Diss. ETH No. 20995

Distributed and Event-Based State Estimation and Control

A dissertation submitted to
ETH Zurich

for the degree of
Doctor of Sciences

presented by

SEBASTIAN TRIMPE
Diplom-Ingenieur Elektrotechnik, Hamburg University of Technology

born July 20, 1981
citizen of Germany

accepted on the recommendation of

Prof. Dr. Raffaello D'Andrea, examiner
Prof. Dr. Manfred Morari, co-examiner
Prof. Dr. Jan Lunze, co-examiner

2013

Institute for Dynamic Systems and Control
ETH Zurich
Switzerland

© 2013 Sebastian Trimpe. All rights reserved.

Acknowledgments

I worked toward this thesis for almost five years – a significant portion of my life – and the fantastic people around me contributed greatly to the meaningfulness of this time period. Without them, I would not have enjoyed my time in Zurich as much as I did, and this thesis would not have been possible in its present form.

I owe my deepest gratitude to my advisor Raffaello D'Andrea. I feel honored having been one of Raff's students and having had the chance to learn from him in many different respects: through his creativity, his enthusiasm, his broad expertise, and his brilliant thinking, all of which had significant impact on my work. At the same time, I thank Raff for giving me the opportunity, the support, and the trust to take leadership and to develop my own research direction. Further I am extremely thankful for the great personal relationship that we have, and I have many fond memories of our years working together. I would sometimes go to see Raff in his office with an idea or a problem, and we would end up arguing and doing math on the white board for one, two, or sometimes three hours to rigorously address the technical or mathematical issue at hand. During these sessions we would often find solutions and gain new insight, and just as often, we would raise new questions too. Inevitably I would walk out of Raff's office with a renewed sense of enthusiasm for my research. These discussions were true highlights of my PhD.

My gratitude extends to my co-examiners Manfred Morari and Jan Lunze. Not only am I thankful to them for providing me with valuable feedback on my doctoral thesis, but also for the years of excellent support I have received from them, which began well before the final steps of my doctoral studies.

When Jan Lunze invited me to visit his lab at the Ruhr-Universität Bochum in late 2009, we had just completed the Balancing Cube as the experimental test bed for my doctoral studies, and I was seeking to define my research agenda for the years to come. My visit in Bochum, as well as many conversations with Prof. Lunze and his students thereafter, stimulated my interest in the area of event-based state estimation and control and were influential for many ideas that I developed during in my PhD research. I would like to thank Prof. Lunze for this excellent scientific exchange and his great support of my work.

Equally, I want to thank Manfred Morari. ETH Zurich is an excellent environment for controls research, and the synergies that exist between the controls groups play an important role in this. My home for the past five years was the Institute for Dynamic Systems and Control (IDSC), and the Automatic Control Laboratory headed by Manfred Morari and John Lygeros was my second home. The various joint seminars and classes, yearly ski-weekends, and many personal interactions served to create strong ties between the two control labs, and were extremely valuable to me, both professionally and personally. My sincerest thanks go to Manfred Morari for attracting me to ETH Zurich, for his valuable feedback on my work, and for his personal support.

Much of the five years has been spent with my colleagues at IDSC – either in the lab, or outside the lab at group events and trips. What an honor to have been part of this wonderful group, where people strive for both individual and collective excellence and support each other with their diverse skills. We have learned from each other in so many ways. Some of these “working” relationships have developed into friendships that will last for life.

First of all, I thank Angela Schoellig, with whom I had the pleasure to share an office for the entire duration of my doctoral studies. Angela, who began her studies at IDSC around the same time I did, was my partner in the PhD process; I’d like to thank her for our many discussions – both scientific and otherwise; for her excellent teamwork when being teaching assistants; for shaping the research group with her ideas and mindset; for her friendship; and, simply, for the great time we had in K33. I am very glad that Philipp Reist was also my office mate for a number of years; I would like to thank Philipp especially for the happiness and care that he brought to the group; for many stimulating discussions; for his expertise in being the only local in our group for a long time; for teaching me to understand Swiss (not just the language); and for being a great friend. Likewise, I thank Sergei Lupashin, Raymond Oung, and Markus Hehn for being fantastic colleagues during all my years at IDSC and friends. I extend my thanks to all other present and past members of our group: Felix Althaus, Federico Augugliaro, Frédéric Bourgault, Dario Brescianini, Daniel Burch, Marc-Andre Corzillius, Matt Donovan, Guillaume Ducard, Carolina Flores, Hans Ulrich Honegger, Mike Hamer, Nico Hübel, Max Kriegleder, Cason Male, Gajamohan Mohanarajah, John Morrell, Mark Mueller, Katharina Munz, Oliver Purwin, Francisco Ramos, Robin Ritz, Geo Robson, Michael Sherback, Hallie Siegel, Igor Thommen, Stefania Tonetti, and Markus Waibel.

The day when our colleague and friend Geo Robson died in an accident was the saddest day of my time at IDSC. I feel deep gratitude that I had the chance to meet and work with Geo. Geo's passion for his dream, his great personality, and his joy in life are what persist.

Building the experimental test bed of the Balancing Cube would not have been possible without the incredible efforts of my collaborators on the project. My special thanks goes to Matt Donovan for his contributions to the concept, the design, and the realization of the Balancing Cube. Furthermore, I am indebted to Daniel Burch, Sergei Lupashin, Hans Ulrich Honegger, and Gajamohan Mohanarajah for their technical support and problem solving skills. I also owe my gratitude to the many students who helped developing the Balancing Cube and who later used the platform for experimental research projects with me.

I would like to thank Katharina Munz, Brigitte Rohrbach, Annina Fattor, Claudia Wittwer, Aleksandra Vukovic, and Stefan Sieber for their administrative support of my work, and Hallie Siegel and Carolina Flores for their excellent help with all types of media design. I especially thank Hallie for her constructive feedback on my writing; I was glad to be able to benefit from her excellent skills in science communication. Finally, I want to thank all other members of IDSC and, in particular, Lino Guzzella and his group for making our institute such a wonderful working environment.

Over the course of my PhD, I had many discussions and interactions with researchers in controls and related fields – during summer schools, conferences, workshops, and when presenting my work at the lab to visitors or when visiting other research groups; these played an important role in my development as a researcher. I am particularly thankful to Bob Bitmead, Karl Henrik Johansson, Jan Lunze, and Richard Murray for fruitful discussions during my visits of their research groups, and also to Karl Johan Åström for giving an inspiring talk on event-based control during his visit at ETH early on in my PhD.

Furthermore, I want to thank my former advisors from the Hamburg University of Technology, Andreas Kwiatkowski, Herbert Werner, Gerwald Lichtenberg, Edwin Kreuzer, and Wolfgang Meyer, as well as Tarek Zohdi from the University of California at Berkeley, for sparking my interest in systems, dynamics, automation, and control.

I also thank my parents, Hannelore and Gerd, and my sisters, Christina and Nora, for supporting me in becoming who I am.

Finally, and most importantly, I thank my wife Britta and my son Joris. There are so many things that I need to thank you for, but let me combine all of these into just one word: thank you for your *love*.

Sebastian Trimpe
Zurich, Spring 2013

Thesis Cover

Carolina Flores is gratefully acknowledged for the photo of the Balancing Cube on the cover of this thesis.

Thesis Template

I thank my colleagues Angela Schoellig and Sergei Lupashin for providing the LaTeX template for this thesis, which builds upon the basic template provided by Leif Andersson from Lund University, who is also gratefully acknowledged.

Financial Support

The Swiss National Science Foundation (SNSF) is gratefully acknowledged for funding this research in part under the grant "Distributed Estimation and Control of Mechatronic Systems."

Abstract

In this thesis, state estimation and control are considered for dynamic systems with multiple distributed control agents (each equipped with sensing, actuation, and computation) that share data with each other over a broadcast network in order to coordinate their actions. With the state estimation algorithms that are developed for this class of systems, each agent can estimate the full state of the networked system based on its own sensor measurements and the sporadically transmitted measurements of the other agents. In order to ensure an efficient use of the shared communication resource, each agent transmits its sensory data only when certain *events* indicate that new data is required to meet a certain estimation performance. A balancing cube serves as the test bed to demonstrate that these event-based estimation algorithms can be used for event-based control when combining them with standard state-feedback controllers. The experimental results show that the event-based control system significantly reduces average network traffic as compared to a system that uses periodic data transmission.

The two main parts of this thesis are 1) the development of distributed and event-based state estimation methods, and 2) the design and construction of the Balancing Cube as a test bed for distributed estimation and control.

The key feature of the algorithms for distributed and event-based estimation is that each agent in the network broadcasts its local sensor measurements to all other agents only if the data is required in order for the other agents to meet a certain estimation performance. To be able to make this decision, each agent implements a state estimator that is connected to the broadcast network (a common bus). Since the state estimate is computed based on data received over the bus only (the local sensor data is used only when also broadcast), the estimates are the same on all agents and represent the common information in the network. The estimator can hence be used to make the transmit decision: if the common estimate of a particular measurement is already “good enough,” it is not necessary to communicate this measurement; if the common estimate is poor, on the other hand, the measurement is transmitted so that all agents can update their estimates.

Two different decision rules for determining whether an estimate is “good enough” are considered in this work: the first compares the real-time measurement to the estimator’s prediction (measurement-based triggering), and the second makes the transmit decision based on the estimation error variance (variance-based triggering). In addition to the different triggering mechanisms, the developed estimator variants differ in their use of the state estimation algorithms (Kalman filters or modified Luenberger observers).

Theoretical guarantees are obtained for some of the *event-based estimation* methods developed herein. For event-based state estimation with variance-based triggering (Paper I), the resulting update equation for the estimation error variance of the Kalman filters is a deterministic equation with switching (the switching modes correspond to the available measurements). This variance update equation represents a new type of Riccati equation, whose iterations typically converge to periodic solutions. This convergence is proven for the special case of a scalar system. In another event-based estimation variant, a suitable measurement-based triggering rule is combined with a linear state estimator that switches between pre-computed static gains (Paper III). Here, it is proven that the event-based state estimator mimics a centralized Luenberger observer with periodic communication of all measurements up to a guaranteed bound.

The usefulness of the developed algorithms for *event-based control* is demonstrated through experiments on an unstable system. The event-based control system consisting of the event-based state estimators and static-gain state-feedback controllers is used to stabilize the Balancing Cube (Paper II and Paper III).

The Balancing Cube (Paper IV) is a dynamic sculpture that can balance autonomously on any of its edges or corners. When standing on a corner, the cube represents a three-dimensional inverted pendulum with six rotating arms mounted on its inner faces that keep the cube in balance. The arms are designed as self-contained modular units (called modules) that are equipped with sensors, actuation, a computer, and a battery. The modules exchange data over a shared communication bus. They constitute the agents in the distributed and networked control system; their joint objective is the stabilization of the cube. The Balancing Cube combines the challenges of nonlinear unstable dynamics with distributed control and networked communication, making it a rich platform for research in dynamics and control.

In addition to serving as motivation and experimental platform for the

event-based estimation and control methods herein, the Balancing Cube also led to research results in other areas. These results are also part of this thesis. For example, the algorithm that was developed to estimate the cube's tilt from multiple inertial sensors (Paper IV) can be applied to any rigid body with only rotational degrees of freedom. Since this estimation method does not rely on a dynamic system model, it works independently of the rigid body dynamics (such as slow or fast motion, or when changing the system's mass configuration). The limiting property of the matrix exponential (Paper V) is another result that was triggered by a concrete problem on the cube. The mathematical result is useful for obtaining simplified models of dynamic systems with sufficiently different time scales, such as the cascaded control system with high-gain inner feedback loops that is used on the cube.

Kurzfassung

In dieser Dissertation werden Verfahren zur Regelung und Zustandsschätzung für dynamische Systeme mit mehreren verteilten Agenten entwickelt. Jeder Agent ist mit eigener Sensorik, Aktorik, und Rechnerleistung ausgerüstet und tauscht mit anderen Agenten Daten über ein gemeinsames Broadcasting Netzwerk aus. Mit Hilfe der entwickelten Zustandsschätzverfahren kann jeder Agent den vollständigen Zustandsvektor des vernetzten Systems basierend auf seinen lokalen Sensormessdaten und auf sporadisch von anderen Agenten übertragenen Daten schätzen. Um eine effiziente Nutzung des Kommunikationsnetzwerks zu gewährleisten, sendet jeder Agent seine Sensormessdaten nur zu bestimmten *Ereignissen*, welche anzeigen, dass neue Messdaten erforderlich sind um eine gewisse Schätzgüte zu erreichen. Ein balancierender Würfel ("Balancing Cube") dient als Versuchsstand zur experimentellen Demonstration, dass die ereignisbasierten Zustandsschätzverfahren bei Kombination mit geeigneter Zustandsrückführung auch zur ereignisbasierten *Regelung* verwenden werden können. Die experimentellen Resultate zeigen, dass mit diesem Ansatz die durchschnittliche Auslastung des Netzwerks gegenüber periodischer Datenübertragung deutlich reduziert wird.

Die Entwicklung von verteilten und ereignisbasierten Schätzverfahren einerseits, und der Entwurf und die Konstruktion des Balancing Cube als ein experimenteller Versuchsstand für verteilte Zustandsschätzung und Regelung andererseits, bilden die beiden Hauptteile dieser Dissertation.

Das Hauptmerkmal der Algorithmen für verteilte und ereignisbasierte Schätzung besteht darin, dass jeder Agent des Netzwerks seine lokalen Sensormessdaten nur dann an alle anderen Agenten überträgt, wenn diese Daten für die anderen Agenten notwendig sind, um eine bestimmte Schätzgüte zu erzielen. Um diese Entscheidung treffen zu können, verwendet jeder Agent einen zweckbestimmten Zustandsschätzer, welcher anhand der über das ge-

meinsame Netzwerk (Bus) kommunizierten Messdaten den Systemzustand schätzt. Da diese Zustandsschätzung somit ausschliesslich auf solchen Daten basiert, die über den Bus empfangen wurden (die lokalen Messdaten werden nur dann verwendet, wenn sie auch über den Bus kommuniziert werden), sind die Schätzungen auf allen Agenten gleich und repräsentieren somit die gemeinsame Information im Netzwerk. Der Schätzer kann daher verwendet werden, um zu entscheiden, ob eine Sensormessung gesendet werden soll oder nicht: Ist die gemeinsame Schätzung einer bestimmten Messung bereits "gut genug", so ist die Übertragung dieser Messung nicht nötig; wenn die gemeinsame Schätzung jedoch schlecht ist, wird die Messung über den Bus gesendet, so dass alle Agenten ihre jeweiligen Schätzungen aktualisieren können.

Für die Entscheidung, ob eine Messung "gut genug" ist, werden zwei unterschiedliche Regeln untersucht: bei der einen, werden die tatsächlichen Sensormessungen mit ihrer Vorhersage durch den gemeinsamen Schätzer verglichen (Messdaten-basiertes Senden, engl. "measurement-based triggering"); wohingegen die andere Regel auf der Schätzfehlervarianz basiert (Varianz-basiertes Senden, engl. "variance-based triggering"). Zusätzlich zu diesen zwei Sendemechanismen unterscheiden sich die in dieser Arbeit entwickelten Varianten für ereignisbasierte Zustandsschätzung noch in den verwendeten Schätzalgorithmen (Kalman Filter oder modifizierter Luenberger Beobachter).

Für einige der hierin entwickelten *ereignisbasierten Schätzverfahren* werden theoretische Garantien hergeleitet. Bei ereignisbasierter Zustandsschätzung mit Varianz-basiertem Senden (Paper I) erfolgt die Iteration der Schätzfehlervarianz gemäss einer deterministischen Gleichung mit schaltendem Verhalten, wobei die Schaltmodi den zum jeweiligen Zeitpunkt zur Verfügung stehenden Messungen entsprechen. Diese Iterationsgleichung stellt einen neuen Typ einer Riccatigleichung dar, deren Lösungen typischerweise asymptotisch periodisches Verhalten aufweisen. Die Konvergenz gegen periodische Lösungen wird für den Spezialfall eines skalaren Systems bewiesen. Für eine weitere Variante der ereignisbasierten Schätzung (schaltender linearer Schätzer mit Messdaten-basiertem Senden, Paper III), wird gezeigt, dass der ereignisbasierte Zustandsschätzer einen zentralen Luenberger Beobachter mit periodischem Zugriff auf alle Messdaten bis auf einen beschränkten Fehler approximiert.

Die Möglichkeit, die entwickelten Schätzalgorithmen für *ereignisbasierte Regelung* zu verwenden, wird anhand von Experimenten mit einem instabilen System demonstriert: Das ereignisbasierte Regelsystem bestehend aus einem ereignisbasierten Zustandsschätzer und einem Regler mit statischer Zustandsrückführung kann zur Stabilisierung des Balancing Cube auf einer seiner Ecken oder Kanten verwendet werden (Paper II und Paper III).

Der Balancing Cube (Paper IV) ist eine dynamische Skulptur, welche selbstständig auf einer beliebigen Kante oder Ecke balancieren kann. Wenn der Würfel auf einer seiner Ecken steht, stellt er ein dreidimensionales inverses Pendel dar, welches von sechs rotierenden Armen auf seinen Innenflächen im Gleichgewicht gehalten wird. Die Arme sind als eigenständige modulare Einheiten (genannt "Module") ausgelegt. Jedes Modul ist mit Sensoren, einem Aktuator, einem Computer und einem Akku ausgerüstet; und die Module tauschen untereinander Daten über einen gemeinsamen Kommunikationsbus aus. Die Module bilden dabei die Agenten des verteilten und vernetzten Regelungssystems; ihr gemeinsames Ziel besteht darin, den Würfel zu stabilisieren. Der Balancing Cube vereint damit die Herausforderungen einer nichtlinearen und instabilen Streckendynamik mit verteilter Regelung und Netzwerk-basierter Kommunikation, und stellt somit eine interessante Problemstellung für regelungstechnische Forschung dar.

Im Rahmen des Balancing Cube Forschungsprojektes sind neben ereignisbasierten Regelungs- und Schätzverfahren noch weitere Forschungsergebnisse erzielt worden, deren Anwendung über die konkrete Problemstellung des Balancing Cube hinausgehen. Zum Beispiel wurde eine Methode zur Schätzung der Neigung des Würfels aus den Messdaten mehrerer Inertialsensoren entwickelt (Paper IV). Mit dieser Methode kann die Neigung beliebiger Starrkörper mit ausschliesslich rotatorischen Freiheitsgraden geschätzt werden. Ausserdem funktioniert dieses Schätzverfahren unabhängig von der Dynamik des Starrkörpers (so zum Beispiel für langsame und schnelle Bewegungen oder bei einer Veränderung der Massenkonfiguration), weil der Algorithmus nicht auf einem Model der Systemdynamik beruht. Die Grenzwerteigenschaft der Matrixexponentialfunktion (Paper V) ist ein weiteres Beispiel für ein Forschungsergebnis, welches aus einem konkreten Problem bei der Entwicklung des Würfels hervorging. Das mathematische Resultat ist nützlich zum Beispiel bei der Berechnung vereinfachter Modelle von dynamischen Systemen mit stark unterschiedlichen Zeitkonstanten, wie beispielsweise der Kaskadenregelung mit schnellen inneren Regelkreisen, die auf dem Balancing Cube verwendet wird.

