_t^j \left(\xi_t^{q,l} \right)$ is an azimuth-elevation pair associated with hypothesis H_l^q and $p(t, \cdot | H_l^q)$. conditional probability of azimuth-elevation pairs given hypothesis H_l^q at time $t \geq 0$
4. We derive hypotheses conditional posterior $p(t, H_l^q | \gamma_t)$ given all measurements Y_t up to the current time $t \geq 0$ by using Bayes' Theorem and then select a set of hypotheses that have posteriors above a given threshold.

Selected results from [14] are shown in Fig. 2.13 demonstrating the effectiveness of the collision detection algorithm. Assuming 50 s between any two consecutive observations (SBV scans), plots (a) and (b) in Fig. 2.13 demonstrate that it takes two measurements to get that probability of non-maneuver to be zero (hypothesis indexed by zero in Fig. 2.13). The rest of the plots (c)–(f) of Fig. 2.13 demonstrate that it takes extra two to four measurements (four to six from the change of the trajectory) to uniquely determine the collision trajectory (hypothesis index one).

2.5.2.3 Sensor Management Algorithm

The details of sensor management algorithms and its effectiveness for LEO collision detection and tracking using a SBV sensor platform are described in [14]. The algorithm is extended to disparate and dispersed sensors (radars and SBV platforms) and also its effectiveness is demonstrated for continuous tracking of LEO satellites in [33]. Our sensor management algorithm is based on maximization of the Posterior Expected Number of Targets of Interest (PENTI) objective function $f_k(\cdot)$ that is information-theoretic representation of the expected number of well localized

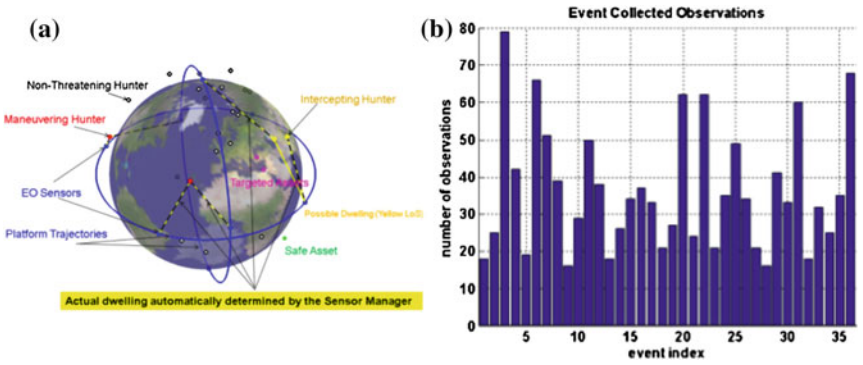

Fig. 2.14 Results of sensor management for collision detection and tracking: (**a**) simulation snapshot example of a typical LEO complex environment; and (**b**) number of observations per event

targets that are of tactical interest [26]. The sensor management is reduced to finding pairs $(j_1, l_1), \ldots, (j_n, l_n)$ chosen from available sensors $(j_1, \ldots, j_n)$ and targets $(l_1, \ldots, l_n)$ which are a subset of all collision satellites of interest.

The sensor management problem then consists in determining the pointing angles $(\gamma_{k,j_1}, \ldots, \gamma_{k,j_n})$ that are approximate solution of the following optimization problem:

$$(\gamma_{k,j_1}, \ldots, \gamma_{k,j_n}) = \underset{\gamma_{j_1},\ldots,\gamma_{j_n}}{\arg\max}\ f_k(\gamma_{j_1}, \ldots, \gamma_{j_n}).$$

The sensor management simulation results are shown in plots (a) and (b) of Fig. 2.14. Plot (a) of Fig. 2.14, is a snapshot of LEO complex environment that consists of EO/IR sensor platforms (blue squares), assets that are not currently in danger of collisions (green diamonds), assets that are in danger of being intercepted (magenta diamonds), collision satellites that are in event interval (red circles), collision satellites that are not currently associated with any event (white circles), and collision satellites that might be on collision trajectories (yellow circles). Possible observation collections and actual observations are represented by yellow and dash black lines. In plot (b) of Fig. 2.14, we show the total number of collected measurements for each event.

We implemented and demonstrated multisensor-multitarget sensor management and multi-hypothesis based collision detection and tracking on the SSCI's LEO environment testbed [35]. The simulation results for sensor constellation in plot (a) of Fig. 2.11, are shown in plots (a)–(c) of Fig. 2.15. Note that the standard deviations in plot (a) of Fig. 2.11 are very low (shades of blue) for all collision satellites during collision intervals which indicates good observability of the collision events and therefore excellent sensor management. Oscillatory property of location and velocity errors indicates an accurate modeling of state dynamics [35]. All collision events are tracked with high accuracy and on a timely basis for effective collision avoidance. A conclusion is given in Sect. 2.6.

2.6 Conclusion

This chapter discusses new guidance and control technologies for planetary landing and guidance, navigation, estimation, and control for swarms of spacecraft in low Earth orbit. In Sect. 2.2, we compared the Lander Vision System (LVS) and Autonomous Landing and Hazard Avoidance Technology (ALHAT) systems for planetary landing applications. The LVS has less challenging requirements because of the day-time landing, small size of the robotic lander, and the significant hazard tolerance of the lander. These enabled a system that could perform TRN with the mature computer vision algorithms and other sensor measurements such as a single flash LIDAR image. On the other hand, the requirements on the ALHAT system are more challenging. The ALHAT system must detect hazards from far away at high resolution over a large area to safely land the crewed lunar lander. This resulted in the design of a gimbaled flash LIDAR and the associated increase in complexity, mass, and power. In Sect. 2.3, the new formation control and phase synchronization strategy for swarms of spacecraft was discussed, in which orbital and attitude motions could be modeled as coupled Langrangian systems moving in elliptical periodic orbits. The adaptive control strategy of automatically computing evolving network topologies eliminated the need for defining a fixed communication or relative sensing topology on a digraph for synchronization stability. Such an evolving communication network gave rise to the adaptive graph Laplacian matrix. The error bound of the proposed synchronization control law with an adaptive graph Laplacian was shown to be smaller than that of an uncoupled tracking control law. This justified the use of a synchronization framework, for application where synchronization errors should be kept smaller than tracking errors, like stellar interferometry applications. In Sect. 2.4, the Probabilistic Guidance Algorithm (PGA) was applied to the problem of guiding swarms of spacecraft, operating under dynamic constraints imposed by being in Earth orbit. The main simplifying assumption was that all agents have nearly circular orbits and they obey Hill's linearized equations of motion. A simulation example showed the basic feasibility of the method. Due to space limitation, the discussion was restricted to in-plane motion only, but references have been provided that generalized all results to the out-of-plane case. Section 2.5 demonstrated that it is possible

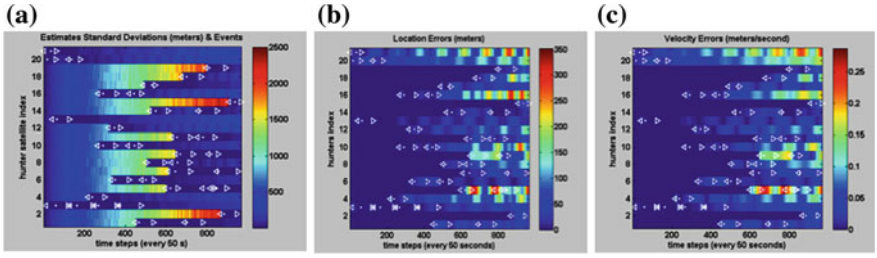

Fig. 2.15 Collision events (*left triangles* and *dots* represent event intervals and *right triangle* represents approximate intercept time) tracking and detection results: (**a**) estimation standard deviations; (**b**) location errors; and (**c**) velocity errors for all satellites

to detect and avoid space collision events using a small number of space-based EO/IR sensors. We implemented realistic scenarios and models for LEO collisions, developed appropriate metrics for the evaluation of different EO/IR sensor constellations, and evaluated the tracking and sensor management performance for different LEO EO/IR sensor constellations. The constellation that is designed on two orbital planes with four satellites on each offers the best compromise between the number of satellites and overall performance. The developed capabilities are expected to lead to significant improvement in Space Situational Awareness (SSA). The Space Based Visible sensors/constellations can be used for enhancing the capabilities of Space Surveillance Network (SSN). Future developments of an effective space surveillance system can utilize a swarm of spacecraft. The algorithms developed during our study will be applicable to those designs and lead to cost-effective implementations.

Acknowledgments The research was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA. The following people are thanked for their contributions: Saptarshi Bandyopadhyay and Daniel Morgan at UIUC; Milan Mandic at JPL; and Adel El-Fallah and Aleksandar Zatezalo at SSCI.

References

1. Acikmese, B., Bayard, D.: A markov chain approach to probabilistic swarm guidance. Am. Control Conf. (ACC) **2012**, 6300–6307 (2012). doi:10.1109/ACC.2012.6314729
2. Acikmese, B., Bayard, D.S.: Probabilistic guidance for swarms of autonomous agents. Technical Report D-73778, Jet Propulsion Laboratory, JPL (2012)
3. Adams, D., Criss, T., Shankar, U.: Passive optical terrain relative navigation using aplnav. In: Aerospace Conference, 2008 IEEE, pp. 1–9 (2008). doi:10.1109/AERO.2008.4526303
4. Amzajerdian, F., Pierrottet, D., Petway, L., Hines, G., Roback, V.: Lidar systems for precision navigation and safe landing on planetary bodies. In: Proceedings of the SPIE, vol. 8192, pp. 819, 202–819 ,202–207 (2011). doi:10.1117/12.904062. http://dx.doi.org/10.1117/12.904062
5. An Introduction to the Mathematics and Methods of Astrodynamics. AIAA education series. American Institute of Aeronautics and Astronautics, Reston (1999). http://opac.inria.fr/record=b1096182
6. Berman, A., Plemmons, R.J.: Nonnegative Matrices in the Mathematical Sciences. SIAM, Philadelphia (1994)
7. Chang, I., Chung, S.J., Blackmore, L.: Cooperative control with adaptive graph laplacians for spacecraft formation flying. In: Decision and Control (CDC), 2010 49th IEEE Conference on, pp. 4926–4933 (2010). doi:10.1109/CDC.2010.5717516
8. Cheng, Y., Clouse, D., Johnson, A., Owen, W., Vaughan, A.: Evaluation and improvement of passive optical terrain relative navigation algorithms for pin-point landing. In: AAS Space Flight Mechanics Meeting (AAS-SFM 2011) (2011)
9. Chung, S.J., Ahsun, U., jacques, E., Slotine, J.: Application of synchronization to formation flying spacecraft: Lagrangian approach. J. Guid. Control Dyn. **32**(2), 512–526 (2009)
10. Chung, S.J., Bandyopadhyay, S., Chang, I., Hadaegh, F.Y.: Phase synchronization control of complex networks of lagrangian systems on adaptive digraphs. Automatica **49**(5), 1148–1161 (2013). doi:http://dx.doi.org/10.1016/j.automatica.2013.01.048. http://www.sciencedirect.com/science/article/pii/S0005109813000496
11. Chung, S.J., Hadaegh, F.Y.: Swarms of femtosats for synthetic aperture applications. In: 4th International Conference on Spacecraft Formation Flying Missions and Technologies (SFFMT). St-Hubert, Quebec, Canada (2011)

12. Chung, S.J., Slotine, J.J.: On synchronization of coupled hopf-kuramoto oscillators with phase delays. In: Decision and Control (CDC), 2010 49th IEEE Conference on, pp. 3181–3187 (2010). doi:10.1109/CDC.2010.5717962
13. Chung, S.J., Slotine, J.J.E.: Cooperative robot control and concurrent synchronization of lagrangian systems. Trans. Rob. **25**(3), 686–700 (2009). doi:10.1109/TRO.2009.2014125. http://dx.doi.org/10.1109/TRO.2009.2014125
14. El-Fallah, A., Zatezalo, A., Mahler, R., Mehra, R.K., Pham, K.: Joint search and sensor management of space based eo/ir sensors for leo event estimation. In: Proceedings of the SPIE, vol. 7330, pp. 73,300N–73,300N–12 (2009). doi:10.1117/12.818292. http://dx.doi.org/10.1117/12.818292
15. Epp, C., Robertson, E., Brady, T.: Autonomous landing and hazard avoidance technology (alhat). In: Aerospace Conference, 2008 IEEE, pp. 1–7 (2008). doi:10.1109/AERO.2008.4526297
16. Fiedler, M.: Special Matrices and Their Applications in Numerical Mathematics. Dover, Mineola (2008)
17. Hoots, F.R., Schumacher, P.W., Glover, R.A.: History of analytical orbit modeling in the u. s. space surveillance system. J. Guid. Control Dyn. **27**(2), 174–185 (2004). doi:10.2514/1.9161. http://dx.doi.org/10.2514/1.9161
18. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, New York (1985)
19. Huertas, A., Cheng, Y., Madison, R.: Passive imaging based multi-cue hazard detection for spacecraft safe landing. In: Aerospace Conference, 2006 IEEE, pp. 14 (2006). doi:10.1109/AERO.2006.1655794
20. Johnson, A., Golombek, M.: Lander vision system for safe and precise entry descent and landing. In: LPI Workshop on Concepts and Approaches for Mars Landing (2012)
21. Johnson, A., Huertas, A., Werner, R., Montgomery, J.: Analysis of on-board hazard detection and avoidance for safe lunar landing. In: Aerospace Conference, 2008 IEEE, pp. 1–9 (2008)
22. Johnson, A., Ivanov, T.: Analysis and testing of a lidar-based approach to terrain relative navigation for precise lunar landing. In: AIAA Guidance Navigation and Control Conference (AIAA-GNC 2011) (2011)
23. Johnson, A., Keim, J., Ivanov, T.: Analysis of flash lidar field test data for safe lunar landing. In: Aerospace Conference, 2010 IEEE, pp. 1–11 (2010). doi:10.1109/AERO.2010.5447025
24. Johnson, A., Montgomery, J.: Overview of terrain relative navigation approaches for precise lunar landing. In: Aerospace Conference, 2008 IEEE, pp. 1–10 (2008). doi:10.1109/AERO.2008.4526302
25. Johnson, A., Willson, R., Cheng, Y., Goguen, J., Leger, C., Sanmartin, M., Matthies, L.: Design through operation of an image-based velocity estimation system for mars landing. Int. J. Comput. Vis. **74**(3), 319–341 (2007). doi:10.1007/s11263-006-0022-z. http://dx.doi.org/10.1007/s11263-006-0022-z
26. Montenbruck, O., Gill, E.: Satellite Orbits, Models, Methods, and Applications. Springer, New York (2005)
27. Morgan, D., Chung, S.J., Blackmore, L., Acikmese, B., Bayard, D., Hadaegh, F.Y.: Swarm-keeping strategies for spacecraft under J2 and atmospheric drag perturbations. J. Guid. Control Dyn. **35**(5), 1492–1506 (2012). doi:10.2514/1.55705. http://dx.doi.org/10.2514/1.55705
28. Mourikis, A., Trawny, N., Roumeliotis, S., Johnson, A., Ansar, A., Matthies, L.: Vision-aided inertial navigation for spacecraft entry, descent, and landing. IEEE Trans. Robot. **25**(2), 264–280 (2009). doi:10.1109/TRO.2009.2012342
29. Poberezhskiy, I., Johnson, A., Chang, D., Ek, E., Natzic, D., Spiers, G., Penniman, S., Short, B.: Flash lidar performance testing: configuration and results. In: SPIE Laser Radar Technology and Applications XVII (2012)
30. Seo, K., Chung, S.J., Slotine, J.J.: Cpg-based control of a turtle-like underwater vehicle. Auton. Robots **28**(3), 247–269 (2010). doi:10.1007/s10514-009-9169-0. http://dx.doi.org/10.1007/s10514-009-9169-0
31. Vallado, D., Clain, W.: Fundamentals of Astrodynamics and Applications. Collège custom series. McGraw-Hill, New York (1997). http://books.google.com/books?id=HuWMPwAACAAJ

32. Villalpando, C., Johnson, A., Some, R., Oberlin, J., Goldberg, S.: Investigation of the tilera processor for real time hazard detection and avoidance on the altair lunar lander. In: Aerospace Conference, 2010 IEEE, pp. 1–9 (2010). doi:10.1109/AERO.2010.5447023
33. Zatezalo, A., El-Fallah, A., Mahler, R., Mehra, R.K., Brown, J.: Dispersed and disparate sensor management for tracking low earth orbit satellites. In: Proceedings of the SPIE, vol. 7336, pp. 73,360I–73,360I–12 (2009). doi:10.1117/12.819299. http://dx.doi.org/10.1117/12.819299
34. Zatezalo, A., El-Fallah, A., Mahler, R., Mehra, R.K., Pham, K.: Optimal constellation design of low earth orbit (leo) eo/ir sensor platforms for space situational awareness. In: Proceedings of the SPIE, vol. 7330, pp. 73,300T–73,300T–11 (2009). doi:10.1117/12.818304. http://dx.doi.org/10.1117/12.818304
35. Zatezalo, A., El-Fallah, A., Mahler, R., Mehra, R.K., Pham, K.: Eo/ir satellite constellations for the early detection and tracking of collision events. In: Proceedings of the SPIE, vol. 7697, pp. 76,970L–76,970L–12 (2010). doi:10.1117/12.851217. http://dx.doi.org/10.1117/12.851217
36. Zatezalo, A., El-Fallah, A., Mahler, R., Mehra, R.K., Pham, K.D.: Multimodel filtering of partially observable space object trajectories. In: Proceedings of the SPIE, vol. 8050, pp. 80, 500K–80,500K–12 (2011). doi:10.1117/12.884609. http://dx.doi.org/10.1117/12.884609

Chapter 3
Aircraft Autonomy

Piero Miotto, Leena Singh, James D. Paduano, Andrew Clare, Mary L. Cummings and Lesley A. Weitz

3.1 Introduction

The word *automatic* is a compound of the Greek words *auto* (αὐτο-) which means "self", and *maō*, which means "to be ready or eager", hence the word automatic literally means "acting on one's will or self-acting." Automation is the discipline behind the design of automatic systems. Important strides have been made in the area of system automation. The word *autonomy*, on the other hand, is formed from compounding the word *auto* with another Greek word *nomos* (νόμος), which means "law", hence autonomy literally means "one who gives oneself its own law." Unlike automation, very little has been done in the area of developing manned and unmanned aerospace systems that are truly autonomous. The focus of this chapter is to explore

P. Miotto (✉) · L. Singh
Draper Laboratory, Cambridge, MA 02139-3563, USA
e-mail: pmiotto@draper.com

L. Singh
e-mail: lsingh@draper.com

J.D. Paduano
Aurora Flight Sciences, Cambridge, MA 02142-1050, USA
e-mail: jpaduano@aurora.aero

A. Clare
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: andrew.clare@mit.edu

M.L. Cummings
Duke University MEMS, Durham, USA
e-mail: m.cummings@duke.edu

L.A. Weitz
Mitre Corporation CAASD, New Jersey, USA
e-mail: lweitz@mitre.org

E. Feron (ed.), *Advances in Control System Technology for Aerospace Applications*, Lecture Notes in Control and Information Sciences 460, DOI 10.1007/978-3-662-47694-9_3

the progress in aircraft automation and the challenges that lay ahead in our path toward the development of truly autonomous aircraft systems.

Automation is increasing in every aspect of our society and in our daily lives. It is being entrusted with the management of complex operations in many wide-ranging sectors such as mass transit, medicine, finance, law enforcement, military, science, etc. In mass transit, we have seen the automation of intra-airport rail systems and even a few city scale subway systems. In finance, banking operations are conducted via Automatic Teller Machines (ATM) without human supervision. In the field of medicine, automation is used in procedures such as the laser eye surgery and the post- surgical care of patients. Unmanned Vehicles (UVs) have also been successfully employed in dangerous and remote environments, which has resulted in important scientific breakthroughs and discoveries. For example, underwater Unmanned Vehicles (UUVs) have explored the deepest trenches of the ocean [75] and NASA's unmanned rovers have traversed the surface of Mars [55]. Scientists have studied global warming by surveying the polar ice caps [68] with UAVs, while civilian agencies have employed Unmanned Aerial Vehicles (UAVs) for border patrol [39] and fighting forest fires [16].

The importance of automation is clearly manifested in the military sector. UAVs have enabled the military to run long endurance missions over hostile territory without placing a human pilot in harm's way. For this reason, they have become a key component of aerial surveillance missions. UAVs are also playing an increasingly vital role in ground attacks. Unmanned Ground Vehicles, for example, have been utilized by soldiers and civilian bomb squads to investigate and defuse explosive devices [56]. The military is now working to incrementally increase the employment of UAVs in cargo delivery, medical evacuation, and supply missions. They are set to operate alongside piloted aircraft in what the Air Force terms "same base, same time, same tempo" operations. Further, automatic landing systems on aircraft have demonstrated the ability to land jets on a moving ship deck in extremely low visibility and turbulent landing conditions with no pilot supervision. Automatic landing has proven so successful and practical that, today, the United States military trains twice as many ground operators for its unmanned aerial vehicles (UAVs) as pilots for its military jets. The military is working to further enhance the autonomous capabilities of their aircrafts. These enhancements include the abilities to perform unmanned mid-air refueling, takeoff and landing of full-scale helicopter on unmapped terrain, low-altitude nap-of-the-earth flight, and pilotless medical-evacuation demonstrations.

In the commercial sector, automation has proven instrumental in reducing the cockpit crew from five to two. Current plans include even further reduction in piloting operations. In the words of the President and CEO of Boeing Commercial Airlines, James Albaugh, when delivering the keynote address at the 2011 AIAA Modeling and Simulation Conference, a "pilotless airliner" is no longer a question of "if" but only a matter of "when". In the coming years, unmanned aircrafts are also expected to increase their footprint in civilian applications particularly in the "dull, dirty and dangerous" mission space. The goal is to employ UAVs to execute air-based storm damage surveillance including infrastructure surveys, refugee evacuation, and transportation of cargo.

3.1.1 Challenges to the Safe Integration of UAVs in the National Airspace

In order to achieve these goals, UAVs, whether fully autonomous or remotely piloted, will have to demonstrate in recognizable and provable terms, not only their ability to perform the most routine missions in civilian airspace but also their ability to operate alongside manned aircraft in full conformance with the rules and constraints of national civil aviation. As of 2015, the FAA has allowed only very restrictive conditions under which UAVs can operate during routine civilian operations. In order to meet the requirements to operate in civilian airspace, it is paramount that we enhance the technical capabilities of UAVs such that we can guarantee the safe integration of UAVs in the civilian and commercial sector. A safe and graceful integration will, as a result, reduce the public's fear about the usage of UAVs and enable wide acceptance among the passengers and partners alike.

The differences between Military UAVs and the civilian ones are stark. Military UAVs leverage owned-and-operated satellite systems supplemented by undersea cables to link their ground controllers (at times, half a world away from the aircraft) with low latency data and high-speed video transmission. Civilian UAVs, on the other hand, are forced to splice into the FCC-directed channels for all communication. Insufficient ground-communication bandwidth caused by the fine frequency allocations spread between mobile phones and other communication infrastructure fundamentally limits the ability to coordinate operations between all aircraft in a common airspace and does not allow a civilian aircraft to transmit its data to ground stations thus preventing human backup takeover of the flight controls.

Because of these severe limitations, as of today, UAVs lack: the ability to automatically detect and avoid nearby aircraft within visual range; a standard communication capability between the ground stations of remotely piloted aircraft and fully automated ones; the logic and software, both onboard and within the ground-based air traffic systems, that would allow autonomous air vehicles to share airspace with manned aircraft. These limitations prevent them from entering mainstream operations and ensuring the necessary safety error tolerance.

The current aviation infrastructure, since it was conceived and developed solely for human operators with multiple tiers of human-in-the-loop safety systems, presents an additional obstacle to the safe integration of UAVs in the civil sector. Modern air traffic management protocols are neither suitable nor reliable enough to ensure safe incorporation into autonomous flight monitoring and support. To simply mandate that any UAV operating within the National Air Space (NAS) be monitored by a pilot who can communicate with existing Air Traffic Control (ATC) systems and issue in-flight trajectory updates or changes, is not a practical solution. This approach would simply overwhelm the communication frameworks, the human infrastructural capability and increase the financial burdens of using UAVs which de facto obliterates the advantages of autonomy. However, the shared goal of the air traffic monitoring and control systems currently under development in the European Union (SESAR or Single

European Sky ATM Research) and the United States (NextGen ATM) include provisions to support both military and civilian, national and international aircraft in the same airspace.

3.1.2 Technical Enhancements for Safe Insertion of UAVs in the NAS

To achieve a safe integration of UAVs in the NAS, the military has identified the following enabling technologies: *sense-and avoid technologies* via communication-based response and collision-avoidance capabilities; *intent estimation* of the other aircraft; *integrity management*; *terminal area sensing* and *Verification and Validation* (V&V) of autonomous systems. Likewise, the FAA has instituted an unmanned aircraft program office to determine the *regulations that will accord UAVs permissible flight corridors*. In Europe, SESAR has been formed to completely overhaul the European airspace and its Air Traffic Management (ATM) system. For both sides of the Atlantic, the next generation of ATM system regulations will need to define capabilities and conventions for handling both manned and unmanned aircrafts.

The need to develop *sense-and-avoid* (SAA) technologies stems from an FAA requirement that expects the aircraft operator to be able to detect an aircraft within visual range, and *maneuver* to prevent collisions. Sense-and-avoid capabilities, therefore, will need to span autonomous aircraft detection, fusing measurements from multiple on-board sensors to detect and identify an "intruder" aircraft, take evasive action from the aircraft (according to regulation minimum separation standards), notify air-traffic control and then land safely. Remotely piloted aircrafts, too, will need to be fitted with some SAA capability since high-bandwidth telecommunication operation is not practical due to bandwidth constraints in the large and the size of the expected fleet. Although such implementations readily fall within present day automation capabilities, they will need to be universally and uniformly instituted.

Notwithstanding such implementations, a minimum threshold on guaranteed communications capability will still need to be instituted in all autonomous aircraft, civilian and military, to allow ground controllers to communicate with the aircraft in the terminal area, and ensure rapid response to controller directives i.e. to receive, acknowledge, execute and request confirmation on all ATC commands according to the same protocols and in the same timely fashion as the manned counterparts.

Intent estimation is a derived capability required by the FAA to support terminal area operations. It involves fusing measurements from all proximity sensors with models of typical airfield operations to estimate the status and possible intent of other aircraft, and to ensure proactive rather than reactive actions conformal with the actions of piloted aircraft.

Finally, once UAVs are included within the NAS, there will be a need for a reformulated ATC to handle a heterogeneous airspace as well as to scale to the numbers of aircraft that will need to be supported.

Under current day practices, under normal flight conditions, military UAVs and autonomous air systems operate with an on-board autopilot and a mission manager flying a preplanned trajectory or performing way-point guidance under the approval of a ground-stationed human operator. Despite the capability of UAVs to perform challenging ship-deck landing operations in uncertain lighting and turbulent weather conditions, a hand off to a human pilot is required under off-nominal conditions such as in rough weather or due to special airport operational requirements. Total autonomous flight control operations throughout the operating envelope and in the presence of systems failures will be necessary to expand the safe and reliable operational envelope of the autonomous pilot; this will necessitate further development of control, diagnostics, and mission planning systems [4, 14, 15, 36, 46, 60, 61, 71].

In the "Unmanned Systems Integrated Roadmap FY2011-2036", the United States Department of Defense has stated that top priorities for future research include increasing UV autonomy and developing advanced techniques for Manned-Unmanned teaming [29]. The road-map has outlined the following: the need for more advanced autonomy, the desire for this autonomy to be flexible and adaptable to dynamic and uncertain environments, the need for collaborative autonomy between multiple UVs, and the need for new Verification and Validation (V&V) approaches to certify increasingly complex autonomy [29, 53]. While the Department of Defense is enthusiastic about autonomy, the road-map also cautioned that "because artificial systems lack the human ability to step outside a problem and independently reevaluate a novel situation based on commander's intent, algorithms that are extremely proficient at finding optimal solutions for specific problems may fail, and fail badly, when faced with situations other than the ones for which they were programmed [29]".

The ensuing three sections of this chapter tackle some of the challenges outlined in this introduction. The first section titled "On-board air autonomy systems needs" discusses advancements in flight control system capabilities and flight-deck-automation that will equip UAVs with the on-board functionality required of all aircraft operating within the NAS. The next section titled "Human-Automation Collaboration" discusses the algorithms required for the development of an effective human-automation collaborative system for managing multiple UAVs in the airspace. The final section discusses issues in the modernization of the air traffic management system, and presents the challenges and the new technologies required to meet ever increasing air traffic demands. All three sections will talk about the key shortcomings in today's autonomous systems, and propose the most promising concepts to overcome these shortcomings. Each section will include a road map for the safe development, testing, and integration of these technologies into fielded systems.

3.2 On-Board Air Autonomy Systems Needs

Challenges for the autonomy and controls community have appeared in the form of Broad Agency Announcements (BAAs), discussions of next-generation warfare scenarios, and concepts that continue to resurface regarding desired capabilities. In the present section, we first attempt to introduce these challenges with some attempt at organization. Then, we translate these challenges into technical challenges or technical approaches that show promise. The final part describes a road-map for moving forward.

3.2.1 Challenges to Integration of UAVs in the NAS

The continued expansion of UAVs in commercial operations is the key challenge identified by Congress, users and providers alike for the next decade. For this to happen, it is universally agreed that UAVs must achieve operation at man -rated levels of safety. To be sure, this translates to different requirements for different types of UAVs. For Predator/Global Hawk class UAVs to operate from civilian airports and to follow flight paths similar to those of civil transports, and thus be fully integrated in the commercial sector, reliability levels will have to be higher than, for example, for a 2 pound foam UAV, which will not need triplex redundancy. A more interesting challenge regarding safety is posed by the 10 to 150 pound. range of vehicles. Although in this case the potential for injury and death is still very high, the cost of the vehicle would be prohibitive were it required to meet the standards for large vehicles. To reach appropriate safety standards across the board, key technical challenges previously mentioned in the introduction include: *V&V for autonomous system*, *Sense & Avoid solutions*, and *FAA integration strategies* and *ConOps*. Other technical challenges are listed below.

Proximity Operations:

This area of expansion in the employment of UAVs includes landing on ship decks or other moving vehicles, urban and indoor operation, autonomous refueling, and airborne launch and recovery. Many of these operations would place UAVs in environments where, to date, they have never operated: in proximity to other vehicles or structures. Sensing, estimation, disturbance rejection, and control take on very different requirements in this regime, and many challenges have yet to be met.

GPS denied operation:

This is related to proximity operations in the sense that ProxOps often cannot rely on GPS due to its lack of sufficient accuracy and the fact that relative position to obstacles is needed. Many of the sensing solutions are related to those that enable ProxOps, such as vision-based control, but others such as signals of opportunity, are unique.

Anti-Access Area Denial (A2AD):

This has been made a top priority by the Air Force. Autonomous systems are a potential solution to achieving the objective of area protection. Anti-Access refers to the ability to prevent large scale access to a region, whereas Area Denial describes the ability to prevent tactical operation in the immediate area of an enemy force. Large teams of UAVs with unique properties could provide a way to overcome enemy capabilities.

Protection against "Swarm" Attacks:

In this context, "Swarm" refers to a set of vehicles (manned or unmanned) that overwhelms the protective "pickets" set up around large vessels at sea. This is an asymmetric threat; were enough low-tech vehicles to attack, just a few breaking through could create a USS Cole-like incident.

3.2.2 Technical Enhancements for Improved In-Air autonomy—Key Technologies

This section translates the high-level programmatic challenges outlined in Sect. 3.2.1 into technical challenges to the autonomy community and puts them in the context of technologies to be pursued and advanced.

Vision and LIDAR based estimation and control:

Vision and LIDAR systems have made great strides in recent years and computational capabilities needed to put them into flight to enable Sense-and-Avoid capabilities are now available. Vision-based methods are divided roughly into optical flow or heuristic methods, feature-based methods, and perception- or semantic- based methods.[1] Optical flow methods are bio-inspired and provide fast, reactive capabilities for guidance through urban environments and obstacle detection and avoidance. Generally they are not quantitative estimation techniques; nevertheless the amazing optical flow -driven capabilities of human and insect eyes, for enabling safe, proximal operations and threat detection, make this an important area of research. Feature-based methods consider 'features' in the environment that are easily detected and tracked by a computer, but otherwise have no identity to the computer. A rich set of such features can be used for GPS-denied navigation, mapping/SLAM, and proximity operations. Semantics in vision refers to the idea of actually recognizing and classifying objects in the environment. Such ability can be used to identify, for instance, a landing site, improve localization accuracy by looking for known objects in the environment, or to determine whether a landing site has suitable rigidity to allow a heavy helicopter to land without sinking into mud, gravel, etc.

[1]This is the author's own taxonomy and terminology, a vision expert might divide it some other way.