Contents

Preface 19

1. Introduction 21
   1.1 Scope and Motivation 22
   1.2 Thesis Outline 30

2. Contributions 33
   2.1 Distributed and Event-Based State Estimation (Part A) 33
   2.2 The Balancing Cube: A Test Bed for Distributed Estimation and Control (Part B) 44
   2.3 List of Publications 49
   2.4 Invited Talks 50
   2.5 Supervised Student Projects 51
   2.6 Outreach 52

3. Future Directions 55
   3.1 Distributed and Event-Based State Estimation (Part A) 55
   3.2 The Balancing Cube: A Test Bed for Distributed Estimation and Control (Part B) 58

References 59

A. DISTRIBUTED AND EVENT-BASED STATE ESTIMATION 63

Paper I. Event-Based State Estimation with Variance-Based Triggering **65**
   1. Introduction 66
   2. Event-Based State Estimator 72
   3. Illustrative Examples 75
   4. Asymptotic Periodicity for Scalar Problem 80

5. Discussion 99
Appendix 101
A. Proof of Lemma 1 101
B. Proof of Lemma 2 106
References 107

Paper II. An Experimental Demonstration of a Distributed and Event-Based State Estimation Algorithm 111
1. Introduction 112
2. State Estimation Algorithm 114
3. Application to the Balancing Cube 120
4. Concluding Remarks 126
Acknowledgements 129
Appendix 129
A. State Space Model and Feedback Gains of the Balancing Cube 129
References 131

Paper III. Event-Based State Estimation with Switching Static-Gain Observers 133
1. Introduction 134
2. Estimation Problem Formulation 137
3. Event-Based State Estimator 139
4. Analysis 140
5. Experiments 143
6. Concluding Remarks 146
Acknowledgements 148
References 148

B. THE BALANCING CUBE: A TEST BED FOR DISTRIBUTED ESTIMATION AND CONTROL 151

Paper IV. The Balancing Cube: A Dynamic Sculpture as Test Bed for Distributed Estimation and Control 153
1. Introduction 154
2. Design 159
3. Operation 164
4. Modeling 165
5. Control System Architecture 175
6. State Estimation 177
7. Control 183
8. Experiments 189

9. Concluding Remarks 192
Acknowledgments 194
References 194
Sidebar 1: Balancing Cube on Tour 200
Sidebar 2: What Is the Cube's Maximal Balancing Range? . . . 201
Sidebar 3: Time Scale Separation Algorithm 203
Sidebar 4: What Is the Effect of Integral Action in the Controller? 205
Sidebar 5: Why Are the Top Modules Used Less? 208
Sidebar 6: How Steady Can the Cube Balance? 213
Sidebar 7: Other Balancing Shapes 218
Paper V. A Limiting Property of the Matrix Exponential . 219
1. Introduction 220
2. Notation and Preliminaries 222
3. Condition Based on the Log-norm of $K(\alpha)$ 223
4. $K(\alpha)$ with Special Functional Dependency 230
5. Concluding Remarks 235
Appendix 235
A. Proof of Fact 1 235
B. Proof of Fact 2 235
References 236

Preface

This thesis reports the results of the author's doctoral studies at the Institute for Dynamic Systems and Control (IDSC) at ETH Zurich from March 2008 until February 2013 under the supervision of Prof. Raffaello D'Andrea.

The thesis is structured as a cumulative dissertation with its main parts consisting of five self-contained research articles (three journal, two conference) that have been published or submitted for publication throughout the doctoral studies. Three introductory chapters provide an introduction to the topics considered in this work, give an overview of the contributions of this thesis, and point to directions for future research.

This thesis contributes to the field of distributed and event-based estimation and control for networked systems. In particular, the main contributions are: the development and theoretical analysis of distributed and event-based state estimation algorithms for networked systems; the development of a test bed for networked and distributed estimation and control (the Balancing Cube); and the experimental validation of event-based control on this test bed by combining the estimation algorithms developed herein with standard state-feedback control. The results on event-based state estimation and their application on the Balancing Cube make up Part A of this thesis. The design, modeling, and control of the test bed are presented in Part B, along with further research contributions that lie outside the area of event-based estimation and control.

1

Introduction

Advances in sensor and computer technology in the last several decades (see [1]–[3]) make it now viable to embed sensors, high performance computing, and communication technology into almost any engineering system. Embedded systems connect with each other, for example, to form large sensor networks whose sensing and monitoring capabilities exceed those of a single sensing device (for example, real-time traffic monitoring [4], and autonomous ocean monitoring systems [5, 6]). To leverage the full potential of the vast amount of available information that comes with inexpensive sensing, computation, and communication, however, engineering systems must be enabled to make decisions and act autonomously based on this data.

This doctoral thesis considers control systems where multiple autonomous agents make their control decisions based on feedback data from their local sensors and data received from other agents. The agents exchange data over a shared communication network in order to coordinate their actions and to achieve a joint objective, such as the stabilization of the interconnected dynamic system. Such systems, where multiple controller, sensor, and actuator units communicate over a shared multi-purpose network, are referred to as *networked control systems* (NCSs) [7]–[11].

Traditional control system design typically assumes a fixed communication structure and periodic data transmission between the controller, the sensors, and the actuators. In contrast, NCSs allow for more flexible communication structures where the entities of the network can connect and exchange data when needed. The communication medium is not longer exclusively dedicated to a single feedback control loop, but shared by multiple controllers. The design of safe and efficient NCSs requires the development of novel design methodologies that take the network into account as a shared resource, and address the design of a network access strategy in tandem with the design of the estimation and control algorithms that rely on the network data. This work focuses on the development and experimental validation of estimation and control methods that ensure (in addition to the control objectives) an efficient use of the shared communication resource by invoking the transmission of feedback data only when certain events indicate that new data is necessary (for example, when an error signal crosses a threshold level).

The thesis is separated into two parts that contain its main contributions. With the *distributed and event-based state estimation* algorithms that are developed in Part A, each agent of an NCS can estimate the complete state of the interconnected dynamic system while, at the same time, sensor data is exchanged between the agents only when required to meet a certain estimation performance. Part B describes the design of the *Balancing Cube* as a test bed for distributed estimation and control, which was developed as part of these doctoral studies. The Balancing Cube is a dynamic sculpture that can balance autonomously on any one of its edges or corners through the joint action of six rotating arms on its inner faces that constitute the control agents of the NCS.

The event-based state estimation algorithms of Part A were successfully used for *event-based control* in experiments on the cube: the state estimates computed on each control agent were used with a state-feedback control law to compute control commands for the agent's actuator. For the event-based state estimation methods in Part A, theoretical guarantees are presented herein. The effectiveness of the methods for event-based control (when the event-based estimators are combined with state-feedback controllers) is demonstrated through experiments on the cube.

The sections below introduce the scope and the motivation of this work in greater detail (Sec. 1.1) and present the outline of this thesis (Sec. 1.2).

1.1 Scope and Motivation

Figure 1.1 depicts the class of the networked control systems that is considered in this thesis. Multiple sensor-actuator agents (sensor, actuator, and algorithm block in Fig. 1.1 are together considered as one agent) are distributed spatially along a dynamic system. The dynamic system may have unstable modes and dynamic coupling between the agents. Each agent observes part of the system state through its sensors, and it computes commands to its local actuator based on the local sensor data, the data received from the other agents over the common bus, and a feedback law. The feedback controller typically consists of a state estimator and a state-feedback law. Each agent is responsible for deciding when to broadcast its local sensor data over the bus. By exchanging data over the network, the agents cooperate with each other to achieve a common objective, for example, stabilization of the dynamic system.

The following list summarizes key properties of the NCSs considered in this work:

Unstable and coupled dynamics. Being a balancing object with multiple actuated arms mounted on the same rigid body, the test bed developed in Part B is unstable and the agents' dynamics are coupled. For the control and estimation algorithms developed herein, we do not impose restrictions on the system dynamics with respect to stability or the dynamic coupling. For the development of the algorithms in Part A, we assume linear dynamics.

Noisy output measurements. The sensor measurements that are available from the system are corrupted by noise. Not all system states are measured.

Distributed control system. There is no hierarchical order among the agents of the NCS and, in particular, there is no central control unit.

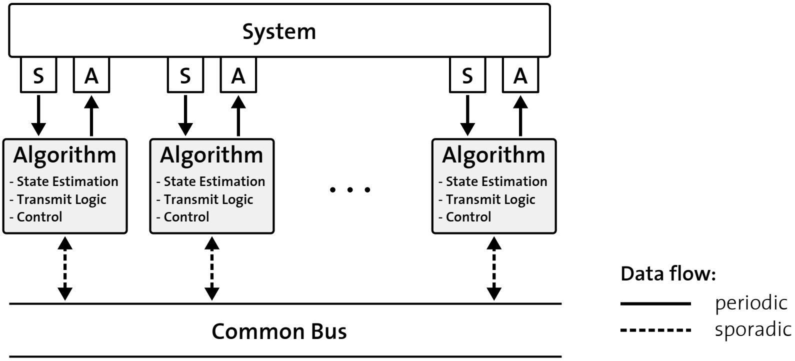


Figure 1.1 Abstraction of the networked control systems considered in this thesis. Multiple sensor (S) and actuator (A) units are distributed spatially along a dynamic system. Each sensor and actuator is associated with an algorithm block; and sensor, actuator, and algorithm together are denoted as an *agent*. Each agent runs control and estimation algorithms, and decides whether or not to transmit its local sensor measurements to its peers over the common bus. Solid lines indicate periodic data flow (at every instant of an underlying discrete-time sampling), and dashed lines indicate sporadic transmission of data (not at every instant).

Stabilizability and detectability. From all inputs (actuators) taken together, the system is assumed to be stabilizable; and from all outputs (sensor measurements) taken together, the system is assumed to be detectable. Yet, the system needs neither be stabilizable nor detectable from the local sensors and actuators of a single agent alone.

Communication network. Since the communication network must support the exchange of feedback data required for stabilization, we use a reliable communication network (Controller Area Network (CAN)) to connect the control agents of the test bed developed in Part B. The shared bus of this network type allows all other agents to receive the data if any agent puts data on the bus. For the development of the event-based estimation algorithms in Part A, we assume an ideal bus network where communication is without delay and data loss.

The framework of the NCS shown in Fig. 1.1 includes other types of networked systems, to which the algorithms developed herein can be equally applied. For example, the agents may be heterogeneous in the sense that some agents may be missing local actuators or sensors. If all actuators are removed, the system in Fig. 1.1 represents a *sensor network* [12, 13], where multiple sensors observe a dynamic process and transmit their measurements over the bus. One or more estimator nodes connected to the bus receive the data and estimate the process state, for example, for the purpose of monitoring the process from a remote location.

The work herein aims to develop algorithms for general systems within the class of NCSs considered. The results therefore contribute to the fundamental research in the field. To validate the assumptions made in the theoretical development, as well as to trigger new research questions, the Balancing Cube was developed as a physical representation of an NCS. The problems addressed in this thesis (such as coordination of multiple control agents, network-based control, management of a shared communication resource) play an important role in many application areas. Examples of today's application areas for multi-agent or networked control include sensing and monitoring systems [4]–[6], transportation systems (for example, vehicle platooning [14]), distribution systems (autonomous warehouse [15]), and industrial automation [16]. Communication networks and their consideration in the design of control and estimation algorithms will become even more important as the number of interconnected entities is expected to increase in future engineering systems such as cyber-physical systems [17, 18] with envisioned application domains like transportation, power systems, smart buildings, mobile robots, and process industry.

Part A. Distributed and Event-Based State Estimation

A key assumption in traditional control system design is the continual flow of data from the sensors to the controller, and the controller to the actuator. In continuous-time control [19, 20], the data signals are continuous in time (for example, an analog voltage signal representing a sensor measurement); thus, the controller receives new information about the plant continuously. In discrete-time or digital control [21, 22], data is exchanged periodically at equidistant sampling instants (for example, a digital sensor transmitting measurements every 10 ms). The periodic sampling rate of a discrete-time control system is a design parameter and usually chosen to be significantly larger than the natural frequency of the process that is controlled. Both paradigms have in common that the instants when data is transmitted depend on time (every time for continuous-time control and at equidistant instants for discrete-time control), and that they are fixed during the system design, rather than adapted to the state of the system during operation. In order to guarantee the continual flow of data in the implementation of traditional control systems, dedicated communication hardware is usually required for each data link between sensors, controllers, and actuators (for example, every sensor is connected to the controller through an individual link, and the link is used exclusively to communicate that sensor's measurements).

In networked control systems, however, data is exchanged between sensor, actuator, and control units over a multi-purpose network. Multiple control loops are typically closed over the same network, and the different entities may exchange data of various content (for example, in addition to sensor data and control commands of typical feedback control systems, data for system health monitoring or higher-level tasks such as adaptation may be exchanged). For the design of cost effective networked control systems, it is vital to consider the communication network as a shared resource and design network access strategies (*who talks to whom and when*) in tandem with the control and estimation algorithms that use the network. This requires the development of design methodologies that extend beyond traditional control design.

Event-based estimation and control approaches fall into this category of algorithms that consider the estimation and control design problem jointly with the design of a communication strategy. Event-based control started with the work in [23, 24] and has since developed into an active field of research (see [25] for an overview). In traditional control systems, communication is time-triggered. In event-based systems, however, data is exchanged only when certain events indicate that new data is required in order to meet some specification of the control system (such as a stability criterion, or control or estimation performance). This way, the transmission of data can be linked to its relevance for control or estimation, which allows for an efficient and adaptive use of the communication resource.

Part A of this thesis focusses on the development of *event-based state estimation* methods for a process that is observed by multiple sensors connected over a common bus (such as in Fig. 1.1). In particular, we seek to develop algorithms that enable each agent to (i) *estimate the full state of the dynamic system*, and (ii) *transmit sensor data only when the data is required for the other agents in order to meet a certain estimation performance.* If objective (i) was considered exclusively, an optimal strategy would be for all sensors to communicate their measurements at every time step (provided the network capacity allowed this), since this would maximize the information available for state estimation. On the other hand, if one only cares about minimizing communication, no communication at all would be the obvious solution. Yet, some inter-agent communication is required to obtain bounded estimation errors for the problem at hand since the system is generally not detectable from the local sensors of an agent. It is therefore clear that considering the objectives (i) and (ii) simultaneously will yield a trade-off between estimation performance and average communication rates. The developed state estimation methods are *distributed* because the transmit decision is made locally by every agent.

While the focus of the development and theoretical analysis in this part of the thesis is on methods for event-based *state estimation*, we also combine the algorithms with state-feedback controllers to obtain an event-based *control* system. The feasibility of this approach for event-based control is demonstrated through experiments on the Balancing Cube test bed. The experimental results for event-based control are also described in Part A of this thesis. An overview of the contributions of this part to the field of event-based state estimation and control is given in Sec. 2.1.

Part B. The Balancing Cube: A Test Bed for Distributed Estimation and Control

An independent goal of these doctoral studies was the development of a three-dimensional dynamic sculpture that can balance autonomously on a single point. Like a troupe of acrobats balancing in formation[1], the sculpture has multiple identical balancing modules that adjust their motion and

---

[1] The first ideas for building a self-balancing dynamic sculpture by Raffaello D'Andrea were inspired by the Cirque du Soleil performance "Statue Act," where two individuals achieve various balance poses through coordination and precise control of their bodies.

coordinate with each other. The balancing modules are designed as self-contained units equipped with sensors, actuation, computation, and power; and they coordinate with each other by exchanging data over a network. The system therefore falls into the class of NCSs depicted in Fig. 1.1 making it a test bed for the research results of Part A as well as further results presented in this part.

The key challenges in the design of such a dynamic sculpture can be summarized as follows:

Balancing. The sculpture needs active stabilization to be able to balance on a single point.

Modular design. The balancing modules are self-contained units equipped with sensing, actuation, computation, communication, and power. The modular design allows their use for balancing different shapes such as a cube, a pendulum, or a tetrahedron (the first two were realized within this work).

Coordination. Since the balancing modules are mounted on the same rigid body, the dynamics of the individual units are coupled. Thus, the modules must coordinate with each other (for example, by exchanging data over a communication network).

Interaction. The sculpture is dynamic and interactive. Users initiate the sculpture's autonomous balancing by placing it on one of its tips. They can interact by pushing the sculpture. The structure must thus be able to withstand repeated falls.

We began by building an inverted pendulum (Fig. 1.2, right) that served as a one-dimensional prototype of the cube. The pendulum's single degree of freedom is stabilized by a single balancing module. Once this proof-of-concept was complete, we built the Balancing Cube – a full 3D multi-body inverted pendulum (Fig. 1.2, left) – which has six balancing modules mounted on its inner faces and is more than two meters tall when standing on one of its corners.

Building the Balancing Cube represented a system design challenge that required the integration of various components including the mechanical structure, the actuation mechanism, the sensor systems, the electronics, the computer hardware, the communication network, and the control system. In addition to the system integration, the main contributions within these doctoral studies were the design and the implementation of the control and estimation algorithms that enable the cube to balance. The control challenges that the above design poses are:

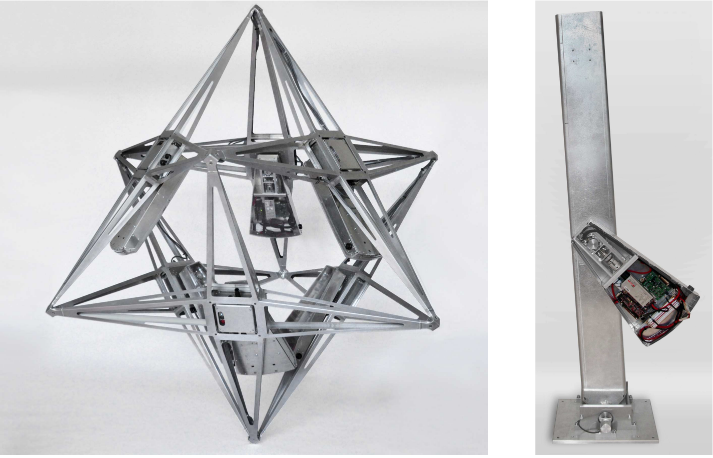


Figure 1.2 The Balancing Cube (left) is a dynamic sculpture that can balance autonomously on any one of its corners or edges through the action of six balancing modules on its inner faces. Each module carries actuation, sensing, computation, and power; and it coordinates with its peers by exchanging data over a communication network. The inverted pendulum (right) is a one-dimensional abstraction and early prototype of the cube. The Balancing Cube in the shown configuration is about 2 m tall; the pendulum body has a length of roughly 1.5 m.

Nonlinear unstable dynamics. The cube standing on one of its edges or corners represents a 1D/3D inverted pendulum with unstable and nonlinear dynamics. The basal control objective is stabilization.

Underactuated mechanical system. Since the degrees of freedom of the cube body are not actuated, the system is underactuated, [26].

Distributed control. Each of the balancing modules controls its angular motion relative to the cube body. There is no centralized unit that controls all actuators.

Networked communication. The balancing modules exchange data with each other over a shared communication bus.

Within these doctoral studies, the Balancing Cube served multiple purposes: the cube served as an experimental platform for testing and verifying the developed estimation and control methods, it triggered new research questions and results, it supported education, and it was used as a means for communicating controls research to the general public.

Experimental platform. The developed test beds (Balancing Cube and inverted pendulum) were used to demonstrate that the control and estimation algorithms developed in this thesis work on physical systems, to verify that assumptions made in the design and development of these algorithms are justified from a practical point of view, and to quantify the performance of the methods in an experimental setting.

All algorithms developed in Part A of this thesis were demonstrated experimentally on the Balancing Cube platform. In particular, these experiments showed that the event-based estimation methods can be used for event-based control.

Motivation for research problems. The challenges that we faced during the design of the Balancing Cube motivated research questions and triggered results that were not anticipated at the start of the project (see Sec. 2.2).

The problem formulation in Part A of this thesis was largely motivated by the Balancing Cube: in order to estimate the full system state on each module (which is then used for state-feedback control), exchange of some sensor data between the modules is necessary; however, the rates at which transmission of this data is required is not obvious and depends on the type of sensor. This motivated the development of the event-based algorithms where the transmit decision on each module is made on-line based on the relevance of the data for state estimation.

Educational platform. Not only did the cube lead to publishable research results, all project participants also gained hands-on experience in the practical design of modern control systems. In addition to representing an engineering challenge for these doctoral studies, the Balancing Cube served as the object of a two-semester-long design class for ten Master-level students, and it supported twelve individual student projects (see Sec. 2.5). Overall, the cube project facilitated learning at all academic levels.

Demonstrations and exhibitions. We have used the Balancing Cube to explain our research result to non-scientific audiences in demonstrations, presentations, and exhibitions (see Sec. 2.6). The Balancing Cube is an excellent means of communicating the principles of dynamic systems and control: because it builds on our intuitive understanding of balance, it allows the audience to visualize concepts such as stability (the cube does not fall) and feedback control (it takes action based on where it is leaning), without requiring an education in engineering or physics.

1.2 Thesis Outline

The contributions of this thesis are summarized in Chapter 2; in particular, the context and the contribution of each paper included in this thesis is explained. In addition, lists are provided of all publications, invited talks, supervised student projects, and outreach activities that were part of these doctoral studies.

Chapter 3 gives a brief account of possible future research directions emanating from this work.

The main part of this thesis consists of five research papers. Each paper can be read as a self-contained document. The notation between any two papers may differ slightly. Related work is reviewed in each paper.

Part A on *Distributed and Event-Based State Estimation* comprises the following papers:

Paper I

S. Trimpe and R. D'Andrea, **Event-based state estimation with variance-based triggering**, submitted to *IEEE Transactions on Automatic Control*.

Paper II

S. Trimpe and R. D'Andrea, **An experimental demonstration of a distributed and event-based state estimation algorithm**, in *Proc. of the 18th IFAC World Congress*, Milan, Italy, Aug. 2011, pp. 8811–8818.

Paper III

S. Trimpe, **Event-based state estimation with switching static-gain observers**, in *Proc. of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Santa Barbara, CA, USA, Sep. 2012, pp. 91–96.

And Part B on *The Balancing Cube: A Test Bed for Distributed Estimation and Control* includes these papers:

Paper IV

S. Trimpe and R. D'Andrea, **The Balancing Cube: A dynamic sculpture as test bed for distributed estimation and control**, *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 48–75, Dec. 2012.

Paper V

S. Trimpe and R. D'Andrea, **A limiting property of the matrix exponential**, submitted to *IEEE Transactions on Automatic Control.*

2

Contributions

This chapter provides an overview of the contributions of this thesis. In Sec. 2.1, the distributed state estimation problem considered in Part A of this thesis is stated, the key ideas of the event-based approach are explained, and the publications of this part are put in context. Section 2.2 summarizes the results and research contributions of Part B of this thesis (development of the Balancing Cube test bed). The sections 2.3 to 2.6 list all publications, invited talks, supervised student projects, and outreach activities that were part of these doctoral studies.

2.1 Distributed and Event-Based State Estimation (Part A)

In this section, we formulate the distributed estimation problem and discuss the key ideas of the event-based approach to address it. The problem formulation and the key ideas are shared by Paper I to Paper III. The contribution of each of these papers is discussed at the end of this section.

Problem Formulation

The first part of this thesis deals with state estimation for networked systems (such as the one shown in Fig. 1.1) where communication of sensor measurements is costly. The state of the dynamic system is to be estimated recursively from measurements by the distributed sensors and a model of the system. Sensor measurements shall, however, only be exchanged sporadically between the network agents in order to keep the network traffic low. Figure 2.1 depicts the considered estimation problem (the NCS in Fig. 1.1 has been reduced to the components essential for the state estimation problem).

All agents in Fig. 2.1 are of the same type: each one has access to local sensors and runs a state estimator. This is motivated by the Balancing Cube

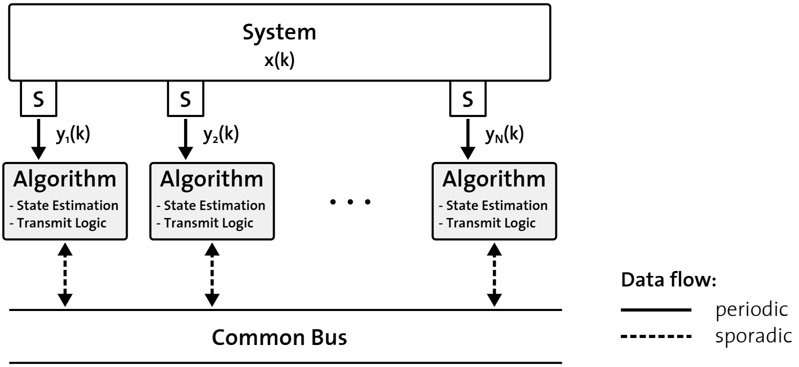


Figure 2.1 Distributed state estimation problem. The graphic is condensed from Fig. 1.1 to include only the components relevant for state estimation. Each agent i estimates the system state $x(k)$ based on its own measurements $y_i(k)$ (received periodically) and on other agents' data communicated sporadically over the bus. Each agent is responsible for making the decision at time k of whether or not to broadcast its local measurement $y_i(k)$.

test bed with six identical balancing modules. Each module is equipped with local sensors and computes state estimates to be used for feedback control. The algorithms developed in this part readily apply to different estimation scenarios, where only some agents are equipped with local sensors. In a remote estimation scenario, for example, one or several pure estimator agents (without local sensors) are connected to the communication bus at a remote location (as shown in Fig. 1 of Paper I). They receive data from the sensor agents and use it to estimate and monitor the system's state. For simplicity, we assume in the description below that every agent has local sensors and a state estimate is to be maintained on every agent. The type of sensors on every agent can be different.

For the analysis of the event-based state estimation method presented herein, we consider a discrete-time, stochastic, linear time-invariant (LTI) system. The key ideas presented below are not confined to this specific class of dynamic systems and can readily be extended to time-varying or nonlinear system. We consider the LTI process

$$x(k) = A\,x(k-1) + B\,u(k-1) + v(k-1), \tag{2.1}$$

where $x(k)$ is the process state vector, $u(k)$ is a known input vector, $v(k)$ is zero-mean process noise with a known covariance, and k is the discrete-time index. The process is observed by N sensors with measurements

$$\begin{aligned} y_1(k) &= C_1\, x(k) + w_1(k) \\ y_2(k) &= C_2\, x(k) + w_2(k) \\ &\vdots \\ y_N(k) &= C_N\, x(k) + w_N(k), \end{aligned} \tag{2.2}$$

where $w_1(k)$ to $w_N(k)$ are zero-mean, mutually independent measurement noise vectors with covariance matrices R_1 to R_N. When all measurements are combined, the system is assumed to be detectable; that is, (A, C) is detectable, where $C := [C_1^{\mathrm{T}}, \dots, C_N^{\mathrm{T}}]^{\mathrm{T}}$. Notice that we do not assume detectability from any individual sensor ((A, C_i) does not need to be detectable for any i), and we do not make any assumption on the dynamic coupling (A is a general square matrix).

The results of this part of the thesis address the following estimation problem for the networked system in Fig. 2.1 with process and measurement equations given by (2.1) and (2.2):

PROBLEM 2.1 On every agent i and at every time step k,

(i) estimate the system state $x(k)$ based on the system model (2.1), (2.2) and measurements received over the common bus and from the local sensors; and

(ii) transmit the local measurement $y_i(k)$ if, and only if, the other agents in the network cannot predict this measurement "well enough" (based on the system model and past measurement data).

■

The considered problem has two parts: in addition to a state estimation algorithm, some *transmit logic* must be designed, which locally decides on every sensor whether or not to transmit a measurement. Different solutions to this problem are developed herein; the methods differ in their choice of the state estimation algorithm (problem part (i)), and/or in the way they assess a prediction as "good enough" (part (ii)), which will yield different implementations of the transmit logic.

The estimation problem given by part (i) only, when neglecting part (ii) and communicating all measurements $y_1(k), \dots, y_N(k)$ at every instant k, is solved by the classic Kalman filter [27, 28]. The Kalman filter is the best unbiased linear estimator that minimizes the estimation error variance. In addition, if process and measurement noise are Gaussian distributed, the

Kalman filter is the optimal Bayesian estimator in the sense that it keeps track of the full conditional probability density function of the state $x(k)$ conditioned on all past measurements. We refer to this Kalman filter (which has access to all measurements) as the *full communication Kalman filter*; it serves as a reference and comparison to the event-based estimator developed herein.

REMARK 2.1 The estimation problem in Problem 2.1 is not formulated as an optimization problem; we do not seek, for example, an optimal transmit logic (as is done for a scalar problem in [25], for example). Instead, we take the full communication case as the baseline (for which the Kalman filter is well-known as the optimal state estimator), and then decide for each measurement if we can do without while not compromising the estimator performance too severely. This design approach results in tractable event-based estimation algorithms, whose design and implementation is a straightforward extension of well-known discrete-time estimation techniques (such as the Kalman filter). ■

REMARK 2.2 For the results herein, we take an abstract view of the network and assume the communication to be ideal; that is, we assume that data transmission is instantaneous (without delay) and that no data is lost. This may be partly ensured by low level communication protocols. The Controller Area Network (CAN) that is implemented on the Balancing Cube is an example of a reliable network, where data loss occurs very rarely. (See Sec. 3.1 for a discussion of extensions of the results herein to unreliable networks.) ■

REMARK 2.3 For the design and analysis of the estimation algorithms herein, we assume that the input $u(k)$ in (2.1) is known to the estimators on all agents. In practice, this may be the case if $u(k)$ is a known reference input or if the inputs are communicated to the estimators at every time step k (required, for example, when $u(k)$ is computed locally from feedback control laws). Section 3.1 briefly discusses how the periodic communication of inputs can be avoided in the latter case.

In Paper II and Paper III, the event-based estimation algorithms are used on the Balancing Cube for event-based feedback control; that is, the inputs $u(k)$ to the actuators are computed from the local event-based state estimates and a state-feedback law. If every agent broadcasts its control input to all other agents at every time step, and if an ideal network with identical estimates on all agents is assumed, the stability of the feedback control system follows from the stability of the event-based estimator and the stability of the state-feedback controller (similar to the separation principle for linear systems, see [21] for example). In any practical situation where the network is not perfectly ideal, however, the estimates of the individual agents are not exactly identical, and stability of the closed-loop system must be analyzed separately from the stability of the controller and the stability of the estimator.

Even though closed-loop stability is not shown herein for the case of combining the event-based estimators with state-feedback controllers, the effectiveness of this approach is validated experimentally by showing that the event-based control system can stabilize the Balancing Cube. (See Sec. 3.1 for a discussion of future work in this direction.) ■

Key Ideas of the Event-Based State Estimation Approach

Figure 2.2 depicts the algorithms that are implemented on every agent from Fig. 2.1 in order to address Problem 2.1. The key ideas of the event-based state estimation approach are explained below by explaining the function of each block in Fig. 2.2.

In order to link the transmit decision of a particular measurement to its relevance for state estimation (Problem 2.1, (ii)), each agent implements a copy of the *common estimator*, which computes a state estimate based on the system model (2.1), (2.2) and measurements that are received over the common bus. Notice from Fig. 2.2 that the common estimator uses local sensor measurements only if they are also transmitted over the bus. Since each agent has access to the data on the bus, the common estimator represents the *common information* in the network. Each agent makes the transmit decision for its local measurements based on the common estimator: generally speaking, if the common estimator's prediction of a measurement is already satisfactory by some measure of estimation performance, this measurement need not be communicated; if the prediction is poor, the measurement is transmitted so that all agents can update their estimates.

In principle, the common estimator can be any recursive state estimation algorithm that fuses model-based predictions with measurement updates (see, for example, [29] for an overview of recursive state estimation techniques). Herein, we consider Kalman filters [27,28] (in Paper I and Paper II) and a Luenberger observer [30] with suitable modifications (Paper III). Both estimators recursively compute $\check{x}(k|k)$ and $\check{x}(k|k-1)$, the common estimate of $x(k)$ based on all measurements received over the network up to and including time k and time $k-1$, respectively. The Kalman filter additionally

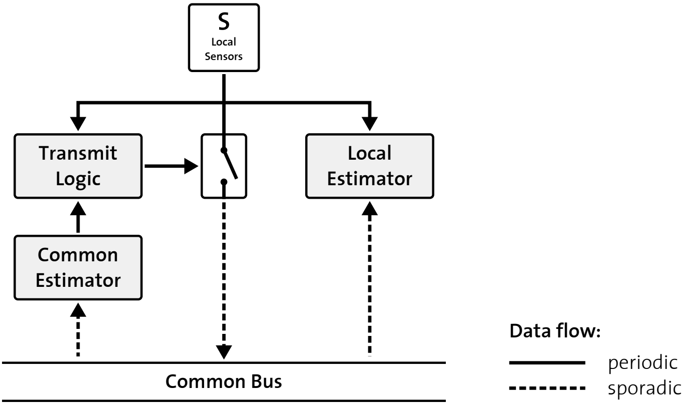


Figure 2.2 Event-based state estimation approach. The block diagram shows the algorithms implemented on a single agent of Fig. 2.1. Each agent runs a copy of the common estimator, which estimates the process state based on data received over the common bus and thus represents the common information in the network. The agent uses the common estimate (or its error variance) together with a transmit logic to decide whether or not to broadcast the local measurements. Optionally, a local estimator can be used to obtain a better estimate by exploiting the local sensor data in addition to the data from the bus. Solid lines indicate periodic data links and dashed lines indicate event-based communication.

computes the variance of the estimation error $\check{e}(k|k) = x(k) - \check{x}(k|k)$ and the variance of the prediction error $\check{e}(k|k{-}1) = x(k) - \check{x}(k|k{-}1)$, which we denote by $\check{P}(k|k)$ and $\check{P}(k|k{-}1)$, respectively.

The design of the *transmit logic* in Fig. 2.2 depends on the way estimation performance is quantified; that is, how one decides whether or not the common prediction of a particular measurement is "good enough." In Paper II and Paper III, we consider a constant threshold logic on the prediction error of a measurement; that is, we use the transmit rule

$$\text{transmit } y_i(k) \quad \Leftrightarrow \quad \|y_i(k) - C_i\,\check{x}(k|k{-}1)\| \geq \delta_i, \tag{2.3}$$

where the scalar δ_i is a tuning parameter and $\|\cdot\|$ is some vector norm. Since the triggering rule (2.3) is based on the real-time measurement, we refer to it as *measurement-based triggering.* Another quantity that represents a measure of prediction quality is the prediction variance of a measurement; it essentially captures the uncertainty in predicting the measurement based on past data and the process model. A triggering rule where the transmission of a measurement is triggered by a condition on the prediction variance

$\check{P}(k|k-1)$ is referred to as *variance-based triggering*; for example,

$$\text{transmit } y_i(k) \quad \Leftrightarrow \quad C_i\,\check{P}(k|k-1)\,C_i^{\mathrm{T}} + R_i \geq \delta_i, \tag{2.4}$$

($C_i\,\check{P}(k|k-1)\,C_i^{\mathrm{T}} + R_i$ is the prediction variance of the scalar measurement $y_i(k)$). A variance-based triggering rule similar to (2.4) is used in Paper I.

The triggering rules (2.3) and (2.4) define the instants at which data is transmitted. As opposed to continuous-time or discrete-time state estimation (see [29]), data transmission does not explicitly depend on time, but is triggered by *events* that are defined by the right-hand sides of (2.3) and (2.4). Following common terminology in event-based control (see [31], for example), the transmit logic block is also called an *event generator* to emphasize its role of generating events (data transmissions). Both the measurement-based triggering rule (2.3) and the variance-based triggering rule (2.4) can be evaluated locally by each agent, which allows for the distributed implementation of the event-based estimation method.

The common estimator together with the transmit logic constitute the event-based state estimator. The common estimate is the same on all agents (the estimator uses the same model and receives the same input data under the assumption of an ideal network). Therefore, the common estimator ensures consistency in the network. To this end, it is important that local sensor data is used to update the common estimate if, and only if, it is also shared with the other agents. In order to improve the state estimation performance as compared to the common estimator, each agent can implement a second estimator, called *local estimator* (see Fig. 2.2). The local estimator exploits all data that is locally available; that is, the local sensor data at every time step (solid line) and the data sporadically received over the network (dashed line). The use of the local estimator is optional; if the performance of the common estimator is satisfactory, the agent may use the common estimate for whatever is the purpose of state estimation on the agent (for example, feedback control).

REMARK 2.4 Setting the transmit thresholds δ_i in (2.3) or (2.4) to zero means that measurement data is transmitted at every time step k; that is, the performance of the standard discrete-time state estimator (Kalman filter or Luenberger observer) is recovered. Therefore, the proposed approach to event-based estimation can be regarded as a direct extension of well-known state estimation methods for discrete-time systems (with periodic communication of sensor data). The resulting algorithms are straightforward to design and implement: building on top of the design of a standard discrete-time state estimator, the only additional tuning parameters are the transmit thresholds δ_i. ∎

Remark 2.5 The two triggering approaches (2.3) and (2.4) yield fundamentally different analysis problems. Because the rule (2.3) depends on the measurement $y_i(k)$, which is a random variable according to (2.2), the transmit decision itself becomes a random variable. When the triggering rule (2.4) is used with a Kalman filter, on the other hand, the evolution of the filter variance $\check{P}(k|k-1)$ (and therefore of the transmit decision) is governed by a deterministic update equation. The variance iteration can be analyzed offline, and its solutions are typically asymptotically periodic as is discussed in Paper I. ■

The key features of the proposed event-based state estimation approach can be summarized as follows:

Event-based communication. Data transmission is not time-based (periodic or continuous), but caused by triggering conditions on real-time measurement data such as in (2.3) or on the estimation variance as in (2.4). Data transmission is thus linked to the relevance of the data for the state estimation task.

Prior model knowledge. The state estimators that are used (Kalman filter and modified Luenberger observer) fuse state predictions based on the model (2.1) with measurement data that is received over the network. Thus, the estimators inherently use model-based predictions to (partly) compensate for not receiving some measurements. Rather than using the last received measurements (or a state estimate based on those measurements), state and measurement estimates are updated using the process model even if no data is received. Transmission of measurement data only occurs if the prior knowledge given by the set of past measurements *and* the model is not sufficient.

Information sets. The state estimates by the common estimator and by the local estimator are based on different information sets. The local estimator computes an estimate based on all measurement data that is locally available (local sensor data and data from the network), while the common estimator uses the data from the network only. The common estimate essentially represents the baseline information: every agent has at least this information about the system state. It is essential to distinguish local information from common information in order to be able to make a measurement transmit decision based on the measurement's *relevance for all other agents.*

Distributed decision making. Each agent is responsible for making the transmit decisions for its local measurements. Since the agent knows what model the other agents use for making their state predictions (in the common estimator), the agent can assess when it is time to transmit new data to its peers. The agents cooperate in that they share the complete system model, and there is an implicit agreement that each agent sends new data whenever relevant.

Overview of Papers

Papers I to III, which include the main results of Part A of this thesis, all follow the basic approach outlined above. They differ in the estimator implementation (Kalman filter or modified Luenberger observer), the triggering rule (measurement-based triggering or variance-based triggering), and whether or not a local estimator is implemented (see Table 2.1 for an overview). The context and the contribution of each of these papers are summarized next.

Table 2.1 Overview of the papers on event-based state estimation that are included in this thesis.

	Estimation Algorithm	**Triggering Rule**	**Local Estimator**	**Focus**
Paper I	Kalman filter	variance-based	no	theory
Paper II	Kalman filter	measurement-based	yes	experiments
Paper III	switching Luenberger-type observer	measurement-based	no	theory & experiments

Paper I

[P1] S. Trimpe and R. D'Andrea, "Event-based state estimation with variance-based triggering," submitted to *IEEE Transactions on Automatic Control.*

Context This paper presents a Kalman filter as state estimator and variance-based triggering (similar to (2.4)) for the transmit decision. The paper focuses of the theoretical analysis and the convergence properties of the event-based state estimator.

Contribution The variance-based triggering condition is introduced as a novel triggering mechanism, as opposed to triggering on real-time state or measurement data (see discussion of related work in the paper). The update equation for the Kalman filter prediction variance $\check{P}(k|k-1)$ is derived; it represents a new type of Riccati equation with switching that corresponds to the available measurements at a time step and that depends on the variance at the previous step. Simulation results of this Riccati equation for the Balancing Cube system and other examples suggest that the variance iteration asymptotically converges to a periodic solution. For the scalar problem (scalar process with a single scalar measurement), we prove the asymptotic periodicity of the variance solution under certain assumptions. Since a periodic sequence for $\check{P}(k|k-1)$ corresponds to a periodic transmit sequence via (2.4), this result links time-triggered (periodic) and event-based state estimation. The event-based approach provides a practical way to design periodic sensor transmit rates for a multi-sensor network by specifying tolerable bounds on the estimator performance.

Related Publications

[R3] S. Trimpe and R. D'Andrea, "Reduced communication state estimation for control of an unstable networked control system," in *Proc. of the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, Dec. 2011, pp. 2361–2368.

[R4] S. Trimpe and R. D'Andrea, "Event-based state estimation with variance-based triggering," in *Proc. of the 51st IEEE Conference on Decision and Control*, Maui, HI, USA, Dec. 2012, pp. 6583–6590.

(The numbering of the publications corresponds to the list of all publications within these doctoral studies provided in Sec. 2.3.)

We first presented event-based state estimation with variance-based triggering in the conference publication [R3]. This publication includes experimental results (not contained herein) of applying the method on the Balancing Cube, where periodic sensor transmit schedules are obtained from a periodic solution to the variance iteration and implemented for the cube's sensors. The event-based state estimators on each agent are used for feedback control to stabilize the cube.

A preliminary version of the convergence result for the switching Riccati-type equation (the variance iteration of the event-based estimator) for the scalar problem was presented in [R4], but all proofs of intermediate results (Propositions 1 to 6 and Lemmas 1 to 3 in Paper I) were omitted.

Paper II

[P2] S. Trimpe and R. D'Andrea, "An experimental demonstration of a distributed and event-based state estimation algorithm," in *Proc. of the 18th IFAC World Congress*, Milan, Italy, Aug. 2011, pp. 8811–8818.

Context This paper demonstrates the use of event-based state estimation for feedback control in experiments on the Balancing Cube.[1] The blocks shown in Fig. 2.2 are implemented on the cube's six balancing modules.[2] The measurement-based triggering rule (2.3) is applied for each of the twelve sensors that are used in this experiment. The estimate by the local estimator in Fig. 2.2 is used as input to a state-feedback controller that computes the commands for the local actuator.