As with many technology areas, the number of approaches and applications is so wide that little or no attempt has been made to organize them in a rational way. Such organization would be appropriate at this stage. Aurora maintains the SPHERES test-bed on the ISS to which a binocular camera system was recently added, as well as an embedded system suitable for MAVs, which performs vision processing. These platforms and associated vision-based ego- motion estimation and optical flow algorithms could be used as baselines to start the process of identifying best approaches to address various Proximity Operations challenges, strengthening those approaches, and making them available to the engineering community at large.

Adaptive Control:

Adaptive control has advanced by leaps and bounds in recent years, to the point that many believe it is 'ready for prime time', that is, ready to become an integral part of the next generation of autonomous vehicles. New vehicle configurations engineered to meet the challenges previously described [2] often do not live up to their potential because conventional control design methods require too much work to bring to bear. A case in point: a hobbyist can hover and transition a foamy aircraft in light gusts, but only a handful of autonomous systems have achieved this capability—and these have either been adaptive, have utilized motion capture, have required costly and time-consuming engineering, and have utilized simplifications that reduced utility. Adaptive control is making its way into industry, but funding is needed to further the transition from universities and NASA to industry. Reliability, and V&V challenges exist, and research is needed in this area. UAVs will benefit from these advancements towards enabling safe flight in the presence of failures and/or executing "safe abort" in the presence of critical failures.

Optimization-Based Control:

Optimization based trajectory generation and following, path planning, and multi-vehicle coordination have all been researched actively over the past 10 or 15 years. Methods are making their way into applications across the board and, for problems like A2AD and Swarms, there are a number of companies and institutions with strong capabilities to address these challenges. For path planning and multi- vehicle coordination, the remaining challenges appear to be logistics and system-integration. For instance, many of the techniques implicitly rely on the existence of an existing ad hoc network and, although many of these techniques are robust to brief outages or range-related drop-outs, they all require the existence of such a network. However the military has not settled on or fielded a standard for the type of radios required to enable network- in-the-sky communication between many vehicles in the air. The logistics of take-off and landing for a large team of UAVs also presents a challenge. This area would need a "killer app", agreed upon by all funding agencies and that does not leave space for alternatives. Swarm protection may prove to be such a killer

[2] Specifically, proximity operations, indoor operations, small vehicles that require severe gust rejection capability, and operations off of unequipped ship decks.

app, and the Navy has made progress in the area of team-based anti-submarine and mine countermeasures. Other ways may, however, exist and should be investigated.

As to the area of trajectory generation and following, which here refers to maneuvering along a non-equilibrium flight path to achieve a goal such as aircraft landing onto a pitching ship deck, maneuvering through an urban environment, or flare-to-land of a helicopter onto a highly sloped surface. Here, too, the tools seem up to the challenge but questions about reliability have still been left unanswered. Advanced methods such as falsification, or other efficient methods to replace Monte Carlo, may be appropriate for bringing these tools into applications.

Verification and Validation:

A recurring concern in the world of control systems is the high cost of turning concepts into practice, due to the expensive nature of software, integration, and testing costs [53]. By way of a number of short-cuts (motion capture, open-source software, one-time integration), Universities have been able to develop and fine-tune a number of significant capabilities that seem ready for transition. Few of these capabilities, however, at the present have found their way into actual air vehicles. The complexities of full-scale aircraft, the cost of hardware-in-the-loop simulation and validation, and the cost of code generation and validation in a flight critical setting continue to be severe obstacles and are perhaps the main obstacles for the realization of capabilities that are so close at hand.

Although the current modus operandi may prove sufficient in the case of the government being willing to pay the industry to implement an important capability, DARPA has prominently articulated the question if the development cycle really needs to be so expensive, manpower hungry, and slow-moving. Some private industries have attained higher levels of flexibility. Universities, too, have achieved important results in flight demonstrations. But, at present, it is still unclear how to lower the cost of the implementation of such capabilities on full-scale vehicles.

Self-Aware Vehicles:

Some of the challenges outlined earlier can be partially addressed by aircraft with proprioception capability, that is, the capability to sense their aerodynamic and/or structural state. New, distributed sensor in and on the wings of small UAVs are giving them gust rejection and maneuver capabilities that should be considered.

3.2.3 Conclusions: A Road-Map to Address the Technical Challenges

Any ideal road-map should have, as its goal, the creation of a codified base of capabilities, that is, a (hopefully diverse) set of companies, research labs, and universities able to perform the engineering tasks associated with a new capability on a regular basis. The development process of miniature flight controls components serves as an example of such an ideal road-map. Through funded research, small business

initiatives, and developments in the electronics industry, the integration of a small flight controller into a UAV is now a routine matter. This can be achieved in one of the following ways: buy a turnkey system from a small company, create an internal capability by hiring a few engineers with know-how and experience, or join an open-source community that maintains a system. Similarly, 6-DOF nonlinear simulation of conventional flight vehicles is a well-established capability both reliable and attainable at a reasonable cost. For these and a number of other control-related technologies, "people know how to do it".

The road-map to creation of such codified capabilities or shared industrial know-how is difficult to pre-specify. Often these capabilities evolve in an organic way, with little or no pre-planned road-map. For important technology efforts, the R&D community has the responsibility of a more rational approach, i.e. we need to have a clear vision of the capability we want to develop and of the specific steps that will lead us to attain such goal. Below are two examples of this model road-map.

Vision and LIDAR based estimation and control:

This is an area where a "capability goal" is relatively easy to define. At present, we lack the ability to systematically engineer vision-based approaches. Although attempts utilize simulations in 3D environments derived from video games, it is yet unclear whether these environments are applicable to the real world. No good tools or guidelines for hardware-in-the-loop or real-time simulation exist; this is especially true for LIDAR. Testing in realistic settings is often prohibitively difficult, except for indoor flight. Thus a road-map to an engineering base for vision- and LIDAR-based estimation and control is urgently needed.

To build such a road-map we first need to focus on the simulation issue and answer the following questions: how can we simulate vision, and how does one validate the simulation? Can we create a framework that is suitable for real-time or faster-than real time operation, how can we capitalize on the gaming industry and apply it tothe real world? The next step, which could be pursued in parallel, and might inform the first step, is the issue of sensor emulation: i.e., how can a simulation environment take on the properties of a LIDAR or a camera, as either a bridge to or a mechanism for hardware-in-the-loop testing? Finally, can a systematic methodology for testing against available video data, or testing under various canonical illumination, shadow, and other environmental conditions, be created?

There are a number of software tools for vision available and, at present, it is relatively routine to do blob and edge detection, SIFT and SURF, and run various other vision algorithms. What is still missing and urgently needed is to bring these tools into a systematic engineering framework of design, build, test, iterate, and fly.

Verification and Validation:

Another area where a clear end state can be formulated is Verification and Validation. Large aerospace prime contractors have the knowledge to perform V&V and have methodologies consistent with certification requirements; in other words, we have a process in place for certifying manned aircraft. No such process, however, exists for UAVs in the 10 to 150 pound class. While UAVs of this class are too heavy to

ignore from a safety perspective, they are too small and low cost to undergo the full manned aircraft approach. We need a road-map to the creation of a framework for engineering certified small UAS.

To develop such a road-map, we first need to define the requirements entailed in certifying small UAVs:

1. what, for example, are the ConOps for its operation;
2. what are the safety considerations;
3. how could confining the operations of UAS make special-purpose certification feasible and acceptable to the public.

Such a road-map, whenever given the FAA stamp of approval, would offer the framework that would enable the development of a vibrant industry. Informed by such a road-map, we could seek verification and validation methods that meet the specific needs of small UAS and are not hampered by current regulations. The goal of such a road-map would be to trade off reliability for cost of V&V for, if and when realistic set of ConOps and certification standards are put into place, the overall safety can be met with a different set of reliability standards. To this end, engineering tools can be adapted to meet the specific needs of the small UAS V&V problem. Finally, we need to develop processes leading to certification consistent with the ConOps and tool envisioned.

Similar road-maps should be developed for the other challenges outlined in Sect. 3.2.1 of this paper. We outline a feasible approach to road-map generation based on the experience of past successes, two of which have been discussed in some detail in the present paper. Previous experience leads us to believe that a successful road-map should have the following components:

1. *A clear vision of the end state being sought*. This vision should be to solidify or strengthen a capability within the industry, rather than a one-off demonstration or research thread with no clear conclusion.
2. *Buy-in from funding agencies and the technical community*. Obviously a long-term road-map requires funding, and this funding is predicated on agreement that the sought after goals are appropriate and the roles of the agencies involved are clear. A technical working group to guide the process is also valuable.
3. *Realism*. Goals and way points toward those goals should be clearly reachable. At present, we too often are able to implement in a one-off fashion, so we know what can be done; we just don't know how to do these things routinely.
4. *Modularization and redundancy*. One of the factors that enabled the success of the miniature flight avionics capability was the fact that many companies were trying to create these devices. Competition and a critical mass within the community are needed to create a 'sub-culture' of people and organizations that can do the engineering involved.

3.3 Human-Automation Collaboration

Although in the past decade UAVs have proven key to a variety of missions and tasks both in the military and in the commercial sector, significant obstacles still prevent their wider use. One problem is the large number of human operators required to operate them, a number that is often well in excess of that needed to operate a comparable manned vehicle [42]. This obstacle can, however, be overcome through an increase in the autonomous capabilities of UVs [24]. Many advanced UVs are able to execute basic movement and navigational tasks autonomously and can collaborate with other UVs to complete higher level tasks, such as surveying a designated area [2, 10]. The United States Department of Defense has for years envisioned inverting the operator -to-vehicle ratio such that a single operator controls multiple UAVs simultaneously [28]. This concept has now been extended to single operator control of multiple heterogeneous (air, sea, land) UVs [54].

This inversion of operators to vehicles will require the computational ability of optimization algorithms combined with the judgment and adaptability of a human supervisor. Identifying the characteristics that make an algorithm suitable for this kind of human-automation collaboration is essential for the development of an effective system. In order to begin to derive requirements for algorithms that can effectively collaborate with humans, a survey of academic and industry publications was conducted focusing on recently developed multiple UV scheduling algorithms [19]. The goal of the survey was to determine the types and frequency of scheduling algorithms that were currently in use in unmanned systems and to classify the characteristics and capabilities of these algorithms in terms of human-automation collaboration. We summarize the results of that survey below to describe the state of the art in multiple UV scheduling algorithms and to identify the important remaining challenges towards the development of an effective human-automation collaborative system for controlling multiple UVs.

3.3.1 Challenges in the Collaborative Human-Automation Scheduling Process

To effectively control multiple semi-autonomous UVs, there exists the need for a systematic method for scheduling tasks. For our purposes, scheduling is defined as creating a temporal plan that assigns tasks among a team of UVs and determines when the tasks are to be completed. While there is no explicit focus on path planning, it is worth noting that path planning is coupled with the scheduling problem, due to the need to estimate the amount of time needed for a UV to travel to a certain location to accomplish a task.

A wide variety of optimization algorithms have been developed to address the problem of scheduling tasks for multiple UVs. While varying in their method of formulating the scheduling problem and solving the optimization, all the approaches

utilize an autonomous scheduler with little to no human input during the development of the schedule. In the presence of unknown variables, possibly inaccurate information, and changing environments, these automated scheduling algorithms do not always perform well [64, 66].

Though fast and able to handle complex computation far better than humans, optimization algorithms are notoriously "brittle" in that they can only take into account those quantifiable variables, parameters, objectives, and constraints identified in the design stages that were deemed to be critical [67]. In a command and control situation such as one requiring the supervision of multiple UVs and where events are often unpredictable (e.g. weather changes, unexpected target movement), automated planners have difficulty accounting for and responding to unforeseen changes in the environment [40, 58]. Additionally, the designers of optimization algorithms often make a variety of assumptions when formulating the optimization problem, determining what information to take into account, or, in the case of receding horizon algorithms, deciding how far into the future to plan [8, 47].

One approach to deal with the "brittleness" of these algorithms is to have a human operator and an algorithm develop schedules collaboratively. A number of studies have shown that humans collaborating with algorithms can achieve higher performance than either the human or the algorithm alone under certain conditions [3, 25, 27, 45, 49, 59, 63]. While extensive research has been conducted to develop better algorithms for planning, comparatively little research has occurred on the methods by which human users utilize these tools, especially when working in dynamic, time-critical situations with high information uncertainty [50]. Additionally, operators can become confused when working with automation, unaware of how the "black box" algorithm has reached its solution or the assumptions made by the algorithm in modeling the problem. Moreover, generally humans play solely the role of decision-maker and only approve or disapprove schedules,without any consideration to the way a human could actually aid an algorithm in improving a solution. Figure 3.1 is a representation of the fact that there are often differences between the real world, the automation/engineer's model, and the human operator's model of the world. Designing an effective human-automation collaborative scheduling system

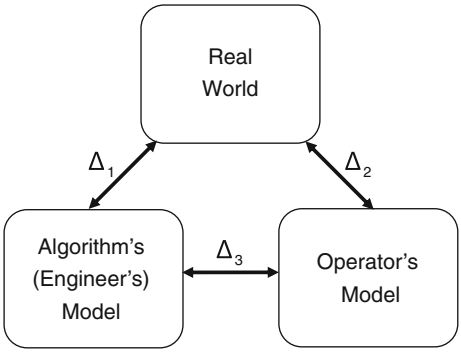

Fig. 3.1 Differences between real world, algorithm's model, and operator's model of the world

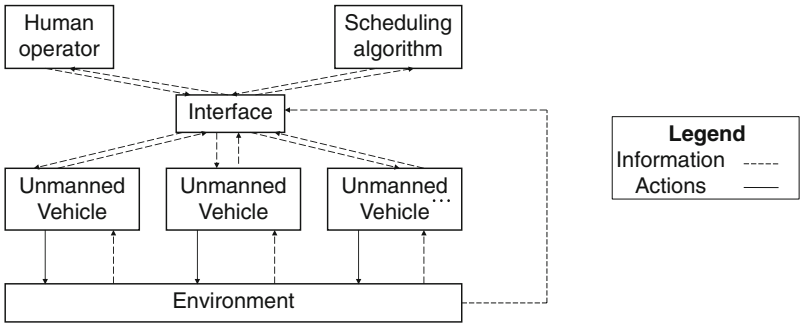

Fig. 3.2 Human-automaton collaborative scheduling system diagram

could provide the ability to supervise multiple UVs while addressing the inherent brittleness and opacity of algorithms. As shown in Fig. 3.2, the system could include the human operator, the graphical interface which displays information to the operator and allows the operator to interact with the system, the scheduling algorithm, and the semi-autonomous UVs which act in the environment, all with information flowing between components. It should be noted that the scheduling algorithm could exist as a stand-alone component, as pictured, or as sub-system of each UV, as in many decentralized systems [44, 76].

In the development of any complex system, the definition of requirements is a fundamental step in the systems engineering process [12]. While others have attempted to develop requirements for the graphical interfaces in human-automation collaboration [23, 37, 44, 67], few have attempted to develop requirements for the scheduling algorithms to be used in such systems.

3.3.2 Candidate Methods in Human-Automation Collaborative Scheduling

A survey of 117 publications within academia and industry on multiple UV scheduling algorithms was completed in order to determine the types and frequency of scheduling algorithms that are currently in use, and to classify the characteristics and capabilities of these algorithms [19]. The survey spanned the years 2006-2011, in order to emphasize what is currently in use or in development [19]. Papers included in the survey were chosen based on searching for resource allocation or scheduling algorithms specifically meant for assigning tasks to multiple UVs in real-time.

A brief summary of the results is presented here to identify the primary algorithms that are available for use in human-automation collaborative scheduling systems and highlight the areas where more research is necessary to enhance the capabilities of such systems.

We began by examining the methods chosen for solving the combinatorial optimization problem and the guarantees of optimality made by these methods. Among

the typical nonlinear optimization search methods, meta-heuristic methods and market-based auction algorithms were the most popular technique to solve a variety of scheduling problems. Also, Dynamic Vehicle Routing (DVR) methods using Voronoi partitions were identified as a potential solution method that could guarantee a certain level of performance without the need to re-plan constantly in a dynamic environment. DVR methods produce policies or decision rules, as opposed to specific task assignments, typically by optimizing the expected value of performance.

Non-deterministic scheduling algorithms, while potentially sub-optimal, have faster computational times and scale better to larger problems. Thus, meta-heuristic and auction-based algorithms can provide a good balance of performance and computational speed for larger missions. Branch and bound or other "iterative" approaches that monotonically improve the schedule with further iterations favor smaller numbers of vehicles and tasks. Thus, deterministic algorithms, which do not necessarily scale well to larger problems, are adequate for simpler missions. These algorithms are likely also easier to certify for systems that must operate with real vehicles.

In general, more centralized algorithms have been developed than decentralized ones, but interest has increased recently in decentralized algorithms for their potential ability to scale to larger problems, reduce communication bandwidth, and maintain robustness to single node failure [41]. Algorithm developers are also shifting their focus towards methods that have the capability to schedule tasks for heterogeneous UVs in order to support cooperation between air, land, and sea UV operations. Finally, there is growing interest in algorithms which can take into account uncertainty, especially algorithms designed for generating robust schedules and providing estimates of the certainty of the schedule. This class of algorithms will be essential for successful operations in dynamic environments with new tasks emerging, changing weather, and potential UV failures.

3.3.3 Technical Enhancements needed for Humans Interactions with Scheduling Algorithms

In comparison to the extensive literature on developing scheduling algorithms for multiple UV control, there is relatively little research on human interaction with these planners. Caves [17] demonstrated that humans guiding a Rapidly-Exploring Random Tree (RRT) algorithm in the development of paths for multiple UVs could decrease time to solution; however, operators had difficulty understanding the constraints that such algorithms would impose on physical paths. Forest et al. [37, 69], found that operators working with a human-guided algorithm to create a pre-mission schedule for multiple UVs gave the highest subjective ratings to the interface where they had the most control of the objective function of the algorithm. Clare et al. [20] found a similar result in an experiment where operators were allowed to modify the objective function of a real-time decentralized scheduling algorithm throughout a mission. While overall system performance did not change, operators had increased

trust and acceptance of the planner [20]. Hanson et al. [43] found that human operators paired with an algorithm for scheduling multiple UVs desired a greater understanding of why the algorithm made certain recommendations. Miller et al. [51] developed the Playbook™ human-automation integration architecture, which identified a set of common tasks performed by semi-autonomous UVs, grouped them into "plays", and provided the operator with a set of play templates to utilize. Cummings et al. [26] demonstrated that when humans are allowed to coach a decentralized multiple UV scheduler, overall system performance can improve up to 50 % as compared to an unassisted planner. Ryan [63] has shown that when coupled with human expert reasoning, an integer linear programming algorithm can produce superior schedules for multiple manned and unmanned vehicles as compared to just the algorithm alone.

From this sparse literature, as well as more literature on human interaction with algorithms in other domains like transportation planning [3, 67], space satellite scheduling [26], and industrial process control [17, 41], five challenges have been identified in understanding how to best design automated schedulers in order to capitalize on the strengths of human induction and judgment, which are critical in resolving uncertainty in dynamic environments.

Emergent Behaviors:

While near-optimal algorithms have obvious performance benefits, the emergent behavior of an algorithm is a significant concern when pairing a human operator with a scheduling algorithm. An open question is how human operators react to the unpredictability of working with these types of algorithms, where the algorithm's behavior is non-deterministic. Caves [17] showed that operators can become quickly frustrated with such an approach to the detriment of overall performance.

A prime concern for any human operator collaborating with a scheduling algorithm is gaining a sufficient understanding of how the algorithm created the solution [57]. Increasing automation transparency may be difficult with the use of probabilistic, meta-heuristic and auction algorithms, whereas deterministic solvers such as greedy algorithms often emulate the method by which humans would choose to solve the problem [9]. Would a human operator collaborating with an "inferior", but understandable algorithm perform better than a human operator collaborating with a more "advanced" but less predictable and more opaque algorithm? More research is needed into how well human operators can understand the method by which the algorithm reached its solution.

Behavior Modeling:

Computational models of human behavior when collaborating with advanced scheduling algorithms are needed. In contrast to purely theoretical or conceptual models, these models typically leverage computer simulations to both promote deeper understanding of how human operators behave and provide testable predictions about human behavior under different circumstances [38]. These models could be used by designers of future UV systems to predict the relative impact of different algorithm architectures, human-automation collaborative strategies, and training methods on system performance, saving time and money in the design process. It is also possible

that these models could help to generate a set of recommendations for the developers of scheduling algorithms, based on model predictions of which algorithm characteristics have the greatest positive impact on the collaborative scheduling process.

Operator Situation Awareness:

In a human-automation collaborative scheduling system, a central node is necessary for a human operator to maintain situation awareness. It remains an open question whether decentralized algorithms maintain their proposed advantages over centralized algorithms when used by a human operator due to the need for a centrally connected node. While decentralized systems may be capable of continuing a mission even with a communication interruption, the need for consistent updates to the human operator and the need for approval for major schedule changes or critical events such as weapons release from the operator may negate some of the proposed advantages of decentralized algorithms.

Trusted Autonomy:

Human trust in the algorithm is a crucial driver of performance in a human-automation collaborative scheduling system [21]. Human trust in the algorithm can fluctuate due both to the operator's initial trust level in the algorithm and the behavior of the algorithm throughout the mission [38]. Providing operators with certainty estimates could be beneficial to both system performance [11] and developing appropriate trust [48] between the operator and scheduling algorithm. Further research is necessary into human trust in advanced scheduling algorithms and how different methods of displaying algorithm certainty estimates impact the operator's ability to effectively make decisions under time pressure.

Vehicle Health Monitoring:

When monitoring multiple UVs, the need to assess the health and status of the vehicles drives much of the operator's workload. Health monitoring is a very significant problem for single operator control of multiple UVs and even with highly automated vehicles may drive workload to unacceptably high levels [21]. More highly automated health monitoring, error detection, and self-repair algorithms are necessary before single operator control of multiple UVs becomes feasible.

3.3.4 Conclusions

The goal of this effort is to begin to identify algorithm characteristics that could support real-time human-automation collaborative schedule creation for multiple UVs in uncertain environments. While some algorithms may provide optimal schedules almost instantaneously, an algorithm may lack certain characteristics that are essential for effective human-automation collaboration. Our goal is to analyze how these scheduling algorithms can successfully be paired with a human operator to operate in real-world scenarios.

We have highlighted five critical areas that should be considered when teaming an automated scheduler with a human operator. Most importantly, the human represents a source of information that can resolve uncertainty for many schedulers and should be treated as a core system component who adds value to the solution quality instead of simply approving it.

Finally, a systems-level significant issue not specifically addressed by examining human or algorithm attributes is the regulatory aspect of the adoption of non-deterministic algorithms for UV scheduling [72]. How does one certify that an algorithm with no guarantee of repeatability is safe or ready for deployment in safety critical missions? While some have begun to propose methods of performing V&V on advanced autonomy [5], further work is necessary before the military and the Federal Aviation Administration (FAA) will certify advanced autonomy for real-world operations.

3.4 Autonomy Evolution for Air Traffic Control

Air Traffic Control (ATC) operations began in 1936 with fifteen controllers managing flights across US airspace to ensure that aircraft were not on conflicting routes. At that time, air traffic controllers managed flights by contacting airline dispatchers to get information about aircraft locations and routes because radar surveillance systems were not yet in place [1]. ATC operations continued to evolve over the next several decades to meet increasing levels of air traffic and enabled by advances in communication, navigation and surveillance (CNS) systems. For several decades a ground-based radar surveillance system has provided air traffic controllers with the necessary information (e.g., position, velocity, and flight-plan data) to safely separate and manage flights across the National Airspace System (NAS). Advances in flight control systems (FCS) on the aircraft have also evolved to better help pilots manage their aircraft in a safe and efficient manner. For example, Flight Management Systems (FMSs) are able to compute precise trajectories that optimize for fuel use and flight time as a function of aircraft weight and predicted wind conditions [70].

Advances in CNS systems are continuing today with the development of some key technologies, such as Controller-Pilot Datalink Communications (CPDLC), global navigation satellite systems (GNSS), and Automatic Dependent Surveillance-Broadcast (ADS-B). CPDLC is a text-based system that alleviates congestion on voice frequencies and is expected to reduce errors in interpreting voice-based communications between air traffic controllers and pilots. GNSS receivers on-board aircraft allow for more precise navigation in the airspace, eliminating the dependence on ground-based navigation aids that have previously constrained aircraft to certain routes. ADS-B is a key component technology in the modernization of US and European ATC operations. ADS-B transmitters on-board an aircraft will transmit the aircraft's position and velocity, derived from a GNSS, for use by air traffic controllers and ground-based automation systems. Additionally, aircraft that are equipped with ADS-B receivers will receive ADS-B messages from other aircraft in the surrounding

airspace enabling the development of avionics that leverage the increased situation awareness in the cockpit [32].

3.4.1 Challenges and Limitations of Current Air Traffic Management System

Despite the continued advancement of key technologies on the ground and on the flight deck intended to increase safety and efficiency, ATC operations remain human centric with air traffic controllers ultimately providing oversight and direction to pilots to safely manage flights and avoid conflicts in the airspace. Whereas new technologies enable increased automation in ground-support tools for air traffic controllers and in flight-deck systems for pilots, the humans in each of those domains must be provided with sufficient information to accurately assess a situation and take action if necessary [65]. Overriding safety objectives prevent excessive reliance on automation alone in a system with many uncertainties, such as weather, differences in aircraft performance, and the potential for automation failures. The Air France 447 accident is one example of human error due to an over-reliance on flight-deck automation and the information provided by the automation [7].

The Federal Aviation Administration's NextGen and Europe's SESAR (Single European Sky ATM Research) programs are efforts to transform and modernize US and European air traffic management operations, respectively, to meet increasing air traffic demands with traffic levels forecasted to double in the next twenty years [33]. Increasing traffic demands will be addressed by developing new ATC operational concepts that leverage advanced CNS technologies while also improving safety and efficiency for aircraft operators. In the US, there are several ongoing efforts to improve ground automation to help air traffic controllers more efficiently manage flows of aircraft, and there are efforts to improve flight-deck technologies to enable the reallocation of air-traffic-controller tasks to the airborne domain.

3.4.2 Enhancements Made Within ATC System

The Time-based Flow Management (TBFM) program is one effort to improve decision-support tools for air traffic controllers during metering operations, where aircraft are being managed to meet a Scheduled Time of Arrival (STA) at a designated point in the airspace [31]. Metering operations are typically used when traffic density reaches a level that operations are constrained by airspace capacity, and the STAs help ensure a consistent flow of traffic through the airspace. The ground-based automation generates a sequence and schedule of aircraft in the airspace using trajectory-modeling and scheduling algorithms, and the air traffic controller is provided with the amount of time that each flight must be delayed or make up to meet its

STA. The air traffic controller would then employ various techniques (e.g., issuing speed changes or path modifications) for each aircraft to meet their STAs. However, uncertainties in the environment, such as convective weather, may cause significant errors in trajectory modeling and other functions causing air traffic controllers to revert to less efficient modes of operation (e.g., placing flights in holding patterns). Proposed enhancements to ground-based automation include 3D-Path Arrival Management (3D-PAM), developed by the National Aeronautics and Space Administration (NASA), which adds the calculation of speed and path-stretching advisories for aircraft to more precisely meet their STAs while reducing air-traffic-controller workload [22]. In the proposed 3D-PAM concept, the ground-automation still provides the speed and path-stretching advisories to the air traffic controller to communicate to the pilots, keeping the air traffic controller in the loop.

Several flight-deck enhancements have been proposed to enable the allocation of some tasks to the pilot. Currently, some aircraft are equipped with a Required Time of Arrival (RTA) functionality in the FMS that determines the speeds and the vertical profile that the aircraft should fly to meet an ATC-provided STA at a point [6]. In that case, the management of the aircraft's speeds to meet an overall operational objective is allocated to the airborne domain (i.e., the pilot and the avionics). The Interval Management (IM) concept is another flight-deck concept that uses avionics to help the flight crew achieve and maintain a spacing interval (in distance or time) relative to an ATC-specified target aircraft [35, 62]. The IM avionics rely on ADS-B technology to receive the target aircraft's state information. Other flight-deck concepts, enabled by the information provided by ADS-B, have explored the use of a Cockpit-Display of Traffic Information (CDTI) to increase the flight crew's awareness of surrounding traffic and to provide guidance to support the flight crew's safe and efficient management of the aircraft [13, 30, 52].

The 3D-PAM, RTA, and IM concepts may all be used to address a similar operational objective: provide precise and consistent delivery of aircraft to a point (e.g., the terminal boundary). However, the control objectives and task allocations are different for the three concepts, as shown in Table 3.1. In each case, the overall operational objective is managed by the ground domain (i.e., the air traffic controller with supporting ground automation), and some tasks may be allocated to the airborne domain.

Table 3.1 Task allocation and control objectives of flight deck enhancements

Concept	Control objective	Task allocation
3D-PAM	Meet STA at a point	Speeds calculated by ground automation and communicated to pilot by air traffic controller
RTA	Meet STA at a point	STA provided to pilot by air traffic controller; speeds (and vertical trajectory, if necessary) calculated by avionics
IM	Achieve spacing interval relative to target aircraft	Target aircraft and relative spacing interval provided to pilot by air traffic controller; speeds calculated by avionics

Furthermore, in the IM concept, the spacing is relative to a target aircraft resulting in a decentralized control solution, which is in contrast to the RTA concept where an aircraft meets its STAs independently of how the preceding aircraft meets its STA. Whereas the IM concept must address issues with string stability [73], that is not a concern for the 3D-PAM and RTA concepts.

3.4.3 Technical Enhancements needed in the Evolution of Airborne and Ground-Based Technologies

There are several challenges that must be addressed as a part of the ATC operational and technological evolution, including:

1. the determination of the appropriate allocation of tasks between the ground domain (i.e., air traffic controllers and supporting ground automation) and the airborne domain (i.e., pilots and supporting avionics);
2. the methods to synchronize information between the ground and airborne domains;
3. the approach to managing mixed-equipage operations;
4. and, the provision of necessary and sufficient information to the human operators to ensure situational awareness.

These challenges are not independent of each other, and as will be demonstrated here, the investigation of one challenge may inform another.

Task Allocation Strategies:

As previously described, advances in CNS systems and flight-deck automation will enable some tasks to be allocated to the aircraft that were previously managed by air traffic controllers. For example, in the IM concept, the controller could identify a target aircraft and request that an appropriately-equipped aircraft follow the target aircraft with a 90 second spacing interval. The flight crew would enter the appropriate information into the avionics and would ensure that the guidance is followed to achieve and maintain a 90 second spacing relative to its target aircraft. In this case, the air traffic controller still identifies the overall operational objective: to ensure consistent and precise spacing between aircraft; however, the management of the spacing relative to the target aircraft, which is one component of the overall objective, would be the responsibility of the flight crew. There are several different proposed concepts currently under development by the FAA, and in some of these concepts, the task allocation remains much as it is today, with controllers leveraging decision-support tools to manage traffic. In other concepts, such as IM, tasks are reallocated because it is believed that improved flight-deck automation will enable more efficient operations.

Information Synchronisation:

A major challenge that must be addressed is the synchronization of information used by the airborne and ground domains. Inconsistent information used by the flight-deck

and ground automation may lead to inconsistent or unexpected behavior, which will likely limit the usability of automation systems and will lead to distrust by human operators. For example, inconsistencies in the wind information used to model aircraft trajectories can lead to significant differences in trajectory times. In RTA flight trials, the ground automation modeled trajectory times to determine feasible STAs for aircraft entering terminal airspace. In the RTA operation, the air traffic controller provided a ground-derived STA to a flight crew, and the flight crew entered the STA into the RTA function in the FMS. In some cases, inconsistencies between the wind information in the ground automation and in the avionics resulted in the avionics declaring the STA to be infeasible because the required speeds were outside of the aircraft's performance envelope [34]. Reference [18] describes a proposed method for synchronizing aircraft trajectories between the airborne and ground domains, leveraging CPDLC and ADS-C technologies, to ensure consistent trajectory predictions.

Mixed Equipage Operations:

Avionics standards that support NextGen operations, such as the RTA or IM concepts, are currently being developed. However, there is no mandate for airline operators to equip their fleets with more advanced flight-deck systems. Without a mandate, equipage is driven by economic incentives; and therefore, it is expected, for the foreseeable future, that air traffic operations will consist of a mix of aircraft: some that are equipped with avionics to support applications like RTA or IM and some that are not. In mixed-equipage operations, the air traffic controllers will need to know the equipage capabilities of the different aircraft in order to provide appropriate instructions (e.g., provide an STA to an RTA-equipped aircraft and a target aircraft and relative spacing interval to an IM-equipped aircraft). Furthermore, the combination of different applications within a single operation, in order to best leverage the available aircraft equipage, must be studied to understand any adverse effects that may arise when combining different concepts. Reference [74] presents some considerations for a mixed-equipage IM operation, where some aircraft are equipped with IM avionics and other aircraft are managed by supporting ground automation. Whereas aircraft that are equipped with IM avionics will implement speeds relative to a target aircraft to support the overall operational objective, the air traffic controller will provide speeds, calculated by the ground automation, to unequipped aircraft to meet STAs at a point. Therefore, the mixed-equipage operation combines two different spacing concepts: relative spacing (i.e., spacing relative to a target aircraft) and absolute spacing (i.e., spacing to an absolute time at a point in space), and performance differences in the two concepts can result in unexpected behavior in the combined operation that should be accounted for in the ground automation or in procedures that the air traffic controllers follow.

Operator Situational Awareness:

The information needs of the human operators to accurately assess the state of the operation are a major part of the research for any new ATC operational concept. Air-traffic-controller and pilot inputs remain a significant part of the entire research and development process of any new concept. Feedback from the human operators,

e.g., through human-in-the-loop simulations and fields tests, helps to determine what information must be displayed to the human to ensure that the guidance provided by the ground automation or the avionics is appropriate and will not adversely affect the safety of the operation. Keeping the human in the loop (i.e., the air traffic controller or the pilot) provides an added level of safety to ensure that uncertainties that result in unacceptable guidance will not get passed on to the aircraft.

3.4.4 Conclusions and Proposed Road-Map

A road-map is proposed for the evolution of ATC operations and the evolution of supporting ground and flight-deck automation.

Step 1 *Develop and select concepts to address different airspace objectives (e.g., provide a consistent flow of aircraft into terminal airspace or precisely space aircraft at the runway threshold).*
Because of differences in traffic demand, airspace design, and weather conditions across the US NAS, some ATC operational concepts may be better suited to address certain airspace objectives than others (i.e., one concept may not be suitable for all operations). Furthermore, a number of concepts may be combined to best leverage the expected variation in aircraft equipage.

Step 2 *Evaluate the mixed-equipage operation to identify challenges in combining different concepts that could require different aircraft equipage.*
The identified challenges in the mixed-equipage operation may be addressed through improvements or enhancements in the ground automation or through procedures, to support the air traffic controller's management of the operation, or through modifications to the avionics.

Step 3 *Define steps in the evolution from today's operation to the proposed operation.*
Intermediate steps in the evolution of the operational concept, where the ground automation is providing increasing levels of information and the flight-deck avionics is providing guidance for situations with increasing complexity, may help the human operators to trust the automation.

Step 4 *Conduct operational evaluation throughout the evolution from today's operation to the final, proposed operation.*
The operational evaluation would be comprised of data gathering to:

(I) evaluate the effectiveness of the ground automation and the avionics, especially in the presence of uncertainties such as weather;
(II) evaluate the effectiveness of the information provided to the human operators to aid in their assessments of the operation.

This proposed road-map seeks to address the challenges identified in the previous section, while acknowledging the challenges of increasing the role of automation in a human-centric system. Hesitation to take the human out of the loop in air traffic

operations does, in fact, slow the evolution to higher levels of autonomy. However, we must ask whether traffic demands, as well as safety and efficiency goals, can be reached without complete autonomy in the foreseeable future.

References

1. At 75, America's air traffic control system keeps getting better. http://fastlane.dot.gov/2011/07/air-traffic-control-at-75.html (2011)
2. Alighanbari, M., How, J.: Robust decentralized task assignment for cooperative UAVs. In: AIAA Conference on Guidance, Navigation, and Control Conference, Keystone, CO, pp. 21–24 (2006)
3. Anderson, D., Anderson, E., Lesh, N., Marks, J., Mirtich, B., Ratajczak, D., Ryall, K.: Human-guided simple search. In: AAAI/IAAI, pp. 209–216 (2000)
4. Atkins, E.M.: The benefits and risks of increased autonomy in air and space systems. In: 9th German-American Frontiers of Engineering Symposium (2006)
5. Atkins, E.M.: Intelligent systems for unmanned aircraft safety certification. In: AIAA Aerospace Sciences Meeting, Nashville, TN (2012)
6. Balaskrishna, M., Becher, T., MacWilliams, P., Klooster, J., Kuiper, W., Smith, P.: Seattle required time of arrival flight trials. In: Proceedings of the 30th Digital Avionics Systems Conference, Seattle, WA, p. 2D4-1 (2011)
7. BEA: Final Report: On the accident on 1st June 2009 to the Airbus A330–203 registered F-GZCP operated by Air France flight AF 447 Rio de Janeiro-Paris. Technical report, Bureau d'Enquêtes et d'Analyses pour la Sécurité de l'Aviation Civile (2012)
8. Bellingham, J., Richards, A., How, J.P.: Receding horizon control of autonomous aerial vehicles. In: American Control Conference, 2002. Proceedings of the 2002, vol. 5, pp. 3741–3746. IEEE (2002)
9. Bertuccelli, L., Beckers, N., Cummings, M.L.: Developing operator models for UAV search scheduling. In: AIAA Conference on Guidance Navigation, and Control Conference, Toronto, Canada (2010)
10. Bertuccelli, L.F., Choi, H.L., Cho, P., How, J.: Real-time multi-UAV task assignment in dynamic and uncertain environments. In: AIAA Conference on Guidance, Navigation, and Control, Chicago, IL (2009)
11. Bisantz, A.M., Cao, D., Jenkins, M., Pennathur, P.R., Farry, M., Roth, E., Potter, S.S., Pfautz, J.: Comparing uncertainty visualizations for a dynamic decision-making task. J. Cogn. Eng. Decis. Mak. **5**(3), 277–293 (2011)
12. Blanchard, B.S., Fabrycky, W.J.: Systems Engineering and Analysis, 3rd edn. Prentice Hall, Upper Saddle River (1998)
13. Bone, R.: Cockpit display of traffic information (CDTI) assisted visual separation (CAVS): Pilot acceptability of a spacing task during a visual approach. In: Proceedings of the 6th USA/Europe Air Traffic Management Research and Development Seminar, Baltimore, MD, vol. 122 (2005)
14. Butterworth-Hayes, P.: Conversations with Patrick Ky. Aerosp. Am. (2012)
15. Butterworth-Hayes, P.: A long road for UAS integration in Europe. Aerosp. Am. (2012)
16. Casbeer, D.W., Kingston, D.B., Beard, A.W., Mclain, T.W., Li, S.M., Mehra, R.: Cooperative forest fire surveillance using a team of small unmanned air vehicles. Int. J. Syst. Sci. **37**, 360 (2006)
17. Caves, A.: Human-automation collaborative RRT for UAV mission path planning. Technical report, Massachusetts Institute of Technology (2010)
18. Chan, D.S.K., Brooksby, G.W., Hochwarth, J., Klooster, J.K., Torres, S.: Air-ground trajectory synchronization? metrics and simulation results. In: Proceedings of the 30th Digital Avionics Systems Conference, Seattle, WA (2011)

19. Clare, A.S., Cummings, M.L., Bertuccelli, L.F.: Identifying suitable algorithms for human-computer collaborative scheduling of multiple unmanned vehicles. In: AIAA Aerospace Sciences Meeting (2012)
20. Clare, A.S., Cummings, M.L., How, J.P., Whitten, A.K., Toupet, O.: Operator object function guidance for a real-time unmanned vehicle scheduling algorithm. J. Aerosp. Comput. Inf. Commun. **9**(4), 161–173 (2012)
21. Clare, A.S., Macbeth, J.C., Cummings, M.L.: Mixed-initiative strategies for real-time scheduling of multiple unmanned vehicles. In: American Control Conference, 2012. Proceedings of the 2012, pp. 676–682. IEEE (2012)
22. Coppenbarger, R., Dyer, G., Hayashi, M., Lanier, R., Stell, L., Sweet, D.: Development and testing for efficient arrivals in constrained airspace. In: Proceedings of the 27th International Council of the Aeronautical Sciences, Nice, France (2010)
23. Cummings, M.L., Bruni, S.: Collaborative human-automation decision making. In: Handbook of Automation LXXVI, pp. 437–448. Springer (2009)
24. Cummings, M.L., Bruni, S., Mercier, S., Mitchell, P.J.: Automation architecture for single operator, multiple UAV command and control. Int. Command. Control. J. **1**(2), 1–24 (2007)
25. Cummings, M.L., Clare, A.S., Hart, C.S.: The role of human-automation consensus in multiple unmanned vehicle scheduling. Hum. Factors **52**(1), 17–27 (2010)
26. Cummings, M.L., How, J., Whitten, A., Toupet, O.: The impact of human-automation collaboration in decentralized multiple unmanned vehicle control. Proc. IEEE **100**(3), 660–671 (2012)
27. Cummings, M.L., Thornburg, K.T.: Paying attention to the man behind the curtain. IEEE Pervasive Comput. **10**(1), 58–62 (2011)
28. DoD: Unmanned aircraft systems (UAS) roadmap, 2005–2030. Technical report, Office of the Secretary of Defense (2005)
29. DoD: Unmanned aircraft systems roadmap 2011–2036. Technical report, Office of the Secretary of Defense (2011)
30. Domino, D.A., Tuomey, D., Mundra, A., Smith, A.: Air ground collaboration through delegated separation: Application for departures and arrivals. In: Integrated Communications Navigation and Surveillance Conference (ICNS), 2010, pp. G5–1. IEEE (2010)
31. Federal Aviation Administration: Concept of use for time-based flow management (TBFM). Technical report, United States Department of Transportation (2009)
32. Federal Aviation Administration: Airworthiness approval of automation dependent surveillance-broadcast (ADS-B) out systems (advisory circular 20–165). Technical report, United States Department of Transportation (2010)
33. Federal Aviation Administration: FAA aerospace forecast: Fiscal years 2011–2031. Technical report, United States Department of Transportation (2011)
34. Federal Aviation Administration: Flight trials assessment for 4 dimensional flight management system trajectory based operations. Technical report, United States Department of Transportation (2011)
35. Federal Aviation Administration: Arrival interval management—spacing (IM-s) concept of operations for the mid-term timeframe, version 3. Technical report, United States Department of Transportation (2012)
36. Finnegan, P.: UAV sector faces sweeping changes. Aerosp. Am. (2012)
37. Forest, L.M., Kahn, A., Thomer, J., Shapiro, M.: The design and evaluation of human-guided algorithms for mission planning. In: Human Systems Integration Symposium (2007)
38. Gao, J., Lee, J.D.: Extending the decision field theory to model operators? reliance on automation in supervisory control situations. IEEE Trans. Syst. Man and Cybern.—Part A: Syst. Hum. **36**(5), 943–959 (2006)
39. Girard, A.R., Hedrick, J.K.: Border patrol and surveillance missions using multiple unmanned air vehicles. In: Proceedings of the IEEE Conference on Decision and Control (2004)
40. Guerlain, S.A.: Using the critiquing approach to cope with brittle expert systems. In: Human Factors and Ergonomics Society 39th Annual Meeting (1995)

41. Choi, H.-L.: Consensus-based decentralized auctions for robust task allocation. IEEE Trans. Robot. **25**(4), 912–926 (2009)
42. Haddal, C.C., Gertler, J.: Homeland security: Unmanned aerial vehicles and border surveillance. Technical report, Congressional Research Service (2010)
43. Hanson, M.L., Roth, E., Hopkins, C.M., Mancuso, V.: Developing mixed-initiative interaction with intelligent systems: Lessons learned from supervising multiple UAVs. In: AIAA 1st Intelligent Systems Technical Conference (2004)
44. How, J.P., Fraser, C., Kulling, K.C., Bertuccelli, L.F., Toupet, O., Brunet, L., Bachrach, A., Roy, N.: Increasing autonomy of UAVs. IEEE Robot. Autom. Mag. **16**(2), 43–51 (2009)
45. Johnson, K., Ren, L., Kuchar, J.K., Oman, C.M.: Interaction of automation and time pressure in a route replanning task, In: International Conference on Human-Computer Interaction in Aeronautics (HCI-Aero) (2002)
46. Keller, J.: Air Force looks for machine autonomy to enable UAVs and piloted aircraft to work and play well together. Mil. Aerosp. Electron. (2010)
47. Layton, C., Smith, P.J., McCoy, C.E.: Design of a cooperative problem-solving system for en-route flight planning—an empirical evaluation. Hum. Factors **36**(1), 94–116 (1994)
48. Lee, J.D., See, K.A.: Trust in automation: designing for appropriate reliance. Hum. Factors **46**(1), 50–80 (2004)
49. Malasky, J., Forest, L.M., Kahn, A.C., Key, J.R.: Experimental evaluation of human-machine collaborative algorithms in planning for multiple UAVs. In: Systems, Man and Cybernetics, 2005 IEEE International Conference on, vol. 3, pp. 2469–2475. IEEE (2005)
50. Marquez, J.J.: Human-automation collaboration: decision support for lunar and planetary exploration. Technical report, Massachusetts Institute of Technology (2007)
51. Miller, C., Funk, H., Wu, P., Goldman, R., Meisner, J., Chapman, M.: The playbook approach to adaptive automation. In: Human Factors and Ergonomics Society 49th Annual Meeting (2005)
52. Mundra, A., Cooper, W., Smith, A., Audenaerd, L., Lunsford, C.: Potential benefits of a paired approach procedure to closely spacing parallel runways in instrument and marginal visual conditions. In: Proceedings of the 27th Digital Avionics Systems Conference, St Paul, MN (2008)
53. National Research Council: Autonomy Research for Civil Aviation: Toward a New Era of Flight. The National Academies Press, Washington (2014)
54. National Safety Board: Autonomous vehicles in support of naval operations. Technical report, National Research Council (2005)
55. Norris, J., Powell, M., Vona, M., Backes, P., Wick, J.: Mars exploration rover operations with the science activity planner. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 4618–4623 (2005)
56. Odedra, S., Prior, S.D., Karamanoglu, M.: Investigating the mobility of unmanned ground vehicles. In: International Conference on Manufacturing and Engineering Systems (2009)
57. Parasuraman, R., Riley, V.: Humans and automation: use, misuse, disuse, abuse. Hum. Factors: J. Hum. Factors Ergon. Soc. **39**(2), 230–253 (1997)
58. Polson, P., Smith, N.: The cockpit cognitive walkthrough. In: 10th Symposium on Aviation Psychology (1999)
59. Ponda, S., Ahmed, N., Luders, B., Sample, E., Hoossainy, T., Shah, D., Campbell, M., How, J.: Decentralized information-rich planning and hybrid sensor fusion for uncertainty reduction in human-robot missions. In: AIAA Conf. on Guidance, Navigation, and Control, Portland, OR, pp. 8–11 (2011)
60. Rediess, H.A., Garg, S.: Autonomous civil aircraft—the future of aviation? Aerosp. Am. (2006)
61. Ross, P.E.: When will we have unmanned commercial airliners?. IEEE Spectr. (2011)
62. RTCA: Safety Performance, and Interoperability Requirements Document for Airborne Spacing—Flight-Deck Interval Management (ASPA-FIM). RTCA, Washington (2011)
63. Ryan, J.C.: Assessing the performance of human-automation collaborative planning systems. Technical report, Massachusetts Institute of Technology (2011)
64. Scott, S.D., Lesh, N., Klau, G.W.: Investigating human-computer optimization. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '02, pp. 155–162. ACM, New York, NY, USA (2002)

65. Sheridan, T.B.: Next generation air transportation systems: Human-automation interaction and organizational risks. In: Proceedings of the Resilience Engineering Symposium, Antibes-Juan-Les-Pins, France (2008)
66. Silverman, B.G.: Human-computer collaboration. Hum.-Comput. Interact. **7**(2), 165–196 (1992)
67. Smith, P., McCoy, E., Layton, C.: Brittleness in the design of cooperative problem-solving systems: the effects on user performance. IEEE Trans. Syst. Man, and Cybern.—Part A, Syst. Hum. **27**(3), 360–371 (1997)
68. Sweeten, B.C., Royer, D., Keshmiri, S.: Meridian UAV flight perfomance anlaysis using analytical and experimental data. In: AIAA Infotech@Aerospace Conference, Seattle, WA, vol. 1899 (2009)
69. Thorner, J.L.: Trust-based design of human-guided algorithms. Technical report, Massachusetts Institute of Technology (2007)
70. Walter, R.: Chapter 15: Flight Management Systems, Chap. 15, pp. 1–25. CRC Press, Boca Raton (2001)
71. Webster, M., Cameron, N., Jump, M., Fisher, M.: Towards certification of autonomous unmanned aircraft using formal model checking and simulation. In: AIAA Infotech@Aerospace Conference, Garden Grove, CA (2012)
72. Weibel, R.E., Hansman, R.J.: An integrated approach to evaluating risk mitigation measures for UAV operational concepts in the NAS. In: AIAA Infotech@Aerospace Conference, Arlington, VA (2005)
73. Weitz, L.A., Hurtado, J.E.: String stability analysis of selected speed control laws for interval management. In: AIAA Conference on Guidance, Navigation, and Control, Minneapolis, MN (2012)
74. Weitz, L.A., Katkin, R., Moertl, P., Penhallegon, W.J., Hammer, J.B., Bone, R.S., Peterson, T.: Considerations for interval management operations in a mixed-equipage environment. In: Proceedings of the AIAA Aviation Technology, Integration, and Operations Conference, Indianapolis, IN (2012)
75. Westwood, J.: Global prospects for AUVs. In: Offshore Technology Conference (2001)
76. Whitten, A.K.: Decentralized planning for autonomous agents cooperating in complex missions. Technical report, Massachusetts Institute of Technology (2010)

Chapter 4
Challenges in Aerospace Decision and Control: Air Transportation Systems

Hamsa Balakrishnan, John-Paul Clarke, Eric M. Feron, R. John Hansman and Hernando Jimenez

4.1 Introduction

The Next Generation Air Transportation System (NextGen) is the FAA's vision of how the nation's aviation system will operate in 2025 and beyond [40]. In 2004, the National Airspace System (NAS) was already operating near capacity, and demand was expected to grow two- to threefold over the following 20 years. The NextGen initiative was established in 2003 in order to meet the challenges presented by the predicted increase in demand. It represents a substantial and long-term change in the management and operation of the US air transportation system, and involves both the leveraging of existing technologies and the development of new ones. This includes satellite-based navigation and control of aircraft, advanced digital communications, advanced infrastructure for greater information sharing, and enhanced connectivity between all components of the air transportation system. The overarching objectives of NextGen are to improve the safety, speed and efficiency, and to mitigate the environmental impacts of air transportation, while accommodating increased demand.

H. Balakrishnan (✉) · R.J. Hansman
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: hamsa@mit.edu

R.J. Hansman
e-mail: rjhans@mit.edu

J.-P. Clarke · E.M. Feron · H. Jimenez
Georgia Institute of Technology, Atlanta, Georgia
e-mail: johnpaul@gatech.edu

E.M. Feron
e-mail: feron@gatech.edu

H. Jimenez
e-mail: hernando.jimenez@asdl.gatech.edu

E. Feron (ed.), *Advances in Control System Technology for Aerospace Applications*, Lecture Notes in Control and Information Sciences 460, DOI 10.1007/978-3-662-47694-9_4

There are similar ongoing efforts in Europe as well, as part of the Single European Sky Air Traffic Management Research (SESAR) initiative [22].

The safe and efficient functioning of the air transportation system requires the smooth integration of the technological (which are generally computer-based) and physical elements of the system. This requirement will be even greater in NextGen, with an increase in the level of automation in all parts of the system, ranging from the aircraft themselves, to the ground infrastructure, communication systems, and air traffic controller decision support tools. Air transportation is a prime example of a system with computational as well as physical elements, also known as a Cyber-Physical System (CPS). Research on fundamental CPS issues is therefore critical to the successful development and implementation of next generation air transportation systems worldwide. Recognizing this need, the US National Science Foundation (NSF) initiated discussions amongst researchers studying air transportation systems [5]. The main objectives of the study were:

- Identifying key challenges related to the NextGen and its successors.
- Determining a logical path from the safety-critical problems faced by NextGen to a list of fundamental research topics.

It is worth noting that many of the fundamental research topics in air transportation systems are closely related to broader CPS problems, further emphasizing the importance of this research, and the need for developing solutions.

4.2 Key NextGen Topics

The initial survey of the research community identified nine key topics of discussion, as shown below in Fig. 4.1. The topics included the decomposition of the system by phases of flight (*Airspace Management*, and *Airport and Terminal-area Operations*), decomposition by function (*Traffic Flow Management*, and *Communication, Navigation and Surveillance*), system-level performance concerns (*Safety*, *Interactions*

Fig. 4.1 The nine key NextGen topics that were identified

between Humans and Automation, and *Metrics*), interoperability in an increasingly global environment (*International Operations*), and the future challenges that will be imposed by the changing face of demand (*New Vehicles in the NAS*). Because many different divisions of the system into such topic areas are possible, similar research issues were raised independently in multiple discussion groups: for example, the need for frameworks to assess system risk and vulnerability was seen as a key issue in the context of safety, CNS systems, international operations, and the introduction of new types of vehicles into the NAS.

4.3 Supporting Technology Research Challenges

Each of the previously-mentioned NextGen topics presents fundamental research problems that need to be tackled. There are many recurring themes, i.e., the same research problems are critical to several topics. This result is not surprising, and it emphasizes the very interconnected nature of air transportation. However, this observation also shows the need to solve the identified research problems in a holistic manner, paying attention to the different elements of the system, in order to obtain solutions that can be implemented in the real world. A bipartite mapping of some of the critical research issues to the NextGen topics is shown in Fig. 4.2.

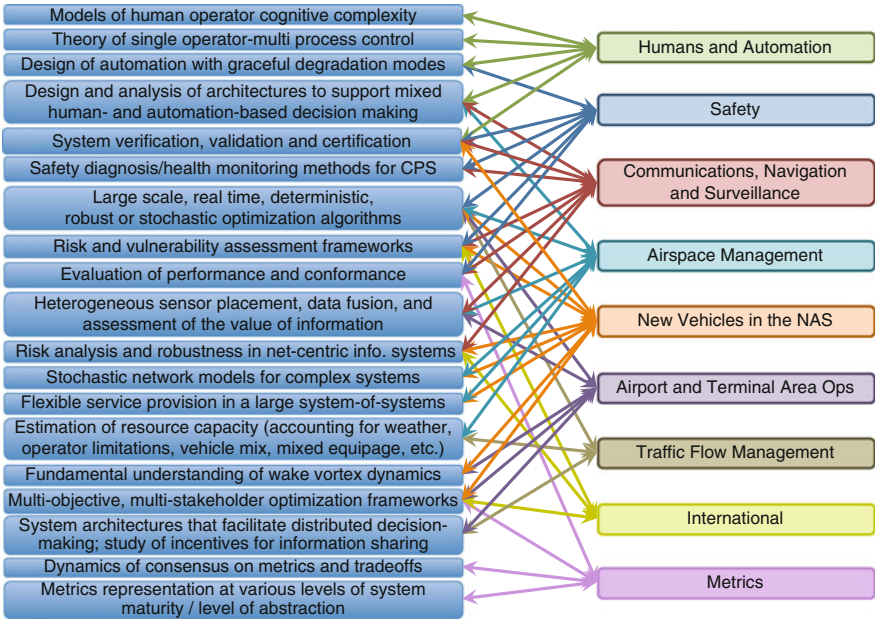

Fig. 4.2 A mapping of research issues (nodes on the *left*) to NextGen topics (nodes on the *right*)

It is useful to consider the mapping of research problems to NextGen topics using different "lenses", each centered on a particular research problem. We now consider the views through four such "lenses", namely, ones centered on the problems of the design of automation with graceful degradation modes, the verification, validation and certification of aviation systems, the development of large scale, real time, deterministic, robust or stochastic optimization algorithms, and the design of multi-objective, multi-stakeholder optimization frameworks. The objective of the following discussions is to describe each research problem, assess its relevance to NextGen, discuss its relevance to other or more general cyber-physical systems, and finally, describe ongoing research efforts within the aviation community to solve this problem and the research directions adopted for the same problem in other CPS.

4.3.1 Design of Automation with Graceful Degradation Modes

Graceful degradation is a core concern for NextGen, and is relevant to several of the identified NextGen topics: Humans and Automation, Safety, Airspace Management, and CNS (Communication, Navigation and Surveillance). Considering CNS and airspace management, a particular challenge arises from hardware breakdowns, whose ability to gracefully degrade may lie with the hardware itself (via some form or redundancy), or with a combination of hardware and operations reconfiguration (for example, existing procedures to handle sudden losses of radar coverage, temporary airport shutdown such as San Francisco in july 2013, or loss of airspace management, such as Chicago's approach control in 2014). Fundamental research topics associated with mitigating such graceful degradation include fault detection and isolation, as well as adaptive control, taken in its most general sense - the sense carried by this research volume and previously covered in [76].

Considering humans, automation, and safety, the notion of graceful degradation is perceived to be intimately connected with human operators and their interaction with automation. In other terms, the entire human + automation system is considered to be degrading, and graceful degradation must be considered with the whole system in mind. In the context of safety, graceful degradation is concerned more with a nominal human being able to properly handling several machine modes, such as nominal or alternative control laws in a jetliner, or handling several aircraft projected on a computer screen [29].

4.3.2 System Verification and Validation (V&V)

This broad topic relates to Humans and Automation, Safety, and Communication, Navigation, and Surveillance (CNS). In the context of humans and automation, the issue is that system breakdown is typically not in human performance or automation per se, or their interface, but rather in work constructs that are too complex, or too

brittle, to provide efficient, repeatable, robust operations. Thus, it is important to develop system verification and validation techniques that apply to both the humans and the environment they interact with. This novel observation has served as the basis for several research projects, including those supported by NASA's airspace safety program.

In the broad context of safety, there are several well-known issues in V&V. The evolutionary nature of NextGen encourages the use of compositional approaches in software and system design/verification. For example, there is the need to be able to integrate/replace subsystems and technologies without having to re-certify the whole system. Such wishful thinking, however, must also accept the limits established by system theory, which has shown long ago that compositionality of system sub-elements to form a system often does not imply compositionality of verification and validation of the same sub-elements to prove the system is safe. The underlying question is whether the air transportation is closer to a large supply chain or the Internet, where modularity of design and analysis have proved particularly efficient, or more like one of these modular, complex flexible structures like the International Space Station or the most recent commercial aircraft, where compositionality of verification and validation has eluded researchers and engineers for years, and re-examination of the entire system is necessary each time a new module is added or a new version is designed [35]. Moreover, the need for certification of embedded and real time software at the algorithmic level, mixed with concerns about cost and development time of that certified software justify the need for research in model-based software validation and certified-by-design software.

In the context of CNS, we observe that the current FAA network has approximately 17,000 network devices, and keeps growing, and that in November 2009 configuration errors brought current FAA network down and air traffic to crawl [67]. It is therefore necessary to come up with new methods to update, verify, and validate configurations on the fly. Moreover, advances in network configuration management technology and techniques are needed, which require the development of a discipline of complex system configuration, verification of system architectures, robust and fault-tolerant systems design.

4.3.3 Large-Scale, Real-Time Optimization Algorithms

This research topic covers the NextGen topics of safety, airspace management, airport and terminal-area operations, new vehicles, and traffic flow management strategies.

There are currently over 35,000 commercial flights a day in the US, and a similar number of military and general aviation flights. Scheduling airport and airspace resources in such a system needs the development of large-scale optimization algorithms, and the incorporation of weather uncertainties motivates the development of robust or stochastic approaches. Safety constraints increase the complexity of the problem. There are a range of ways in which the system can be modeled, ranging from discrete models, which require the solution of large integer programs [8, 9],

to continuous, flow-based, Eulerian models, which can often be solved using linear programs, but lose individual, flight-scale characteristics [7, 51, 62, 90]. Both approaches hold promise at different time-scales, and research into the tradeoffs between the approaches for tactical and strategic decision-making is needed.

4.3.4 Multi-Objective, Multi-Stakeholder, Optimization Frameworks

The presence of multiple stakeholders and competing interests in the air transportation system also poses an interesting research challenge. The objectives themselves may differ among stakeholders; for example, on a weather-impacted day at a congested airport, air traffic control may be interested in maximizing the rate at which aircraft arrive and depart (the throughput), the airlines may be interested in minimizing either fuel costs, crew costs, or the total delays incurred by them, or the delay incurred by high-priority flights, while travelers may care about the delay per passenger or the number of missed connections, or, simply, the possibility of arriving home today rather than tomorrow. These objectives are not necessarily aligned; for example, the schedule that maximizes throughput may not be the fuel-optimal or delay-optimal schedule. It is necessary to develop techniques that determine the trade-offs among the different objectives to support traffic managers and airport operators in their decision-making [52, 66]. Similarly, the study of the incentive properties of resource allocation mechanisms is an important step to handling such a multi-stakeholder environment.

4.4 Domain-Specific Research Challenges

In addition to the overarching issues mentioned above, the design and implementation of domain-specific solutions also pose important research challenges. A few such problems are discussed below.

4.4.1 Airport Arrival Management

Arrival management consists mainly of making sure that the flow of arriving aircraft is timely and orderly. In this process, the primary consideration is runway capacity.

4.4.1.1 Ground Delay Programs

Sometimes, optimizing arrival flows occurs *before* the arriving aircraft has even taken off from its origin airport. Under these conditions, for a given set of aircraft bound to an airport, it has been observed that an excessive number of flights at departure results, a few hours later, into excess arrival demand at the destination airport and delays. Given the lower fuel and crew costs associated with waiting on the ground at the departure location, rather than waiting in the air at destination, *ground delay programs* aim at delaying aircraft departures sufficiently so as to avoid traffic saturation at arrival [6, 94]. For completely saturated airports, such as London Heathrow international airport, such ground delay programs must be very carefully weighted against the extreme, round-the-clock demand for landings, which cannot tolerate any missed landing opportunity. In this case, residual airborne delay is deliberately maintained close to arrival so as for arrival traffic to always be available to meet the landing runway capacity. The key progress in ground delay programs has been, unchallenged, the introduction of *Collaborative Decision Making* (CDM) [96], whereby airlines recover some independent decision-making ability despite temporary capacity limitations. This success can be equally attributed to the phenomenal evolution in communication technology, and to the strong spirit of collaboration among all airspace operators, most notably air traffic flow managers and airline dispatchers.

Recently, the collaborative principles of ground-delay programs have been tentatively extended to handle airborne capacity limits. The *Collaborative Trajectory Options Program* (CTOP) aimed at formalizing the already existing collaboration between airline dispatchers, pilots, and air traffic flow managers when desired routes and enroute capacity drop. Although similar in spirit to CDM, the foundations of the problem addressed are very different, adding a strong spatial dimension to the temporal dimension successfully handled by CDM.

4.4.1.2 Green Arrivals

Traditional arrival procedures are built upon the availability of 1950's navigation and control technology. Such arrival procedures are very inefficient as far as fuel and noise go, as they include several descent and power-on level flight segments, whereas, in principle, environmentally friendlier, less noisy, power-off descents are possible. The main challenge associated with power-off, green descents is the partial loss of controllability over the aircraft trajectory *timing*. Consider for example the standard approach shown in Fig. 4.3 together with a green approach. The standard approach is staged with many leveled-off segments allowing air traffic control and pilots to tightly control their 4D position. The green approach, in comparison, offers little control over the trajectory timing, having been already carefully optimized to be performed with idle, or near idle, engines. The challenge of green descents is therefore to meet the stringent timing requirements allowing the arrival runway to perform at full capacity, despite the more limited control action. In recent work [74], it was shown that, given essential information, such as aircraft type, aircraft mass,

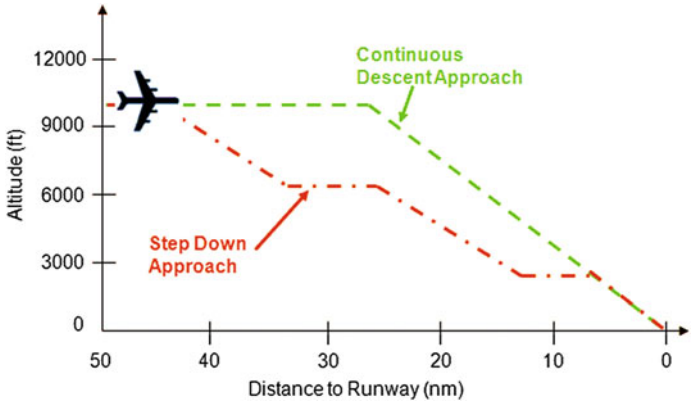

Fig. 4.3 Conventional versus continuous approaches (from [64])

and the aircraft time of arrival at the top of descent point, it is possible to estimate the time of arrival at the final approach fix with sufficient precision and therefore reduce environmental impact without impacting airport capacity. Today, green arrivals are implemented at many airports around the world.

4.4.2 Airport Departure Processes

Aircraft taxiing on the surface contribute significantly to the fuel burn and emissions at airports. This observation has motivated several efforts to identify opportunities to reduce airport congestion, design and field-test surface management strategies, and estimate the benefits of these strategies [24, 63, 71, 84, 89].

4.4.2.1 Modeling

The modeling of airport operations is essential for understanding the underlying challenges, and for the development of mitigation strategies. To this end several models have been developed, ranging from queuing models [38, 49, 56, 72, 83], to various attempts to characterize airport capacity [31, 73]. These models have been further used to predict taxi-out times at a specific airport [36, 37, 81, 82], and on occasion, as part of NAS-wide modeling efforts [17, 55, 99].

A third body of related work involves the microscopic modeling of all airport components. These tools model the layout of an airport, the operating rules for every aircraft type, and the dynamics of every gate, taxiway and runway with high fidelity. As would be expected, there are tradeoffs among the level of modeling fidelity, effort, and computational times [18, 34, 53].