Contribution The basic approach for distributed and event-based state estimation as outlined in the introduction of this section was first presented in this paper. The paper's focus is on the experimental demonstration of using the event-based estimation method for event-based control on the Balancing Cube by combining it with a static state-feedback controller on every agent (a theoretical analysis of convergence or stability properties is not contained). The experimental results show that the cube can balance with the distributed and event-based controllers, and a significant reduction of the average communication rates is achieved compared to periodic communication. The results exemplify the expected trade-off between communication and control performance: in the presented experiment, a reduction of average communication rates by roughly a factor 16 is accompanied by a decrease in control performance (measured as the state RMS) by a factor of 1.5.

Paper III

[P3] S. Trimpe, "Event-based state estimation with switching static-gain observers," in *Proc. of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Santa Barbara, CA, USA, Sep. 2012, pp. 91–96.

---

[1] In these experiments, the cube balances on one of its edges rather than on a corner, which avoids the discussion of secondary issues in the paper such as a more complex model and control design. The cube can, however, also balance on a corner using event-based control as we demonstrated during the interactive presentation of the paper at the 2011 IFAC World Congress (see Sec. 2.6 and Fig. 2.3).

[2] The paper uses slightly different terminology. The local estimator is described in Sec. 2.2 as the "receiver algorithm," and the common estimator with transmit logic is described in Sec. 2.3 as the "sender algorithm."

Context In contrast to Paper I and Paper II, this paper uses a linear estimator that switches between static gains, which are obtained from the design of a centralized Luenberger observer with periodic communication of all measurements. Combined with the measurement-based triggering rule (2.3), the resulting event-based estimator mimics the Luenberger observer with full communication. The paper presents theoretical guarantees for the estimation error, as well as experimental results of using the method for event-based control on the cube.

Contribution The event-based estimator proposed in this paper is compared to the Luenberger observer with periodic communication of all measurements: the difference between the state estimates of the two estimators is shown to be bounded (even for the case that $v(k)$ and $w_i(k)$ in (2.1) and (2.2) are unbounded). The state estimator switches between pre-computed gains and is therefore computationally less expensive than the Kalman filter used in Paper II. The reduction of average sensor communication rates achieved by using the switching Luenberger-type observer for feedback control is demonstrated in experiments on the Balancing Cube; the experimental results are qualitatively similar to those in Paper II.

2.2 The Balancing Cube: A Test Bed for Distributed Estimation and Control (Part B)

The design challenge of building a self-balancing dynamic sculpture as described in Sec. 1.1 was successfully met: the Balancing Cube shown in Fig. 1.2 (left) was realized as part of these doctoral studies.

The cube is stabilized by six rotation arms, called the balancing modules, which are mounted on the cube's inner faces. Each balancing module is equipped with embedded sensing, actuation, computation, and power; and it is connected to a communication bus. The Balancing Cube is therefore a concrete realization of the networked control system (NCS) in Fig. 1.1. The individual components of the cube's NCS are summarized in Table 2.2. The details of the design, the modeling, and the control of the cube are described in Paper IV. A video of the system is available in [32].

The development of the Balancing Cube started in fall 2007 within the project-based design class *!And Yet It Moves* taught by Raffaello D'Andrea and Matt Donovan at ETH Zurich. The author of this thesis joined the class in spring 2008 as a teaching and research assistant. At the end of this two-semester class, the team (consisting of the instructors, the author, two technicians, and ten Master-level students from electrical engineering,

Table 2.2 Technical realization on the Balancing Cube of the components of the networked control system (NCS) shown in Fig. 1.1.

NCS Component	Technical realization on the Balancing Cube
System	Balancing Cube (a multi-body 3D inverted pendulum)
Common bus	Controller Area Network (CAN)
Control agent:	Balancing module (rotating arm):
Sensors (S)	Absolute encoder (measurement of arm angle) 3-axis accelerometer (mounted on the cube body) 3-axis rate gyro (mounted on the cube body)
Actuator (A)	DC motor (rotating the arm relative to cube body)
Algorithm block	Single-board computer, 200 MHz ARM9 processor

mechanical engineering, and computer science) completed the conceptual design of the Balancing Cube and the construction of a functional, one-dimensional prototype (an earlier iteration of the pendulum shown in Fig. 1.2 (right)), which included most of the components to be used in the final design of the cube. After another year, the Balancing Cube was completed as part of these doctoral studies in August 2009, when it first balanced autonomously on one of its corners. As intermediate steps in the development process, the cube was balanced (first on an edge, then on a corner) using state feedback from the motion capture system of the institute's Flying Machine Arena [33].

Since its completion, the Balancing Cube has served as a research platform for the results herein, it has supported twelve student projects (see Sec. 2.5), and it was used as a means of communicating controls research to the general public, including three public exhibitions (see Sec. 2.6).

Overview of Papers

The design, modeling, and control of the Balancing Cube as a test bed for distributed estimation and control is described in Paper IV. The cube represents a multi-body 3D inverted pendulum [34] and, as such, is a rich platform for research in dynamics an control (see references in Paper IV for other results on the control of 3D pendulum systems). At its time of completion, the Balancing Cube was the only cube that could balance autonomously on a corner. At that time, the only other self-balancing cube that the author was aware of was sold by Quanser Inc., [35]. Quanser's cube can balance on a fixed edge using a single actuation mechanism. Since that time, the *Cubli* (Swiss German for "small cube") has been developed at IDSC as a follow-up project of the Balancing Cube. The Cubli is equipped with three momentum wheels, which it uses together with a braking mechanism to jump up and to balance, [36].

The Balancing Cube's main function within this thesis is as a test bed for distributed and networked estimation and control. Being a concrete realization of the NCS in Fig. 1.1 (see Table 2.2), the cube motivated the problem formulation investigated in Part A of this thesis, and it served as an experimental test bed for validating and evaluating the obtained results (see Sec. 2.1). In particular, we were able to show through experiments on the cube that the event-based estimation method developed in Part A can be used for event-based control.

In addition to supporting the research on distributed and event-based state estimation and control, the development of the cube led to research results in other areas. An example is the accelerometer-based tilt estimation algorithm presented in Paper IV, which can be used to estimate the tilt of any rigid body with only rotational degrees of freedom. The estimate is independent of the system dynamics and no dynamic model is required for the estimator design. Another example is the matrix exponential result in Paper V, which was motivated by the concrete problem of modeling the cascaded control system on the cube. The individual contributions of these two papers are summarized below.

Additional research topics that are not contained in this thesis were explored on the Balancing Cube as part of student projects. These topics include system identification, modeling of gear backlash, trajectory learning, and self-tuning control (see Sec. 2.5).

Paper IV

[P4] S. Trimpe and R. D'Andrea, "The Balancing Cube: A dynamic sculpture as test bed for distributed estimation and control," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 48–75, Dec. 2012.

Context This paper describes the design, modeling, and control of the Balancing Cube. The control system design is presented for the case where all modules share their sensor data over the network at every time step (i.e. the dashed lines connecting the algorithm blocks with the common bus in Fig. 1.1 would be solid). Since every agent has access to all measurements, the design of a controller and a state estimator can be considered as centralized problems. The cube's balancing performance is demonstrated with experimental data.

Contribution The Balancing Cube is introduced as a test bed for distributed estimation and control. The system represents a multi-body 3D inverted pendulum. The cube distinguishes itself from other inverted pendulum systems by its distributed control agents (the balancing modules), and from many other test beds for multi-agent or distributed control by possessing unstable and coupled dynamics.

The control system design for the cube is separated into the design of a standard Linear Quadratic Regulator (LQR) as state-feedback controller and a state estimator that is tailored to the specific problem. The state estimator exploits the facts that the cube has only rotational degrees of freedom, and that measurements from multiple inertial sensors are available, to generate an estimate of the cube's tilt that is independent of the rigid body dynamics. In particular, the estimator provides an accurate tilt estimate for whatever motion of the cube (slow or fast), and no assumption on near equilibrium configuration is made. As opposed to the event-based estimators in Part A, this estimator relies on the periodic transmission of all sensor data, but it does not require a dynamic system model. It is thus inherently robust to modeling errors or changes in the system (for example, in the mass or module configuration). The developed tilt estimation algorithm is applicable to any rigid body with only rotational degrees of freedom that is equipped with multiple inertial sensors.

The results in this paper show that the design of a dynamic sculpture as described and motivated in Sec. 1.1 (Part B) is actually feasible. The modeling and design techniques developed in this paper can readily be extended to build other balancing sculptures.

Related Publications

[R2] S. Trimpe and R. D'Andrea, "Accelerometer-based tilt estimation of a rigid body with only rotational degrees of freedom," in *Proc. of the IEEE International Conference on Robotics and Automation*, Anchorage, AK, USA, May 2010, pp. 2630–2636.

The estimation algorithm for estimating the tilt of the Balancing Cube from measurements of multiple inertial sensors was first introduced in [R2]. The paper also presents the experimental validation of the algorithm using a motion capture system.

Paper V

[P5] S. Trimpe and R. D'Andrea, "A limiting property of the matrix exponential," submitted to *IEEE Transactions on Automatic Control.*

Context This paper presents a limiting property of the matrix exponential, which can be used to obtain simplified models of dynamic systems that exhibit significantly different time scales, such as a cascaded control system with fast inner feedback loops. The Balancing Cube uses such a cascaded control system with a fast velocity feedback loop implemented locally on each balancing module. The motivation for investigating the limiting property of the matrix exponential originated from the concrete problem of modeling the cascaded control system of the Balancing Cube.

Contribution We state conditions on the matrix function $K(\alpha)$ (α a scalar parameter), for which the following limiting property holds (for $t > 0$ and any complex-valued matrix A):

$$\lim_{\alpha\to\infty} \exp\left(\begin{bmatrix} A_{11}-K(\alpha) & A_{12} \\ A_{21} & A_{22} \end{bmatrix} t\right) = \begin{bmatrix} 0 & 0 \\ 0 & e^{A_{22}t} \end{bmatrix}. \tag{2.5}$$

The property (2.5) holds if $-K(\alpha)$ becomes arbitrarily small ("negative infinite") in one of the following ways: (i) the log-norm of $-K(\alpha)$ approaches negative infinity; or (ii), for certain polynomial functions $-K(\alpha)$, the matrix associated with the largest power of α is stable (its eigenvalues have negative real part). The result (ii) is obtained as an extension of established results by Campbell et al., [37,38]. The result (i) is proven independently of these.

If matrix A is the system matrix of a linear system with state vector (x_1, x_2), and if $K(\alpha)$ is a static feedback gain matrix applied on the states x_1, then the result (2.5) means a decoupling of the states in the limit as the controller gains become arbitrarily large (see introductory example in Paper V, equations (3)–(5)).

Related Publications

[R1] S. Trimpe and R. D'Andrea, "A limiting property of the matrix exponential with application to multi-loop control," in *Proc. of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, China, Dec. 2009, pp. 6419–6425.

We first derived the condition (i) (the log-norm of $-K(\alpha)$ approaching negative infinity) for the result (2.5) to hold in [R1]. Therein, we also present a time-scale separation algorithm that makes use of the property (2.5) to obtain a simplified model of a cascaded control system with fast inner loops, and we use this modeling approach in the design of a controller that stabilizes the inverted pendulum in Fig. 1.2 (right). The time-scale separation algorithm is summarized and applied to the Balancing Cube in Paper IV.

2.3 List of Publications

This section lists all articles and conference contributions that were published or submitted during these doctoral studies. The publications [P1]–[P5] are included in this thesis (Part A and Part B). The publications [R1]–[R4] are related results that were (partly) integrated into the journal articles [P1, P4, P5].

The journal manuscripts [P1] and [P5] are currently under review. Preliminary results of [P1] are published in [R1]; and the results in [P5] are based on the conference publications [R3, R4].

Publications Included in this Thesis

(order as appearing in this thesis)

[P1] S. Trimpe and R. D'Andrea, "Event-based state estimation with variance-based triggering," submitted to *IEEE Transactions on Automatic Control*.

[P2] S. Trimpe and R. D'Andrea, "An experimental demonstration of a distributed and event-based state estimation algorithm," in *Proc. of the 18th IFAC World Congress*, Milan, Italy, Aug. 2011, pp. 8811–8818.

[P3] S. Trimpe, "Event-based state estimation with switching static-gain observers," in *Proc. of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Santa Barbara, CA, USA, Sep. 2012, pp. 91–96.

[P4] S. Trimpe and R. D'Andrea, "The Balancing Cube: A dynamic sculpture as test bed for distributed estimation and control," *IEEE Control Systems*, vol. 32, no. 6, pp. 48–75, Dec. 2012.

[P5] S. Trimpe and R. D'Andrea, "A limiting property of the matrix exponential," submitted to *IEEE Transactions on Automatic Control*.

Related Publications

(chronological order)

[R1] S. Trimpe and R. D'Andrea, "A limiting property of the matrix exponential with application to multi-loop control," in *Proc. of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, China, Dec. 2009, pp. 6419–6425.

[R2] S. Trimpe and R. D'Andrea, "Accelerometer-based tilt estimation of a rigid body with only rotational degrees of freedom," in *Proc. of the IEEE International Conference on Robotics and Automation*, Anchorage, AK, USA, May 2010, pp. 2630–2636.

[R3] S. Trimpe and R. D'Andrea, "Reduced communication state estimation for control of an unstable networked control system," in *Proc. of the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, Dec. 2011, pp. 2361–2368.

[R4] S. Trimpe and R. D'Andrea, "Event-based state estimation with variance-based triggering," in *Proc. of the 51st IEEE Conference on Decision and Control*, Maui, HI, USA, Dec. 2012, pp. 6583–6590.

Technical Report

[TR1] S. Trimpe and R. D'Andrea, "Numerical models and controller design parameters for the Balancing Cube," IDSC, ETH Zürich, Tech. Rep., 2012. [Online].
Available: `http://dx.doi.org/10.3929/ethz-a-007343301`

2.4 Invited Talks

Below is a list of seminars and workshops where the author of this thesis was invited to present research results of his doctoral studies. Presentations at international conferences of the conference papers in Sec. 2.3 are not included. Invited talks addressing a general audience are listed in Sec. 2.6 as part of the author's outreach activities.

[T1] Hamburg University of Technology (TUHH), Hamburg, Germany, "The Balancing Cube," *Alumni-Workshop at the Institute of Control Systems (Prof. H. Werner)*, Nov. 2009.

[T2] Ruhr-Universität Bochum (RUB), Bochum, Germany, "The Balancing Cube," *Seminar at the Institute of Automation and Computer Control (Prof. J. Lunze)*, Dec. 2009.

[T3] Ruhr-Universität Bochum (RUB), Bochum, Germany, "Reduced communication state estimation for networked control systems," *Seminar at the Institute of Automation and Computer Control (Prof. J. Lunze)*, Nov. 2011.

[T4] Royal Institute of Technology (KTH), Stockholm, Sweden, “Event-based state estimation for networked control systems,” *Seminar at the Automatic Control Laboratory (Prof. K. H. Johansson)*, May 2012.

[T5] University of Southern California (USC), Los Angeles, CA, USA, “Event-based state estimation for networked control systems,” *Seminar at the Center for Robotics and Embedded Systems (Prof. F. Valero-Cuevas, Prof. G. S. Sukhatme)*, Sep. 2012.

[T6] California Institute of Technology (Caltech), Pasadena, CA, USA, “Event-based state estimation for networked control systems,” *Seminar at the Control and Dynamical Systems group (Prof. R. M. Murray, Prof. J. C. Doyle)*, Sep. 2012.

[T7] Technische Universität München (TUM), Munich, Germany, “Event-based state estimation for networked control systems,” *Workshop on “Event-based control and optimization” (Deutsche Forschungsgemeinschaft Schwerpunktprogramm 1305)*, Oct. 2012.

2.5 Supervised Student Projects

Below is a complete list of student projects that were supervised at ETH Zurich as part of the author’s doctoral studies.

Master Thesis

six months, full-time research project

[M1] D. Akay, “Alternative modeling tools and system identification for the Balancing Cube,” Fall 2009.

[M2] M. Spirig, “A learning strategy for swinging motions of an inverted pendulum,” Fall 2010.

[M3] S. Dössegger, “Improving the balancing performance of the cube – adaptation and learning to make the cube balance more steadily,” Spring 2012.

Master Semester Project

semester-long, part-time project

[S1] G. Mohanarajah, “Single-board computer operating system,” Fall 2008.

[S2] V. Baumann, “Modeling and simulation of the Balancing Cube,” Spring 2009.

[S3] L. Wunderli, “Determining the mode of the Balancing Cube based on local sensor information,” Fall 2009.

[S4] A. Köberl, “Backlash modeling for the Balancing Cube,” Spring 2010.

[S5] N. Voellmy, “User interface for the Balancing Cube,” Fall 2010.

[S6] K. Nottensteiner, “Moving the Balancing Cube – a gain scheduling approach,” Fall 2010.

[S7] K. Schindler, “Event-based state estimation on the cube – experimental study of recent event-based state estimation algorithms,” Fall 2012.

Studies on Mechatronics

semester-long, part-time literature study

[L1] L. Wunderli, “Attitude estimation for the Balancing Cube,” Spring 2009.

[L2] A. Widmer, “Networked and distributed control systems – physical control systems similar to the Balancing Cube,” Fall 2011.

2.6 Outreach

Throughout the doctoral project, the Balancing Cube and corresponding research were presented to the general public and to the media in exhibitions, festivals and lab demonstrations.

Exhibitions

Nacht der Forschung (Researchers’ Night), Zurich, Switzerland, Sept. 25, 2009.

Organized by ETH Zurich and the University of Zurich as part of the *European Researchers’ Night*, more than 25’000 visitors joined approximately 600 researchers for this event around lake Zurich to gain insights in the worlds of science, technology, medicine, and industry.
(`http://www.nachtderforschung.ethz.ch/en`)

Festival Della Scienza, Genoa, Italy, Oct. 23–25, 2009.

The annual *Festival Della Scienza* is one of the largest science festivals in

Italy.
(`http://www.festivalscienza.eu`)

18th IFAC World Congress, Milan, Italy, Aug. 31, 2011.

The triennial IFAC (International Federation of Automatic Control) World Congress is one of the largest international conferences for automatic control. The Balancing Cube was shown in one of the interactive sessions (see Fig. 2.3).
(`http://www.ifac2011.org/`)

Lab Demonstrations

From 2009 to 2013, the Balancing Cube was shown in more than 120 lab demonstrations to guests from academia, schools, media, government, and industry.

Invited Talks to a General Audience

Swiss Science Center Technorama, Winterthur, Switzerland, "The Balancing Cube," General assembly of Schweizerische Gesellschaft Pro Technorama, May 2011.

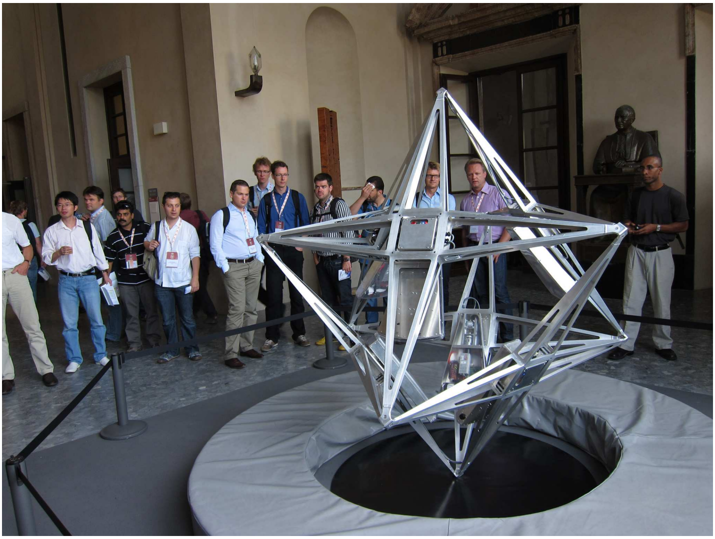


Figure 2.3 The Balancing Cube at the 2011 IFAC World Congress in Milan, Italy. (Photo by Raffaello D'Andrea.)

IEEE Conference on Decision and Control (CDC), Orlando, FL, USA, “A cube that balances itself on a corner,” *Workshop for high school students and teachers on “Ideas and Technology of Control Systems,”* Dec. 2011.

IEEE Conference on Decision and Control (CDC), Maui, HI, USA, “A cube that balances itself on a corner,” *Workshop for middle and high school students and teachers on “Ideas and Technology of Control Systems,”* Dec. 2012.

Selected Media Coverage

NZZ Format, “Die Intelligenz der Roboter,” Swiss TV (SF1), Switzerland, Apr. 2010.

IEEE Automaton Blog, “Amazing Robotic Sculpture Balances Itself on One Corner,” May 2010.

Daily Planet, Discovery Channel, Canada, Jan. 2011.

World Radio Switzerland, Switzerland, Jun. 2011.

c’t Magazin, magazine for computer technology, Germany, Jul. 2012.

3

Future Directions

Two main themes in this work are the consideration of the communication network in a networked control system (NCS) as a shared resource, and the design of control and estimation algorithms in tandem with a network access strategy. While managing the communication cost is important even for today's NCS, it will be vital for the next generation of engineering systems that will tightly integrate the physical world with computation and communication. Referred to as cyber-physical systems (CPSs) [17,18], these highly integrated systems will extend present-day networked systems in the number of interconnected entities, size, and complexity. As a consequence, communication costs will become an even more significant factor which must be taken into account in the control system design if CPSs are to meet their potential in envisioned application areas such as transportation, power systems, smart buildings, mobile robots, and process industry.

The results in this thesis demonstrate the potential of event-based estimation and control strategies to ensure an effective use of the shared communication resource in NCS. Open questions and promising directions for future research that arise from the results herein are briefly discussed in this chapter.

3.1 Distributed and Event-Based State Estimation (Part A)

Open questions or direct extensions of the event-based state estimation algorithms presented in Part A of this thesis are pointed out in the respective papers. Here, we discuss extensions and directions for future research in a broader sense.

Event-Based Output-Feedback Control

The experimental results on the Balancing Cube showed that the event-based estimation methods developed herein can be combined with state-feedback controllers to form an event-based output-feedback control system. The analysis in this work, however, focuses on theoretical guarantees for the pure estimation problem (such as boundedness of the estimation error). The theoretical analysis of the event-based output-feedback system as used to balance the cube in Paper II and Paper III is an interesting direction for future research. In particular, the question of whether theoretical stability results can be obtained for the event-based control scheme used herein is an obvious candidate for further investigation.

When the control input $u(k)$ is computed by the individual agents based on the agent's local estimate from the event-based estimator, the full input vector is not known by all agents. In order to satisfy the assumption of known inputs $u(k)$, which is used in the analysis of the event-based estimation schemes in Paper I and Paper III, each agent must broadcast its input at every time step over the network. While this simplifies the analysis, it may be counterproductive for the overall objective of saving communication (depending on the number of inputs relative to the number of measurements). It seems thus reasonable to use a similar event-triggering scheme also for the inputs; that is, to communicate the inputs only if they differ by more than a threshold from the previously communicated value or a common prediction of the inputs. Another approach to avoid the communication of the inputs at every time is used in Paper II: the inputs of all other agents are estimated based on the agent's local estimate and the (known) control laws used by the other agents. Whether stability for this approach can be proven is another open question.

Related problems on event-based control with output measurements have been considered, for example, in [39]–[42] for a single event-based control loop and, for a distributed setting where triggering decisions are made by multiple entities, in [43] (deterministic output measurements without noise) and in [44] (partial state measurements). Results on the optimal structure of event-based control systems with a single event-triggered loop (the combination of a certainty-equivalence state estimator with a controller, similar to what is used herein) are reported in [39, 45].

Unreliable Networks

The use of a reliable, wire-based communication network on the Balancing Cube was a deliberate design decision since the network must support the communication of feedback data necessary for stabilization. The reliable broadcast network (a common bus) allows an efficient implementation of the distributed transmit decision making: since data on the bus is received by every agent, the estimates by the common estimator are the same on all agents, and they thus reflect the common information in the network. In other words, the common bus ensures consistency in the network.

When broadening the event-based state estimation method to other types of networks, such as wireless networks that may exhibit packet drops, the principle ideas of this work still apply. When making a transmit decision, for example, it is still useful to distinguish information sets such as local information (all data available to a single agent) and common information (data available to every agent). However, when one cannot rely on data reaching every agent due to delays and packet drops, the key problem becomes how the common information can be captured locally on every agent with minimal overhead. The common estimator, if implemented as in Fig. 2.2, would then only *approximate* the common information. The lack of perfect information may partly be compensated for by making more conservative transmit decisions. For this approach to work, the consistency of the common estimators (within bounds) will have to be guaranteed; that is, one must ensure that the difference between any two agents' common estimates does not drift.

Large Scale Systems

The implementations of the common and local estimators (see Fig. 2.2) considered herein use the full process model (2.1) for making state predictions. For networks with a medium number of agents and states, this is effective since the predictions are as accurate as the model permits, and arbitrary dynamic couplings are feasible. However, for large scale systems with a large number of agents and states (such as in future CPSs), simulating the full dynamics is infeasible.

The extension of the approach herein to such large scale systems will require the use of reduced complexity models for making the predictions in the estimators. By decoupling the full system into subsystem (while possibly neglecting weak couplings) and using the subsystem models as the basis for the estimator design, the complexity of the estimator implementations can be reduced. In addition, the subsystem dynamics may be approximated by lower complexity models. One could imagine that each agent (or subnetwork consisting of multiple agents) communicates an approximate model of its dynamics to those agents whose states are affected by this agent. The other agents use this approximate model for making their predictions, and they will receive new measurement data from the originating agent whenever these predictions are off. In general, the use of approximate low complexity models will be at the expense of more frequent communications.

3.2 The Balancing Cube: A Test Bed for Distributed Estimation and Control (Part B)

The insights that were gained from the design of the Balancing Cube led to a follow-up project at the IDSC: the *Cubli* is a small cube (edge length of 15 cm compared to 1.2 m for the Balancing Cube) that can jump up and balance on a corner, [36]. It uses momentum wheels mounted on three of its faces as actuation mechanisms. To jump up, the wheels are rotated at high angular velocities, and then suddenly braked, causing the cube to jump up using conservation of angular momentum. While the focus of the Balancing Cube platform is on distributed estimation and control, the Cubli is controlled by a centralized unit and a research focus is on nonlinear control strategies. The Cubli uses the same tilt estimation algorithm that was developed for the Balancing Cube (see Paper IV).

The Balancing Cube has proven to be a robust platform for demonstrations and exhibitions (Sec. 2.6) and will continue to be used in this capacity at IDSC and external events.

References

[1] M. Hilbert and P. López, “The world’s technological capacity to store, communicate, and compute information,” *Science*, vol. 332, pp. 60–65, Apr. 2011.

[2] O. Kanoun and H.-R. Tränkler, “Sensor technology advances and future trends,” *IEEE Transactions on Instrumentation and Measurement*, vol. 53, no. 6, pp. 1497–1501, Dec. 2004.

[3] C.-Y. Chong and S. P. Kumar, “Sensor networks: evolution, opportunities, and challenges,” *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, Aug. 2003.

[4] J. C. Herrera, D. B. Work, R. Herring, X. J. Ban, Q. Jacobson, and A. M. Bayen, “Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century field experiment,” *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 4, pp. 568–583, 2010.

[5] Liquid Robotics, Inc., [accessed 19.06.2013]. [Online]. Available: http://liquidr.com

[6] J. Appelgren, “Robotic adventure across the Pacific: 313 days, 17,486 nautical km, one cyclone, millions of data points,” *IEEE Robotics Automation Magazine*, vol. 20, no. 2, pp. 24–30, Jun. 2013.

[7] A. Bemporad, M. Heemels, and M. Johansson, *Networked Control Systems*, ser. Lecture Notes in Control and Information Sciences. Springer, 2011, vol. 406.

[8] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, “A survey of recent results in networked control systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[9] T. Yang, “Networked control system: a brief survey,” *IEE Proceedings - Control Theory and Applications*, vol. 153, no. 4, pp. 403–412, Jul. 2006.

[10] Y. Tipsuwan and M.-Y. Chow, “Control methodologies in networked control systems,” *Control Engineering Practice*, vol. 11, no. 10, pp. 1099–1111, 2003.

[11] W. Zhang, M. Branicky, and S. Phillips, “Stability of networked control systems,” *IEEE Control Systems Magazine*, vol. 21, no. 1, pp. 84–99, Feb. 2001.

[12] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[13] G. J. Pottie, “Wireless sensor networks,” in *Information Theory Workshop*, Jun. 1998, pp. 139–140.

[14] A. A. Alam, A. Gattami, and K. H. Johansson, “An experimental study on the fuel reduction potential of heavy duty vehicle platooning,” in *Proc. of the 13th International IEEE Conference on Intelligent Transportation Systems*, Sep. 2010, pp. 306–311.

[15] P. R. Wurman, R. D’Andrea, and M. Mountz, “Coordinating hundreds of cooperative, autonomous vehicles in warehouses,” *AI Magazine*, vol. 29, no. 1, pp. 9–19, 2008.

[16] P. Neumann, “Communication in industrial automation – what is going on?” *Control Engineering Practice*, vol. 15, no. 11, pp. 1332–1347, 2007.

[17] K.-D. Kim and P. Kumar, “Cyber-physical systems: A perspective at the centennial,” *Proceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1287–1308, May 2012.

[18] M. Broy, E. Geisberger, M. Cengarle, P. Keil, J. Niehaus, C. Thiel, and H.-J. Thönnißen-Fries, *Cyber-Physical Systems: Driving force for innovation in mobility, health, energy and production*, acatech, Ed. Springer, Berlin, Dec. 2011, vol. 8.

[19] K. Ogata, *Modern Control Engineering*, 4th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2002.

[20] J. Lunze, *Regelungstechnik 1*, 6th ed. Springer, 2007.

[21] K. Åström and B. Wittenmark, *Computer-controlled systems: theory and design*, ser. Prentice-Hall information and system sciences series. Prentice Hall, 1997.

[22] J. Lunze, *Regelungstechnik 2*, 4th ed. Springer, 2006.

[23] K. J. Åström and B. Bernhardsson, "Comparison of periodic and event based sampling for first-order stochastic systems," in *Proc. of the 14th IFAC World Congress*, Beijing, China, 1999, pp. 301–306.

[24] K. Årzén, "A simple event-based PID controller," in *Proc. of the 14th IFAC World Congress*, Beijing, China, 1999, pp. 423–428.

[25] M. Lemmon, "Event-triggered feedback in control, estimation, and optimization," in *Networked Control Systems*, ser. Lecture Notes in Control and Information Sciences, A. Bemporad, M. Heemels, and M. Johansson, Eds. Springer, 2011, vol. 406, pp. 293–358.

[26] M. W. Spong, "Underactuated mechanical systems," in *Control Problems in Robotics and Automation*, ser. Lecture Notes in Control and Information Sciences, B. Siciliano and K. Valavanis, Eds. Springer, 1998, vol. 230, pp. 135–150.

[27] R. Kalman, "A new approach to linear filtering and prediction problems," *ASME Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[28] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* Mineola, New York: Dover Publications, 2005.

[29] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches.* Wiley-Interscience, 2006.

[30] D. Luenberger, "An introduction to observers," *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 596–602, Dec. 1971.

[31] J. Lunze and D. Lehmann, "A state-feedback approach to event-based control," *Automatica*, vol. 46, no. 1, pp. 211–215, Jan. 2010.

[32] G. Robson, S. Trimpe, M. Donovan, and R. D'Andrea. (2009) The Balancing Cube (video). Institute for Dynamic Systems and Control, ETH Zurich. [Online]. Available: http://www.youtube.com/watch?v=gbT_XoSIlEo

[33] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The Flying Machine Arena," *Mechatronics*, to appear.

[34] J. Shen, A. K. Sanyal, N. A. Chaturvedi, D. S. Bernstein, and N. H. McClamroch, "Dynamics and control of a 3D pendulum," in *Proc. of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004, pp. 323–328.

[35] Quanser Inc., "Cube," [accessed 19.06.2013]. [Online]. Available: http://www.quanser.com

[36] M. Gajamohan, M. Merz, I. Thommen, and R. DAndrea, "The Cubli: A cube that can jump up and balance," in *Proc. of the IEEE/RSJ International Conference on in Intelligent Robots and Systems*, Vilamoura-Algarve, Portugal, Oct. 2012, pp. 3722–3727.

[37] S. L. Campbell and N. J. Rose, "Singular perturbation of autonomous linear systems," *SIAM Journal on Mathematical Analysis*, vol. 10, no. 3, pp. 542–551, 1979.

[38] S. L. Campbell, "Singular perturbation of autonomous linear systems II," *Journal of Differential Equations*, vol. 29, no. 3, pp. 362–373, 1978.

[39] A. Molin and S. Hirche, "Structural characterization of optimal event-based controllers for linear stochastic systems," in *49th IEEE Conference on Decision and Control*, Atlanta, GA, USA, Dec. 2010, pp. 3227–3233.

[40] L. Li and M. Lemmon, "Weakly coupled event triggered output feedback control in wireless networked control systems," in *Proc. of the 49th annual Allerton Conference on Communication, Control, and Computing*, University of Illinois at Urbana-Champaign, IL, USA, Sep. 2011.

[41] D. Lehmann and J. Lunze, "Event-based output-feedback control," in *19th Mediterranean Conference on Control Automation*, Corfu, Greece, Jun. 2011, pp. 982–987.

[42] G. A. Kiener, D. Lehmann, and K. H. Johansson, "Actuator saturation and anti-windup compensation in event-triggered control," *Discrete Event Dynamic Systems*, pp. 1–25, Sep. 2012.

[43] M. C. F. Donkers and W. P. M. H. Heemels, "Output-based event-triggered control with guaranteed L_∞-gain and improved and decentralized event-triggering," *IEEE Transactions on Automatic Control*, vol. 57, no. 6, pp. 1362–1376, Jun. 2012.

[44] C. Stoecker and J. Lunze, "Event-based control with incomplete state measurement and guaranteed performance," in *Proc. of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Santa Barbara, CA, USA, Sep. 2012, pp. 49–54.

[45] A. Molin and S. Hirche, "On the optimality of certainty equivalence for event-triggered control systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 470–474, Feb. 2013.

Part A

Distributed and Event-Based State Estimation

Paper I

Event-Based State Estimation with Variance-Based Triggering

Sebastian Trimpe · Raffaello D'Andrea

Abstract

An event-based state estimation scenario is considered where multiple distributed sensors sporadically transmit observations of a linear process to a time-varying Kalman filter via a common bus. The triggering decision is based on the estimation variance: each sensor runs a copy of the Kalman filter and transmits its measurement only if the associated measurement prediction variance exceeds a tolerable threshold. The resulting variance iteration is a new type of Riccati equation, with switching between modes that correspond to the available measurements and whose choice depends on the variance at the previous step. Convergence of the Riccati-type iteration to periodic solutions is typically observed in simulations. A periodic solution facilitates a straightforward implementation of the transmit decision: each sensor transmits its measurements with a fixed periodic sequence. We prove asymptotic periodicity of the Riccati-type equation for a scalar process.

Submitted to *IEEE Transactions on Automatic Control*, 2012.

1. Introduction

Novel control strategies and improvements in sensor, actuator and network technology will allow the next generation of control systems to tightly integrate the physical world with computation and communication. Referred to as cyber-physical systems (CPSs) [1], these highly integrated systems will extend present-day networked systems (such as networked control systems (NCSs) [2] and wireless sensor networks (WSNs) [3]) in both size and complexity.

As the number of interconnected entities in future CPSs increases, the cost of communication will become a significant factor. Communication is costly even in today's networked systems. In NCSs, where a multi-purpose communication network is shared by many different control, sensor and actuator units, a sensor node cannot transmit its measurement without preventing other units from using the network or causing load-induced delays. In WSNs, the transmission of a sensor measurement to a remote estimator consumes energy that is often a significant fraction of the system's overall energy balance.

While the future of CPSs in areas such as transportation, power systems, smart buildings, mobile robots and process plants is promising, the cost of communication must be managed if CPSs are to meet their potential. It will be vital, for example, to consider the communication network as a shared resource, and to design the control and estimation algorithms in tandem with the network access strategy.

This article considers the problem of estimating the state of a dynamic system from multiple distributed sensors in a scenario where the communication of sensor measurement is costly. We propose a method where data is transmitted only when certain *events* indicate that the data is required to meet constraints on the estimator performance (expressed as tolerable bounds on the error variance). Thus the transmit decision is linked to its contribution to the estimator performance and data is exchanged only when needed.

Figure 1 depicts the distributed state estimation problem and the event-based communication strategy used to address it. The key idea is that each agent transmits its local sensor measurement *only if it is required in order to meet a certain estimation performance.* To be able to make this decision, each agent implements a state estimator that is connected to the common bus. Since the state estimate is computed based on data received over the bus only (the local sensor data is used only when also broadcast) and since we assume a loss and delay-free network, the estimates are the same on all agents and represent the common information in the network. The estimator

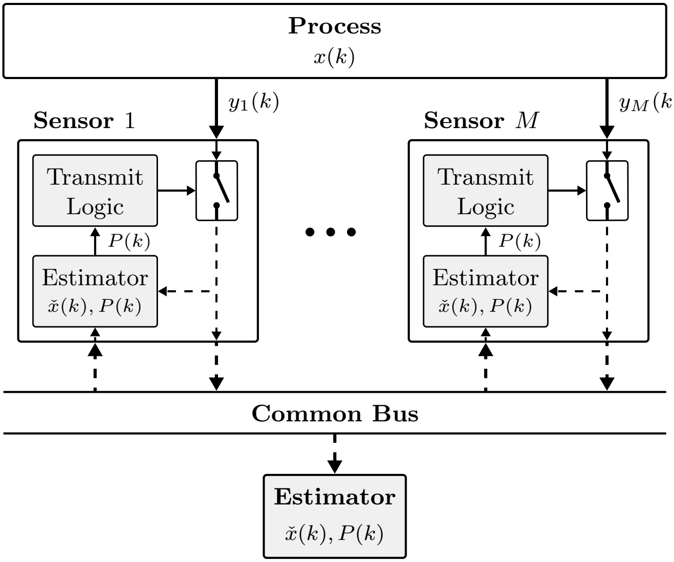


Figure 1. Distributed event-based state estimation problem. The state $x(k)$ of a linear process is observed by M sensor agents, which sporadically transmit their measurements $y_j(k)$ over a common bus. (Solid lines denote continuous flow of data, dashed lines indicate event-based communication, and communication is assumed without delay and data loss.) Estimator nodes connected to the bus receive the measurements and keep track of the conditional state mean $\check{x}(k)$ and variance $P(k)$. Each sensor makes the decision whether to transmit its local measurement based on the estimation variance $P(k)$, thus linking the transmit decision to the estimation performance. The gray blocks constitute the event-based state estimator to be designed herein. The depicted scheme can be applied in different scenarios where communication is costly. In a monitoring application, a remote estimator centrally fuses all sensor data received from the bus (as shown here). In a networked control system, where the agents are also equipped with actuation, the state estimates can be used locally for feedback control.

can hence be used to make the transmit decision: if the other agents' estimate of a particular measurement is already "good enough," it is not necessary to communicate this measurement; if the common estimate is poor, on the other hand, the measurement is transmitted so that all agents can update their estimates. The estimators are implemented as time-varying Kalman filters, which compute the mean and variance of the state conditioned on the received measurements.

Different decision rules for determining whether an estimate is "good enough" are conceivable. In [4, 5], for example, a constant threshold logic on the difference between the actual measurement and its prediction mean is used. Herein, we consider a different approach where the decision is based on the variance: a measurement is broadcast if its prediction variance exceeds a tolerable bound, which indicates that the uncertainty when predicting the measurement is too large. If a transmission is triggered by a condition on the estimation variance, we refer to this as *variance-based triggering*.

As opposed to making the transmit decision based on real-time measurement data (where, for a stochastic process, the transmit decision is a random variable), the approach herein permits an off-line analysis of the (deterministic) estimation variance iteration, provided that the observed process is stationary and its statistics are known in advance. Specifically, if a periodic solution of the variance iteration is found, it corresponds to a periodic sending sequence for each sensor, and hence allows for a straightforward implementation of the resulting communication logic. The periodic sending sequence can be computed in advance and fixed for each sensor.

The variance iteration for the event-based state estimator, which is derived in Sec. 2, represents a Riccati-type equation with switching that corresponds to the available measurements at a time step and depends on the variance at the previous step. We applied the event-based estimation method to the NCS of the Balancing Cube [6], and present the results of simulating the Riccati-type equation in Sec. 3. These results suggest that the variance iteration converges to a periodic solution (observed to numerical accuracy). In Sec. 4, we investigate the special case of a scalar process and prove a theorem that guarantees the asymptotic convergence of the Riccati-type equation to a periodic solution with a known period. We conclude with a discussion of the applicability and extensions of the method in Sec. 5.

We first presented the event-based state estimation method with variance-based triggering in [7], where we focused on the experimental application on the Balancing Cube (not repeated herein). This article includes the proofs for the convergence result of the scalar Riccati-type equation, which were omitted in a preliminary version of the result in [8].

1.1 Related Work

Event-based strategies are a popular means of ensuring efficient use of the communication resource in NCSs or CPSs (see [9] and references therein). As opposed to traditional time-triggered transmission of data, event-based approaches transmit data only when required to meet a certain specification of the control system (e.g. closed-loop stability, control or estimator performance). Event-based state estimation problems with a single sensor and a single estimator node have been studied in [9]–[16], for example. Event-based state estimation problems for distributed or multi-agent systems have been looked at in [4, 5, 17].