4.4.2.2 Congestion Mitigation Strategies

The simplest form of airport congestion mitigation mechanism is to create a *virtual queue*, whereby aircraft ready for pushback wait for favorable conditions to do so [24]. A state-dependent pushback policy, such as the *N-Control* strategy [12, 15, 16, 71], is based on the typical variation of departure throughput with the number of departures on the surface (denoted N): As more aircraft pushback from their gates onto the taxiways, the throughput of the departure runway initially increases, as seen in Fig. 4.4 and first observed by Shumsky [82]. This curve is the *fundamental diagram of airport traffic* and is the exact counterpart to the *fundamental diagram of networked ground traffic*, extensively discussed by Daganzo [30], and shown in Fig. 4.5. As the number of taxiing departures exceeds a given threshold, denoted N^*, the departure runway capacity becomes the limiting factor, and there is no additional increase in throughput. Any additional aircraft that pushback simply incur taxi-out delays [63]. A similar heuristic, based on the concept of an Acceptable Level of Traffic (ALOT), has been observed to be employed by Air Traffic Controllers at BOS during extreme congested situations [19]. The N-Control policy is also closely related to the *constant work-in-process* (CONWIP) policy used in manufacturing systems. The main benefits of CONWIP systems are their simplicity, implementability and controllability [88]. They present an efficient way to control congestion by accepting an adjustable risk of capacity loss. Such congestion control strategies are the direct counterpart of road access control strategies implemented in many highways around

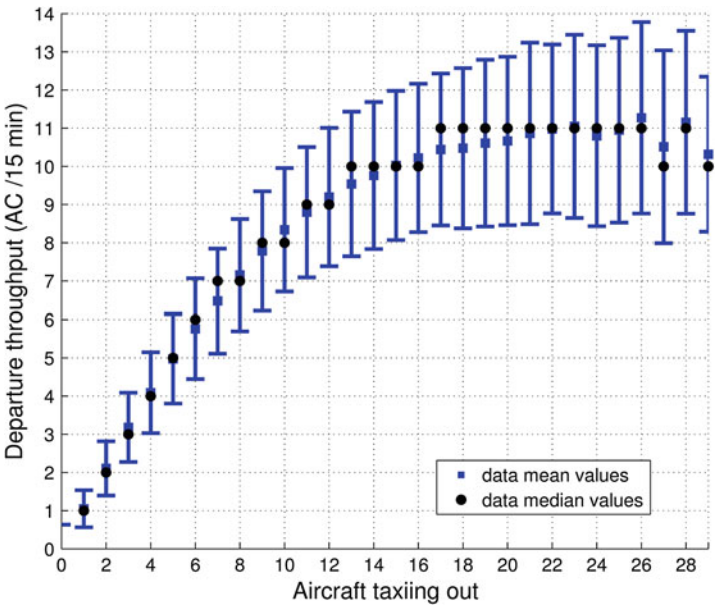

Fig. 4.4 Airport throughput versus traffic density

the world. It is worth mentioning that airports are usually not subject to the type of instabilities resulting from the unimodal geometry of the 'fundamental curve of ground traffic' (cite appropriate people here) that makes road traffic congestion so challenging to address, although evidence of gridlock can be observed experimentally at certain exceptionally busy airports, such as Newark Airport, see Fig. 4.6.

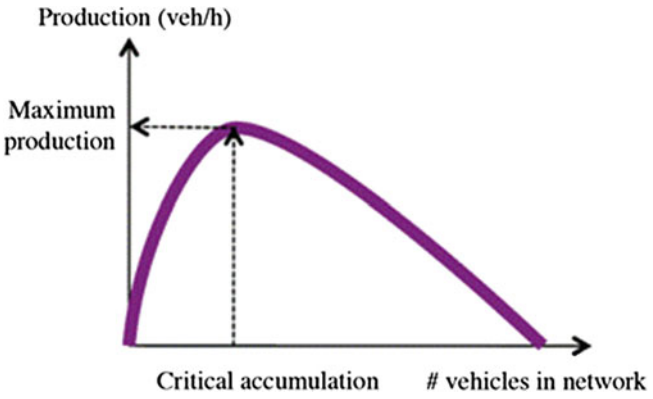

Fig. 4.5 Fundamental diagram of single or networked traffic [93]

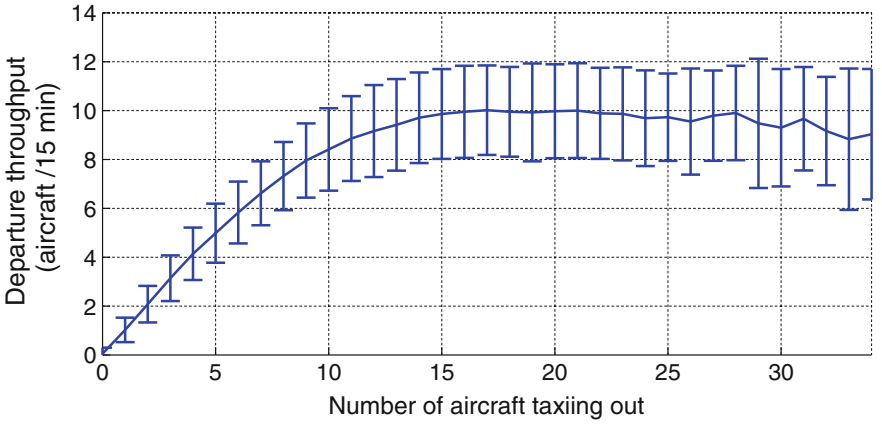

Fig. 4.6 Fundamental diagram of airport traffic at New York International Airport; when the airport is very highly loaded (in excess of 20 aircraft taxiing at the same time), the airport throughput goes down, showing evidence of a growing gridlock

4.4.2.3 Implementation

The N-Control strategy can be adapted to the human environment in the air traffic control tower to obtain a rate-based control strategy by predicting the throughput in a future time period, and deciding on the number of pushbacks in that time period needed to maintain a certain level of ground traffic. Indeed, it was found that a rate-based control strategy is much better accepted by human operators than a pure threshold strategy. Generalized versions of pushback control policies, based on full-state feedback using surface surveillance, have also been considered for surface traffic management [13, 14]. Recently, a refined Pushback Rate Control strategy has been developed, which uses the transient analysis of $D(t)/E_k(t)/1$ queuing systems to determine the optimal rate of pushbacks in a given time period, based on the operating environment, the number of aircraft taxiing out, and the length of the departure runway queue [84].

Several approaches to departure metering have been field-tested, including the Ground Metering Program at New York's JFK airport [63, 89], the field-tests of the Collaborative Departure Queue Management concept at Memphis (MEM) airport [11], the human-in-the-loop simulations of the Spot and Runway Departure Advisor (SARDA) concept at Dallas Fort Worth (DFW) airport [41], the trials of the Departure Manager (DMAN) concept [10] in Athens International airport (ATH) [80], and the field-tests of Pushback Rate Control at Boston's Logan International airport (BOS) [85, 86].

Pushback control can also be formulated as a network congestion control problem and solved efficiently using approximate dynamic programming techniques [92]. This approach has been shown to effectively address practical resource constraints, such as limited gate availability [43, 44].

4.4.2.4 Metrics to Characterize Airport Operational Performance

The availability of detailed surface surveillance datasets from sources such as the Airport Surface Detection Equipment, Model-X (ASDE-X) have the potential to be used for assessing airport performance, in addition to their primary purpose of enhancing safety [23]. In particular, the data can be used to characterize surface flows, including identification of congestion hotspots, queue dynamics and departure throughput, and to develop metrics to evaluate the daily and long-term operational performance of an airport under different operating conditions. These metrics can provide useful feedback on operational performance to airport operators, and therefore have the potential to improve the efficiency of surface operations at airports [42, 45].

4.4.3 The Trip is Not Over: Passenger Management in the Terminals

The passenger perspective on airports resource management differs noticeably from that of the aircraft operator: While aircraft operations happen within the *airside*, the passenger experience, which includes part of the airside, is also considerably influenced by *landside* activities. Aircraft and passengers meet at the interface between the two spaces, which consists of the boarding gate.

Passenger traffic is driven by trajectory optimization, system capacity and queuing behaviors, just like airspace dynamics or, for that purpose, any transportation system.

There are several steps that may be undertaken to mitigate long transit times and excessive queuing for passengers: Both phenomena result in lost time, which may be either reduced, or reused to improve passenger experience and airport earnings alike: For example, passengers may use excess transit time to either engage in shopping, or enjoy the services of a restaurant, all activities also beneficial to retailers and the airport authority.

4.4.3.1 Minimizing Passenger Unimpeded Transit Times

Passenger transit times within a given terminal usually grow with the distance separating the passenger point of entry to his destination. Three cases arise: Either the passenger checked in at the airport, or the passenger reached his destination, or the passenger is connecting from one flight to the next. In all cases, gate assignment is key to influencing passenger transit time, since a gate close to the airport terminal results in reduced transit time for checking-in and exiting passengers. On the other hand, two aircraft close to each other will see passengers connecting from either aircraft to the other enjoy a lower gate-to-gate transit time. Placement of aircraft at gates is therefore very important and can be very dynamic.

4.4.3.2 Queuing Effects

Queuing effects arise in two contexts:

First, couplings and trade-offs exist between the competing goals of minimizing passenger unimpeded transit time and ‘queuing at the gate’, a common phenomenon, whereby an arriving aircraft finds its arrival gate to still be occupied by a departing aircraft. Figure 4.7 shows how expected wait time due to occupied gate decays as a function of scheduled gate idle time between occupancies.

A strategy aimed at optimizing passenger transit tends naturally to cluster aircraft within a small, convenient set of gates, and a density increase in scheduled occupancy times for a given gate will result in increased ‘gate queuing’. Casual observations show that ‘gate queuing’ is badly tolerated by passengers. We can explore the pareto front showing passenger transit time from gate to gate (or from gate to exit) vs.

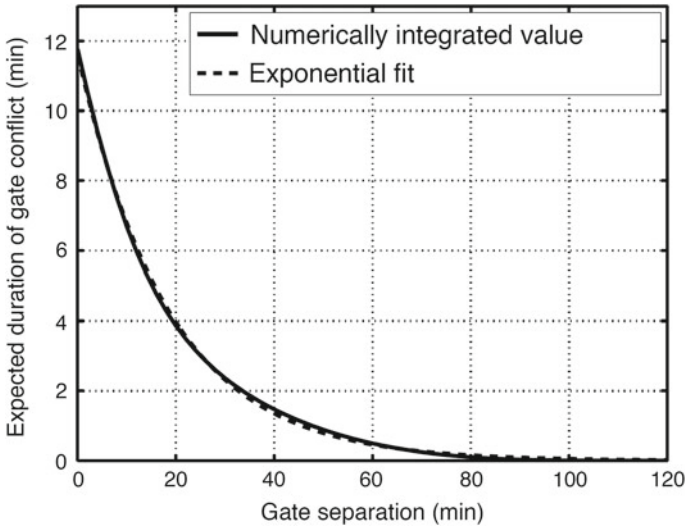

Fig. 4.7 Evolution of average gate delay as a function of planned idle time between occupancies [48]

passenger time waiting for the gate to be free by optimizing

$$J(\alpha) = \alpha \times (\text{passenger transit cost}) + (1 - \alpha) \times (\text{waiting for gate to free up}).$$

for various values of α between 0 and 1 [47]. The resulting tradeoff curve is given in Fig. 4.8. It shows that indeed transit costs and 'wait for gate' costs must be traded against each other. It is up to the airline to adjust what is the equivalent of a 'cost index' to obtain acceptable trade-offs.

Second, queuing occurs at key locations in the airport terminal: Entering passengers experience queuing first at check-in, especially if they need to check in luggage. Then, the same passengers experience queuing at security gates. Last, these passengers experience queuing upon boarding the aircraft. Exiting passengers usually experience delays at customs and border control. Just as for ground delay programs and congested departure operations, intelligent queue management techniques can alleviate queueing problems arising in the terminal. Unlike airport arrival and departure capacity, however, passenger serve capacity at bottlenecks can be adjusted very fast via intelligent staffing policies. One of the keys to matching demand against capacity is appropriate demand monitoring. The recent work by Nikoue and Clarke [48] shows that smart phone positioning techniques using triangulation or other signal localization techniques can be used to anticipate demand at border control and support appropriate booth staffing policies. To support this task, there also exists a fundamental diagram of single or networked pedestrian traffic inside airport terminals. The one shown in Fig. 4.9 reflects that of Sydney airport, based on triangulated signals from cell phones carried by passengers.

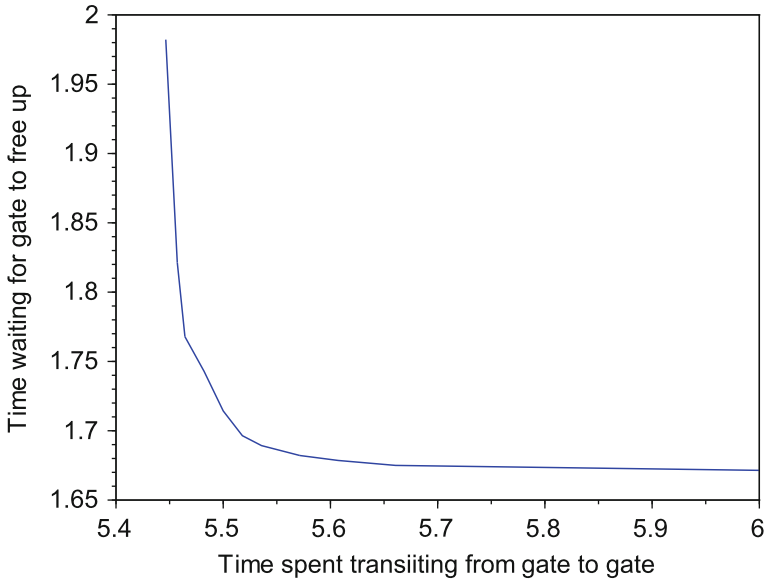

Fig. 4.8 Trade-off between time spent transiting from gate to gate/exit and time spent waiting for arrival gate to be free. All units are in minutes

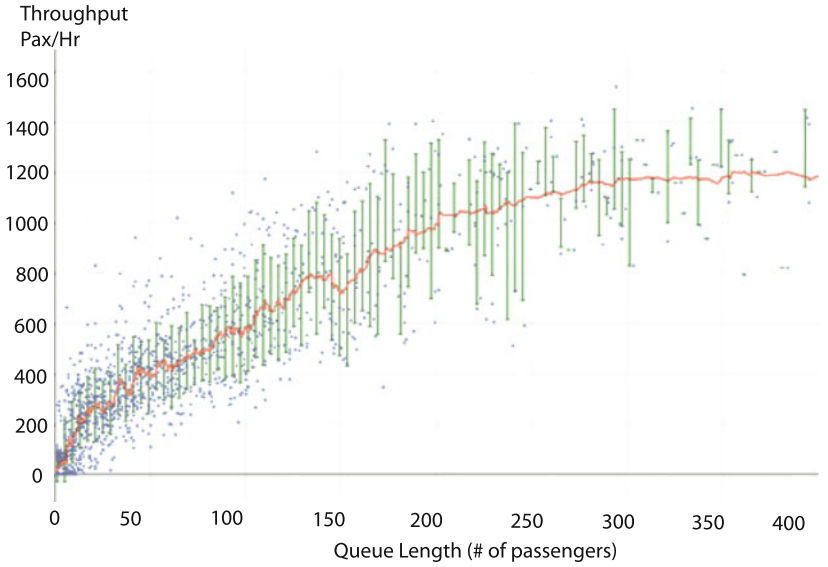

Fig. 4.9 Fundamental *curve* of airport pedestrian traffic approaching passport control at Sydney International Airport (courtesy Harold Nikoue). *Red* Average flow. *Green* 90 % confidence interval. When the facility is highly loaded (in excess of 300 passengers present in the passport control area at the same time), the passenger throughput saturates

Focusing on Sydney international airport, Nikoue and Clarke, have demonstrated the significant improvements that can be obtained by properly managing passenger queuing in real time and feeding back anticipated passenger demand (which is correlated with flights expected flight landing times). Figures 4.10 and 4.11 shows the radically different evolutions of queuing delays, depending on whether feedback is provided and acted upon by the immigration staffing unit.

The effect of feedback is to limit the length of queuing at any time, and eliminate, for example, the massive queuing peak occurring near 11:30 when the nominal staffing profile is applied without feedback.

4.4.4 Domain-Specific Contributions: Abstract Modeling Approaches

The combination of physical aviation infrastructure and automation in NextGen presents a variety of opportunities and challenges related to systems and control theory. The context of traffic control can be the source of many new abstract problems to solicit the curiosity of mathematically inclined researchers in the decision and control sciences. The value of these abstract problems is not only for intellectual amusement: Microscopic simulations project are essential to the validation of the underlying models by confronting their output against the opinion of system opera-

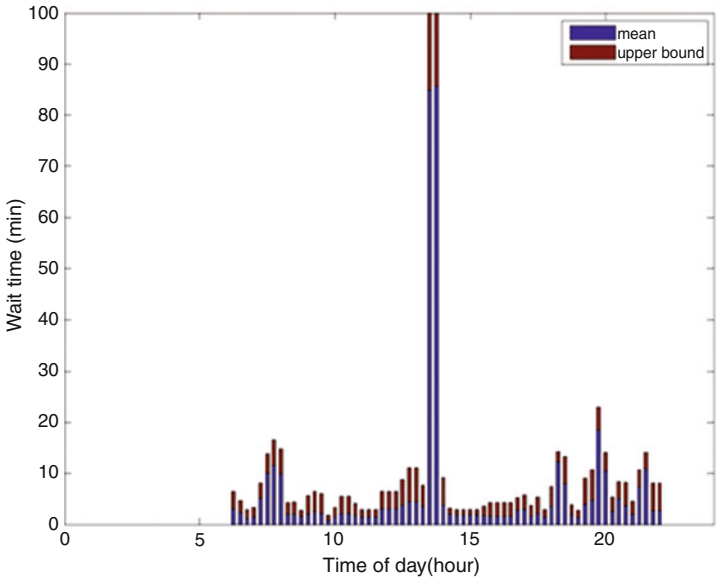

Fig. 4.10 Actual passenger delay generated by mismatched immigration staffing and actual flight arrivals times

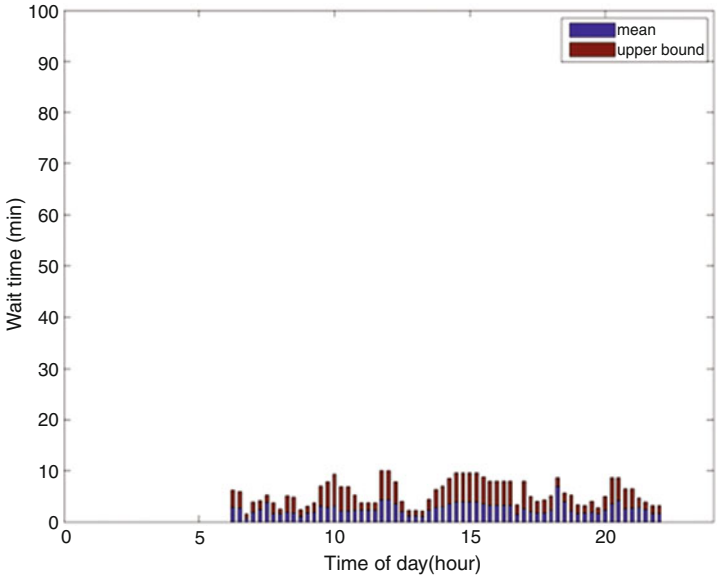

Fig. 4.11 Simulated passenger delay generated by mismatched immigration staffing and actual flight arrivals times

tors. But proper abstract models of traffic can make preliminary analyses of air traffic operations much faster and cost-effective, allowing the investigator to quickly turn several design options around.

4.4.4.1 Models of Air Traffic Based on Statistical Mechanics

Perhaps the greatest success of abstract thinking applied to air traffic control is the generation of closed form analytical models inspired by gas particle dynamics to characterize airspace conflicts arising from properties such as structure, directionality, or density. Some of the earliest published work featuring the gas particle collision model dates to the early 1960's. In a report to the Federal Aviation Administration, Arad et al. [3] proposed a mathematical approach to model air traffic control with the primary purpose of quantifying possible en-route airspace conflicts as a proxy of controller load to guide better sector design. The model relates the control function load to the expected number of conflicts C arising in a sector over a unit of time. C is estimated as

$$C = \frac{2\bar{a}\bar{V}N^2}{g_0 S} \tag{4.1}$$

where $\bar{a}$ is a linear measure of separation minima, V is the average traffic speed, g_0 is a non-dimensional flow organization factor, and S is the (top-view) 2-D area of the

airspace sector. As would be done in most subsequent developments of the particle collision model, expressions for cases interest (e.g. airway crossings or overtaking) are derived from a generalized model form, and the 2D model is extended to formulate one for the 3D case.

In two subsequent studies [25, 32] the same model was applied to the terminal area environment, explicitly incorporating the concept of uniformly distributed random traffic position and heading. As shown by Graham and Orr [33] expressions for the expected number of conflicts can be derived from a few fundamental concepts including the average relative velocity, the conflict volume swept by the moving aircraft, and traffic density. Alexander [1] is perhaps one of the first authors to explicitly refer to this type of model as a *gas model*, arguing that it is a reasonable analogue for air traffic phenomena given the quadratic relationship of conflict rate to traffic density as supported by empirical airspace observations. The model assumes a random distribution of aircraft and flight directions to estimate the probability that an aircraft will not come within a distance r of another aircraft having already traveled a distance L as a function of the average traffic density σ and number of non-interacting altitude layers n. The corresponding mean-free path L_2 and frequency F_2 of 2-body interactions at a distance r with aircraft velocity V are estimated accordingly. The resulting interaction frequency per unit surface area is

$$C_2 = \frac{1}{2} F_2 \sigma = \frac{r\sigma^2 V}{n} \tag{4.2}$$

Using representative values for airspace density σ, aircraft speed V, and aircraft separation minima r, Alexander draws meaningful conclusions regarding inherent conflict risk as a function of density.

In a similar way May [59] builds upon the technique originally suggested by Marks [58] to estimate the probability of conflict with a random particle collision model. The relative velocity $\mathbf{V}_R$ between two aircraft can be used for one so that the other can be assumed immovable. A NMAC cylindrical volume, defined with horizontal separation distance M_H and a vertical separation distance M_V, sweeps a total NMAC volume as the aircraft moves on a straight line in time interval t:

$$R_{NMAC} = 4 M_H M_V V_R t \tag{4.3}$$

The probability of a NMAC is estimated as the the probability that a second aircraft will be in the swept volume R_{NMAC}. Assuming a uniform time-invariant distribution of traffic, the estimate is reduced to be directly proportional to the swept NMAC volume. Following the work by Graham and Orr [33] May estimates the average relative velocity as:

$$\bar{V}_R = \frac{1}{\pi} \int_0^{\pi} \sqrt{V_1^2 + V_2^2 - 2V_1 V_2 cos(\delta)} d\delta \tag{4.4}$$

May applies this general technique to yield analytical estimates for the expected number of NMACs in a number of particular cases of interest, for instance for an aircraft with velocity V_1 intruding airspace where aircraft fly at V_2, in once case with random headings and in another organized as a traffic stream.

Other seminal work focused on extensions and generalizations of the theory, treatment of special cases, and cross-examination against empirical air traffic conflict data. For instance, Anno [2] compared midair collisions from the standpoint of random collision theory with midair collision data to estimate the effectiveness of ATC control.

Endoh's (1982) generalization and extension of the gas model constitutes one of the most significant contributions to the modeling paradigm. For instance, the estimate of the expected relative velocity considers the probability distribution of the velocity values and the vector angle, resulting in the expression below for independent distributions.

$$E(V_r) = \int_{V_1} \int_{V_2} \int_0^{2\pi} \sqrt{V_1^2 + V_2^2 - 2V_1V_2cos(\beta)}\rho(\beta)\rho(V_2)\rho(V_1)d\beta dV_2 dV_1 \tag{4.5}$$

Endoh [21] also examines the implications of the above definition of expected relative velocity against definitions in statistical mechanics and prior midair collision models. The special cases that Endoh derives from the generalized model are also noteworthy because they account for important airspace features that are often not captured in earlier work by virtue of oversimplifying assumptions. For instance, the non-uniform distribution of flight headings as observed in large airspace volumes, an IFR airway immersed in unstructured VFR airspace.

Much of the work by Endoh was later incorporated and expanded upon in reference materials by Professor Robert Simpson [87] for the Engineering of Air Traffic Control Systems course at MIT, contributing to the growing popularity and adoption of the generalized method throughout the 1990s. In this course material Simpson revisits the generalized 2D formulation and provides closed form solutions of the horizontal encounter rate for special cases including a multi-directional (unstructured) traffic flow, unstructured unidirectional flow with varying speeds, airways with overtaking, unidirectional airway in multidirectional flow (IFR airway in VFR airspace), and two-airway intersection. He also examines multi-direction 3-D traffic at constant speed, and the case where traffic is assigned to fly nominal paths, not necessarily straight, in 3 dimensions. Building upon concepts treated by Alexander [1], he also examines the collision rate for traffic assigned to different altitudes.

Today the published literature features a wealth of applications and variations of the gas model for midair collisions, in many cases to study novel airspace concepts and technologies. For instance, the integration of unmanned aircraft systems (UAS) into the national airspace and the midair collision risks it presents have been studied extensively with the gas collision model [4, 20, 60, 61, 95, 97].

Over the last few decades considerable effort has been dedicated to the development of solutions to environmental impact of continued growth of aviation activity.

Technical solutions have been proposed in many forms, the vast majority comprised by advanced concepts and technologies for both aircraft/vehicles and airspace operations. While most efforts will typically address only one or the other, in recent years there has been increasing emphasis on integrated analyses bridging both perspectives. Many advance aircraft concepts are fundamentally predicated on unconventional vehicle configurations for which mission profile and overall performance characteristics are considerably different from more traditional aircraft. These differences have profound implications for airspace operations, particularly when considering the rich variety of aircraft in a mixed global fleet. Rather than examining the implications that unique flight characteristics have on airspace operations as an afterthought, as had occurred often in the past, recent advances have leveraged on the simplicity and power of the gas model to incorporate airspace safety and operational considerations into the core analysis effort for advanced vehicle concepts.

In recent efforts by Feron, Jimenez, and Clarke the impact on airspace conflict resulting from the introduction of advanced aircraft concepts into the operating fleet was examined. This characterization was conducted through encounter rate estimates given the same interpretation of seminal work reviewed earlier in this section, namely a measure of the latent risk providing an upper bound on potential conflicts, or as a measure of the required control load inferred from the level of air traffic activity and associated conflict phenomena. Assessments are made over a 2010 to 2050 time period while incorporating considerations of changing fleet and operations growth forecasts with appropriate fleet-level models [26, 39, 70]. Encounter rate estimates across two fleet scenarios capture the impact of advanced vehicle concept introductions. The reference scenario offers a baseline fleet technology level and assumes no additional improvements beyond the current generation (e.g. A320NEO, B737MAX, B787, A350). The alternate scenario assumes performance improvements for tube and wing aircraft beyond the current generation, single aisle aircraft with open rotor propulsion starting in 2030, and hybrid wing body aircraft for the replacing the traditional wide-body/twin aisle conventional starting in the mid 2030s.

An implementation of the gas particle model as reported by Simpson is used to produce encounter rate estimates. Flight speeds for different flight phases for all aircraft obtained from vehicle performance analyses, in combination with sufficiently detailed sets of operations, are explicitly captured in this implementation of the model via numerical integration of the underlying weighted distributions of speed and heading. The assessment is conducted at three levels of airspace system abstraction.

First, the entire national airspace system is considered only for cruise conditions using the corresponding speed distribution and uniformly random headings. Results predict an increasing rate of conflicts over time that is most directly attributed to the corresponding increase in airspace density. However, there is no significant difference between scenarios suggesting that at an aggregate level for cruise conditions the introduction of advance concepts does not significantly impact air traffic conflict phenomena.

A second examination considers structured IFR airspace at the center level. The Cleveland center (ZOB), one of the busiest in the nation, was selected for this purpose. A simplified representation of the center airspace structure containing its dominant

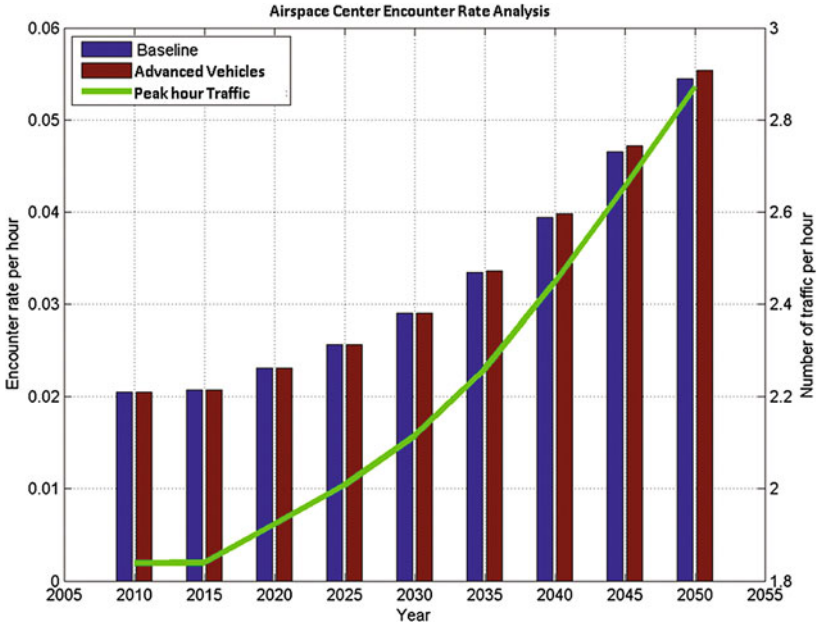

Fig. 4.12 Encounter rate for airway overtaking and crossing, increase trend for traffic projection growth and differences due to fleet mix scenarios

characteristics was developed using a large set of radar track data (for FL350) to identify turning points, cluster them into waypoints, and map each flight into a corresponding sequence of segments [28]. Overtaking conflicts within the busiest airway segment, and crossing segments in the busiest airway segment intersection were evaluated with the model, all the while utilizing the fleet mix for operations conducted in these segments. Results, shown in Fig.4.12, indicate an increase of the encounter rate over time following the increase in airspace density. Only a very small increase in the encounter rate is observed for the scenario with advanced vehicle concepts past the year 2040. This effect, however modest, is attributed to the fleet composition observed for the segments of interest, which is notably different form that for the entire NAS.

The third assessment considers the highly structured and coordinated airspace of a class B airport arrival stream. With about 30 % of all arrivals the FLCON standard arrival is the busiest for Atlanta airport, itself the busiest in the world. The study looked at the arrival peak of the Atlanta 2013 statistical design day, which corresponds to some 95 arrivals per hour. In this busy stream altitude and speed are strictly controlled at the arrival fix which offers an ideal reference point beyond which the encounter rate gas model may be exercised. All aircraft are reasonably assumed to follow the same linear altitude descent profile, as well as a linear velocity deceleration profile, between the arrival fix (12,000 ft, 250 kts CAS) and the final approach fix (1,000 ft, V_{App}) consistent with highly-coordinated Next-Gen operations. In this

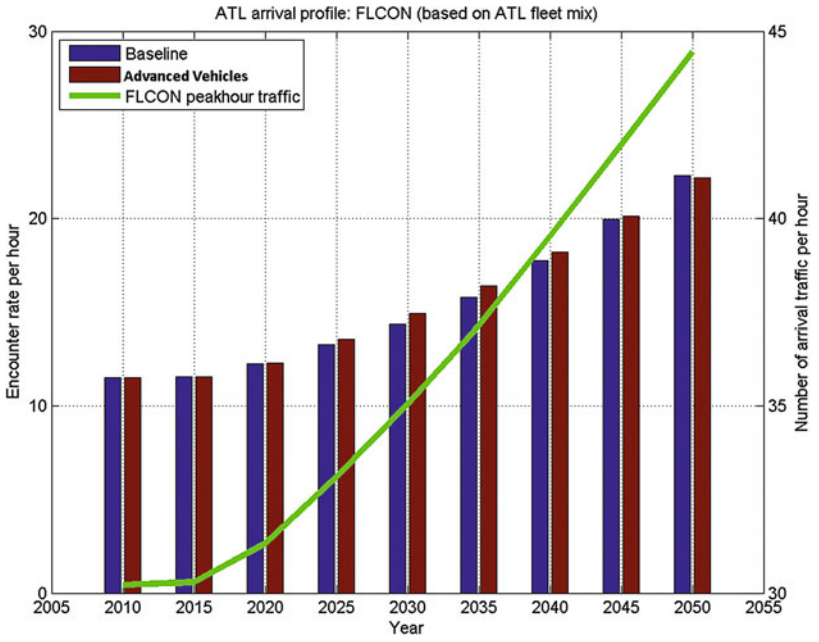

Fig. 4.13 Encounter rate for arrival stream, increase trend for traffic projection growth and differences due to fleet mix scenarios

manner traffic coordination upstream of the arrival fix is implicitly captured without compromising the fundamental modeling approach of the gas model; at the same time it isolates conflict phenomena associated with differences in approach speed for the fleet mix under study within a linear deceleration profile, as would be captured by the model. A 3 NM separation distance is used in this case given that it takes place within the TRACON. Because velocity varies with position, separation minima at the metering fix is not guaranteed for the remainder of the approach path. In the past gas model implementations have assumed a constant speed for aircraft, often times using some average of the arrival speed profile; alternatively the average of the speed differential between consecutive aircraft can be used, for example estimating it as

$$E(V_{Diff}(h)) = E(V_{trailing}(h) - V_{leading}(h)). \tag{4.6}$$

This approach does not capture traffic compression and consequently significantly underestimates the encounter rate. A novel modification to the encounter rate model was implemented to account for the compression effect by calculating the conflict range in consideration of the speed change. This modification to the model remains chiefly analytical, and shows good agreement against direct numerical simulation with Monte Carlo sampling for a Poisson process. Results for the arrival stream, shown in Fig. 4.13, confirm that prior modeling approaches greatly underestimate

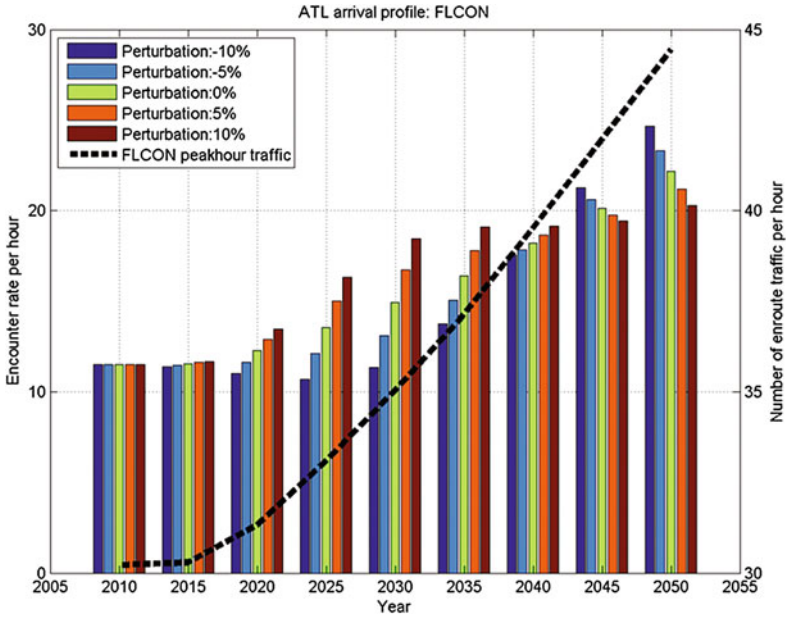

Fig. 4.14 Sensitivity of arrival stream encounter rate to advanced vehicle percent speed perturbations

the encounter rate, and suggest that introduction of advanced vehicle concepts modestly increases the encounter rate over the midterm and then decreases it in the longer term, past the year 2045. This trend reversal is attributed to two effects: the composition of the fleet at any given time, and the approach velocity values of the aircraft in said fleet mix. In the midterm the fleet composition has comparable proportions of conventional tube and wing aircraft with higher approach speeds and unconventional HWB configuration aircraft with much lower approach speeds, resulting in the greatest average speed differentials and higher encounter rate. In the long term the speed differential decreases with the unconventional aircraft comprising a greater proportion of the fleet. The modest magnitude of the impact can itself be attributed to the small proportion of wide-body capacity aircraft operations relative to more dominant narrow-body aircraft operations.

With the encounter rate model for the arrival stream sensitivity analyses can be conducted to examine the extent to which more pronounced velocity discrepancies of advanced vehicles relative to conventional ones result in changes to the encounter rate. Representative results are illustrated in Fig. 4.14.

4.4.4.2 Geometric Analyses of Conflict Detection and Resolution Algorithms

Conflict detection and resolution has rapidly become the canonical problem through which the research community enters the field air transportation. This fact is amply justified by the fact that conflict detection and resolution is the core task and main reason why air traffic management came to be in the first place (cite papers about 1958 grand canyon accident). While there is a great deal of accumulated knowledge about conflict detection and resolution algorithms, the very idea of *proving* that these algorithms do, in fact, eliminate all possible conflicts, is far from exhaustion. Going as far as the development of TCAS in 1992, there has always been a desire to build assurance about collision and conflict avoidance problems (here, cite Lincoln Lab efforts), even though the most common practice has been, up until recently, to rely on exhaustive simulations to slowly build evidence of proper system behavior, and confidence that this behavior will remain so in the future. The very concept of *proving* the correctness of conflict resolution algorithms can be traced back as far back as [65], and possibly further back if one embeds the conflict/collision avoidance problem within the corresponding mobile robotics subject [46, 50, 91], and the concept has slowly gained momentum in the research community, relying on increasingly sophisticated computer-aided verification tools [27].

Bringing a solid mathematical basis to *prove* the absence of collision/conflict among aircraft (under properly stated assumptions) gives the researcher the opportunity to build a complete *axiomatic theory of air traffic control*, whereby proven elementary conflict management blocks can be combined to form a complete engineered system. For example, the initial construct by Mao, Bilimoria and Feron [57], whereby pairs of aircraft flows can be proven to intersect without loss of separation under stated control laws, can be used to build a 'correct-by-design' air transportation network, where conflicts are guaranteed to always be solved. Of course, the experienced air traffic management specialist will immediately recognize many weaknesses in the initial assumptions supporting these constructs. However, such an axiomatic approach can come as a useful complement to operational control procedures to measure the gap that exists between today's system, which originates from a long legacy of pragmatic and incremental improvements, and a more elusive, 'mathematically correct' model built from scratch.

4.4.4.3 Design of Control/Communications Protocols and Architectures

Automatic Dependent Surveillance—Broadcast (ADS-B) is a NextGen surveillance and communication technology by which aircraft can broadcast onboard flight information via datalink to ground stations or other aircraft within range [75, 77, 78]. The position estimates from ADS-B are will be more accurate and have lower latency than traditional ground radar systems [54].

A fundamental systems design decision is the level of decentralization that balances safety and efficiency. While ADS-B can be used to shift air traffic control

to a more distributed architecture, channel variations and interference with existing secondary radar replies can affect ADS-B transmissions. These observations have motivated the development of hybrid control-communication protocols that account for interactions between new and legacy infrastructure, and balance the tradeoffs between safety and efficiency [69].

Along with the increase in efficiency, there is a concern that decentralization may make malicious behavior easier [79, 98]. Erroneous information introduced by malicious entities may be retransmitted by other aircraft, and infect the rest of the system. With an increase in automation in all levels of the system, cybersecurity becomes critical. These challenges also present opportunities to use innovative data fusion techniques to detect errors in broadcast data, build redundancy into the system, and improve overall performance [68].

Acknowledgments This work was performed under partial support from the Federal Aviation Administration under the NEXTOR II program task 0025, the National Aeronautics and Space Administration under Contract No. NNL12AA14C, and Société Internationale de Télécommunications Aériennes.

References

1. Alexander, B.: Aircraft density and midair collision. In: Proceedings of the IEEE, vol. 58 (1970)
2. Anno, J.: Estimate of human control over mid-air collisions. J. Aircr. **19**, 86–88 (1982)
3. Arad, B.A., Golden, B., Grambard, J., Mayfield, C., Saun, H.V.: Control load, control capacity and optimal sector design. Technical Report No. RD64-16, Federal Aviation Administration, William J. Hughes Technical Center, Atlantic city, NJ (1963)
4. Awad, A.I.: An analysis of the risk from UAS missions in the national airspace. Master's thesis, University of Washington (2013)
5. Balakrishnan, H., Feron, E.: (a sample of the) US academic community's views on research contributions to NextGen. http://cps-vo.org/content/nitrdpresentationpptx (2015). Accessed Feb 2015
6. Ball, M., Lulli, G.: Ground delay programs: optimizing over the included flight set based on distance. Air traffic control. Q. (2004)
7. Bayen, A., Raffard, R., Tomlin, C.: Adjoint-based control of a new eulerian network model of air traffic flow. IEEE Trans. Control. Syst. Technol. **14**(5), 804–818 (2006)
8. Bertsimas, D., Lulli, G., Odoni, A.: The air traffic flow management problem: an integer optimization approach. In: Integer Programming and Combinatorial Optimization, pp. 34–46 (2008)
9. Bertsimas, D., Stock Patterson, S.: The air traffic flow management problem with enroute capacities. Oper. Res. **46**(3), 406–422 (1998)
10. Böhme, D.: Tactical departure management with the Eurocontrol/DLR DMAN. In: 6th USA/Europe Air Traffic Management Research and Development Seminar, Baltimore, MD (2005)
11. Brinton, C., Provan, C., Lent, S., Prevost, T., Passmore, S.: Collaborative departure queue management: an example of collaborative decision making in the United States. In: 9th USA/Europe Air Traffic Management Research and Development Seminar (ATM2011), Berlin, Germany (2011)
12. Burgain, P., Feron, E., Clarke, J., Darrasse, A.: Collaborative Virtual Queue: Fair Management of Congested Departure Operations and Benefit Analysis. Arxiv preprint arXiv:0807.0661 (2008)

13. Burgain, P., Kim, S.H., Feron, E.: Valuating surface surveillance technology for collaborative multiple-spot control of airport departure operations. IEEE Trans. Intell. Trans. Syst. **15**(2), 710–722 (2014)
14. Burgain, P., Pinon, O.J., Feron, E., Clarke, J.P., Mavris, D.N.: Optimizing pushback decisions to valuate airport surface surveillance information. IEEE Trans. Intell. Trans. Syst. **13**(1), 180–192 (2012)
15. Carr, F.: Stochastic modeling and control of airport surface traffic. Master's thesis, Massachusetts Institute of Technology (2001)
16. Carr, F., Evans, A., Feron, E., Clarke, J.: Software tools to support researchon airport departure planning. In: Digital Avionics Systems Conference. IEEE, Irvine CA (2002)
17. Clarke, J., Melconian, T., Bly, E., Rabbani, F.: MEANS MIT extensible air network simulation. Simulation **83**(5), 385 (2007)
18. Clarke, J.P., Li, L., Balakrishnan, H., Feron, E., Griffin, K., Kim, B., Kim, S., Lee, H., Robeson, I.J., Solveling, G., Yu, P., Brooks, J.: Surface traffic optimization in the presence of uncertainties (2010). Final report, NASA Project NNX07AU34A
19. Clewlow, R., Michalek, D.: Logan Control Tower: Controller Positions, Processes, and Decision Support Systems. Technical report, Massachusetts Institute of Technology (2010)
20. Dalamagkidis, K., Valavanis, K., L.A., Piegl: Evaluating the risk of unmanned aircraft ground impacts. In: 16th IEEE Mediterranean Conference in Control and Automation, pp. 709–716 (2008)
21. Endoh, S.: Aircraft collision models, Technical report, FTL Report R82–2, Massachusetts Institute of Technology, Flight Transportation Laboratory (1982)
22. EUROCONTROL: SESAR website. http://www.eurocontrol.int/sesar/
23. Federal Aviation Administration: Fact Sheet of Airport Surface Detection Equipment, Model X (ASDE-X). http://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=6296 (2010)
24. Feron, E.R., Hansman, R.J., Odoni, A.R., Cots, R.B., Delcaire, B., Hall, W.D., Idris, H.R., Muharremoglu, A., Pujet, N.: The Departure Planner: a conceptual discussion. Technical report, Massachusetts Institute of Technology (1997)
25. Flanagan, P.: Chapter frequency of airspace conflicts in the mixed terminal environment. Report of the Department of Transportation Air Traffic Control Advisory Committee, vol. 2, pp. 137–144. Federal Aviation Administration, Oklahoma (1969)
26. Frank, C., Jimenez, H., Pfaender, H., Mavris, D.: Scenario development to evaluate system-wide environmental benefits of aircraft technologies and concepts. In: Proceedings of the 2013 Aviation Technology, Integration, and Operations Conference. American Institute of Aeronautics and Astronautics (2013)
27. Galdino, A., Munoz, C., Ayala-Rincón, M.: Formal verification of an optimal air traffic conflict resolution and recovery algorithm. Log. Lang. Inf. Comput. 177–188 (2007)
28. Gariel, M.: Toward a graceful degradation of air traffic management systems. Ph.D. thesis, Georgia Institute of Technology (2010)
29. Gariel, M., Feron, E.: Graceful degradation of air traffic operations: airspace sensitivity to degraded surveillance systems. Proceedings of the IEEE, special issue on aviation information systems (2009)
30. Geroliminis, N., Daganzo, C.: Existence of urban-scale macroscopic fundamental diagrams: some experimental findings. Trans. Res. Part B **42**(9), 759–770 (2008)
31. Gilbo, E.: Airport capacity: representation, estimation, optimization. IEEE Trans. Control. Syst. Technol. **1**(3), 144–154 (1993)
32. Graham, W., Orr, R.: Chapter terminal air traffic model with near midair collison and midair collison comparison. Report of the Department of Transportation Air Traffic Control Advisory Committee, vol. 2, pp. 151–164. Federal Aviation Administraiton, Oklahoma (1969)
33. Graham, W., Orr, R.: Terminal air traffic flow and collison exposure. In: Proceedings of the IEEE, vol. 58 (1970)
34. Griffin, K.J., Yu, P., Rappaport, D.B.: Evaluating surface optimization techniques using a fast-time airport surface simulation. In: AIAA Aviation Technology, Integration and Operations (ATIO) Conference. Fort Worth, TX (2010)

35. How, J., Glaese, R., Grocott, S., Miller, D.: Finite element model-based robust controllers for the middeck active control experiment (mace). Control. Syst. Technol. **5**(1) (1995)
36. Idris, H., Clarke, J.P., Bhuva, R., Kang, L.: Queuing model for taxi-out time estimation. Air Traffic Control. Q. (2001)
37. Idris, H.R.: Observation and analysis of departure operations at Boston Logan International Airport. Ph.D. thesis, Massachusetts Institute of Technology (2001)
38. Jacquillat, A., Odoni, A.R.: A Study of Airport Congestion at JFK and EWR. In: 5th International Conference on Research in Air Transportation (2012)
39. Jimenez, H., Pfaender, H., Mavris, D.: System-wide assessment of nasa era concept vehicles and technologies for fuelburn and co2. J. Aircr. Am. Inst. Aeronaut. Astronaut. **49**, 1913–1930 (2012)
40. Joint Planning and Development Office: Concept of Operations for the Next Generation Air Transportation System (2007)
41. Jung, Y., Hoang, T., Montoya, J., Gupta, G., Malik, W., Tobias, L.: Performance evaluation of a surface traffic management tool for dallas/fort worth international airport. In: 9th USA/Europe Air Traffic Management Research and Development Seminar (ATM2011). Berlin, Germany (2011)
42. Khadilkar, H., Balakrishnan, H.: A multi-modal Unscented Kalman filter for inference of aircraft position and taxi mode from surface Surveillance data. In: AIAA Aviation Technology, Integration and Operations (ATIO) Conference. Virginia Beach, VA (2011)
43. Khadilkar, H., Balakrishnan, H.: Optimal control of airport operations with gate capacity constraints. In: European Control Conference. Zurich, Switzerland (2012)
44. Khadilkar, H., Balakrishnan, H.: Network congestion control of airport surface operations. J. Guid. Control. Dyn. (2014). To appear
45. Khadilkar, H., Balakrishnan, H., Reilly, B.: Analysis of airport performance using surface surveillance data: A case study of BOS. In: AIAA Aviation Technology, Integration and Operations (ATIO) Conference. Virginia Beach, VA (2011)
46. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. Int. J. Robot. Res. **5**(1), 90–98 (1986)
47. Kim, S.: Airport control through intelligent gate assignment. Ph.D. thesis, Georgia Inst. Technology (2013)
48. Nikoue H., Marzuoli, A., Clarke, J.-P., Feron, E., Peters, J.: Passenger flow predictions at sydney international airport: a data-driven queuing approach. arXiv:1508.04839v1 (2015)
49. Kivestu, P.A.: Alternative methods of investigating the time dependent M/G/k queue. Master's thesis, Massachusetts Institute of Technology (1976)
50. Latombe, J.: Robot Motion Planning. Kluwer, Dordrecht (1995)
51. Le Ny, J., Balakrishnan, H.: Feedback control of the National Airspace System. J. Guid. Control. Dyn. **34**(3) (2011)
52. Lee, H., Balakrishnan, H.: A study of tradeoffs in scheduling terminal-area operations. Proc. IEEE **96**(12) (2008)
53. Lee, H., Balakrishnan, H.: Fast-time simulations of Detroit airport operations for evaluating performance in the presence of uncertainties. In: Digital Avionics Systems Conference (DASC). IEEE (2012)
54. Lester, E.A., Hansman, R.J.: Benefits and Incentives for ADS-B Equipage in the National Airspace System. Technical report, MIT (2007)
55. Long, D., Lee, D., Johnson, J., Gaier, E., Kostiuk, P.: Modeling Air Traffic Management Technologies with a Queuing Network Model of the National Airspace System. Technical report, NASA Langley Research Center, Hampton, VA. Technical Report NASA/CR-1999-208988 (1999)
56. Malone, K.M.: Dynamic queueing systems : behavior and approximations for individual queues and for networks. Ph.D. thesis, Massachusetts Institute of Technology (1995)
57. Mao, Z.H., Feron, E., Bilimoria, K.: Stability and performance of intersecting aircraft flows under sequential conflict resolution. IEEE Trans. Intell. Trans. Syst. (2001)

58. Marks, B.L.: Air traffic control separation standards and collision risk. Technical Note 91, Royal Aircraft Establishment (1963)
59. May, G.: A method for predicting the number of near mid-air collisions in a defined airspace. Oper. Res. Q. **22**, 237–251 (1971)
60. Melnyk, R., Schrage, D., Volovoi, V., Jimenez, H.: Sense and avoid requirements for unmanned aircraft systems using a target level of safety approach. Risk Anal. **34**, 1894–1906 (2014)
61. Melnyk, R., Schrage, D., Volovoi, V., Jimenez, H.: A third-party casualty risk model for UAS operations. In: Reliability Engineering and System Safety, vol. 124, pp. 105–116. Elsevier (2014)
62. Menon, P.K., Sweriduk, G.D., Bilimoria, K.D.: New approach for modeling, analysis, and control of air traffic flow. J. Guid. Control. Dyn. **27**(5), 737–744 (2004)
63. Nakahara, A., Reynolds, T., White, T., Dunsky, R.: Analysis of a surface congestion management technique at New York JFK Airport. In: AIAA Aviation Technology, Integration and Operations (ATIO) Conference. Virginia Beach, VA (2011)
64. University of New South Wales, C.: Air traffic simulation—atoms. http://www.cs.adfa.edu.au/research/details2.php?page_id=543 (2015). Accessed Feb 2015
65. Oh, J., Feron, E.: Safety certification of air traffic conflict resolution algorithms involving more than two aircraft. In: American Control Conference, vol. 5, pp. 2807–2811. Philadelphia, PA (1998)
66. O'Neill, M.G., Dumont, J., Hansman, R.: Use of hyperspace trade analyses to evaluate environmental and performance tradeoffs for cruise and approach operations. In: AIAA Aviation Technology, Integration and Operations (ATIO) Conference. IEEE (2012)
67. Federal Aviation Administration telecommunication infrastructure review panel: Report on nov 19, 2009 outage. http://www.faa.gov/air_traffic/publications/media/FTI_Phase1.pdf (2015). Accessed Feb 2015
68. Park, P., Khadilkar, H., Balakrishnan, H., Tomlin, C.: High confidence networked control for next generation air transportation systems. IEEE Trans. Autom. Control. (2014). To appear
69. Park, P., Khadilkar, H., Balakrishnan, H., Tomlin, C.: Hybrid communication protocols and control algorithms for NextGen aircraft arrivals. IEEE Trans. Intell. Trans. Syst. (2014). To appear
70. Pfaender, H., Jimenez, H., Mavris, D.: Effects of technology R&D investments on system level performance. In: Proceedings of the 2013 Aviation Technology, Integration, and Operations Conference. American Institute of Aeronautics and Astronautics (2013)
71. Pujet, N., Delcaire, B., Feron, E.: Input-output modeling and control of the departure process of congested airports. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Portland, OR pp. 1835–1852 (1999)
72. Pyrgiotis, N., Malone, K., Odoni, A.: Modelling delay propagation within an airport network. Trans. Res. Part C: Emerg. Technol. (2011)
73. Ramanujam, V., Balakrishnan, H.: Estimation of Arrival-Departure Capacity Tradeoffs in Multi-Airport Systems. In: Proceedings of the 48th IEEE Conference on Decision Control (2009)
74. Ren, L., Clarke, J.P., Ho, N.T.: Achieving low approach noise without sacrificing capacity. In: Digital Avionics Systems Conference, 2003. DASC'03. The 22nd, vol. 1, pp. 1-E. IEEE (2003)
75. RTCA: Minimum Aviation System Performance Standard for Automatic Dependent Surveillance Broadcast (ADS-B) (2002). DO-242A
76. Samad, T., Balas, G. (eds.): Software-Enabled Control: Information Technology for Dynamical Systems. Wiley, Hoboken (2003)
77. Sampigethaya, K., Poovendran, R., Bushnell, L.: Secure operation, control and maintenance of future e-enabled airplanes. In: Proceedings of the IEEE, Special issue on Aviation Information Systems, vol. 96(12), pp. 1992–2007 (2008)
78. Sampigethaya, K., Poovendran, R., Bushnell, L.: A framework for securing future e-enabled aircraft navigation and surveillance. In: AIAA Infotech at Aerospace Conference. Seattle, WA (2009)

79. Sampigethaya, K., Poovendran, R., Shetty, S., Davis, T., Royalty, C.: Future e-enabled aircraft communications and security: the next 20 years and beyond. Proc. IEEE **99**(11), 2040–2055 (2011)
80. Schaper, M., Tsoukala, G., Stavrati, R., Papadopoulos, N.: Departure flow control through takeoff sequence optimisation: Setup and results of trials at Athens airport. In: 2011 IEEE/AIAA 30th Digital Avionics Systems Conference (DASC), pp. 2B2-1. IEEE (2011)
81. Shumsky, R.: Real-time forecasts of aircraft departure queues. Air Traffic Control. Q. **5**(4) (1997)
82. Shumsky, R.A.: Dynamic statistical models for the prediction of aircraft take-off times. Ph.D. thesis, Massachusetts Institute of Technology (1995)
83. Simaiakis, I.: Analysis, Modeling and Control of the Airport Departure Process. Ph.D. thesis, Massachusetts Institute of Technology (2013)
84. Simaiakis, I., Balakrishnan, H.: Dynamic control of airport departures: Algorithm development and field evaluation. In: American Control Conference (2012)
85. Simaiakis, I., Khadilkar, H., Balakrishnan, H., Reynolds, T.G., Hansman, R.J.: Demonstration of reduced airport congestion through pushback rate control. Trans. Res. Part A: Policy and Pract. **66**, 251–267 (2014)
86. Simaiakis, I., Sandberg, M., Balakrishnan, H.: Dynamic control of airport departures: algorithm development and field evaluation. IEEE Trans. Intell. Trans. Syst. **15**(1), 285–295 (2014)
87. Simpson, R.: Engineering of air traffic control systems. Technical report, Massachusetts Institute of Technology, Flight Transportation Laboratory (1993)
88. Spearman, M., Zazanis, M.: Push and pull production systems: Issues and comparisons. Oper. Res. 521–532 (1992)
89. Stroiney, S., Levy, B., Khadilkar, H., Balakrishnan, H.: Assessing the impacts of the JFK Ground Metering Program. In: IEEE Digital Avionic Systems Conference (DASC) (2013)
90. Sun, D., Bayen, A.: Multicommodity Eulerian-Lagrangian large-capacity cell transmission model for en route traffic. J. Guid. Control. Dyn. **31**(3) (2008)
91. Lozano-Pérez, T.: T., Wesley, M.: An algorithm for planning collision- free paths among polyhedral obstacles. Commun. ACM **22**(10), 560–570 (1979)
92. Tsitsiklis, J.: Neuro-Dynamic Programming. Athena Scientific, Cambridge (1996)
93. Delft, T.U., Planning, D.T.: Why traffic management works. http://www.citg.tudelft.nl/en/about-faculty/departments/transport-and-planning/traffic-management-and-traffic-flow-theory/challenges/why-traffic-management-works/ (2015). Accessed Feb 2015
94. Vranas, P., Bertsimas, D., Odoni, A.: The multi-airport ground-holding problem in air traffic control. Oper. Res. (1994)
95. Waggoner, B.: Developing a risk assessment tool for unmanned aircraft system operations. Ph.D. thesis, University of Washington (2010)
96. Wambsganss, M.: Collaborative decision making through dynamic information transfer. Air traffic Control. Q. (1996)
97. Weibel, R., Hansman, R.: An integrated approach to evaluating risk mitigation measures for UAV operational concepts in the nas. In: Proceedings of InfoTech at Aerospace: advancing contemporary aerospace technologies and their integration, pp. 509–519 (2005)
98. Weiss, J.: Protecting Industrial Control Systems From Electronic Threats. Momentum Press, New York (2010)
99. Wieland, F.: The detailed policy assessment tool (DPAT). In: Proceedings of the Spring INFORMS Conference (1997)

Chapter 5
From Design to Implementation: An Automated, Credible Autocoding Chain for Control Systems

Timothy Wang, Romain Jobredeaux, Heber Herencia, Pierre-Loïc Garoche, Arnaud Dieumegard, Éric Feron and Marc Pantel

5.1 Introduction

A wide range of today's real-time embedded systems, especially their most critical parts, relies on a control-command computation core. The control-command of an aircraft, a satellite, a car engine, is processed into a global loop repeated during the activity of the controlled device. This loop models the acquisition of new input values via sensors, either from environment mesures (wind speed, acceleration, engine RPM, …) or from human feedback through, for example, the brakes, the accelerator, the stick or wheel control.

T. Wang (✉) · R. Jobredeaux · É. Feron
Georgia Institute of Technology, Atlanta, GA, USA
e-mail: timothy.wang@gatech.edu

R. Jobredeaux
e-mail: romain.jobredeaux@gatech.edu

É. Feron
e-mail: feron@gatech.edu

H. Herencia
General Electric Global Research, Clifton Park, NY, USA
e-mail: heber.herencia-zapana@ge.com

P.-L. Garoche
ONERA—The French Aerospace Lab, Toulouse, France
e-mail: pierre-loic.garoche@onera.fr

A. Dieumegard · M. Pantel
ENSEEIHT, Toulouse, France
e-mail: arnaud.dieumegard@enseeiht.fr

M. Pantel
e-mail: marc.pantel@enseeiht.fr

E. Feron (ed.), *Advances in Control System Technology for Aerospace Applications*, Lecture Notes in Control and Information Sciences 460, DOI 10.1007/978-3-662-47694-9_5

The cost of failure of such systems is significant, and examples of such failures are numerous, in spite of increasingly high certification requirements. In addition, as the Federal Aviation Administration (FAA) works on defining the proper frame to open the airspace to Unmanned Aerial Systems (UAS), a major market is about to bloom and will benefit from automated and simple tools to facilitate product certification. Current analysis tools focus mainly on simulations. One obvious shortcoming is the impossibility to simulate all the possible scenarios the system will be subject to. More advanced tools include static analysis modules, which derive properties of the system by formally analyzing its semantics. However, in the specific case of control systems, analyzing the computational core can prove arduous for these tools, whereas the engineers who designed the controller have a variety of mathematical results that can greatly facilite this analysis, and evince more subtle properties of the implemented controller.

There are many modern techniques to analyse software. Model checking is one that endeavors to automatically prove safety-properties of finite-state systems [1]. It is widely used in industry as recent developments have made SAT solvers and SMT solvers much more efficient and scalable [2, 3]. Unfortunately, control software remains subject to an explosion of the state-space, making the use of these techniques difficult for this research.

Abstract interpretation has proven to be a powerful, scalable technique to prove low-level properties of code. It was successfully applied on the Airbus A380 code to prove the absence of runtime errors caused by buffer overflow or index out-of-bound failures [4]. The choice of a proper abstract domain and good widening/narrowing heuristics remains a difficult one. In particular, there is no good lattice structure on the domain of ellipsoids, crucial to many results of control theory. Finally, some control systems require highly non-linear Lyapunov functions in their proof of stability, involving transcendental functions that no current domain encompasses, to our knowledge. Feret's work [5, 6] is a practical approach to the problem of extracting quadratic invariants in an abstract interpretation framework. Its goal is to address the need by Astrée [7] to handle the linear filters present in Airbus' real time software. Previous work [8] by Monniaux addressed the same class of systems but not on actual code. Both of these efforts address a strict subset of the systems we consider in this work.

This chapter, following previous efforts aimed at demonstrating how control-systemic domain knowledge can be leveraged for code analysis [9, 10], describes a practical implementation of a fully automated framework, which enables a control theorist to use familiar tools to generate credible code, that is, code delivered with certificates ensuring certain properties will hold for all executions.

The focus is on a specific class of controllers and properties, in order to achieve full automation; it also explores various possible extensions.

The chapter is structured as follows: We first present a high level view of the general framework in Sect. 5.2. We then proceed to describe how control semantics can be expressed at different levels of design, in Sect. 5.3. Section 5.4 describes the translation process by which graphical synchronous languages familiar to the

control theorist can be turned into credible code. Section 5.12 demonstrates how a proof of correctness can be automatically extracted from the generated code and its annotations.

5.2 Credible Autocoding Framework

Autocoding is an automated programming process that transforms a system expressed in a high-level modeling language such as Simulink or SCADE into a low-level implementation language such as C. In *credible autocoding*, the code is generated along with mathematically verifiable guarantees of functional correctness. The concept of credible autocoding is analogous to credible compilation in [11]. Both processes generate formally verifiable evidences that the output correctly preserves certain semantics of the input. The evidences can be independently checked for correctness by the certification authorities. Unlike credible compilation of Rinard, the formally verifiable evidences of interest in this research are the high-level functional properties of control systems which include stability, robustness and performance. An alternate approach towards producing guarantees for autocoder is building a formally verified autocoder. In a formally verified autocoder, each block transformations are mathematically proved to be correct. This approach is technically challenging and has yet to be demonstrated to be feasible.

Data-flow modeling languages such as Simulink or SCADE are the default industry choice for Model-based development of safety-critical control systems. Within this framework of software development, systems are build using a language of high-level abstraction in order to facilitate rapid design and prototyping. The source code is then generated automatically from the input model using an automated code generation tool or an autocoder. The trustworthiness of the autocoder has often been questioned in the industry [12]. In a data-flow language environment such as Simulink, there are two major elements: “blocks”, and “lines.” The blocks are functions that perform some operations on its input(s) and then output the result(s). The lines are directed edges that flow from an origin block’s output to a destination block’s input. This type of connectivity specifies that the origin output is equivalent to the destination input. The blocks are organized into sets of blocks, forming a library of blocks. Some of the blocks have unpredictable variations in their semantics. For example, the precise semantics of the Simulink product block depends on its input types, input dimensions, number of input/output, product operator selected, etc. The variations in Simulink block semantics, and the lack of formal documentation about them, present an obstacle in its wide adoption for safety-critical production. A related work [13], which is complementary to this research used a model-based approach, to assign provably correct semantics to a set of Simulink blocks. The result of that research is a library of trustworthy blocks—the *BlockLibrary* language—with precise semantics, that can be reasoned about formally.

In the framework of credible autocoding, instead of proving that individual block transformations are correct i.e. building the library of trustworthy blocks, the goal is to

be able to show that the output code also satisfies the high-level functional properties of the input model. The functional properties of the input model is dependent on the domain of the input model. In the domain of control systems, a strong functional property is an exponential stability of the closed-loop system with a global domain of attraction, and a weaker functional property is simply the boundedness of the system. In the domain of convex optimization, an example property is the linear convergence of the duality gap function to zero. The verification of the code against high-level functional property imparts an additional layer of guarantee on the correctness of the code. For example, if a gain in a Simulink model was inverted accidentally before autocoding, the output code while correct in the sense of each individual block transformations, is not likely to satisfy a pre-computed property such as the Lyapunov stability of the system.

The complete credible autocoding process from Simulink model to verified code, for the domain of control systems, is illustrated in Fig. 5.1. In this process, the credible autocoding portion (left half of Fig. 5.1) is performed by the code producers, who generate the code and provide evidence that the generated code is correct. The proof-checking portion (right half of Fig. 5.1) are performed by the certification authorities who are independent from the code producers. The only shared knowledge between these two parties is a common language used to express the mathematical proofs on the code. The *control semantics* include stability property of the system and the plant models used in the stability analysis for the closed-loop cases. The framework adds, on top of a classic model-based development cycle, another layer of translations, that converts quadratic invariant sets, computed using Lyapunov-based methods, into code annotations on the code, which form a proof of the correctness of the output code.