The basic idea of implementing state estimators on the agents of an NCS in order to reduce communication of sensor data was first presented in [18]. Therein, each agent uses a model to predict the other agents' measurements at times when these are not transmitted (because the prediction error is below a threshold), and the agent *resets* parts of the state vector when new measurement data becomes available. In contrast, the Kalman filters used herein *fuse* model-based predictions with the received measurements. Communication schemes like these where, in order to reduce network traffic, sensor data is not sent at every time step, are also referred to as *controlled communication*, [2, 13, 19].

In most of the above-mentioned references for the single sensor/single estimator case, the sensor node transmits a local state estimate (obtained from a Kalman filter on the sensor) to the remote estimator, rather than the raw measurement. While this seems to be the method of choice for the single sensor agent case (the local state estimate contains the fused information of all past measurements), communicating raw measurements has a practical advantage for the case of multiple agents with coupled dynamics. For an agent to fuse another agent's measurement with its local state estimate, it must know the variance of the measurement conditioned on the state. This is usually known in form of a sensor model. To optimally fuse another agent's state estimate (with coupled dynamics), on the other hand, the variance associated with the estimate would have to be known. Since this variance is, however, only known to the agent that generated the estimate, it would have to be communicated over the network as well, hence, increasing the network load. The method herein makes no assumptions on the dynamic coupling between the system parts that are observed by the various sensors.

In the above-mentioned references on event-based estimation, an event is triggered by some condition on real-time data (measurement or state); that is, in a stochastic framework, data transmission is a random event. In contrast, the variance-based trigger used herein depends on the prediction variance at the previous step. The resulting variance iteration is deterministic and depends on the problem data only. A condition on the variance to trigger sensor transmissions is considered in [20] in a slightly different framework. Therein, the authors consider two heterogeneous sensors: a condition on the estimator variance is used to decide which of the sensors will transmit its measurement to a remote estimator at any given time step. Whereas the average communication rate is constant in [20], we seek to reduce the average sensor transmission rate, and to have the option to not transmit any data at a time step. The authors in [20] also observe convergence of the estimator variance to periodic sequences in their scenario, but they do not prove this convergence.

The Riccati iteration obtained for the event-based estimation problem herein is related to Riccati equations of other well-known Kalman filtering problems with different sensor transmission policies.

- Full communication. If the estimator has access to all sensor measurements at every time step, the considered problem reduces to the classic Kalman filtering problem for linear time-invariant systems, [21]. The variance iteration of the filter then becomes the *discrete-time Riccati equation*, whose convergence properties are well known.

- Fixed periodic transmission. With a-priori fixed periodic transmission of sensor data, the problem can be cast as a linear periodic system with a periodically varying measurement model. The estimator variance evolves according to the *discrete-time periodic Riccati equation*, whose convergence properties are studied in [22], for example. The problem considered herein is different in that we do not assume a-priori a periodic transmit sequence, but we show that a periodic sequence results from the event-based estimation problem.

- Data arrival as random process. In Kalman filtering with intermittent observations [23], measurement data arriving at the filter is subject to random data loss modeled as a Bernoulli process. Hence, the estimation variance itself becomes a random variable. In [23], the authors show that there exists a critical value for the data loss probability, beyond which the variance becomes unbounded. In contrast to measurement transmissions being governed by an external (random) process, the measurement transmission herein is triggered "internally" by the estimator whenever new data is needed.

1.2 Notation and Preliminaries

We use $\mathbb{R}$, $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{N}^+$ to denote real numbers, integers, nonnegative integers, and positive integers, respectively. By $\mathrm{E}[\cdot|\cdot]$ and $\mathrm{Var}[\cdot|\cdot]$, we denote the conditional expected value and the conditional variance. A normally distributed random variable z with mean m and covariance matrix V is denoted by $z \sim \mathcal{N}(m, V)$.

For $i, j \in \mathbb{Z}$ and $N \in \mathbb{N}^+$, we define the binary operator '$-_N$' as follows:

$$i -_N j = \begin{cases} \mathrm{mod}(i-j, N) & \text{if } \mathrm{mod}(i-j, N) > 0 \\ N & \text{if } \mathrm{mod}(i-j, N) = 0, \end{cases} \tag{1}$$

where $\mathrm{mod}(i, N) \in \{0, \dots, N{-}1\}$ is the (nonnegative) remainder of i divided by N. Hence, '$-_N$' is the subtraction with subsequent modulo N operation, except that a resulting 0 is replaced by N.

The matrix $I_{n\times n}$ denotes the n-by-n identity matrix; the subscript is omitted if the dimension is clear from context. For a matrix $A \in \mathbb{R}^{m\times n}$, $A_{j:}$ denotes the jth row, and $[A_{j:}]_{j\in J}$ with $J \subseteq \{1, \dots, m\}$ denotes the matrix constructed from stacking the rows $A_{j:}$ for all $j \in J$. Further, $\mathrm{diag}[A_{jj}]_{j\in J}$ denotes the diagonal matrix with entries A_{jj}, $j \in J$, on its diagonal.

We define the binary indicator function $\mathbf{1}_X$ such that $\mathbf{1}_X = 1$ if statement X is true, and $\mathbf{1}_X = 0$ otherwise.

Consider the iteration

$$p(k{+}1) = h(p(k)), \quad p(0) = p_0 \geq 0, \tag{2}$$

with a function $h : D \to \mathbb{R}^n$, $D \subseteq \mathbb{R}^n$. For h being applied m times, we write h^m; that is, for $m \in \mathbb{N}$,

$$p(k{+}m) = h^m(p(k)) = \underbrace{h(h(\dots(h}_{m}(p(k))\dots))), \tag{3}$$

where $h^0(p(k)) := p(k)$. For the domain of a function h, we write $\mathrm{dom}(h)$. We use the following definitions to characterize periodic solutions of (2):

Definition 1—adapted from [24] Let $p^* \in \mathrm{dom}(h)$. Then p^* is called an *N-periodic point* of (2) if it is a fixed point of h^N, that is, if

$$h^N(p^*) = p^*. \tag{4}$$

The periodic orbit of p^*, $\{p^*, h(p^*), h^2(p^*), \dots, h^{N-1}(p^*)\}$, is called an *$N$-cycle*, and N is called the *period.* ■

Definition 2 A solution to (2) is called *asymptotically N-periodic* for the initial condition $p(0) = p_0$ if

$$\lim_{m\to\infty} h^{mN}(p_0) = p^*, \tag{5}$$

where p^* is an N-periodic point of (2). ■

For a function h and collections of intervals $\mathcal{I}_1$ and $\mathcal{I}_2$, we write $\mathcal{I}_1 \xrightarrow{h} \mathcal{I}_2$ to indicate that each interval from $\mathcal{I}_1$, when mapped by h, is contained in an interval in $\mathcal{I}_2$; that is,

$$\mathcal{I}_1 \xrightarrow{h} \mathcal{I}_2 \quad \Leftrightarrow \quad \forall I_1 \in \mathcal{I}_1, \exists I_2 \in \mathcal{I}_2 \;:\; h(I_1) \subseteq I_2. \tag{6}$$

2. Event-Based State Estimator

We consider the stochastic linear time-invariant system

$$x(k) = A\,x(k-1) + v(k-1) \tag{7}$$

$$y(k) = C\,x(k) + w(k), \tag{8}$$

where k is the discrete time index, $x(k) \in \mathbb{R}^n$ is the state, $y(k) \in \mathbb{R}^M$ its observation (M the number of sensors), and all matrices are of corresponding dimensions. The process noise, the measurement noise, and the initial state $x(0)$ are assumed mutually independent, normally distributed with $v(k) \sim \mathcal{N}(0, Q)$, $w(k) \sim \mathcal{N}(0, R)$, $x(0) \sim \mathcal{N}(x_0, P_0)$, $Q \geq 0$, $R > 0$, and $P_0 \geq 0$. We assume that (A, C) is detectable (i.e. the process is detectable when measurements from all sensors are combined, but not necessarily from an individual sensor), (A, Q) is stabilizable, and R is diagonal. The latter assumption means that the measurement noise is mutually independent for any two sensors considered, which is often the case in practice. The presented state estimation method can, however, be readily extended to the case of block diagonal R by sending blocks of correlated measurements at once.

REMARK 1 For ease of notation, we consider an unforced system; that is, no input $u(k-1)$ in (7). Provided the inputs are known by all sensor agents at all times (such as when they represent a known reference signal or when they are shared over the network), the extension of the event-based state estimator to this case is straightforward, and the analysis of the estimator variance in Sec. 3 and 4 remains unchanged. ■

We seek an algorithm to recursively compute an estimate of the state $x(k)$ from measurements received up to time k, and the problem parameters given by (A, C, Q, R, P_0). If the full measurement vector $y(k)$ is available at time k, the problem is solved by the standard Kalman filter (Sec. 2.1). In Sec. 2.2, we present the event-based state estimator, which consists of a Kalman filter that uses a reduced set of measurements as an input and a rule for deciding whether to transmit a measurement.

2.1 Full Communication Kalman Filter

It is well known that the Kalman filter is the optimal Bayesian state estimator for the process (7), (8) because it keeps track of the Gaussian conditional probability distribution of the state $x(k)$ conditioned on all measurements up to time k, $\mathcal{Y}(k) := \{y(1), \ldots, y(k)\}$ (see [21], for example). To distinguish this Kalman filter from the event-based filter derived below, we refer to it as the *full communication Kalman filter*. Under the above assumptions, the state prediction variance $\mathrm{Var}\,[x(k)|\mathcal{Y}(k{-}1)]$ converges to $\bar{P} > 0$, which is the unique positive solution to the discrete algebraic Riccati equation (DARE):

$$\bar{P} = A\bar{P}A^{\mathrm{T}} + Q - A\bar{P}C^{\mathrm{T}}(C\bar{P}C^{\mathrm{T}} + R)^{-1}C\bar{P}A^{\mathrm{T}}. \tag{9}$$

We write $\bar{P} = \mathrm{DARE}(A, C, Q, R)$.

2.2 Event-Based Kalman Filter

Denote by $J(k) \subseteq \{1, \ldots, M\}$ the subset of sensors that transmit their measurement at time k. We make precise how we choose $J(k)$ later in this section. Since communication is assumed to be instantaneous and without data loss, $J(k)$ is also the set of measurements available at the estimator at time k. The corresponding measurement equation is then given by

$$\tilde{y}(k) = \tilde{C}(k)\, x(k) + \tilde{w}(k), \tag{10}$$

where $\tilde{y}(k) = [y_j(k)]_{j \in J(k)}$ is the vector of those measurements available at time k, $\tilde{w}(k) \sim \mathcal{N}(0, \tilde{R}(k))$, and output and measurement noise variance matrices are constructed as

$$\tilde{C}(k) = [C_{j:}]_{j \in J(k)}, \quad \tilde{R}(k) = \mathrm{diag}[R_{jj}]_{j \in J(k)}. \tag{11}$$

Notice that $\tilde{y}(k) \in \mathbb{R}^{m(k)}$, $\tilde{w}(k) \in \mathbb{R}^{m(k)}$, $\tilde{C}(k) \in \mathbb{R}^{m(k) \times n}$, and $\tilde{R}(k) \in \mathbb{R}^{m(k) \times m(k)}$ have time varying dimensions with $m(k) \leq M$, which includes the case $m(k) = 0$; that is, at time k there is no measurement available at the estimator. In order to avoid special treatment of this case, we use the convention that the measurement update step in the Kalman filter below is omitted in case $m(k) = 0$.

For any given sequence of $\tilde{C}(k)$ and $\tilde{R}(k)$, the distribution of the state $x(k)$ conditioned on the set of available measurements $\tilde{\mathcal{Y}}(k) = \{\tilde{y}(l) \mid 0 \leq l \leq k\}$ is Gaussian, [21]. The Kalman filter keeps track of the conditional means and variances, $\check{x}(k|k{-}1) = \mathrm{E}\,[x(k)|\tilde{\mathcal{Y}}(k{-}1)]$, $\check{x}(k|k) = \mathrm{E}\,[x(k)|\tilde{\mathcal{Y}}(k)]$, $\check{P}(k|k{-}1) = \mathrm{Var}\,[x(k)|\tilde{\mathcal{Y}}(k{-}1)]$, and $\check{P}(k|k) = \mathrm{Var}\,[x(k)|\tilde{\mathcal{Y}}(k)]$. The filter equations are

$$\check{x}(k|k{-}1) = A\check{x}(k{-}1|k{-}1) \tag{12}$$

$$\check{P}(k|k{-}1) = A\check{P}(k{-}1|k{-}1)A^{T} + Q \tag{13}$$

$$\check{K}(k) = \check{P}(k|k{-}1)\, \tilde{C}^{T}(k)\big(\tilde{C}(k)\check{P}(k|k{-}1)\tilde{C}^{T}(k) + \tilde{R}(k)\big)^{-1} \tag{14}$$

$$\check{x}(k|k) = \check{x}(k|k-1) + \check{K}(k)\big(y(k) - \tilde{C}(k)\check{x}(k|k-1)\big) \tag{15}$$

$$\check{P}(k|k) = \big(I - \check{K}(k)\tilde{C}(k)\big)\check{P}(k|k-1). \tag{16}$$

The filter is initialized with $\check{x}(0|0) = x_0$ and $\check{P}(0|0) = P_0$.

For notational convenience, we use $P(k) := \check{P}(k|k-1)$ for the state prediction variance. The prediction variance captures the uncertainty about $x(k)$ given all measurements up to the previous time step $k-1$. Similarly, $\mathrm{Var}\,[y_j(k)|\tilde{\mathcal{Y}}(k-1)] = C_j P(k) C_j^{\mathrm{T}} + R_{jj}$ captures the uncertainty in predicting the measurement $y_j(k)$. A measurement $y_j(k)$ is transmitted and used to update the estimator if, and only if, its prediction variance exceeds a tolerable bound. Since the event-based Kalman filter cannot do better than the full communication filter, we use a threshold δ on the difference $\mathrm{Var}\,[y(k)|\tilde{\mathcal{Y}}(k-1)] - \lim_{k\to\infty} \mathrm{Var}\,[y(k)|\mathcal{Y}(k-1)] = C_j(P(k) - \bar{P})C_j^{\mathrm{T}}$ for the transmit decision. Hence, we use the transmit rule

$$\text{transmit } y_j(k) \Leftrightarrow \; C_j\big(P(k) - \bar{P}\big)C_j^{\mathrm{T}} \geq \delta_j \tag{17}$$

for sensor j, where the design parameter δ_j captures the tolerable deviation of the jth sensor measurement prediction variance from the full communication, steady-state variance. For ease of notation, we introduce the transmit function

$$\gamma_j(k) := \mathbf{1}_{C_j(P(k)-\bar{P})C_j^{\mathrm{T}} \geq \delta_j}. \tag{18}$$

Having established the transmit rule (17), we can now make the set $J(k)$ of all sensors transmitted at time k precise:

$$J(k) = \{j \mid 1 \leq j \leq M,\, C_j\big(P(k) - \bar{P}\big)C_j^{T} \geq \delta_j\}. \tag{19}$$

The matrices $\tilde{C}(k)$ and $\tilde{R}(k)$ for $k \in \mathbb{N}$ are well defined by (11), (19), and knowledge of $P(k) = \check{P}(k|k-1)$.

The Kalman filter (12)–(16) together with the variance-based transmit decision (17) is referred to as the *event-based state estimator with variance-based triggering*. Since the Kalman filter (12)–(16) is the optimal Bayesian state estimator for any sequences $\tilde{C}(k)$ and $\tilde{R}(k)$, it is also optimal for those sequences given by (11) and (19). In other words, given the rule (17) (which captures the objective to use relevant measurements only), the filter (12)–(16) is the optimal state estimator for the estimation problem given by (7), (10), (11), and (19). Clearly, if $\delta_j = 0$ for all sensors, the full communication Kalman filter is recovered.

2.3 Switching Riccati-Type Iteration

The update equation for the estimator prediction variance $P(k)$ is obtained by combining (11), (13), (14), (16), and (19):

$$\begin{aligned} P(k{+}1) &= A\,P(k)\,A^{\mathrm{T}} + Q \\ &\quad - A\,P(k)\,\big(\check{C}(P(k))\big)^{\mathrm{T}}\Big(\check{C}(P(k))\,P(k)\,\big(\check{C}(P(k))\big)^{\mathrm{T}} + \check{R}(P(k))\Big)^{-1} \\ &\quad\quad \cdot\check{C}(P(k))\,P(k)\,A^{T} \\ &=: H(P(k)), \end{aligned} \tag{20}$$

where $\check{C}(P(k)) := \tilde{C}(k)$ and $\check{R}(P(k)) := \tilde{R}(k)$ have been introduced to emphasize their dependence on $P(k)$ by (11) and (19); and $H(\cdot)$ denotes the map from $P(k)$ to $P(k{+}1)$. The system given by (7), (10), and (11) can be regarded as a switching system with modes given by the possible values of $J(k)$. The modes of the system are switched as a function of the prediction variance at the previous step through (19). Thus, (20) is a Riccati-type iteration with switching that depends on the variance at the previous step.

According to (20), the sequence $P(k)$ for $k \in \mathbb{N}^{+}$ can be computed from the problem data (A, C, Q, R, P_0), and the tuning parameters δ_j. Notice that this is fundamentally different from approaches such as [4], where the decision whether to transmit a measurement is based on the actual real-time measurement. Since the measurement is a random variable, the Kalman filter variables $P(k)$ and $\check{P}(k|k)$ become random variables themselves; whereas here, they are deterministic and can be computed off-line from the problem data. This will allow the analysis of the Riccati-type iteration (20) in the following sections.

3. Illustrative Examples

We provide two examples to illustrate the behavior of the Riccati-type iteration (20) for the event-based state estimator. The solutions are asymptotically periodic in both examples. Periodic solutions of the Riccati iteration correspond to periodic transmit sequences, which gives rise to a time-triggered implementation of the event-based design with low complexity (the sensor nodes do not need to run a copy of the estimator then). (Matlab files to reproduce the simulation results of this section are provided as supplementary material.)

3.1 Scalar Problem

Consider the system (7), (8) with a single sensor ($M = 1$), a scalar process ($n = 1$), and the parameters (small letters are used to indicate scalar quantities):

EXAMPLE 1 $a = 1.2$, $c = q = r = 1$, $\delta = 3$, $p_0 = \bar{p}$. ■

Figure 2(a) shows the results from simulating (20).

As expected, the prediction variance $p(k)$ grows at times where no measurement is available. Once the threshold is exceeded, a measurement is transmitted ($\gamma(k) = 1$) and the estimator variance drops. The solution in Fig. 2(a) asymptotically converges to a periodic solution with period $N = 3$.

Figures 2(b) and 2(c) illustrate that, for different values of δ (all other parameters are the same as in Example 1), asymptotically periodic solutions with very different periods can be obtained. Notice that the period does not vary monotonically with δ.

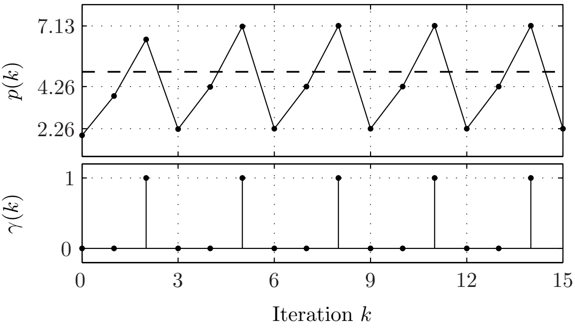
3.2 Multivariate Problem of the Balancing Cube

The event-based state estimation method was used in [7] to reduce the average communication in the networked control system of the Balancing Cube [6]. Here, we present the simulation results of (20) that were used to find periodic transmit sequences for this system.

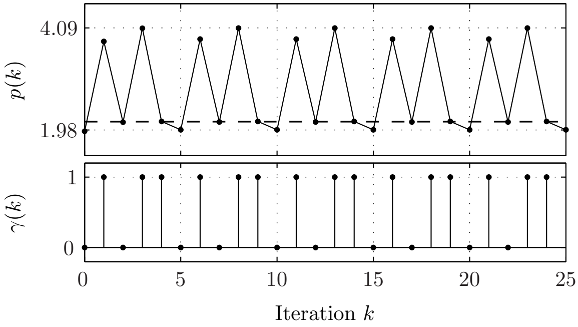
The Balancing Cube is a multi-body system consisting of a rigid cube-shaped aluminum structure (with an edge length of 1.2 m) and six rotating arms that are mounted at the centers of the cube's inner faces. The arms constitute the control agents, each one carrying sensing, actuation, computation, and power. They coordinate by exchanging data over a wire-based broadcast network; thus, the system has the architecture of Fig. 1 (with the sensor agents also equipped with actuation). By controlling the actuated arms, the cube can balance on any of its corners or edges. In [7], the event-based state estimator from Sec. 2.2 is implemented on each agent, and the state estimates are used for feedback control.