For credible autocoding of control software, compared against the traditional model-based development, the only additional requirement on control engineers is

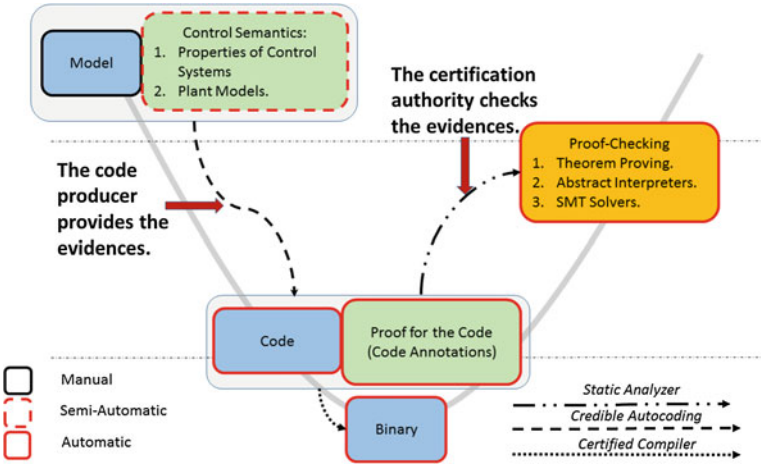

Fig. 5.1 Automated credible autocoding/compilation chain for control systems

that they provide the Lyapunov function. The procedure for generating Lyapunov-type certificate of stability and performance properties of control systems can be automated using Linear Matrix Inequalities [14] (LMIs) and the Integral Quadratic Constraint [15] (IQC) framework. Each of the stability and performance properties generated can be encoded using an ellipsoid invariant, which can then be transformed into an invariant for the code.

The advantages of the framework developed in this chapter can be summarized as follows:

1. All the advantages of model-based development are inherited.
2. The correctness of the autocoder is more or less guaranteed by the correctness of its output code. This is based on the idea of credible compilation in [11]. This reduces the need to formally verify the autocoder.
3. The framework provides guarantees are of high-level functional properties, which provides a potentially more useful characterization of the correctness of the system to the certification authorities.
4. The framework can provide feedback information to the domain expert so errors in the construction of the model could be diagnosed more rapidly.
5. In the context of certification, credible autocoding could potentially reduce the number of tests required for certification of the control software. In traditional unit testing, a piece of code, such as the control software is tested repeatedly for many possible different inputs and scenarios. This is extremely time consuming. The credible autocoding framework enables a meta-testing procedures, in which the software, are tested for all possible inputs and scenarios, in one iteration.

5.2.1 Input and Output Languages of the Framework

The input language of the framework should be a graphical data-flow modeling language such as Simulink that is familiar to control engineers. The exact choice for the input language is up to the domain users' preference and does not affect the utility of the framework as it can be eventually adapted to other modeling languages such as SCADE [16]. For the prototype tool-chain developed in this research, the choice of the input language is a discrete-time subset of Simulink blocks. The subset of Simulink language include basic blocks: unit delays, gain, input, output, plus, minus, multiplication, divide, min, and max.

Likewise, for the output language, the choice is likely to depend on the preferences of the industry and the certification authority. For the experimental prototype described in this chapter, the output language was chosen to be C because of its industrial popularity and the wide availability of static analyzers tailored for C code.

The set of annotations in the output source code contains both the functional properties inserted by the domain expert and the proofs, which can be used to automatically prove these properties. For the analysis of the annotated output, a prototype annotation checker that is based on the static analyzer frama-C and the theorem prover

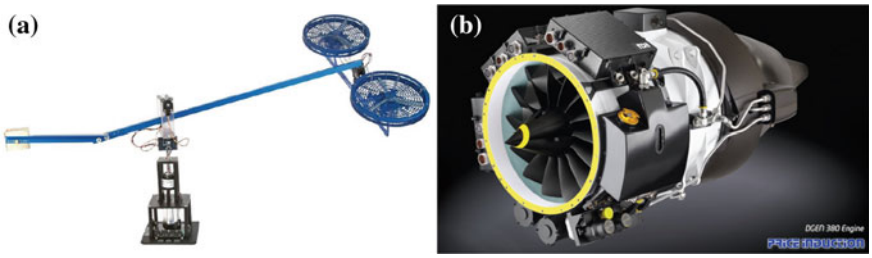

Fig. 5.2 Test platforms. **a** Quanser helicopter (© Quanser), **b** DGEN 380 lightweight turbofan engine (© Price Induction)

PVS is built. For automating the proof-checking of the annotated output, a set of linear algebra definitions and theories were integrated into the standard NASA PVS library [10].

In this chapter, the fully automated process from the input model to the verified output is showcased for the property of close-loop stability. At this point, we restrict the input space to only linear controllers with possible saturations in the loop. The running example that we use in this paper is described by the state-space difference equation in Example 5.1. This example is chosen because it has enough complexity to be representative of many controllers used in the industry, and is simple enough such that we can show in this paper, the output annotated code. The example system is consisted of states $x \in \mathbb{R}^2$, input $y \in \mathbb{R}$, output $u \in \mathbb{R}$, the state-transition function in (5.1), and the output function in (5.2).

Example 5.1

$$x_+ = \begin{bmatrix} 0.4990 & -0.05 \\ 0.01 & 1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} y \tag{5.1}$$

$$u = \begin{bmatrix} 564.48 & 0 \end{bmatrix} x + \begin{bmatrix} 1280 \end{bmatrix} y. \tag{5.2}$$

In addition to the two dimensional example used in this chapter, we have worked with several larger systems shown in Fig. 5.2, which include the Quanser 3-degree-of-freedom Helicopter, and a FADEC control system of a small twin jet turbofan engine [17]. The state-space size of the engine FADEC controller is 11.

5.3 Control Semantics

The set of control semantics that we can express on the model includes stability and boundedness.

Remark 5.1 The types of systems in which we can express closed-loop stability properties for are not just limited to simple linear systems like Example 5.1. They

also include nonlinear systems that can be modeled as linear systems with bounded nonlinearities in the feedback loops.

5.3.1 Control System Stability and Boundedness

A simple linear system such as Example 5.1 is commonly represented using the following state-space formalism. For matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{k\times n}$, and $D \in \mathbb{R}^{k\times m}$, we have

$$\begin{aligned} x_+ &= Ax + By \\ u &= Cx + Dy \end{aligned} \tag{5.3}$$

This state-space system has the state-transition function $\delta : (x, y) \to Ax + By$ and the output function $\omega : (x, y) \to Cx + Dy$. We also consider linear systems with bounded nonlinearities in their feedback interconnections. This model is a closer representation of the real control systems when there are saturations, time-delays, anti-windup mechanisms, hysteresis, or noise in the loop. For $\tilde{y} = \sigma(t, y)$ with the time-varying nonlinear function $\sigma(t, y)$, in which $\tilde{y}_i \leq M_i y$ for $M_i > 0$, we have

$$\begin{aligned} x_+ &= Ax + B\tilde{y} \\ u &= Cx + D\tilde{y} \end{aligned} \tag{5.4}$$

Analogous to system (5.3), the state-transition function for (5.4) is $\delta : x, y \to Ax + B\sigma(t, y)$, and the output function is $\omega : (x, y) \to Cx + D\sigma(t, y)$. For any systems described by (5.3) and (5.4), we can compute efficiently the answer to the following problem.

Problem 5.1 1. Does there exist a symmetric matrix $P \in \mathbb{R}^{n\times n}$ such that the quadratic function $q : x \to x^{\mathrm{T}}Px$ is non-increasing along the system trajectories as $t \to +\infty$?

Generally speaking such a problem can be transformed into a linear matrix inequality (LMI). The details of these transformations are skipped here as they are well-established in the control literature [14, 18]. The algorithms used to solve LMIs are based on semi-definite programming, which is tractable up to large sizes [19].

5.3.2 Prototype Tool-Chain

In this chapter, we describe a prototype tool-chain, which has been developed for the demonstration of credible autocoding. The prototype tool-chain is split into a credible autocoder front-end and a proof-checker back-end. The credible autocoder front-end translates the model into annotated code. The proof-checking back-end

analyzes the annotated code produce by the front-end and decides whether or not the proof is coherent.

Gene-Auto [20–23] is an existing, open-source, autocoding prototype for embedded systems. The front-end prototype in our tool-chain, which we dubbed Gene-Auto+, is based on Gene-Auto. For the front-end, we have several extensions to the language or language environments Simulink, Gene-Auto and ACSL. ACSL [24] is a formal specification language for C programs. More details on ACSL is described in Sect. 5.3.7. The language extensions are summarized as follows:

1. A library of Annotation blocks in Simulink and Gene-Auto.
2. An ACSL-like language within Gene-Auto.
3. Abstract types and their operators in ACSL: matrix, vector and quadratic predicates.

The language extensions in Simulink and Gene-Auto are described in Sect. 5.3.3. The language extensions in ACSL are described in Sect. 5.3.7.

5.3.3 Control Semantics in Simulink and Gene-Auto

An observer (see [25]) in Simulink takes an input signal and returns an output of 1 if the input satisfies a specific property, and 0 if otherwise. Both boundedness and stability can be expressed, for example, using an observer with inputs x_i, $i = 1, \ldots, n$, and the boolean-valued function

$$x \to \sum_{i,j=1,\ldots,n} x_i P_{ij} x_j \leq 1. \tag{5.5}$$

To express the types of observers in (5.5) as annotations on the Simulink model, we extended the Simulink language and the Gene-Auto environment with a set of annotation blocks.

Annotation blocks are structurally the same as any other Simulink blocks. The key difference is that they do not translate into code. Our prototype annotation block library has been built to contain a minimal set of blocks needed to express the properties of control systems that are currently verifiable from Simulink to C code.

The prototype annotation block library contains four symbols: *vamux*, *constant*, *quadratic*, and *system*. Each annotation symbol denotes an annotation block type, To illustrate the annotations blocks, we have Fig. 5.3, which shows a Simulink model of an engine controller, along with 6 annotation blocks. The annotation blocks are highlighted in red for the purpose of clarity.

In Simulink, the *vamux* block type takes n scalar or vector inputs x_i, and outputs a concatenated signal $y = \left[x_1^\mathsf{T}, \ldots, x_n^\mathsf{T}\right]^\mathsf{T}$. In Fig. 5.3, there are three *vamux* blocks, labeled as *nh*, $xc(t)$ and $yd(t)$. The *vamux* block type only accepts one parameter, which determines the number of inputs to the block type. The *vamux* block does not express any property of the system. In Gene-Auto+, the main functionality of the

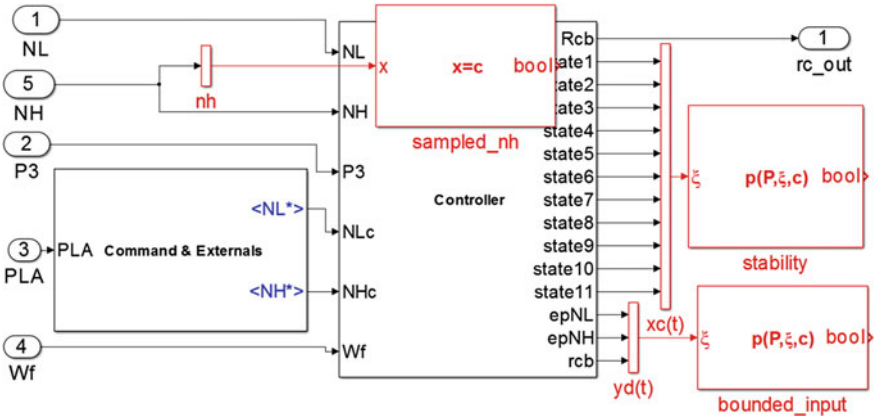

Fig. 5.3 Simulink model with annotation blocks

vamux block is to establish equivalence relations between its inputs and the *i*th entry of its output. i.e. $x_i == y_i$. This enables the prototype to replace the pseudo-variables in the templates within the other annotation blocks with the actual variables from the code.

The *constant* block type accepts one scalar, vector, or matrix input x, and a constant matrix parameter $[c_1]$ or $[c_1, \ldots, c_n]$ for $n \in \mathbb{N}$. The type of the constants c_i are constrained to be the same type as the input x. The output of the block is the boolean value $x == c_1$ or $\bigvee_{i=1}^{n} (x == c_i)$, which implies n sets of *behaviors* for the code.

Definition 5.1 A behavior is a set of unique assumptions on the parameters and input, and output of the model.

This block type is useful for expressing the semantics of parameter varying systems such as a gain-scheduled controller. For example, the scheduling parameter of the controller in Fig. 5.3 is the input *NH*, which is annotated with a *constant* block labeled *sampled_nh*. In Gene-Auto+, the *constant* block type generates a set of assumption(s) $\{x == c_i\}$, $i = 1, \ldots, n$.

The *quadratic* block type accepts one input vector of n variables x, a matrix parameter $P \in \mathbb{S}^{n \times n}$, a logic connective symbol $\diamond \in \{<=, <, >, ==\}$, a level-set constant $c \in \mathbb{R}$, and outputs the boolean value of $x^{\mathsf{T}} P x \diamond c$. The *quadratic* block type can be used, for example, to express ellipsoidal invariant sets, sector-bound inequalities, 2-norm squared, sum of squares polynomial sets, etc. The *quadratic* block also accepts a positive scalar parameter *mu*. This is used to indicate the multiplier computed in stability analysis. The *quadratic* block type behaves like an ellipsoid observer from (5.5) in Simulink. In Gene-Auto+, the *quadratic* block type generates a predicate defined on its inputs: $\forall x . x^{\mathsf{T}} P x \leq c$. For example, in Fig. 5.3, the *quadratic* block labeled *stability* represents a claim that $xc(t)$ does not violate a quadratic constraint. The other *quadratic* block *bounded_input* is used to express a bound on the input $yd(t)$ to the controller.

The *system* block type is parameterized by 4 matrices A, B, C, and D. An example of a Simulink model annotated with the *system* block can be found in Sect. 5.3.5. The *system* block type accepts two vector inputs u and y. The output of the *system* block type is the state x of the dynamical system

$$\begin{aligned} x_+ &= Ax + Bu, \quad x(0) = x_0 \\ y &= Cx + Du. \end{aligned} \tag{5.6}$$

The semantics of the *system* block in Gene-Auto include the semantics of the discrete-time linear state-space system in (5.6), and a set of relations $\{\tilde{y}_i == y_i, u_i = \tilde{u}_i\}$ that establish equivalence between the annotation variables y and u and their corresponding variables $\tilde{y}$ and $\tilde{u}$ from the controller model. The *system* block type can be used, for example, to express a model of the plant the controller is expected to interact with. The same controller model can be annotated with multiple *system* blocks, which results in multiple sets of predicates for the code, which can be annotated using the *behavior* keyword from ACSL.

5.3.4 Annotation Blocks and Behaviors in the Model

In a model, multiple *system* blocks $s_1, \ldots, s_n$ can be connected to the same set of *vamux* blocks. This results in a set of n behaviors expressed by the formula $\bigvee_i^n s_i$. If there are n *system* blocks connected to the controller model, then there are n behaviors in the model.

If there are also k *constant* blocks in the model, each connected to a different *vamux* block, and each with m behaviors, then we have a total of m^k behaviors resulting from the *constant* blocks: $\bigwedge_i^k \left(\bigvee_i^m c_i\right)$. The complete set of behaviors in the model resulting from both the *system* and *constant* blocks is described by the formula

$$\left(\bigwedge_i^k \left(\bigvee_i^m c_i\right)\right) \wedge \left(\bigvee_i^n s_i\right) \tag{5.7}$$

or a total of nm^k possible behaviors.

Lastly, if there are w *quadratic* blocks in the model as well, and all of them are connected to the same set of *vamux* block, then we have w number of behaviors $\vee_i^w q_i$ due to the *quadratic* blocks. Combining this set of behaviors conjunctively with the set of behaviors generated by the *system* and *constant* blocks results in

$$\left(\bigwedge_i^k \left(\bigvee_i^m c_i\right)\right) \wedge \left(\bigvee_i^n s_i\right) \wedge \left(\bigvee_i^w q_i\right) \tag{5.8}$$

for a possible total of wnm^k behaviors in the model. However, each of the *quadratic* blocks that encode an inductive property such as stability are typically assigned a

behavior generated by a *system* block. This is true for many examples, in which the quadratic invariant is computed based on some plant model, using independent LMI-based tools. For example, if there are n quadratic invariants and each is assigned a behavior from a *system* block, then there are only n behaviors in the model:

$$\bigvee_{i}^{n} (s_i \wedge q_i) \,. \tag{5.9}$$

This is far less than the explosion in the number of of behaviors predicted by (5.8).

Next, some annotated examples are given. Each example contains a different possible set of control semantics.

5.3.5 *Closed-Loop Stability with Bounded Input*

For the running example, the closed-loop stability of the system with bounded input is expressed with a set of the *system* and *quadratic* blocks. For example, as displayed in Fig. 5.4, the close-loop stability of the Simulink model of the control systems is expressed using:

1. a *quadratic* block *stability* to express the ellipsoidal invariant set that encodes the closed-loop stability of the system.
2. another *quadratic* block *bounded_input* to express a 2-norm bound on the input,
3. a *system* block *plant*, which expresses the discrete-time linear state-space system used in the closed-loop stability analysis

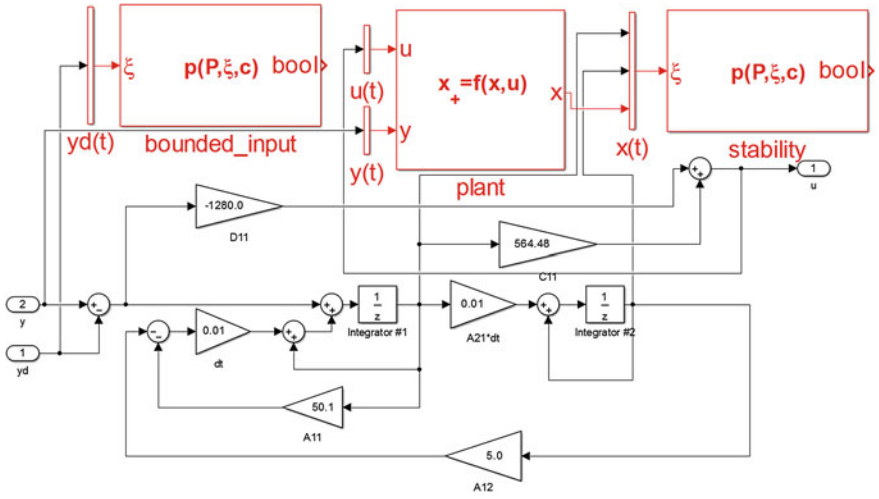

Fig. 5.4 Control system model annotated with control semantics

5.3.6 *Expressing the Observer-Based Fault-Detection Semantics*

In an observer-based fault-detection system, the dynamics of the observer are designed such that the output of the observer changes due to specific faults in the plant. Once the change exceeds a certain pre-defined threshold, the system is said to be in the faulty mode. To express the faulty and nominal behavior of a fault-detection system, one can use two different *system* blocks. One *system* block is the model of the faulty plant that is predicted to trigger the faulty mode and the other is the nominal plant. This is displayed in Fig. 5.5. The *quadratic* blocks connected to the *vamux* blocks $xf(t)$ and $xn(t)$ express the closed-loop stability of the system. They are assigned behaviors based on their physical connections to the *system* block. For example, as displayed in Fig. 5.5, the block *cl_faulty* is connected to the *system* block *quanser_faulty* using the *vamux* block $xf(t)$. The two *quadratic* blocks

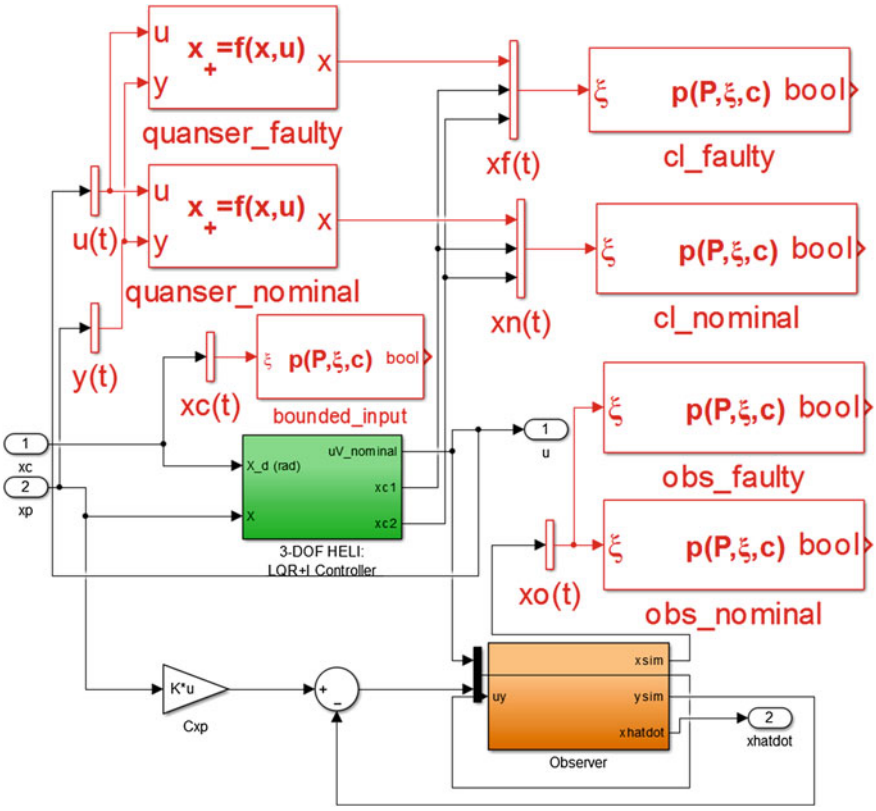

Fig. 5.5 Expressing multiple behaviors: fault-detection system

connected to the *vamux* block $xo(t)$ are used to express the stability of the observer dynamics. They are assigned the behaviors *faulty* and *nominal*, based on the labels in their names.

5.3.7 Control Semantics at the Level of the C Code

For the specific problem of open loop stability, the expressiveness needed at the C code level is twofold. On the one hand, one needs to express that a vector composed of program variables belongs to an ellipsoid. This entails a number of underlying linear algebra concepts. On the other hand, one needs to provide the static analysis tools with indications on how to proceed with the proof of correctness.

The ANSI/ISO C Specification Language (ACSL), is an annotation language for C [24]. It is expressive enough to fulfill our needs, and its associated verification tool, Frama-C [26], offers a wide variety of back-end provers that can be used to establish the correctness of the annotated code.

5.3.7.1 Linear Algebra in ACSL

A library of ACSL symbols has been developed to express concepts and properties pertaining to linear algebra. In particular, types have been defined for matrices and vectors, and predicates expressing that a vector of variables is a member of the ellipsoid $\mathscr{E}_P$ defined by $\{x \in \mathbb{R}^n : x^{\mathrm{T}} P x \leq 1\}$, or the ellipsoid $\mathscr{G}_X$ defined by $\left\{x \in \mathbb{R}^n : \begin{bmatrix} 1 & x^{\mathrm{T}} \\ x & X \end{bmatrix} \geq 0\right\}$. For example, expressing that the vector composed of program variables v_1 and v_2 is in the set $\mathscr{E}_P$ where $P = \begin{pmatrix} 1.53 & 10.0 \\ 10.0 & 507 \end{pmatrix}$, can be done with our ACSL extensions using the annotations in Fig. 5.6.

The invariance of ellipsoid $\mathscr{E}_P$ throughout any program execution can be expressed by the *loop invariant* in Fig. 5.7. This annotation expresses that before and after every execution of the loop, the property $\begin{bmatrix} v_1 & v_2 \end{bmatrix}^{\mathrm{T}} \in \mathscr{E}_P$ will hold. In terms of expressiveness, it is all that is required to express open loop stability of a linear controller. However, in order to facilitate the proof, intermediate annotations are added within the loop to propagate the ellipsoid through the different variable assignments, as suggested in [9] and expanded on in Sect. 5.4. For this reason, a loop body instruction can be annotated with a local contract, as in Fig. 5.8.

```
1 /*@ logic matrix P = mat_of_2x2_scalar(1.53,10.0,10.0,507);
2   @ assert in_ellipsoid(P,vect_of_2_scalar(v_1,v_2)); */
```

Fig. 5.6 Asserting that a vector with components v_1 and v_2 belongs to ellipsoid $\mathscr{E}_P$ in ACSL

```
1 //@ loop invariant in_ellipsoid(P,
2 //@                              vect_of_2_scalar(v_1,v_2));
3 while (true){
4   //loop body
5 }
```

Fig. 5.7 Expressing the invariance of $\mathscr{E}_P$ on a loop in ACSL

```
1 /*@ requires in_ellipsoid(P,vect_of_2_scalar(v_1,v_2));
2   @ ensures in_ellipsoid(Q,vect_of_3_scalar(v_1,v_2,v_3));*/
3 {
4 // assignment of v_3
5 }
```

Fig. 5.8 A local contract to assist the proof process

```
1 /*@ requires in_ellipsoid(P,vect_of_2_scalar(v_1,v_2));
2   @ ensures in_ellipsoid(Q,vect_of_3_scalar(v_1,v_2,v_3));
3   @ PROOF_TACTIC (use_strategy (AffineEllipsoid));*/
4 {
5 // assignment of v_3
6 }
```

Fig. 5.9 Adding proof tactics to a contract to guide the proof back-end

5.3.7.2 Including Proof Elements

An extension to ACSL, as well as a plugin to Frama-C, have been developed. They make it possible to indicate the proof steps needed to show the correctness of a contract, by adding extra annotations. For example, the syntax in Fig. 5.9 signals Frama-C to use the strategy `AffineEllipsoid` to prove the correctness of the local contract considered. Section 5.12 expands on this topic.

5.3.8 Closed Loop Semantics

In order to express properties pertaining to the closed loop behavior of the system, one needs to introduce a model for the plant, to be able to refer to the plant variables. The most accurate way to do so would require a hybrid system representation, given that the plant is commonly a continuous system, while the digital controller is a discrete one. A large body of work is devoted to proving meaningful properties of hybrid systems. In order to obtain actionable results, on which proof can be carried out, we made the choice of representing the plant as a linear system, discretized at the same period as the controller. To achieve this, we use ACSL's ghost code feature.

Ghost code is a way to introduce variables and operations on these variables without affecting the semantics of the code. Any valid C code can be written in ghost code as long it does not introduce a change in the actual variables.

At the end of the control loop, we use these variables to express the state update of the plant that results from the computed control signal value. We also enforce axiomatically the fact that the input read from the sensors equals the output of the plant.

For each state variable in the plant, we introduce a global ghost variable. Within the update function of the controller, we introduce ghost code describing the state update resulting from the control output. A template of the structure of the code is given in Fig. 5.10.

5.3.9 Control Semantics in PVS

Through a process described in Sect. 5.12, verifying the correctness of the annotated C code is done with the help of the interactive theorem prover PVS. This type of prover normally relies on a human in the loop to provide the basic steps required to prove a theorem. In order to reason about control systems, linear algebra theories have been developed. General properties of vectors and matrices, as well as theorems specific to this endeavor have been written and proven manually within the PVS environment [10].

5.3.9.1 Basic Types and Theories

Introduced in [10] and available online[1] as part of the larger NASA PVS library, the PVS linear algebra library allows one to reason about matrix and vector quantities, by defining relevant types, operators and predicates, and proving major properties. To name a few, we have defined:

- A vector type.
- A matrix type, along with all operations relative to the algebra of matrices.
- Various matrix subtypes such as square, symmetric and positive definite matrices.
- Block matrices
- Determinants
- High level results such as the link between Schur's complement and positive definiteness

5.3.9.2 Theorems Specific to Control Theory

In [10], a theorem was introduced, named the ellipsoid theorem. A stronger version of this theorem, along with a couple other useful results in proving open loop stability of a controller, have been added to the library. The theorem in Fig. 5.11 expresses in

[1] http://shemesh.larc.nasa.gov/fm/ftp/larc/PVS-library/.

```
/*@     ghost REAL xp_0;
  @     ghost REAL xp_1;
  @     ...
  @     ghost REAL xp_0_tmp;
  @     ghost REAL xp_1_tmp;
  @     ...*/
/*@ requires in_ellipsoidQ(Q,vect_of_n_scalar(xc[0],
                                          xc[1],...,xp_0,xp_1
                                              ,...));
  @ ensures in_ellipsoidQ(Q,vect_of_n_scalar(xc[0],
                                          xc[1],...,xp_0,xp_1
                                              ,...)); */
void update_fun(t_example_io *_io_, t_example_state *xc){
...
/*@ requires pre_i
  @ ensures post_i
  @ PROOF_TACTIC (use_strategy ( strategy_i ) )*/
{
instruction i;
}
...
/*@          behavior Plant_N:
             requires in_ellipsoidQ(QMat_N,vect_of_m_scalar(xc[0],
                 xc[1],
                                          ...,xp_0,xp_1,...,u));
             ensures in_ellipsoidQ(QMat_{N+1},vect_of_n_scalar(xc
                 [0],xc[1],
                                            ...,xp_0,xp_1,...));
             @ PROOF_TACTIC (use_strategy (AffineEllipsoid));*/
{
    /*@
             ghost xp_0_tmp = xp_0;;
             ghost xp_1_tmp = xp_1;;
             ... */
    /*@
             ghost xp_0 = a_11 * xp_0_tmp + a_12*xp_1_tmp +.. + b1
                 *u;;
             ghost xp_1 = a_21 * xp_0_tmp + a_22*xp_1_tmp +.. + b2
                 *u;;
             ...*/
}
    /*@ behavior Plant_{N+1}:
             requires in_ellipsoidQ(QMat_{N+1},vect_of_n_scalar(xc
                 [0],xc[1],
                                              ...,xp_0,xp_1,...))
                                                ;
             ensures in_ellipsoidQ(Q,vect_of_n_scalar(xc[0],xc[1],
                                       ...,xp_0,xp_1,...));
             @ PROOF_TACTIC (use_strategy (PosDef));
    */
{
}
}
```

Fig. 5.10 Template of the update function with added plant semantics in ghost code. Note that often, the ghost code and the annotations are much larger than the code actually executed

PVS
```
ellipsoid_general: THEOREM
 ∀ (n:posnat,m:posnat, Q:SquareMat(n),
           M: Mat(m,n), x:Vector[n], y:Vector[m]):
              in_ellipsoid_Q?(n,Q,x)
              AND y = M*x
         IMPLIES
         in_ellipsoid_Q?(m,M*Q*transpose(M),y)
```

Fig. 5.11 Affine ellipsoid transformation theorem in PVS

PVS
```
ellipsoid_combination: THEOREM
 ∀ (n,m:posnat, lambda_1, lambda_2: posreal, Q_1: Mat(n,n),
       Q_2: Mat(m,m), x:Vector[n], y:Vector[m], z:Vector[m+n]):
              in_ellipsoid_Q?(n,Q_1,x)
              AND in_ellipsoid_Q?(m,Q_2,y)
              AND lambda_1+ lambda_2 <= 1
              AND z = Block2V(V2Block(n,m)(x,y))
  IMPLIES
  in_ellipsoid_Q?(n+m,Block2M(M2Block(n,m,n,m)(1/lambda_1*Q_1,
                  Zero_mat(m,n),Zero_mat(n,m),1/lambda_2*Q_2)),z)
```

Fig. 5.12 Ellipsoid combination through S procedure theorem in PVS

the PVS syntax how a generic ellipsoid $\mathcal{G}_Q$ is transformed into $\mathcal{G}_{MQM^T}$ by the linear mapping $x \mapsto Mx$.

The theorem in Fig. 5.12: expresses how, given 2 vectors x and y in 2 ellipsoids $\mathcal{G}_{Q_1}$ and $\mathcal{G}_{Q_2}$, and multipliers $\lambda_1, \lambda_2 > 0$, such that $\lambda_1 + \lambda_2 \leq 1$, it can always be said that $\begin{pmatrix} x \\ y \end{pmatrix} \in \mathcal{G}_Q$, where $Q = \begin{pmatrix} \frac{Q_1}{\lambda_1} & 0 \\ 0 & \frac{Q_2}{\lambda_2} \end{pmatrix}$

These 2 theorems are used heavily in Sect. 5.12 to prove the correctness of a given Hoare triple. While they are not particularly novel, their proof in PVS was no trivial process and required close to 10000 manual proof steps from the authors.

5.4 Autocoding with Control Semantics

The translation process in the credible autocoding prototype Gene-Auto+, is now described in more details with a demonstration on the running example. From the input model to the verified output, the property of open-loop and closed-loop stability for a linear system with a nonlinear, but bounded input is expressed.

5.5 Building the Input Model

The model with the closed-loop stability semantics is already displayed in Fig. 5.4. The model expressing open-loop stability is displayed in Fig. 5.13.

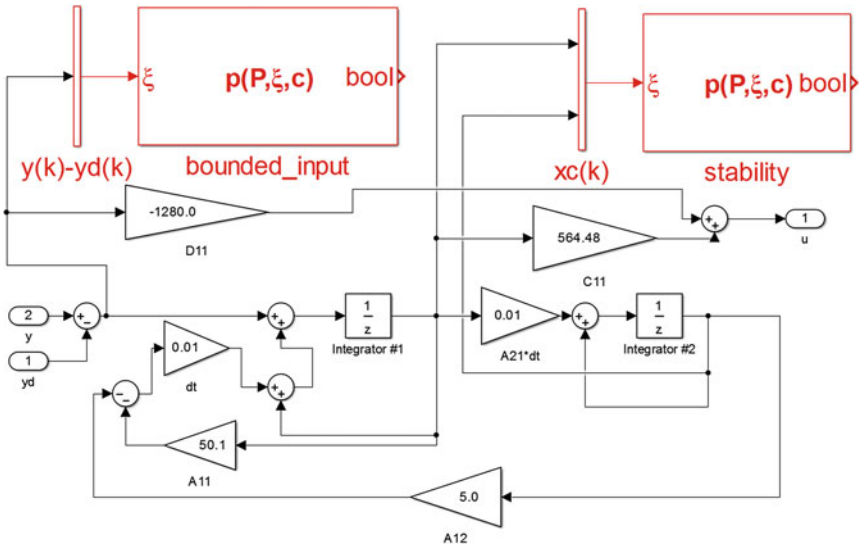

Fig. 5.13 Open-loop stability

In either case, an assumption of boundedness is made on the input to the model and it is expressed by the *quadratic* block *bounded_input*. For the closed-loop case, the assumption of boundedness is made on the signal y_d (see Fig. 5.4). For the open-loop case, a similar boundedness assumption is made on the signal $y - y_d$ (see Fig. 5.13). The closed-loop quadratic invariant, expressing stability, is defined by the multiplier $mu = 0.991$, and $P \succ 0$,

$$P = \begin{bmatrix} 0.1878 & 0.1258 & -0.0813 & 0.0149 \\ 0.1258 & 0.3757 & -0.0220 & 0.0100 \\ -0.0813 & -0.0220 & 0.0660 & -0.0063 \\ 0.0149 & 0.0100 & -0.0063 & 0.0012 \end{bmatrix}. \tag{5.10}$$

Likewise, the stability analysis is also done for the open-loop case. The quadratic invariants are inserted into their respective Simulink model using *quadratic* blocks. Both of them are labeled as *stability*.

5.6 Basics of Program Verification

In the translation process, we use several notions from formal program verification. First we have the following predicate notations for the annotations expressed in this section. The ellipsoid sets are denoted using one of the following symbols:

$$\begin{aligned} p(P,x,c) &\triangleq \left\{x \in \mathbb{R}^n \mid x^{\mathsf{T}} P x \leq c\right\} \\ q(Q,x,c) &\triangleq \left\{x \in \mathbb{R}^n \mid \begin{bmatrix} c & x^{\mathsf{T}} \\ x & Q \end{bmatrix} \succ 0\right\}. \end{aligned} \tag{5.11}$$

Without loss of generality, the sublevel set parameter c is set to 1 unless described otherwise.

The control semantics are translated into axiomatic semantics on the code. Axiomatic semantics is one of several approaches in theoretical computer science to assign mathematical meanings to a program [27]. In axiomatic semantics, the semantics or mathematical meanings of a program are defined using the logic predicates that hold before the execution of the code and the ones that hold the execution of the code. The main structure of axiomatic semantics is a *Hoare triple* [28].

Definition 5.2 A Hoare triple is the 3-tuple $(\{P\}, C, \{Q\})$, in which P is a predicate or a set defined by a formula in some logic, and Q is also another predicate, and C denotes a block of code.

The symbol P denotes a post-condition and the symbol Q denotes a pre-condition.

Definition 5.3 A Hoare triple $\{P\}, C, \{Q\}$ is interpreted to be *partially correct*, if P holds before the execution of C, and Q holds after the execution of C.

Remark 1 The termination of C needs to be proved for correctness. For the rest of this chapter, correctness refers to the notion of partial correctness.

The pre and post-conditions are expressed on the code as comments before and after the block of code. For example, given the simple `while` program in Fig. 5.14, If the statement $|x| \leq 1$ holds before the execution of the loop, then it should hold for all executions of the loop.

Definition 5.4 An *invariant* is a predicate that holds for all executions of the loop.

The statement $|x| \leq 1$ is an invariant. It can be inserted into the code as both the pre-condition and the post-condition, see the ACSL comments in Fig. 5.14. The Hoare triple in Fig. 5.14, therefore is $\{|x| \leq 1\}$ `while` a do C end $\{|x| \leq 1\}$.

Here we give an illustration of Hoare triples on a Matlab implementation of $x_+ = Ax+By$ (see Fig. 5.15). A Matlab example is used in this section and further on only for the sake of brevity and clarity. In practice, a language like C is the typical choice for the implementation of real-time control systems. We assume the Matlab example satisfies

```
1 // abs(x)<=1;
2 while (x*x>0.5)  {
3   x=0.9*x;
4 }
5 // abs(x)<=1;
```

Fig. 5.14 A `while` program in C

```
% x'*P*x<=1;
while (t<5000)
  u=C*x+D*y;
  x=A*x+B*y;
end
% x'*P*x<=1;
```

Fig. 5.15 Annotated lead/lag compensator in matlab

some ellipsoidal invariant $p(P, x, 1)$, computed from a stability analysis [14]. The invariance of $p(P, x, 1)$, which is the property of interest, is translated into axiomatic semantics for the Matlab code. This is done by translating the set $q(P, x, 1)$, using type matching, into the Matlab formula `x*P*x<=1` and then inserting that formula as pre and post-condition for the program. The result is the annotated Matlab program in Fig. 5.15. Next, the basics of deductive program verification are described.

5.6.1 Hoare Logic and Deductive Verification

Hoare logic is a formal proof system that comes with a set of axioms and inference rules for reasoning about the correctness of Hoare triples on various structures of an imperative programming language i.e. `if-else` statements, `assignment` statements, `while` statements, `for` statements, *empty* statements, etc.

For example, an axiom in Hoare logic for the `while` program construct is

$$\frac{\{P \wedge a\}\, C;\, \{P\}}{\{P\}\, \texttt{while}\ a\ \text{do}\ C\ \text{end}\ \{\neg a \wedge P\}}. \tag{5.12}$$

Syntactically speaking, the axioms and inference rules can be interpreted as follows: the formula above the horizontal line implies the formula below that line. In the `while` axiom in (5.12), note that pre and post-conditions of the loop has to be same formula. This means to verify program loops, an invariant is necessary. Some of the basic inferences rules for reasoning about imperative programs using Hoare logic are listed in Table 5.1. The consequence rule in (5.13) is useful whenever a stronger pre-condition or weaker post-condition is needed. The term stronger here means the set defined by the predicate is smaller. The term weaker means precisely the opposite. The substitution rule in (5.16) are used when the code is an *assignment* statement. The weakest pre-condition expression $P[x/expr]$ in (5.16) means P with all free occurrences of the expression *expr* replaced by x. For example, given a post-condition `y<=1` for the line of code `y=x+1`, one can deduct that `x+1<=1` is a weakest pre-condition using the backward substitution rule in (5.16). The skip rule in (5.15) can be used when the executing piece of code does not change any variables in the pre and post-conditions.

Table 5.1 Hoare logic inference rules for a imperative language

$$\frac{\{P_1 \Longrightarrow P_2\}C\{Q_1 \Longrightarrow Q_2\}}{\{P_1\}C\{Q_2\}} \quad (5.13) \qquad \frac{\{P\}C_1\{R\};\{R\}C_2\{Q\}}{\{P\}C_1;C_2\{Q\}} \quad (5.14)$$

$$\overline{\{P\}SKIP\{P\}} \quad (5.15) \qquad \overline{\{P[e/x]\}x{:=}expr\{P\}} \quad (5.16)$$

1. $\{p(P,x,1)\}$ `while` a do C end $\{p(P,x,1)\}$.
2. $\{p(P,x,1)\}C\{p(P,x,1)\}$ by the `while` axiom in (5.12).
3. $\{p(P,A*x+B*y,1)\}x=A*x+B*y\{p(P,x,1)\}$ by the backward substitution rule in (5.16).
4. $\{p(P,A*x+B*y,1)\}u=C*x+B*y\{p(P,Ax+By,1)\}$ by the skip rule in (5.15).
5. $\{p(P,x,1),p(P,A*x+B*y,1)\}C\{p(P,x,1)\}$ by the composition rule in (5.14).
6. if $p(P,x,1) \Longrightarrow p(P,A*x+B*y,1)$, then $\{p(P,x,1)\}C\{p(P,x,1)\}$ by the consequent rule in (5.13).

Fig. 5.16 Correctness of the program using Hoare logic deduction

To verify the Hoare triple in Fig. 5.15, use the inference rules from Table 5.1 on the code, starting from the post-condition `x*P*x<=1`.The process produces an alternate pre-condition $q(P, A*x+B*y, 1)$ for the loop body. By the consequent rule, the correctness of the initial Hoare triple can be checked by checking if $p(P,x,1) \Longrightarrow p(P,A*x+B*y,1)$. The process in Fig. 5.16 is deductive. An algorithmic reformulation of it is Dijstra's work on Predicate transformers [29]. By using the Predicate transformers, the deductive process of Fig. 5.16 is reduced to a computational process that checks the correctness of first order formulas.

5.6.2 Predicate Transformers

The Hoare triples on the code are computed using a form of the *weakest pre-condition* calculus. The weakest pre-condition of C is a function wp that maps any post-condition Q to a pre-condition. The output of the weakest pre-condition function $wp(C, Q)$ is the largest set such that, after the execution of C, Q holds. For example, the correctness of a Hoare triple, for a set of variables x in the code C, is determined by checking if the logic formula $\forall x, P \Longrightarrow wp(C, Q)$ holds. The wp function can be applied to various constructs in an imperative programming language. Some examples are given in Table 5.2. The sequence of I_i in (5.20) can be replaced by a single I if I is an invariant of the loop. Denote the `while` program as $\mathscr{P}$, in the case of partial correctness, $wp(\mathscr{P}, Q) = I$ is the *weakest literal pre-condition* if $I \Longrightarrow wp(C, I)$. In the case of total correctness, $wp(\mathscr{P}, I) = I$ is the weakest pre-condition, if $I \Longrightarrow Q$ and the loop terminates. Recall the control program in Matlab from 5.15, which is comprised of a while loop and satisfies the invariant $p(P, x, 1)$, Apply wp-calculus to that program i.e. $wp(\mathscr{P}, p(P, x, 1)) = p(P, x, 1)$ leads to two logic formulas:

Table 5.2 Weakest Pre-condition Calculus

$$wp(C_1; , ..., C_N, Q) = wp(C_1, wp(C_2, wp(C_3, ..., wp(C_N, Q)) \ldots) \quad (5.17)$$

$$wp(\textbf{skip}, Q) = Q \quad (5.18)$$

$$wp(x := e, Q) = Q[e/x] \quad (5.19)$$

$$wp(\texttt{while}\ a\ \text{do}\ C\ \text{end}, Q) = \forall i \in \mathbb{N}, I_i$$

$$I_0 = true \quad (5.20)$$

$$I_{i+1} = (\neg a \implies Q) \wedge (a \implies wp(C, I_i))$$

1. $I \implies Q$ and the loop terminates. The loop in 5.15 terminates after a finite amount of iterations and clearly $p(P, x, 1) \implies p(P, x, 1)$ is true.
2. $I \implies wp(C, I)$ i.e. $p(P, x, 1) \implies wp(C, p(P, x, 1))$.

The second condition is harder to verify since the set $wp(c, p(P, x, 1))$ need to be computed. Notice the formula $p(P, x, 1) \implies wp(C, p(P, x, 1))$ is equivalent to the Hoare triple

$$\{p(P, x, 1), wp(C, p(P, x, 1))\ C; \{p(P, x, 1)\}\,, \quad (5.21)$$

which means that $p(P, x, 1)$ can be inserted as the pre and post-conditions of the loop body C in Fig. 5.15. Applied additional *wp*-calculus on the loop body results in the annotated code in 5.17. The set of pre-conditions generated by *wp*-calculus i.e. the displayed Matlab comments inside the loop in Fig. 5.17, along with the Matlab code itself, forms the translated proof of stability. Verifying this proof implies that $q(P, x, 1)$ is an invariant of the program, which is a strong evidence that the implementation is good.

```
1 % x'*P*x<=1;
2 while (t<5000)
3 % x'*P*x<=1, wp(u=C*x+D*y,wp(x=A*x+B*y,x'*P*x<=1))=(A*x+B*y)'*P*(
      A*x+B*y)<=1
4   u=C*x+D*y;
5 % wp(x=A*x+B*y,x'*P*x<=1)=(A*x+B*y)'*P*(A*x+B*y)<=1
6   x=A*x+B*y;
7 % x'*P*x<=1
8 end
9 % x'*P*x<=1;
```

Fig. 5.17 Annotated lead/lag compensator in matlab

5.6.3 Strongest Post-condition

The dual of weakest pre-condition is the *strongest post-condition*. The strongest post-condition of C is a function that maps any pre-condition P to a post-condition. The output of the strongest post-condition function $sp(C, P)$ is the smallest set that holds after the execution of C, given that P holds before the execution of C. To verify a Hoare triple $\{P\}\, C\, \{Q\}$ using sp-calculus, first compute $sp(C, P)$ and then check that $sp(C, P) \rightarrow Q$.

5.7 Translation Process for a Simple Dynamical System

This section describes the credible autocoding process for a simple dynamical system, using a mixture of mathematics, C and ACSL.

The process starts with computing a quadratic invariant set for the system. Given a dynamical system $\mathscr{G}$ defined by $x_+ = Ax$, the ellipsoid set $p(P, x, 1)$, constructed by solving $A^{\mathsf{T}}PA - P \prec 0$ for $P \succ 0$, is also invariant w.r.t to $\mathscr{G}$. The invariant property of $p(P, x, 1)$ is the key that allows us to know a priori that the Hoare Triple $\{p(P, x, 1)\}\, \mathscr{P}_2\, \{p(P, x, 1)\}$, in which $\mathscr{P}_2$ is a code implementation of $\mathscr{G}$ in Fig. 5.18, is correct. Since P is invertible, then $q(Q, x, 1)$ with $Q = P^{-1}$ is equivalent to $p(P, x, 1)$. The credible autocoder inserts $q(Q, x, 1)$ as the pre and post-conditions of the program.

Using the weakest pre-condition function from (5.20) on $q(Q, x, 1)$, one obtains $q(Q, x, 1)$ as the pre and post-conditions of the loop body in $\mathscr{P}_2$. Note that the set $q(Q, x, 1)$ is inserted into the code as pre and post-condition of the loop body. This is displayed in lines 7 and 8 of Fig. 5.18, with the loop body enclosed in curly braces.

```
1  /*@
2    requires q(Q,x,1);
3    ensures  q(Q,x,1);
4  */
5  while (true) {
6  /*@
7    requires q(Q,x,1);
8    ensures  q(Q,x,1);
9  */
10 {
11   y1=0.4990*x1+0.1*x2;
12   y2=0.01*x1+1.0*x2;
13   x1=y1;
14   x2=y2;
15   }
16 }
```

Fig. 5.18 $\mathscr{P}_2$: code implementation of $\mathscr{G}$

Next, given the pre-condition $q(Q, x, 1)$ on the loop body, the strongest post-condition computations i.e. *sp*-calculus is performed on the code. Denote the body of the while loop in $\mathscr{P}_2$ as B, the credible autocoding process computes $sp(B, q(Q, x, 1))$ and then checks that $sp(B, q(Q, x, 1)) \rightarrow q(Q, x, 1)$ to ensures the correctness of $\{q(Q, x, 1)\}\, B\, \{q(Q, x, 1)\}$.

The *sp*-calculus process uses ellipsoidal calculus. One of the techniques from ellipsoidal calculus is the following regarding linear transformation of ellipsoidal sets.

Lemma 5.1 *Given a set* $q(Q, x, 1)$*, and given a linear transformation* T*, the image* $T(q(Q, x, 1))$ *is the set* $q(TQT^{\mathsf{T}}, x, 1)$.

Using the formula TQT^{T}, we can compute a strongest post-condition for every line of code in B. Define C_i as the ith line of code in B. Denote x_i as the state vector after the execution of C_i. For example, the state vector starts with $x = \begin{bmatrix} x1 \\ x2 \end{bmatrix}$ before the execution of C_1. The 2 lines of code C_1 and C_2 respectively assigns some values to the variable $y1$ and $y2$. The state vector's dimension increases and becomes $x_2 = \begin{bmatrix} x1 \\ x2 \\ y1 \\ y2 \end{bmatrix}$ after the execution of C_2. The state vector is x again after the execution of C_4. Because the variables $y1$ and $y2$ are discarded from the state vector when they are not used in the code again. Next, given state vectors x_{i-1} and x_i, and given the line of code C_i, the affine semantics of C_i is computed and then used in the construction of a linear transformation T_i from x_{i-1} to x_i. For example, for C_1, the code computes the expression $0.4990 * x1 + 0.1 * x2$ and assigns it to the variable $y2$. The affine semantics of C_1 is therefore $y1 = Lx$, in which $L = \begin{bmatrix} 0.4990 & 0.1 \end{bmatrix}$. The state vector x_0 is x and the state vector x_1 is $x_1 = \begin{bmatrix} x1 \\ x2 \\ y1 \end{bmatrix}$. Hence $T_1 = \begin{bmatrix} I \\ L \end{bmatrix}$. Applying Lemma 5.1, the strongest post-condition for C_m is

$$q(\prod_{i=m}^{1} T_i Q \prod_{i=1}^{m} T_i^{\mathsf{T}}, x_m, 1). \tag{5.22}$$

Hence the strongest-post condition for B i.e. $sp(B, q(Q, x, 1))$ is $q(Q_4, x, 1)$, in which

$$Q_4 = T_4 T_3 T_2 T_1 Q T_1^{\mathsf{T}} T_2^{\mathsf{T}} T_3^{\mathsf{T}} T_4^{\mathsf{T}} \tag{5.23}$$

The computed post-conditions are inserted into $\mathscr{P}_2$ (see Fig. 5.19) as the necessary evidence for the proof-checking of $\mathscr{P}_2$. To verify that $sp(B, q(Q, x, 1)) \implies q(Q, x, 1)$, the inclusion condition $q(Q_4, x, 1) \subseteq q(Q, x, 1)$ is checked. This can be done using a Cholesky decomposition algorithm to check that $Q - Q_4 \succ 0$.

```
while (1) {
/*@
  requires q(Q,x,1);
  ensures  q(T1*Q*T1',x1,1);
*/
{
  y1=0.4990*x1+0.1*x2;
}
/*@
  requires q(Q,x,1);
  ensures  q(T2*T1*Q*T1'*T2',x2,1);
*/
{
  y2=0.01*x1+1.0*x2;
}
/*@
  requires q(Q,x,1);
  ensures  q(T3*T2*T1*Q*T1'*T2'*T3',x3,1);
*/
{
  x1=y1;
}
/*@
  requires q(Q,x,1);
  ensures  q(T4*T3*T2*T1*Q*T1'*T2'*T3'*T4',x,1);
*/
{
  x2=y2;
}
  }
```

Fig. 5.19 $\mathscr{P}_2$ annotated

5.8 Gene-Auto+: A Prototype Credible Autocoder

In this section, some details of the prototype credible autocoder are given. The current prototype is capable of translating control semantics, described in Sect. 5.3.3 into verifiable ACSL annotations on the code.

5.8.1 Gene-Auto: Translation

Gene-Auto's translation architecture is comprised of sequences of independent model transformation stages. This classical, modular approach to code generator design has the advantage of allowing relatively easy insertion of additional transformation and formal analysis stages, such as the annotation generation stage in the prototype. The translation process goes through two layers of intermediate languages. The first one, called the *GASystemModel*, is a data-flow language that is similar to Simulink.

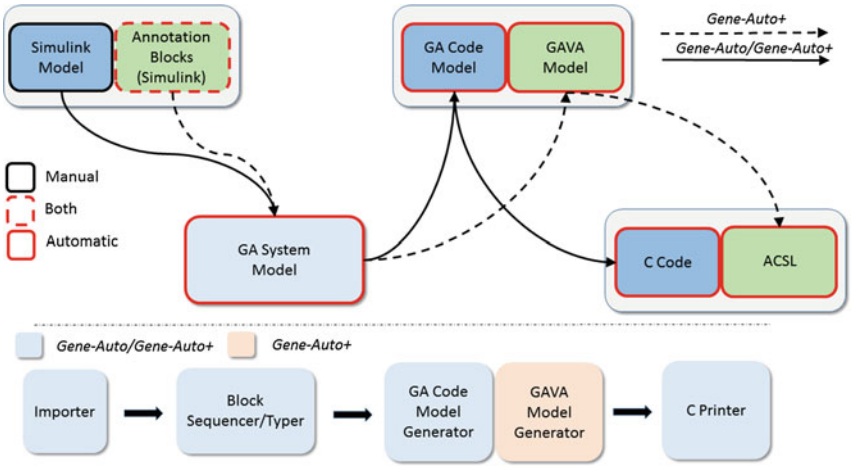

Fig. 5.20 Translation in gene-auto+ versus gene-auto

The input Simulink model, after being imported, is first transformed into the system model. The system model, which is expressed in the *GASystemModel* language, is then transformed into the code model. The code model is in the *GACodeModel* language representation, which has many similarities with imperative programming languages, such as C or Ada. The main translation modules within Gene-Auto, are the importer, the block sequencer and typer, the *GACodeModel* generator, and the C printer. For the prototype, we have recycled much of the transformation modules up to the *GACodeModel* generator. For the translation of the control semantics, we added a sub-module, dubbed the *GAVAModel* generator, to the *GACodeModel* generator. The *GAVAModel* is the ACSL-like language extension in Gene-Auto+. For more details about it, including its meta-model, please see [30]. The *GAVAModel* language enables common ASCL constructs such as: *behavior*, *assumes*-statement, *function contract*, *require*-statement, *ensure*-statement, and *ghost code* to be expressed within an intermediate representation in Gene-Auto+.

Figure 5.20 summarizes the key differences between the translation process of Gene-Auto and Gene-Auto+. The upper half of the figure shows the process in terms of languages and intermediate representations while the bottom part of the figure shows the translation modules. Of the four language representations in the translation process, only the *GASystemModel* representation remains unchanged. This is because, structurally speaking, the annotation blocks are identical to the non-annotation blocks.

5.8.2 Translation of Annotative Blocks

The annotation blocks are also first transformed into a *GASystemModel* representation. This transformation step is unchanged from the original Gene-Auto as the

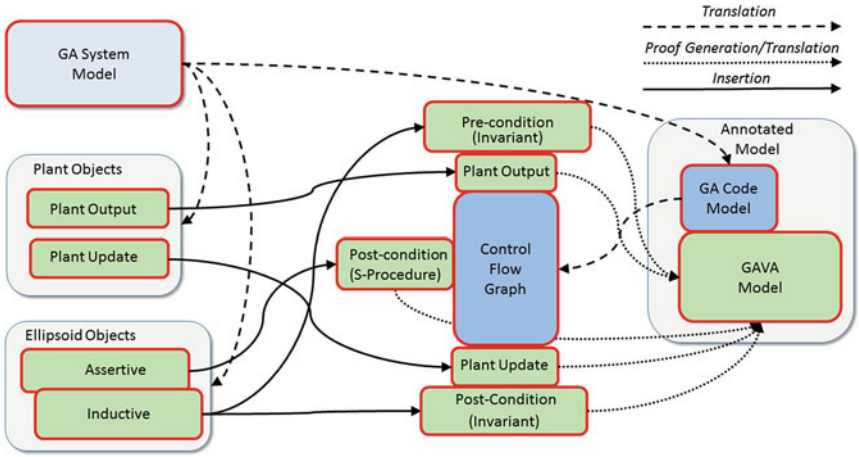

Fig. 5.21 Transformation of control semantics from *GASystemModel* to *GAVAModel*

same language is used to express both regular blocks and annotation blocks. In the *GACodeModel* generation stage, the blocks that express the control semantics are skipped since they are categorized as annotations. They are imported into the *GAVAModel* generation sub-module. This sub-module first translates the annotative blocks into a set of Hoare triple objects on the code model, and then translates the Hoare triple objects into a *GAVAModel* representation. This new representation of the code model with axiomatic semantics is dubbed the *annotated model*.

A high-level overview of the *GAVAModel* generator sub-module is summarized in Fig. 5.21. Following Gene-Auto's modular transformation architecture, the *GAVAModel* sub-module is added as an independent stage within the *GACodeModel* generation module. The major stages in the translation of the annotation blocks into the annotated model are the following:

1. The code model is converted into a control-flow graph structure $\mathscr{X}$.
2. The *constant* blocks are inserted into $\mathscr{X}$.
3. Constant propagation is executed with the definitions provided by the *constant* blocks.
4. The system block is translated into two *plant* objects. A *plant* object is comprised of an affine transformations and a set of ghost code templates expressed in *GAVAModel*. The plant objects are inserted into the beginning of $\mathscr{X}$ and the end of $\mathscr{X}$. The first plant object corresponds to the output function of the state-space system $y = Cx$. The second plant object corresponds to the state-transition function $x_+ = Ax + Bu$.
5. The *quadratic* blocks are grouped based on their inputs as either inductive or assertive. They are translated into *ellipsoid* objects and inserted into appropriate locations within $\mathscr{X}$.

6. The strongest post-condition is computed using *sp*-calculus. In this process, *ellipsoid* objects are generated for almost every line of code and then inserted into the code model.
7. The ellipsoid and plant objects in the annotation model are translated into annotations expressed in *GAVAModel*.

5.9 Translation and Insertion of the *System Block*

The *system* block, which represents the model of the plant, is split into two plant objects representing two linear transformations: $y = Cx$ and $x_+ = Ax + Bu$. One object is inserted into the beginning of the *compute* function and the other part is inserted afterwards. The *compute* function is the function that implements the controller loop body. The two linear transformations are used in the *sp*-calculus to be described later. The *GAVAModel* templates, contained within the two plant objects, are translated into a set of ACSL ghost code statements. The set of ACSL ghost code statements, generated from the closed-loop example, is displayed in Fig. 5.22.

```
1  /*@
2          requires _io_->y == 1.0 * Plant_0_1[0];
3  */
4  void cl_result_compute(t_cl_result_io *_io_, t_cl_result_state *
       _state_) {
5  .
6  .
7  .
8  }
9   {
10     /*@
11             ghost Plant_xp_0_tmp = Plant_0_1[0];
12     */
13     /*@
14             ghost Plant_xp_1_tmp = Plant_0_1[1];
15     */
16     /*@
17             ghost Plant_0_1[0] = 1.0 * Plant_xp_0_tmp + 0.01 *
                   Plant_xp_1_tmp + 5.0E-5 * _io_->u;
18     */
19     /*@
20             ghost Plant_0_1[1] = -0.01 * Plant_xp_0_tmp + 1.0 *
                   Plant_xp_1_tmp + 0.01 * _io_->u;
21     */
22
23   }
```

Fig. 5.22 Ghost code representation of the plant dynamics

5.10 Translation of the *Quadratic* Blocks

A short description of the typing of the *quadratic* blocks and their translations is given here. The semantics of closed-loop stability are structured in such way that there is one inductive ellipsoid set (the Lyapunov function), on the model with another ellipsoid set on the input (bounded input).

5.10.1 Types of Quadratic Blocks

The *quadratic* blocks are separated into two main groups. The first group include the blocks that are inductive. These encode the stability property of the system. To determine if a *quadratic* block is inductive, the following properties must be computed:

1. Every one of the *quadratic* block's input ports must be connected to a port of an unit delay block or to an output port of a *system* block.
2. Given a set $\mathscr{U}$ that contains all unit delay blocks connected to the *quadratic* block. For every unit delay blocks in $\mathscr{U}$, there exists a path from its output node to its input node on the system model.

The second group contains the assertive blocks. These blocks are used to either express boundedness of inputs or sector-bound conditions. Any *quadratic* blocks with one or more input connected to a block that is neither unit delay nor *system* is categorized as an assertive block. The sector-bound blocks are detected by checking to see if the level-set parameter c in the block is set to 0.

After the *quadratic* blocks have been categorized, the bounded-input and inductive blocks are translated into ellipsoid objects containing the Schur form of $p(P, x, 1)$ i.e. $q(Q, x, 1)$ such that $Q = P^{-1}$. This conversion is necessary as all subsequent *sp*-calculus are done in the Schur form due to the possibility of Q_i in $q(Q_i, x_i, 1)$ being singular.

5.10.2 Insertion of Ellipsoid Objects

An assertive ellipsoid invariant $q(Q, x, 1)$ is inserted into a location that is dependent on x. If any of the variable in x is an input argument of the `compute` function, then algorithm will back propagate using *wp*-calculus until x only contains either variables that are input arguments of the compute function, or affine expressions of the variables that are input arguments of the `compute` function.

The *wp*-calculus starts, if needed, after the assertive ellipsoid $q(Q, x, 1)$ has been inserted as a post-condition for the last line of code, in which, a variable belonging to x is assigned. For example, consider the annotation block *bounded_input* in the open-loop case, which expresses a boundedness assumption on the signal $y - y_d$.

```
1  /*@
2      requires in_ellipsoidQ(QMat_0,vect_of_1_scalar(_io_->y - _io_
           ->input));
3  */
4  void simple_olg_compute(t_simple_olg_io *_io_, t_simple_olg_state
       *_state_) \{
5  .
6  .
7  .
8      /*@
9         requires in_ellipsoidQ(QMat_0,vect_of_1_scalar(_io_->y -
              _io_->input));
10        ensures in_ellipsoidQ(QMat_0,vect_of_1_scalar(_io_->y -
              simple_olg_input));
11     */
12   {
13     simple_olg_input = _io_->input;
14   }
15
16     /*@
17
18         requires in_ellipsoidQ(QMat_0,vect_of_1_scalar(_io_->y -
               simple_olg_input));
19    ensures in_ellipsoidQ(QMat_4,vect_of_1_scalar(_io_->y -
          simple_olg_input));
20
21     */
22 .
23 .
24 .
25    /*@
26       requires in_ellipsoidQ(QMat_14,vect_of_1_scalar(
              simple_olg_y - simple_olg_input));
27       ensures in_ellipsoidQ(QMat_14,vect_of_1_scalar(Sum4));
28     */
29 {
30     Sum4 = simple_olg_y - simple_olg_input;
31 }
```

Fig. 5.23 *wp*-calculus on an ellipsoids expressed in ACSL

The signal $y - y_d$ also corresponds to the variable **Sum4** in Fig. 5.23. The annotation block is translated into an assertive ellipsoid object and is inserted into the code as a post-condition of `Sum4=simple_olg_y-simple_olg_y_input`. This post-condition is displayed in the last ACSL contract of Fig. 5.23. Since the variable `Sum4` is not an argument of the `compute` function, the insertion algorithm starts the *wp*-calculus, until x_{-n} in $Q(Q, x_{-n}, 1)$ only contains variables that are input arguments of the `compute` function. For this case, the *wp*-calculus terminated when the ellipsoid in line 2 of Fig. 5.23 is generated.

The insertion of an inductive ellipsoid is more straightforward. The inductive ellipsoid is duplicated three times and inserted as pre and post-conditions respectively at the beginning and end of the `compute` function body. It is also inserted as a pre

```
/*@
   requires in_ellipsoidQ(QMat_1,vect_of_2_scalar(_state_->
       Integrator_1_memory,_
              state_->Integrator_2_memory));
   ensures in_ellipsoidQ(QMat_1,vect_of_2_scalar(_state_->
       Integrator_1_memory,_
              state_->Integrator_2_memory));
*/
void simple_olg_compute(t_simple_olg_io *_io_, t_simple_olg_state
     *_state_) {
.
.
.
 /*@
   requires in_ellipsoidQ(QMat_1,vect_of_2_scalar(_state_->
       Integrator_1_memory,_
              state_->Integrator_2_memory));
   ensures in_ellipsoidQ(QMat_2,vect_of_2_scalar(_state_->
       Integrator_1_memory,_
              state_->Integrator_2_memory));
 */
 {
    simple_olg_input = _io_->input;
 }
 .
 .
 .
 /*@
   requires in_ellipsoidQ(QMat_29,vect_of_2_scalar(_state_->
       Integrator_1_memory,
              _state_->Integrator_2_memory));
   ensures in_ellipsoidQ(QMat_1,vect_of_2_scalar(_state_->
       Integrator_1_memory,
          _state_->Integrator_2_memory));
 */
  {

  }
 }
 // end of the function
```

Fig. 5.24 Inductive ellipsoids in ACSL

and post-conditions on the function itself. These ellipsoids are the ones defined by the matrix variable `QMat_1` in Fig. 5.24.

5.11 Computing the Strongest Post-condition

The *sp*-calculus has been automated in Gene-Auto+ using a set of transformation rules from ellipsoidal calculus, which are used to compute $sp(q(Q, x, 1), C)$ or its

over-approximation for various block of code C. The set of transformation rules can be divided into two categories: affine transformations, and S-Procedure transformations.