Consider the system (7), (8) with $n = 8$, $m = 12$,

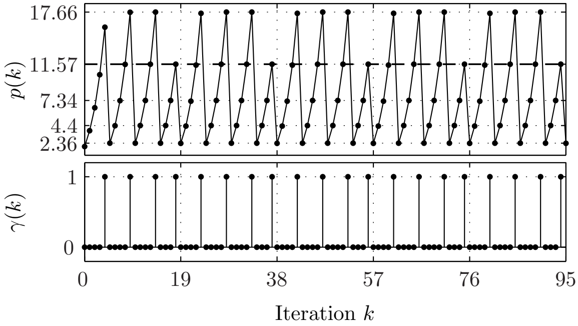
$$A = \begin{bmatrix} I_{6\times 6} & 0 \\ A_{21} & A_{22} \end{bmatrix}, \quad \text{and} \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{21}$$



(a) $\delta = 3$



(b) $\delta = 0.2$



(c) $\delta = 9.6167$

Figure 2. Simulation results for the scalar Example 1 and different values of the threshold parameter δ. The top graph of each sub-figure shows the variance iterates $p(k)$ (dots) and the transmit threshold $\bar{p} + \delta/c^2$ (dashed). The bottom graph shows the corresponding transmit sequence $\gamma(k)$. All solutions are asymptotically periodic with periods $N = 3, 5, 19$ from (a) to (c).

Table 1. States and sensors of the Balancing Cube model (21).

State	Physical Meaning	Meas.	Sensor
x_1	angle arm 1	y_1	angle encoder arm 1
x_2	angle arm 2	y_2	rate gyro arm 1
x_3	angle arm 3	y_3	angle encoder arm 2
x_4	angle arm 4	y_4	rate gyro arm 2
x_5	angle arm 5	y_5	angle encoder arm 3
x_6	angle arm 6	...	...
x_7	angle cube	y_{11}	angle encoder arm 6
x_8	ang. vel. cube	y_{12}	rate gyro arm 6

which represents a model of the cube (when balancing on edge) that is relevant for state estimation. All states and measurements of the model (21) are summarized in Table 1. Since the system inputs are irrelevant for the estimation problem, they are omitted in (21). For each arm, there is an encoder measuring the arm angle relative to the cube structure, and a rate gyro (mounted on the cube body) measuring the angular rate of change of the cube body. Please refer to [6] for a detailed description of the system and to [7] for details on the model (including the numerical values of A_{21} and A_{22}).

In the Kalman filter, we use

$$Q = \operatorname{diag}\left([1\ 1\ 1\ 1\ 1\ 1\ 0.01\ 1]\right) \tag{22}$$

$$R = \operatorname{diag}\left([0.1\ 1\ 0.1\ 1\ 0.1\ 1\ 0.1\ 1\ 0.1\ 1\ 0.1\ 1]\right), \tag{23}$$

which were found to yield satisfactory performance in experiments using the full communication filter (Sec. 2.1) for feedback control. The steady-state solution (9) of the full communication Kalman filter is (rounded to two decimal places):

$$\bar{P} = \begin{bmatrix} 1.09 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.09 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.09 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.09 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.09 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.09 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.72 & 0.13 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.13 & 1.17 \end{bmatrix}. \tag{24}$$

For the design of the event-based state estimator (Sec. 2.2), the only additional design parameters are the threshold parameters δ_j in (17): we chose $\delta_{\text{enc}} = 48$ for the angle encoders and $\delta_{\text{gyro}} = 4$ for the rate gyros.

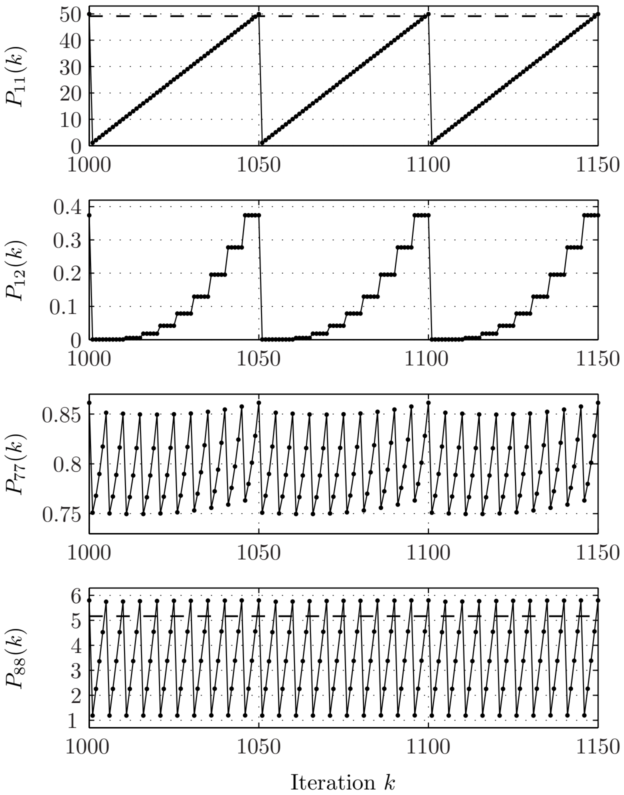


Figure 3. Simulation results of the variance iteration (20) for the Balancing Cube model (21) after 1000 steps. Shown are some elements of $P(k)$ (dots) and the transmit threshold $\bar{P}_{jj}+\delta_j$ (dashed). Notice that there is no explicit threshold on $P_{12}(k)$ and $P_{77}(k)$. The solution approaches an N-cycle with period $N = 50$.

The result for simulating (20) starting at $P(0) = \bar{P}$ is shown in Fig. 3. The obtained solution converges to an N-cycle with period $N = 50$. In fact, the iteration $\tilde{P}(k+1) = H^{50}(\tilde{P}(k))$ yields the fixed point

$$\tilde{P}^* = \begin{bmatrix} 1.10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.10 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.75 & 0.14 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.14 & 1.19 \end{bmatrix}. \tag{25}$$

Interpretation of Periodic Solution A periodic solution of the variance iteration (20) corresponds to periodic sensor transmission via (17). In this example, the following transmit sequences $\gamma_j(k)$ are obtained:

$$\gamma_j(k) = \mathbf{1}_{k=50}(k), \quad \text{for } j = 1, 3, \dots, 11 \text{ (encoder)} \tag{26}$$

$$\gamma_j(k) = \mathbf{1}_{k=5}(k) + \mathbf{1}_{k=10}(k) + \cdots + \mathbf{1}_{k=50}(k), \quad \text{for } j = 2, 4, \dots, 12 \text{ (gyro).} \tag{27}$$

Instead of running the full event-based state estimator given by (12)–(16) and (17) on each sensor (see Fig. 1), the periodic transmit schedule $\gamma_j(k)$ can be implemented for the transmit logic block at lower computation and memory requirements. This approach was applied to the Balancing Cube in [7]. Periodic solutions thus allow for the recovery of a time-triggered (periodic) state estimator implementation. The sampling periods of the sensors are, however, not design parameters as in usual time-triggered estimation, but they result from the event-based approach where the designer specifies tolerable bounds on the estimator performance.

4. Asymptotic Periodicity for Scalar Problem

For the specific problems in the previous section, we observed in simulations that the Riccati-type iteration (20) approaches periodic solutions (i.e. convergence to periodic solutions was interpreted from simulation results rather than proven). In this section, we address the convergence problem and derive a theorem for the scalar version of (20) that guarantees asymptotic periodicity of the solution under certain assumptions to be derived in this section as well.

The question, under what conditions is the periodic transmission of sensor data the optimal solution for the event-based estimation problem posed by (7), (10), (11), and (19), is of theoretical interest for understanding the connection between time-triggered and event-based estimation. On the other hand, checkable conditions for asymptotic periodicity are also of practical value as they provide a means of identifying periodic solutions other than by simulating (20) and having to interpret the result (where it may happen that one has not simulated long enough to find a solution with a larger period).

The subject of study in this section is the scalar version of (20); that is, the nonlinear recursive equation

$$p(k+1) = a^2\, p(k) + q - \mathbf{1}_{c^2(p(k)-\bar{p}) \geq \delta}\, \frac{a^2\, c^2\, p^2(k)}{c^2\, p(k) + r} \tag{28}$$

$$p(0) = p_0 \geq 0, \tag{29}$$

for the parameters $|a| > 1$, $c \neq 0$, $q > 0$, $r > 0$, $\delta > 0$ (small letters are used to indicate scalar quantities). In particular, we consider unstable dynamics ($|a| > 1$), which is the more challenging case, since communication of measurements is required for the estimation error variance to be bounded. We derive conditions that guarantee the solution of (28) to be asymptotically N-periodic, and give an algorithm to compute the period N. After some preliminaries in Sec. 4.1, we use an illustrative example in Sec. 4.2 to outline the convergence proof, which then follows in Sec. 4.3 to 4.6 with the main result being stated in Sec. 4.6 (Theorem 2).

4.1 Preliminaries

Since $q > 0$ and $r > 0$, (28) can equivalently be written as

$$\frac{p(k{+}1)}{q} = a^2 \frac{p(k)}{q} + 1 - \mathbf{1}_{\frac{p(k)}{q} - \frac{\bar{p}}{q} \geq \frac{\delta}{c^2 q}} \frac{a^2 \frac{c^2 q}{r} \left(\frac{p(k)}{q}\right)^2}{\frac{c^2 q}{r} \frac{p(k)}{q} + 1}. \tag{30}$$

By redefining $p(k)$, c^2, and δ as $p(k)/q$, $c^2 q/r$, and $\delta/(c^2 q)$, respectively, we can assume without loss of generality that $q = r = 1$. Henceforth, we study the iteration

$$p(k{+}1) = h(p(k)), \quad p(0) = p_0 \geq 0, \tag{31}$$

with h defined by

$$\begin{aligned} h \,:\, [0,\infty) &\to [0,\infty) \\ p &\mapsto a^2\, p + 1 - \mathbf{1}_{p \geq \bar{p} + \delta} \frac{a^2\, c^2\, p^2}{c^2\, p + 1} \end{aligned} \tag{32}$$

with parameters $|a| > 1$, $c \neq 0$, $\delta > 0$; and with $\bar{p} = \mathrm{DARE}(a, c, 1, 1)$. The graph of h is shown in Fig. 4 together with the graph of the function g,

$$\begin{aligned} g \,:\, [0,\infty) &\to [0,\infty) \\ p &\mapsto a^2\, p + 1 - \frac{a^2\, c^2\, p^2}{c^2\, p + 1}, \end{aligned} \tag{33}$$

which represents the variance iteration of the full communication Kalman filter.

We summarize some properties of h, which will be useful later.

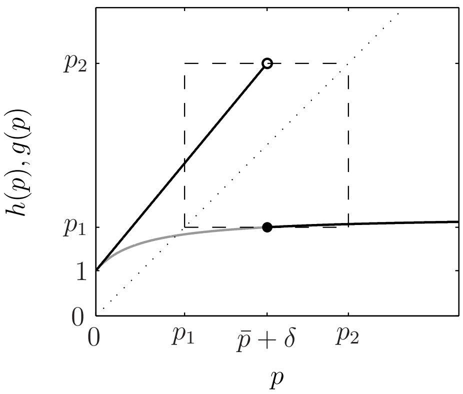


Figure 4. The functions h (black) and g (gray). The filled circle indicates a closed interval boundary, whereas the unfilled circle indicates an open interval boundary. The dotted diagonal represents the identity map $p = p$. The intersection of g with the identity diagonal represents the solution $\bar{p}$ to the DARE (9). The dashed box represents the set $[p_1, p_2)$, which is invariant under h. For $p \geq \bar{p} + \delta$, $h(p) = g(p)$.

PROPOSITION 1 Let $p_1 := h(\bar{p}+\delta)$ and $p_2 := a^2(\bar{p}+\delta)+1$. The following properties of h hold:

(i) h is continuous and strictly increasing on $[p_1, \bar{p}+\delta)$ and on $[\bar{p}+\delta, p_2]$.

(ii) h is differentiable on $(p_1, \bar{p}+\delta)$ and on $(\bar{p}+\delta, p_2)$.

(iii) h is injective on $[p_1, p_2)$.

(iv) $h([p_1, p_2)) = [p_1, h(p_2)) \cup [h(p_1), p_2) \subseteq [p_1, p_2)$.

(v) $\forall p \in [0, \infty), \exists m \in \mathbb{N} : h^m(p) \in [p_1, p_2)$.

■

Proof. From the definitions of h and g in (32) and (33), we get

$$p_1 = h(\bar{p}+\delta) = g(\bar{p}+\delta) < \bar{p}+\delta < a^2(\bar{p}+\delta)+1 = p_2. \tag{34}$$

We first show that

$$h(p_1) > h(p_2), \tag{35}$$

which will be useful later. Let $\tilde{p} := \bar{p} + \delta$. Then, with (34),

$$\begin{aligned} h(p_1) &= a^2 p_1 + 1 = a^2 g(\tilde{p}) + 1 = a^4\tilde{p} + a^2 + 1 - \frac{a^4c^2\tilde{p}^2}{c^2\tilde{p}+1}, \\ h(p_2) &= g(p_2) = a^4\tilde{p} + a^2 + 1 - \frac{a^2c^2(a^2\tilde{p}+1)^2}{c^2(a^2\tilde{p}+1)+1}. \end{aligned}$$

Hence,

$$\begin{aligned} h(p_1) - h(p_2) &= -\frac{a^4c^2\tilde{p}^2}{c^2\tilde{p}+1} + \frac{a^2c^2(a^2\tilde{p}+1)^2}{c^2(a^2\tilde{p}+1)+1} \\ &= \frac{a^4c^4\tilde{p}^2 + a^4c^2(a^2-1)\tilde{p}^2 + a^2c^4\tilde{p} + 2a^4c^2\tilde{p} + a^2c^2}{(c^2\tilde{p}+1)(a^2c^2\tilde{p}+c^2+1)}, \end{aligned}$$

which is strictly greater than zero because of $\tilde{p} \geq 0$, $|a| > 1$, $c \neq 0$; and (35) follows. Next, we prove the statements (iii)–(v) ((i) and (ii) are omitted as they follow directly from (32) and (33)).

(iii): h is injective on each of the intervals $[p_1, \bar{p}+\delta)$ and $[\bar{p}+\delta, p_2)$ separately by (i). Furthermore, by (i),

$$h([p_1, \bar{p}+\delta)) = [h(p_1), \lim_{p \nearrow \bar{p}+\delta} h(p)) = [h(p_1), p_2), \tag{36}$$

$$h([\bar{p}+\delta, p_2)) = [h(\bar{p}+\delta), \lim_{p \nearrow p_2} h(p)) = [p_1, h(p_2)), \tag{37}$$

where $p \nearrow \bar{p}+\delta$ denotes the left-sided limit (p approaches $\bar{p}+\delta$ from below). From (35), $[p_1, h(p_2)) \cap [h(p_1), p_2) = \emptyset$. Therefore, h is injective on $[p_1, p_2)$.

(iv): Follows from (36), (37), and (35).

(v): We consider four cases for $p \in [0, \infty)$.

First case: $p \in (0, p_1)$. We first show that the sequence $h^k(p), k \geq 0$ is eventually greater than p_1. For $h^{k-1}(p) \in (0, p_1)$, $h^k(p) = a^2h^{k-1}(p) + 1 > a^2h^{k-1}(p)$. Hence, for $p, h(p), \ldots, h^{k-1}(p) \in (0, p_1)$, $h^k(p) > a^{2k}p$. Since $\lim_{k\to\infty} a^{2k}p = \infty$, there exists an $m \in \mathbb{N}$ such that

$$h^{m-1}(p) \in (0, p_1) \quad \text{and} \quad h^m(p) \in [p_1, \infty). \tag{38}$$

Next, notice that $h((0, p_1)) = (h(0), h(p_1)) = (1, h(p_1)) \subseteq [1, p_2)$ because $h(p_1) < p_2$ by (iv). Since $h^{m-1}(p) \in (0, p_1)$, it follows that $h^m(p) = h(h^{m-1}(p)) \in [1, p_2)$. Together with (38), this implies that $h^m(p) \in [p_1, \infty) \cap [1, p_2) = [p_1, p_2)$.

Second case: $p = 0$. After one iteration, $h(p) = h(0) = 1 \in (0, p_1)$; that is, the claim follows from the first case.

Third case: $p \in [p_1, p_2)$. Follows with $m = 1$ from (iv).

Fourth case: $p \in [p_2, \infty)$. Since $h(p) = g(p)$, the sequence $h^k(p) = g^k(p)$, $k \geq 0$, with $p, h(p), \dots, h^{k-1}(p) \in [p_2, \infty)$ evolves as for the full communication Kalman filter. By the convergence properties of the full communication Kalman filter, [21], $\lim_{k\to\infty} g^k(p) = \bar{p}$ and, by (34), $\bar{p} < \bar{p} + \delta < p_2$. Hence, there exists an $m \in \mathbb{N}$ such that $h^{m-1}(p) \in [p_2, \infty)$ and $h^m(p) \in [0, p_2)$. Since $h([p_2, \infty)) \subseteq [h(p_2), \infty) \subseteq [p_1, \infty)$ by (iv), we have $h^m(p) = h(h^{m-1}(p)) \in [p_1, \infty)$. Therefore, $h^m(p) \in [0, p_2) \cap [p_1, \infty) = [p_1, p_2)$. ☐

For $h([p_1, p_2)) \subseteq [p_1, p_2)$ in Proposition 1, (iv), we also say that $[p_1, p_2)$ is an invariant set under h. By Proposition 1, (iv) and (v), every solution of (31) enters $[p_1, p_2)$ for some m and remains within for all $k \geq m$. Therefore, attention can be restricted to the interval $[p_1, p_2)$ for studying conditions for asymptotic periodicity in the following.

From Proposition 1, (iii), the inverse of h exists on the range of h on $[p_1, p_2)$, which is $h([p_1, p_2)) = [p_1, h(p_2)) \cup [h(p_1), p_2)$ by (iv). Hence, we define the inverse h^{-1} as

$$\begin{aligned} h^{-1} \ &: [p_1, h(p_2)) \cup [h(p_1), p_2) \to [p_1, p_2) \\ y \ &\mapsto \ h^{-1}(y) \text{ such that } h(h^{-1}(y)) = y. \end{aligned} \tag{39}$$

The convergence proof in the following subsections is based on the *contraction mapping theorem* (also known as *Banach's fixed point theorem*).

Theorem 1—Contraction Mapping Theorem, [25] Let $\|\cdot\|$ be a norm for $\mathbb{R}^n$ and S a closed subset of $\mathbb{R}^n$. Assume $f : S \to S$ is a contraction mapping: there is an L, $0 \leq L < 1$, such that $\|f(p) - f(\tilde{p})\| \leq L\|p - \tilde{p}\|$ for all $p, \tilde{p}$ in S. Then f has a unique fixed point p^* in S. Furthermore, if $p(0) \in S$ and we set $p(k{+}1) = f(p(k))$, then

$$\|p(k) - p^*\| \leq \frac{L^k}{1 - L}\|p(1) - p(0)\| \quad (k \geq 0). \tag{40}$$

■

Equation (40) implies that $p(k)$ converges to p^* as $k \to \infty$ for any $p(0) \in S$.

4.2 Illustrative Example and Outline of the Proof

We now illustrate, by means of Example 1, the main ideas that are used in Sec. 4.3 to 4.6 to prove asymptotic periodicity of (31).

The graph of h for the parameters of Example 1 is shown in Fig. 5. Since there is no intersection with the identity diagonal, h has no fixed point, as expected. The graph of h^3, which is depicted in Fig. 6, does, however, have three intersections in $[p_1, p_2)$ with the identity diagonal. Hence, h^3 has three fixed points in this interval corresponding to the 3-cycle shown in Fig. 2(a).

We illustrate below how Theorem 1 can be used to systematically prove that h^3 has these three fixed points, and that they are (locally) attractive. This approach is then generalized in Sec. 4.3 to 4.6 to general solutions of (31). To be able to apply Theorem 1 (with $n = 1$, $f = h^3$, and $\|\cdot\| = |\cdot|$), there are two key requirements:

(R1) a suitable closed set S that is invariant under h^3 must be constructed, and

(R2) h^3 must be a contraction mapping on S.