5.11.1 Affine Transformation

The basics of affine transformation have been described using the example $x_+ = Ax$ in Sect. 5.7 and proven in PVS (see Fig. 5.11). For automating the proof-checking of the affine transformations of ellipsoids, we define a proof tactic denoted *AffineEllipsoid*, which corresponds to a proof strategy of the same name defined in PVS. This rule is applied whenever a linear abstraction of the code can be computed. Recall from Sect. 5.7, given the pre-condition $q(Q_i, x_i, 1)$ and the code $z = Lx_i$, then the linear transformation from x_i to x_{i+1} is $T_i = \begin{bmatrix} I \\ L \end{bmatrix}$. The strongest post-condition is therefore $q(T_i Q_i T_i^{\mathsf{T}}, x_{i+1}, 1)$.

In the more general case, let the affine semantic of a block of code be $z := Ly$, where $y \in \mathbb{R}^m$ is vector of program states and $L \in \mathbb{R}^{1\times m}$. Let $\mathscr{Q}_i(x) := q(Q_i, x, 1)$, then the *AffineEllipsoid* tactic is

$$\frac{}{\{\mathscr{Q}_n(x)\}\, z := a\, \{\mathscr{Q}_{n+1}(x \cup z)\}}, Q_{n+1} = \mathscr{F}\left(Q_n, \psi(L, y, x), \phi(z, x)\right), \tag{5.24}$$

and the function $\mathscr{F}$ is defined as follows: given the functions $\psi : (L, y, x) \to \mathbb{R}^{1\times n}$ and $\phi : (z, x) \to \mathbb{Z}$, we have

$$\begin{aligned}
\mathscr{F} &: (Q_n, \psi(L, y, x), \phi(z, x)) \to T\left(\psi(L, y, x), \phi(z, x)\right)^{\mathrm{T}} Q_n T\left(\psi(L, y, x), \phi(z, x)\right) \\
T\left(\psi(L, y, x), \phi(z, x)_{i,j}\right) &:= \begin{cases} 1, & 0 \le i, j \le n \wedge i = j \wedge i \ne \phi(z, x) \\ 0, & 0 \le i, j \le n \wedge i \ne j \wedge i \ne \phi(z, x) \\ \psi(y, x)_{1,j}, & i = \phi(z, x) \wedge 0 \le j \le n \end{cases} \\
\psi(L, y, x)_{1,j} &:= \begin{cases} L(1, k), & 0 \le j, k \le n \wedge x_j \in y \wedge y_k = x_j \\ 0, & 0 \le j \le n \wedge x_j \notin y \end{cases} \\
\phi(z, x) &:= \begin{cases} i, & z \in x \wedge z = x_i \\ n+1, & z \notin x \end{cases}
\end{aligned} \tag{5.25}$$

The *ReduceEllipsoid* tactic is used, when the state x_i of the program is reduced in dimensions from the previous state x_{i-1}. Let $\mathscr{Q}(x) := q(Q, x, 1)$, then the *ReduceEllipsoid* tactic is

$$\frac{}{\{\mathscr{Q}_n(x)\}\, \mathbf{SKIP}\, \{\mathscr{Q}_{n+1}(x \setminus \{z\})\}}, Q_{n+1} = \mathscr{G}\left(Q_n, \theta(z, x)\right), \tag{5.26}$$

and the function $\mathscr{G}$ is defined as the following: given the function $\theta : (z, x) \to \mathbb{Z}$, we have

$$\begin{gathered} \mathscr{G} : (Q_n, \theta(z,x)) \to T\,(\theta(z,x))^{\mathrm{T}}\, Q_n T\,(\theta(z,x)) \\ T\left(\theta(z,x)_{i,j}\right) := \begin{cases} 1, & 0 \le i,j \le n-1 \wedge ((i < \theta(z,x) \wedge i = j) \vee (i \ge \theta(z,x) \wedge j = i+1)) \\ 0, & 0 \le i,j \le n-1 \wedge ((i < \theta(z,x) \wedge i \ne j) \vee (i \ge \theta(z,x) \wedge j \ne i+1)) \end{cases} \\ \theta(z,x) := \{\, i,\ z = x_i \end{gathered} \tag{5.27}$$

Before the insertion of the Ellipsoid objects into the code model, each line of code is analyzed for its affine semantics. A linear transformation matrix L is extracted from the abstract semantics and stored in the control flow graph. For example, if we have $x = y + 2z$, then the affine algorithm returns a linear function represented by the matrix $L = \begin{bmatrix} 1 & 2 \end{bmatrix}$. For the plant objects, their affine semantics are computed from the templates stored in the semantics of the *system* blocks.

Figure 5.25 shows an example of the *AffineEllipsoid* usage in the open-loop example. In this example, the pre-condition is the ellipsoid defined by the matrix variable `QMat_21`, and the ensuing line of code assigns the expression `dt_+x1` to the variable `Sum2`. The affine transformation matrix is $L = \begin{bmatrix} 1 & 1 \end{bmatrix}$, and by applying the *AffineEllipsoid* rule, the ellipsoid transformation matrix T is

$$T = \begin{cases} T_{ij} = 1.0, & (i \le 4 \wedge i = j) \vee (i = 6 \wedge (j = 6 \vee i = 6)) \vee (i = 5 \wedge j = 6) \\ T_{ij} = 0.0, & \text{otherwise.} \end{cases} \tag{5.28}$$

5.11.2 S-Procedure

The *S-Procedure* tactic, proven in PVS (see Fig. 5.12), is used to compute an over-approximation of the strongest post-condition for the nonlinear portion of the code. It is based on the well-known principle of Lagrangian relaxation for quadratic forms [31]. For the bounded input stability problem, the LMI solution also yields a small positive multiplier $1 >> \lambda > 0$ that is associated with the bounded input quadratic form. This small multiplier proves to be useful in the *sp*-calculus in the following sense. Consider the line of code `yc=yd-z` with two pre-conditions. One of the two pre-conditions is $q(Q_b, yd, 1)$, which is translated from the *quadratic* block *bounded_input* in the closed-loop model. The other pre-condition $q(Q_i, x_i, 1)$ $y \in x_i$, is generated from the *sp*-calculus. The multiplier $\lambda > 0$ enables us to combine in a convex fashion the two pre-conditions $q(Q_b, yd, 1)$ and $(Q_i, x_i, 1)$ into a single post-condition $q(Q_{i+1}, x_{i+1}, 1)$, $x_{i+1} = x_i \cup \{yd\}$, in which

$$Q_{i+1} = \begin{bmatrix} \frac{1}{1-\lambda} Q_i & 0 \\ 0 & \frac{1}{\lambda} Q_b \end{bmatrix} \tag{5.29}$$

```
/*@
        logic matrix QMat_21 = mat_mult(mat_mult(
  mat_of_6x7_scalar
      (1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,
        0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
        1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,
        0.0,0.0,0.0,0.0,0.0,0.01),QMat_20),
        transpose(mat_of_6x7_scalar(1.0,0.0,0.0,0.0,0.0,0.0,
        0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,
        0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,
        0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.01)));
*/
/*@
        logic matrix QMat_22 = mat_mult(mat_mult(
  mat_of_6x6_scalar
      (1.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,
        0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,
        0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,1.0,0.0,0.0,1.0),QMat_21)
            ,
    transpose(
  mat_of_6x6_scalar
      (1.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,
        0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,
        0.0,0.0,0.0,0.0,0.0,1.0, 0.0,0.0,1.0,0.0,0.0,1.0)));
*/
    /*@
            behavior ellipsoid17_0:
            requires in_ellipsoidQ(QMat_21,
      vect_of_6_scalar(_state_->Integrator_1_memory,
      _state_->Integrator_2_memory,x1,Sum3,Sum1,dt_));
            ensures in_ellipsoidQ(QMat_22,
      vect_of_6_scalar(_state_->Integrator_1_memory,
      _state_->Integrator_2_memory,x1,Sum3,dt_,Sum2));
            @ PROOF_TACTIC (use_strategy (AffineEllipsoid));
    */
{
    Sum2 = dt_ + x1;
}
```

Fig. 5.25 Application of *AffineEllipsoid*

Let $\mathscr{Q}_n(x_i) := q(Q_n, x_i, 1)$, the *S-Procedure* tactic is

$$\frac{\{\mathscr{Q}_1(x_1) \wedge \mathscr{Q}_2(x_2) \wedge \ldots \wedge \mathscr{Q}_N(x_N)\}\,\textbf{SKIP}\,\{\mathscr{Q}_{n+1}(x_0 \cup x_1 \cup \ldots \cup x_n)\}}{Q_{n+1} = \sum_{i=1}^{N} \mu_i \mathscr{H}(Q_i)\,,} \tag{5.30}$$

and $\mathscr{H} : \mathbb{R}^{n_i \times n_i} \to \mathbb{R}^{Nn \times Nn}$ is defined as follows: given the function dim : $R^{n\times n} \to n$, and the function $\rho : n \in \mathbb{Z}^+ \to \sum_{i=1}^{n} \dim(Q_i)$, we have

$$\mathscr{H}(Q_i)(n,m) = \begin{cases} Q_i(n-\rho\,(i-1)\,, m-\rho\,(i-1)), & \rho\,(i-1) \le n, m \le \rho\,(i) \\ 0.0, & \text{otherwise} \end{cases} \tag{5.31}$$

Given the pre-condition $\{\mathscr{Q}_i(x_i)\}$ and code C such that $[\![C]\!] \vDash (y := Lz)$, the *SProcedure* rule is activated only when all the following conditions are satisfied:

1. For each $\mathscr{Q}_i\,(x_i)$, the *AffineEllipsoid* rule does not apply.
2. For the set $\{\mathscr{Q}_i\,(x_i)\}\,, i = 1, \ldots, N, z \subseteq \bigcup_{i=1}^{N} x_i$.
3. For $\mathscr{Q}_i\,(x_i)\,, i = 1, \ldots, N, z \nsubseteq x_i \wedge z \cap x_i \neq \{\varnothing\}$.

The multipliers are computed beforehand using the S-Procedure theory to ensure that the *sp*-calculus, which uses the *S-Procedure* rule at some point, does not result in a strongest post-condition that violates the initial pre-condition. For the closed-loop example in Fig. 5.26, there is one ellipsoid pre-condition defined by the matrix variable `QMat_12`. This ellipsoid is translated from the *quadratic* block *bounded_input*. The other ellipsoid pre-condition, is defined by the matrix variable `QMat_13`. This ellipsoid is computed by the *sp*-calculus. These two ellipsoids are combined to form a post-condition ellipsoid using the *S-Procedure*. The matrix variable `QMat_14`, which defines the post-condition ellipsoid, is expressed using the pre-defined ACSL block matrices function `block_m`.

```
/*@
        logic matrix QMat_14 = block_m(
  mat_scalar_mult(1.0009008107296566,QMat_13),
  zeros(6,2),
  zeros(2,6),
  mat_scalar_mult(1111.111111111111,QMat_12));
*/
    /*@
            behavior ellipsoid9_0:
            requires in_ellipsoidQ(QMat_13,
      vect_of_6_scalar(_state_->Integrator_1_memory,
      _state_->Integrator_2_memory,x1,C11,Integrator_2,Sum3));
            requires in_ellipsoidQ(QMat_12,vect_of_2_scalar(Sum4,
                D11));
            ensures in_ellipsoidQ(QMat_14,
      vect_of_8_scalar(_state_->Integrator_1_memory,
      _state_->Integrator_2_memory,x1,C11,Integrator_2,Sum3,Sum4,
          D11));
            @ PROOF_TACTIC (use_strategy (SProcedure));
    */
{

}
```

Fig. 5.26 Application of the *S-Procedure* tactic

```
1 /*@
2       behavior Plant_22:
3             requires in_ellipsoidQ(QMat_32,vect_of_4_scalar(
                  _state_->Integrator_1_memory,_state_->
                  Integrator_2_memory,Plant_0_1[0],Plant_0_1[1]));
4             ensures in_ellipsoidQ(QMat_1,vect_of_4_scalar(_state_
                  ->Integrator_1_memory,_state_->
                  Integrator_2_memory,Plant_0_1[0],Plant_0_1[1]));
5             @ PROOF_TACTIC (use_strategy (PosDef));
6     */
7 {
8
9 }
```

Fig. 5.27 Verifying the strongest post-condition

5.11.3 Verification of the Strongest Post-condition

After application of the *sp*-calculus, the alternative post-condition generated must be checked against the initial pre-condition. For the closed-loop example displayed in Fig. 5.27, this means checking if `Q_1` is a "bigger" matrix than `Q_32`. If this verification condition can be discharged, then one can claim the code satisfies the stability property. For the closed-loop example, because of a subtle error introduced into the model that went unnoticed until this point, the last verification condition could not be discharged until the sign error was corrected in the Simulink model.

5.12 Automatic Verification of Control Semantics

As part of the credible autocoding process, the annotated C code which generation process we described in Sect. 5.4, must be independently verifiable. Indeed, we now describe and implement a tool that can be used by the certification authority in order automatically check that the annotations are correct with respect to the code. This is achieved by checking that each of the individual Hoare triples hold. Figure 5.28 presents an overview of the checking process. First, the WP plugin of Frama-C generates verification conditions for each Hoare triple, and discharges the trivial ones with its internal prover QeD. Then, the remaining conditions are translated into PVS theorems for further processing, as described in Sect. 5.12.1. It is then necessary to match the types and predicates introduced in ACSL to their equivalent representation in PVS. This is done through theory interpretation [32] and explained in Sect. 5.12.2. Once interpreted, the theorems can be generically proven thanks to custom PVS strategies, as described in Sect. 5.12.3. In order to automatize these various tasks and integrate our framework within the Frama-C platform, which provides graphical support to display the status of a verification condition (proved/unproved), we wrote a Frama-C plugin named pvs-ellipsoid, described in Sect. 5.12.4. Finally, one verification condition does not fall under either `AffineEllipsoid` of `SProcedure` strategies. It is discussed in Sect. 5.12.5.

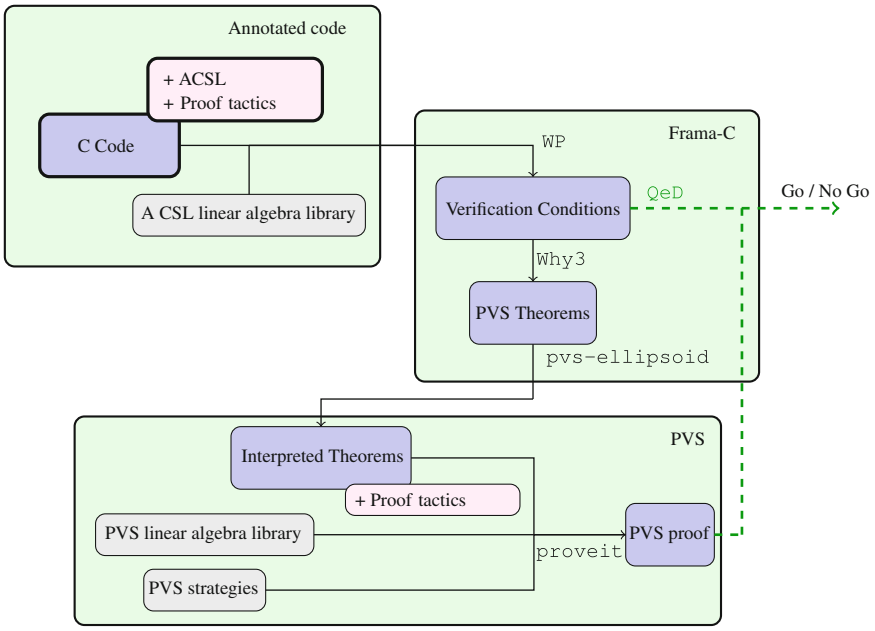

Fig. 5.28 General view of the automated verification process described and implemented in this section

5.12.1 From C Code to PVS Theorems

The autocoder described in the previous Section generates two C functions. One of them is an initialization function, the other implements one execution of the loop that acquires inputs and updates the state variables and the outputs. It is left to the implementer to write the main function combining the two, putting the latter into a loop, and interfacing with sensors and actuators to provide inputs and deliver outputs. Nevertheless, the properties of open and closed loop stability, as well as state-boundedness, can be established by solely considering the update function, which this section now focuses on. The generated function essentially follows the template shown in Fig. 5.29.

Frama-C is a collaborative platform designed to analyze the source code of software written in C. The WP plugin enables deductive verification of C programs annotated with ACSL. For each Hoare tripe $\{pre_i\}\text{inst}_i\{post_i\}$, it generates a first order logic formula expressing $pre_i \implies wp(\text{inst}_i, post_i)$.[2] Through the Why3 platform, these formulas can be expressed as theorems in PVS, so that, for example, the ACSL/C triple shown in Fig. 5.30, taken directly from our running example, becomes the theorem shown in Fig. 5.31.

[2]Given a program statement S and a postcondition Q, $wp(S, Q)$ is the weakest precondition on the initial state ensuring that execution of S terminates in a state satisfying Q.

```
/*@ requires in_ellipsoidQ(Q,
                           vect_of_n_scalar(_state_->s_1,
                                            _state_->s_2,
                                            ...));
  @ ensures in_ellipsoidQ(Q,vect_of_n_scalar(_state_->s_1,
                                           _state_->s_2,
                                           ...));*/
void example_compute(t_example_io *_io_,
                     t_example_state *_state_){
...
/*@ requires pre_i
  @ ensures post_i
  @ PROOF_TACTIC (use_strategy ( strategy_i ) )*/
{
instruction i;
}
...
}
```

Fig. 5.29 Template of the generated loop update function

```
/*@
requires in_ellipsoidQ(QMat_4,
                       vect_of_3_scalar(_state_->
                           Integrator_1_memory,
                                        _state_->
                                            Integrator_2_memory,
                                        Integrator_1));
ensures in_ellipsoidQ(QMat_5,
                      vect_of_4_scalar(_state_->
                          Integrator_1_memory,
                                       _state_->
                                           Integrator_2_memory,
                                       Integrator_1,
                                       C11));
PROOF_TACTIC (use_strategy (AffineEllipsoid));
*/
{
    C11 = 564.48 * Integrator_1;
}
```

Fig. 5.30 Typical example of an ACSL Hoare triple

Note that, for the sake of readability, part of the hypotheses of this theorem, including hypotheses on the nature of variables, as well as hypotheses stemming from Hoare triples present earlier in the code, are ommitted here. Note also that in the translation process, functions like `malloc_0` or `mflt_1` have appeared. They describe the memory state of the program at different execution points.

PVS

```
wp: THEOREM
FORALL (integrator_1_0: real):
  FORALL (malloc_0: [int -> int]):
    FORALL (mflt_2: [addr -> real], mflt_1: [addr -> real],
            mflt_0: [addr -> real]):
     FORALL (io_2: addr, io_1: addr, io_0: addr, state_2: addr,
             state_1: addr, state_0: addr):
             ...
     => p_in_ellipsoidq(l_qmat_4,
                        l_vect_of_3_scalar(mflt_2(shift
                                                  (state_2, 0)),
                                           mflt_2(shift
                                                  (state_2, 1)),
                                           integrator_1_0))
     => p_in_ellipsoidq(l_qmat_5,
                        l_vect_of_4_scalar(mflt_2(shift
                                                  (state_2, 0)),
                                           mflt_2(shift
                                                  (state_2, 1)),
                                           integrator_1_0,
                                           (14112/25 *
                                             integrator_1_0)))
```

Fig. 5.31 Excerpt of the PVS translation of the triple shown in Fig. 5.30

5.12.2 Theory Interpretation

At the ACSL level, a minimal set of linear algebra symbols has been introduced, along with axioms defining their semantics. Section 5.3 describes a few of them. Each generated PVS theorem is written within a theory that contains a translation 'as is' of these definitions and axioms, along with some constructs specific to handling the semantics of C programs. For example, the ACSL axiom expressing the number of rows of a 2 by 2 matrix (in Fig. 5.32) becomes the axiom shown in Fig. 5.33 after translation to PVS.

In order to leverage the existing results on matrices and ellipsoids in PVS, theory interpretation is used. It is a logical technique used to relate one axiomatic theory to another. It is used here to map types introduced in ACSL, such as vectors and matrices, to their counterparts in PVS, as well as the operations and predicates on these types. To ensure soundness, PVS requires that what was written as axioms in the ACSL library be proven in the interpreted PVS formalism.

```
1 /*@ axiom mat_of_2x2_scalar_row:
2   @ \forall matrix A, real x0101, x0102, x0201, x0202;
3   @ A == mat_of_2x2_scalar(x0101, x0102, x0201, x0202) ==>
4   @ mat_row(A) == 2; /*
```

Fig. 5.32 ACSL axiomatization of 2 by 2 matrix row-size

```
PVS
q_mat_of_2x2_scalar_row:
 AXIOM FORALL (x0101_0:real, x0102_0:real,
               x0201_0:real, x0202_0:real):
          FORALL (a_0:a_matrix):
   (a_0 = l_mat_of_2x2_scalar(x0101_0, x0102_0,
                              x0201_0, x0202_0)) =>
   (2 = l_mat_row(a_0))
```

Fig. 5.33 Translation of the ACSL axiom from Fig. 5.32 into PVS

The interpreted symbols and soundness checks are the same for each proof objective, facilitating the mechanization of the process. Syntactically, a new theory is created, in which the theory interpretation is carried out, and the theorem to be proven is automatically rewritten by PVS in terms of its own linear algebra symbols. These manipulations on the generated PVS code are carried out by a frama-C plugin called pvs-ellipsoid, which is described below.

5.12.3 Generically Discharging the Proofs in PVS

Once the theorem is in its interpreted form, all that remains to do is to apply the proper lemma to the proper arguments. Section 5.4 describes two different types of Hoare triples that can be generated in ACSL. Two PVS strategies were written to handle these possible cases. A PVS proof strategy is a generic function describing a set of basic steps communicated to the interactive theorem prover in order to facilitate or even fully discharge the proof of a lemma. The `AffineEllipsoid` strategy handles any ellipsoid update stemming from a linear assignment of the variables. Recall `ellipsoid_general`, a theorem introduced in Sect. 5.3:

```
PVS
ellipsoid_general: LEMMA
 ∀ (n:posnat,m:posnat, Q:SquareMat(n),
           M: Mat(m,n), x:Vector[n], y:Vector[m]):
              in_ellipsoid_Q?(n,Q,x)
              AND y = M*x
        IMPLIES
        in_ellipsoid_Q?(m,M*Q*transpose(M),y)
```

To apply this theorem properly, the first step of the strategy consists of parsing the objectives and hypotheses of the theorem to acquire the name and the dimensions of the relevant variables, and to isolate the necessary hypotheses. The second step consists of a case splitting on the dimensions of the variable: they are given to the prover in order to complete the main proof, and then established separately using the proper interpreted axioms. Next, it is established that $y = Mx$ through

a case decomposition and numerous calls to relevant interpreted axioms. All the hypotheses are then present for a direct application of the theorem. The difficulties in proof strategy design lie in intercepting and anticipating the typecheck constraints (tccs) that PVS introduces throughout the proof. A third strategy was specifically written to handle them.

The S-Procedure strategy follows a very similar pattern, somewhat simpler since the associated instruction in the Hoare triple is a `skip`, using `ellipsoid_combination`, the other main theorem presented in Sect. 5.3.

5.12.4 The `pvs-ellipsoid` Plugin to Frama-C

The pvs-ellipsoid plugin to Frama-C automatizes the steps mentionned in the previous subsections. It calls the WP plugin on the provided C file, then, whenever QeD fails to prove a step, it creates the PVS theorem for the verification condition through Why3 and modifies the generated code to apply theory interpretation. It extracts the proof tactic to be used on this specific verification condition, and uses it to signify to the next tool in the chain, `proveit` [33], what strategy to use to prove the theorem at hand. `proveit` is a command line tool that can be called on a PVS file and attempts to prove all the theories in it, possibly using user guidance such as the one just discussed. When the execution of `proveit` terminates, a report is produced, enabling the plugin to decide whether the verification condition is discharged or not. If it is, a proof file is generated, making it possible for the proof to be replayed in PVS.

5.12.5 Checking Inclusion of the Propagated Ellipsoid

One final verification condition falls under neither the `AffineEllipsoid` nor the `S-Procedure` categories. It expresses that the state remains in the initial ellipsoid $\mathscr{G}_P$. Through a number of transformations, we have proof that the state lies in some ellipsoid $\mathscr{G}'_P$. The conclusion of the verification lies in the final test $P - P' \geq 0$. The current state of the linear algebra library in PVS does not permit to make such a test, however a number of very reliable external tools, like the INTLAB package of the MATLAB software suite, can operate this check. In the case of our framework, the pvs-ellipsoid plugin intercepts this final verification condition before translating it to PVS, and uses custom code from [34] to ensure positive definiteness of the matrix, with the added benefit of soundness with respect to floating point computations.

5.13 Related Works

Although the efforts described in this chapter explore a new intersection between control theory and computer science, a few notable earlier works are mentioned here. Ursula Martin and her team developed a limited Hoare logic framework to reason on frequency domain properties of linear controllers at the Simulink level [35]. Jerome Ferret was the first to focus on the static analysis of digital filters in [5]. It was this work that initiated the connections made between the control-theoretic techniques and software analysis methods in [9]. Parallel work by Roux et al. [34] uses policy interation to generate and refine ellipsoid invariants. We would like to thank Eric Goubault and Sylvie Putot for the useful discussions, and mention their work on zonotopal domain for static analyzers [36].

5.14 Conclusion

The prototype tools and various examples described in this chapter can be found online.[3] We have demonstrated in this chapter a set of prototype tools that is capable of migrating high-level functional properties of control systems down to the code level. In addition we have developed a tool which can independantly verify the correctness of those properties for the code, in an automatic manner. While the nature of controllers and properties supported is relatively restricted, this effort demonstrates the feasability of a paradigm where domain specific knowledge is leveraged and automatically assists code analysis. This opens the way for numerous directions of research. As the mathematical breadth of theorem provers increase, increasingly complex code invariants can theoretically be handled, and thus increasingly complex controllers. In particular, generating verifiable optimization algorithms, e.g. for the purpose of path planning, is a promising direction. Soundness of the results with respect to floating point computations is another issue that requires attention.

The toolchain had been applied not only to the running example, but also on industry size systems, such as the Quanser 3 degree-of-freedom helicopter, and a very light jet turbofan engine controller from Price Induction.

Acknowledgments The authors would like to thank Pierre Roux for his contribution to the PVS-ellipsoid plugin, Gilberto Perez and Pablo Ascariz for their invaluable help on the PVS linear algebra library.

This chapter was prepared under support from NSF Grant CPS Medium 1135955 (CrAVES), CPS Synergy 1446758 (SORTIES), the Army Research Office under MURI W911NF-11-1-0046, the National Aeronautics and Space Administration under NASA Cooperative Agreement NNL09AA00A, activity 2736, the French Fond Unique Interministeriel 2011 project P, ITEA2 OPES, FNRAE project CAVALE, ANR INS project CAFEIN and ANR ASTRID project VORACE.

[3] http://www.feron.org/Eric/Credible.

References

1. Clarke Jr, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press, Cambridge, MA, USA (1999)
2. De Moura, L.: Bjørner, N.: Z3: An efficient SMT solver. In: Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'08/ETAPS'08, pp. 337–340. Springer, Berlin, Heidelberg (2008)
3. Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient SAT solver. In: Proceedings of the 38th Annual Design Automation Conference. DAC '01, pp. 530–535. ACM, New York, NY, USA (2001)
4. Souyris, J.: Industrial experience of abstract interpretation-based static analyzers. In: Jacquart, R. (ed.) Building the Information Society, IFIP International Federation for Information Processing, vol. 156, pp. 393–400. Springer, US (2004). doi:10.1007/978-1-4020-8157-6_31
5. Feret, J.: Static analysis of digital filters. In: European Symposium on Programming (ESOP'04), no. 2986 in LNCS, pp. 33–48. Springer, Berlin (2004)
6. Feret, J.: Numerical abstract domains for digital filters. In: International workshop on Numerical and Symbolic Abstract Domains (NSAD) (2005)
7. Cousot, P., Cousot, R., Feret, J., Mauborgne, L., Miné, A., Monniaux, D., Rival, X.: The ASTRÉE analyzer. In: Proceedings of the 14th European Symposium on Programming, Lecture Notes in Computer Science, vol. 3444 (2005)
8. Monniaux, D.: Compositional analysis of floating-point linear numerical filters. In: CAV (2005)
9. Feron, E.: From control systems to control software. IEEE Control Syst. **30**(6), 50–71 (2010)
10. Herencia-Zapana, H., Jobredeaux, R., Owre, S., Garoche, P.L., Feron, E., Perez, G., Ascariz, P.: Pvs linear algebra libraries for verification of control software algorithms in c/acsl. In: NASA Formal Methods, pp. 147–161 (2012)
11. Rinard, M.: Credible compilation. Techincal report, In: Proceedings of CC 2001: International Conference on Compiler Construction (1999)
12. Denney, E.: Certifying auto-generated flight code. http://shemesh.larc.nasa.gov/LFM2008/slides/Session3/Denney.ppt (2008)
13. Dieumegard, A.: Formal guarantees for safety critical code generation: the case of highly variable languages. Ph.D. thesis, INP Toulouse (2015)
14. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V.: Linear Matrix Inequalities in System and Control Theory, Studies in Applied Mathematics, vol. 15. SIAM, Philadelphia, PA (1994)
15. Megretski, A., Rantzer, A.: System analysis via integral quadratic constraints. IEEE Trans. Autom. Control **42**(6), 819–830 (1997)
16. Berry, G., Gonthier, G., Gonthier, A.B.G., Laltte, P.S.: The esterel synchronous programming language: Design, semantics, implementation (1992)
17. Pakmehr, M., Wang, T., Jobredeaux, R., Vivies, M., Feron, E.: Verifiable control system development for gas turbine engines. CoRR abs/1311.1885 (2013)
18. Yakubovich, V.A.: The solution of certain matrix inequalities in automatic control theory. Soviet Math. Dokl **3**, 620–623 (1962)
19. Nesterov, Y., Nemirovskii, A., Ye, Y.: Interior-point polynomial algorithms in convex programming, In: SIAM, vol. 13. (1994)
20. Bordin, M., Naks, T., Toom, A., Pantel, M.: Compilation of heterogeneous models: Motivations and challenges. In: ERTS. Société des Ingénieurs de l'Automobile, http://www.sia.fr (2012)
21. Izerrouken, N., Pantel, M., Thirioux, X.: Ssi Yan Kai, O.: Integrated formal approach for qualified critical embedded code generator. Formal Methods for Industrial Critical Systems. Lecture Notes in Computer Science, vol. 5825, pp. 199–201. Springer, Berlin (2009)
22. Toom, A., Izerrouken, N., Naks, T., Pantel, M., Ssi-Yan-Kai, O.: Towards reliable code generation with an open tool: Evolutions of the gene-auto toolset. In: ERTS. Société des Ingénieurs de l'Automobile, http://www.sia.fr (2010)

23. Toom, A., Naks, T., Pantel, M., Gandriau, M., Wati, I.: Gene-auto–an automatic code generator for a safe subset of simulink-stateflow and Scicos. In: ERTS. Société des Ingénieurs de l'Automobile, http://www.sia.fr (2008)
24. Baudin, P., Filliâtre, J.C., Marché, C., Monate, B., Moy, Y., Prevosto, V.: ACSL: ANSI/ISO C Specification Language. http://frama-c.cea.fr/acsl.html (2008)
25. Whalen, M.W., Murugesan, A., Rayadurgam, S., Heimdahl, M.P.: Structuring simulink models for verification and reuse. In: Proceedings of the 6th International Workshop on Modeling in Software Engineering, pp. 19–24. ACM (2014)
26. Cuoq, P., Kirchner, F., Kosmatov, N., Prevosto, V., Signoles, J., Yakobowski, B.: Frama-c: A software analysis perspective. In: Proceedings of the 10th International Conference on Software Engineering and Formal Methods. SEFM'12, pp. 233–247. Springer, Berlin, Heidelberg (2012)
27. Scott, M.L.: Programming Language Pragmatics. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000)
28. Hoare, C.A.R.: An axiomatic basis for computer programming. Commun. ACM **12**, 576–580 (1969)
29. Dijkstra, E.W.: Guarded commands, nondeterminacy and formal derivation of programs. Commun. ACM **18**(8), 453–457 (1975)
30. Geneauto metamodel with verification annotations documentation. http://block-library.enseeiht.fr/html/Progress/geneautoAnnot.html
31. Jönsson, U.T.: A Lecture on the S-Procedure. Lecture Note at the Royal Institute of technology, Sweden (2001)
32. Owre, S., Shankar, N.: Theory interpretation in pvs. Techincal report, SRI International (2001)
33. Munoz, C.: Batch proving and proof scripting in pvs. NIA-NASA Langley. National Institute of Aerospace, Hampton, VA, Report NIA Report (2007)
34. Roux, P., Jobredeaux, R., Garoche, P.L., Feron, E.: A generic ellipsoid abstract domain for linear time invariant systems. In: HSCC, pp. 105–114 (2012)
35. Arthan, R., Martin, U., Oliva, P.: A hoare logic for linear systems. Formal Aspects Comput. **25**(3), 345–363 (2013)
36. Goubault, E., Putot, S.: A zonotopic framework for functional abstractions. CoRR abs/0910.1763 (2009)
37. dof helicopter: http://www.quanser.com/Products/3dof_helicopter
38. Price Induction: DGEN 380 turbofan engine. http://www.price-induction.com/en (2013)