As it shall be seen later, the discontinuities of the function h^3 play a crucial role in the development. The function h^3 has two discontinuities, which can be seen as follows:

- $h(p)$ is continuous for all $p \in [p_1, p_2)$ except at $d_1 := \bar{p} + \delta$.
- $h^2(p) = h(h(p))$ is continuous at p if h is continuous at p and if h is continuous at $h(p)$, [26]. Hence, points of discontinuity are d_1 (discontinuity of h); and $d_2 \in [p_1, p_2)$ such that $\bar{p} + \delta = d_1 = h(d_2)$. Since $d_1 \in \mathrm{dom}(h^{-1})$, the inverse h^{-1} exists and $d_2 = h^{-1}(d_1)$.
- Similarly, $h^3(p) = h(h^2(p))$ is continuous at p if h^2 is continuous at p and if h is continuous at $h^2(p)$. Points of discontinuity are d_1 and d_2 (discontinuities of h^2); and (potentially) $d_3 \in [p_1, p_2)$ such that $\bar{p}+\delta = d_1 = h^2(d_3) \Leftrightarrow d_2 = h(d_3)$. But since $d_2 \notin \mathrm{dom}(h^{-1})$ (cf. Fig. 5), such a d_3 does not exist. Hence, h^3 has the discontinuities d_1 and d_2.

The discontinuities d_1 and d_2 subdivide $[p_1, p_2)$ into three disjoint subintervals: $[p_1, p_2) = I_3 \cup I_2 \cup I_1$ with $I_3 := [p_1, d_2)$, $I_2 := [d_2, d_1)$, and $I_1 := [d_1, p_2)$. Figure 7 illustrates where one of the subintervals, I_1, is mapped by repeated application of h. It can be seen that $h^3(I_1) \subseteq [d_1, p_2) = I_1$. Furthermore, since $h(p_2) < d_2$ (cf. Fig. 5), the same property holds for the closure of I_1; that is, $[d_1, p_2]$ is invariant under h^3,

$$h^3([d_1, p_2]) \subseteq [d_1, p_2]. \tag{41}$$

Notice that $h([d_1, p_2])$ and $h^2([d_1, p_2])$ (the same intervals as $h(I_1)$ and $h^2(I_1)$ in Fig. 7, but with closed right bounds) are closed intervals contained in I_3 and I_2, respectively. It can be shown that, under h^3, they are invariant and attractive for any point in I_3 and I_2, respectively. Hence, we can construct closed sets invariant under h^3 (requirement (R1)).

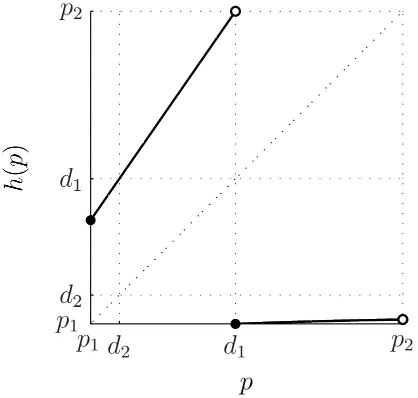


Figure 5. The function h for $a = 1.2$, $c = 1$, $\delta = 3$ on the interval $[p_1, p_2) = [2.20, 8.13)$. The function has a discontinuity at $d_1 = \bar{p} + \delta = 4.95$. The slope of h is a^2 on (p_1, d_1) and bounded by $g'(d_1)$ on (d_1, p_2) (cf. Fig. 4).

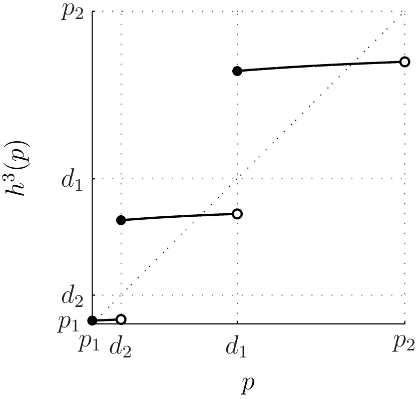


Figure 6. The function h^3 for $a = 1.2$, $c = 1$, $\delta = 3$ on the interval $[p_1, p_2) = [2.20, 8.13)$. The function has two discontinuities at $d_1 = \bar{p} + \delta = 4.95$ and $d_2 = 2.74$.

For (R2), we focus again on the interval I_1. Consider the derivative of h^3 on (d_1, p_2). By the chain rule, for $p \in (d_1, p_2)$,

$$\frac{d(h^3)}{dp}(p) = h'(h^2(p)) \cdot h'(h(p)) \cdot h'(p), \tag{42}$$

where h' refers to the first derivative $\frac{dh}{dp}$. Similar to the argumentation in

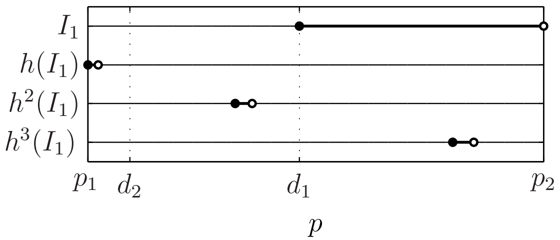


Figure 7. Mapping of the interval I_1 under repeated application of h. On the top line, the interval $I_1 = [d_1, p_2)$ is shown as a thick line. This interval is mapped to $h(I_1) = [h(d_1), h(p_2)) = [p_1, h(p_2))$ (cf. Fig. 5), shown on the second line from above. Notice that the obtained interval is significantly shorter due to the slope of h being significantly less than one on $[d_1, p_2)$ (cf. Fig. 5). The intervals $h^2(I_1) = h(h(I_1))$ and $h^3(I_1) = h(h^2(I_1))$ (third and fourth line from above) are obtained accordingly. The interval length increases for the latter two mappings, since the slope of h is greater than one on $[p_1, d_1)$. Still, after one cycle of three mappings, the resulting interval is contained in the original one, i.e. $h^3(I_1) \subseteq I_1$.

Fig. 7, one can see that $h((d_1, p_2)) \subseteq (p_1, d_2)$ and $h^2((d_1, p_2)) \subseteq h((p_1, d_2)) \subseteq (d_2, d_1)$. From Fig. 5, it can be seen that $h'(p) = a^2$ for all $p \in (p_1, d_2) \cup (d_2, d_1)$ and that $h'(p) < g'(d_1)$ for all $p \in (d_1, p_2)$. Therefore, for $p \in (d_1, p_2)$, we get from (42) the following:

$$\frac{d(h^3)}{dp}(p) < a^2 \cdot a^2 \cdot g'(d_1) = a^4 g'(\bar{p} + \delta) = 0.084. \tag{43}$$

From this, it follows (by the application of the mean value theorem, [26]) that for any closed interval $S \subseteq (d_1, p_2)$, the contraction mapping property in Theorem 1 holds with $L = a^4 g'(\bar{p} + \delta) < 1$. Even though the closed interval $[d_1, p_2]$ is not contained in (d_1, p_2), $\tilde{I}_1 := h^3([d_1, p_2])$ *is* contained (see Fig. 7). Furthermore, $\tilde{I}_1$ is itself invariant under h^3, which follows directly from (41): $h^3(\tilde{I}_1) = h^3(h^3([d_1, p_2])) \subseteq h^3([d_1, p_2]) = \tilde{I}_1$. Theorem 1 thus ensures that there exists a unique fixed point in $\tilde{I}_1$, and that every starting point in $\tilde{I}_1$ converges to this fixed point. Furthermore, since

$$h^3(I_1) = h^3([d_1, p_2)) \subseteq h^3([d_1, p_2]) = \tilde{I}_1, \tag{44}$$

the fixed point is attractive for all points in the original interval I_1.

For the intervals I_2 and I_3, one can proceed similarly and, hence, show that every point in $[p_1, p_2)$ converges to a fixed point of h^3. Furthermore, we know by Proposition 1, (iv) and (v), that every solution to (31) ends up in $[p_1, p_2)$. Therefore, the solution to (31) for the considered example is asymptotically 3-periodic for any initial value $p_0 \geq 0$.

To treat the general case in the remainder of this section, we proceed analogously to this example. The construction of N closed subintervals of $[p_1, p_2)$ that are invariant under h^N proceeds in two steps. First, half-closed intervals I_i are generated that cover $[p_1, p_2)$ and possess the sought invariance property (Sec. 4.3). Second, closed intervals $\tilde{I}_i \subseteq I_i$ are constructed that inherit the invariance property from their supersets (Sec. 4.4). In Sec. 4.5, we show that h^N is a contraction mapping on these intervals, which then allows us (in Sec. 4.6) to apply Theorem 1 and conclude that solutions to (31) are asymptotically N-periodic.

4.3 Invariant Subintervals (Left-Closed, Right-Open)

Motivated by the example of the previous subsection, the left-closed, right-open intervals I_i are obtained by splitting up $[p_1, p_2)$ through a sequence of points $\{d_1, d_2, \dots\}$, $d_i \in [p_1, p_2)$, which represent discontinuities of h^N and are obtained by iteratively applying h^{-1} until some $d_i \notin \mathrm{dom}(h^{-1})$. We give an algorithm to compute these discontinuities:

Algorithm 1

```
$d_1 := \bar{p} + \delta$
while $d_i \in \mathrm{dom}(h^{-1})$
    $d_{i+1} := h^{-1}(d_i)$
    increment $i$
end while
$N := i + 1$.
```

■

If there exists an $m \in \mathbb{N}$ such that $d_m \notin \mathrm{dom}(h^{-1})$, Algorithm 1 terminates, and the obtained sequence $\{d_1, d_2, \dots\}$ is finite. For all problems of an exhaustive search that we have conducted, this has actually been the case. For the purpose of this article, we assume henceforth that the algorithm terminates.

Assumption 1 Algorithm 1 terminates. ■

The assumption is essentially checked by running Algorithm 1 for a concrete problem; if the algorithm terminates, the assumption is true.

Proposition 2 Let $\mathcal{D}_i := \{d_1, \dots, d_i\}$ with d_i defined by Algorithm 1. The following statements hold:

(i) $\forall d_i, d_j \in \mathcal{D}_{N-1}$ with $i \neq j$, $d_i \neq d_j$.

(ii) $d_i \notin [h(p_2), h(p_1)]$, $\forall i < N-1$,
$d_{N-1} \in [h(p_2), h(p_1))$.

(iii) h^i is continuous on $[p_1, p_2) \setminus \mathcal{D}_i$, $\forall i \le N-1$,
h^N is continuous on $[p_1, p_2) \setminus \mathcal{D}_{N-1}$.

■

Proof. (i): Proof by contradiction. Assume there exist $d_i, d_j \in \mathcal{D}_{N-1}$ with $i \neq j$ and $d_i = d_j$. Assume w.l.o.g. $j > i$ and let $m := j - i \le N-2$. Then, from Algorithm 1, $d_i = d_j = h^{-1}(d_{j-1}) = h^{-2}(d_{j-2}) = \cdots = h^{-m}(d_i)$; that is, the sequence of d_i's is periodic with period m, and Algorithm 1 never terminates, which contradicts with Assumption 1.

(ii): By Assumption 1, the sequence $\{d_1, d_2, \dots\}$ defined by Algorithm 1 is finite and equal to $\mathcal{D}_{N-1}$. Therefore, $d_i \in \operatorname{dom}(h^{-1})$ for all $i < N-1$ and $d_{N-1} \notin \operatorname{dom}(h^{-1})$. From $\operatorname{dom}(h^{-1}) = [p_1, h(p_2)) \cup [h(p_1), p_2)$ (see (39)), it follows that $d_i \notin [h(p_2), h(p_1))$ for all $i < N-1$. Furthermore, $d_i \neq h(p_1)$ can be seen by contradiction: assuming $d_i = h(p_1)$, it follows from $h(p_1) \in \operatorname{dom}(h^{-1})$ and $p_1 \in \operatorname{dom}(h^{-1})$ that there is $d_{i+2} \in \mathcal{D}_{N-1}$ with $d_{i+2} = h^{-2}(d_i) = h^{-1}(p_1) = \bar{p} + \delta = d_1$, which contradicts with (i).

Since h^{-1} maps to $[p_1, p_2)$ (see (39)), we have $d_{N-1} = h^{-1}(d_{N-2}) \in [p_1, p_2)$. Together with $d_{N-1} \notin \operatorname{dom}(h^{-1})$, this implies that $d_{N-1} \in [p_1, p_2) \setminus \big([p_1, h(p_2)) \cup [h(p_1), p_2)\big) = [h(p_2), h(p_1))$.

(iii): First, we prove by induction that h^i is continuous on $[p_1, p_2) \setminus \mathcal{D}_i$ for all $i \le N-1$. From Proposition 1, (i), it follows that the statement is true for $i = 1$. Assume the statement holds for some $i \le N-2$ (induction assumption (IA)); and consider

$$h^{i+1}(p) = h(h^i(p)), \quad p \in [p_1, p_2). \tag{45}$$

If h^i is continuous at p and h is continuous at $h^i(p)$, then the composition h^{i+1} is continuous at p, [26]. Hence, h^{i+1} is guaranteed to be continuous on $[p_1, p_2)$ except for the points $\mathcal{D}_i$ and the point $\tilde{p}$ with $h^i(\tilde{p}) = d_1$ (d_1 is the discontinuity of h). But $h^i(\tilde{p}) = d_1 \Leftrightarrow \tilde{p} = h^{-i}(d_1) = d_{i+1}$ (since $i \le N-2$, the i-times application of the inverse map, h^{-i}, is defined). Therefore, h^{i+1} is continuous on $[p_1, p_2) \setminus (\mathcal{D}_i \cup \{d_{i+1}\}) = [p_1, p_2) \setminus \mathcal{D}_{i+1}$.

Next, we prove that h^N is continuous on $[p_1, p_2) \setminus \mathcal{D}_{N-1}$. For this, consider

$$h^N(p) = h(h^{N-1}(p)), \quad p \in [p_1, p_2). \tag{46}$$

By the same argument as above, h^N is guaranteed to be continuous on $[p_1, p_2)$ except for the points $\mathcal{D}_{N-1}$ and the point $\tilde{p}$ with $h^{N-1}(\tilde{p}) = d_1 \Leftrightarrow$

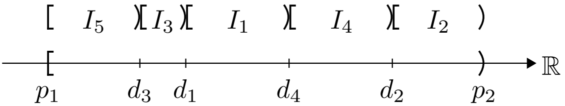


Figure 8. The left-closed, right-open subintervals $\mathcal{I} = \{I_1, I_2, I_3, I_4, I_5\}$ generated by the points $\mathcal{D}_4 = \{d_1, d_2, d_3, d_4\}$ cover the interval $[p_1, p_2)$.

$h(\tilde{p}) = h^{-(N-2)}(d_1) = d_{N-1}$. But a point $\tilde{p}$ with $h(\tilde{p}) = d_{N-1}$ does not exist in $[p_1, p_2)$ since $d_{N-1} \in [h(p_2), h(p_1))$ (by (ii)), which is not in the range of h (by Proposition 1, (iv)). Therefore, h^N is continuous on $[p_1, p_2) \setminus \mathcal{D}_{N-1}$. $\square$

The points $\mathcal{D}_{N-1}$ divide the interval $[p_1, p_2)$ into N subintervals $\mathcal{I} := \{I_1, \ldots, I_N\}$ as illustrated in Fig. 8. The intervals are named such that I_i has d_i as a lower bound for $i \leq N-1$, and I_N has the lower bound p_1. A formal definition of the intervals is given next. Let $\Pi : \{1, \ldots, N\text{–}1\} \to \{1, \ldots, N\text{–}1\}$ be a permutation of the d_i's such that

$$d_{\Pi(i)} < d_{\Pi(i+1)}, \quad \forall i \in \{1, \ldots, N-2\}. \tag{47}$$

Furthermore, let $\underline{i}$ and $\bar{i}$ be the indices of the smallest and greatest d_i, i.e. $\Pi(1) = \underline{i}$ and $\Pi(N-1) = \bar{i}$. Then define

$$I_i := [d_i, d_{\Pi(\Pi^{-1}(i)+1)}) \quad \forall i \leq N-1, i \neq \bar{i} \tag{48}$$

$$I_{\bar{i}} := [d_{\bar{i}}, p_2) \tag{49}$$

$$I_N := [p_1, d_{\underline{i}}); \tag{50}$$

that is, interval I_i has d_i as a lower bound (closed) and the next bigger element from $\mathcal{D}_{N-1}$ as an upper bound (open) (except for the intervals at the boundaries of $[p_1, p_2)$). Since each interval is uniquely specified from (48)–(50) by either its lower or its upper bound, we sometimes omit either one of them and write $[d, *)$ or $[*, d)$. For the interior (the largest contained open interval) of I_i, we write $\mathrm{int}(I_i)$.

Proposition 3 All intervals $I_i \in \mathcal{I}$ are mutually disjoint and nonempty. $\blacksquare$

Proof. Disjointness of the intervals is given by their construction and Proposition 2, (i). Furthermore, because of Proposition 2, (i), the intervals (48) are not empty. Since $d_{\bar{i}} \in \mathcal{D}_{N-1}$, it follows that $d_{\bar{i}} \in [p_1, p_2)$ and $d_{\bar{i}} < p_2$; therefore, interval $I_{\bar{i}}$ in (49) is not empty. To see that I_N in (50) is not empty, we assume that it is and lead this to a contradiction. From $[p_1, d_{\underline{i}}) = \emptyset$ it follows that $p_1 = d_{\underline{i}}$ ($p_1 > d_{\underline{i}}$ is not possible since $d_{\underline{i}} \in [p_1, p_2)$). From $d_{\underline{i}} = p_1 \in \mathrm{dom}(h^{-1})$, it follows that $d_{\underline{i}+1}$ is defined by Algorithm 1 and $d_{\underline{i}+1} = h^{-1}(d_{\underline{i}}) = h^{-1}(p_1) = h^{-1}(h(\bar{p} + \delta)) = \bar{p} + \delta = d_1$. But $d_{\underline{i}+1} = d_1$ (with $\underline{i} \geq 1$) contradicts Proposition 2, (i). □

PROPOSITION 4 The following statements hold:

(i) $h(I_N) \subseteq I_{N-1}$, $h(I_{N-1}) \subseteq I_{N-2}$, $\ldots$, $h(I_2) \subseteq I_1$, and $h(I_1) \subseteq I_N$.

(ii) $h(\mathrm{int}(I_N)) \subseteq \mathrm{int}(I_{N-1})$, $h(\mathrm{int}(I_{N-1})) \subseteq \mathrm{int}(I_{N-2})$, $\ldots$, $h(\mathrm{int}(I_2)) \subseteq \mathrm{int}(I_1)$, and $h(\mathrm{int}(I_1)) \subseteq \mathrm{int}(I_N)$.

■

The following lemma is used in the proof of this proposition (statements (i) and (ii)) and later in Sec. 4.4 ((iii) and (iv)).

LEMMA 1 Consider the collection $\mathcal{I} = \{I_1, \ldots, I_N\}$ of intervals I_i defined by (48)–(50); and let $\mathcal{I}_{\mathrm{int}} := \{\mathrm{int}(I_1), \ldots, \mathrm{int}(I_N)\}$. The following statements hold:

(i) $\mathcal{I} \xrightarrow{h} \mathcal{I}$.

(ii) $\mathcal{I}_{\mathrm{int}} \xrightarrow{h} \mathcal{I}_{\mathrm{int}}$.

(iii) $I_{\bar{i} -_N 1} = \begin{cases} [d_{\bar{i}-1}, d_{N-1}) & \bar{i} > 1 \\ [p_1, d_{N-1}) & \bar{i} = 1. \end{cases}$

(iv) $\mathrm{int}(I_{N-1}) = \begin{cases} (d_{N-1}, d_{\underline{i}-1}) & \underline{i} > 1 \\ (d_{N-1}, p_2) & \underline{i} = 1. \end{cases}$

■

Proof. The proof is given in Appendix A. □

Proof of Proposition 4. We present the proof of (i) and (ii) simultaneously (where required, we distinguish (i) from (ii) by placing the latter in square brackets). From Lemma 1, (i) and (ii), we know that, for any $I \in \mathcal{I}$ $[I \in \mathcal{I}_{\text{int}}]$, $h(I)$ is contained in an interval of $\mathcal{I}$ $[\mathcal{I}_{\text{int}}]$. Since the intervals are disjoint (Proposition 3), there is exactly one interval that contains $h(I)$. Therefore, it suffices to consider only the lower bound of an interval to identify where the interval is mapped to.

Notice that, since h is strictly increasing (Proposition 1, (i)), for all $[a, b) \in \mathcal{I}$ $[(a, b) \in \mathcal{I}_{\text{int}}]$, $h([a, b)) = [h(a), \lim_{p \nearrow b} h(p))$ $[h((a, b)) = (h(a), \lim_{p \nearrow b} h(p))]$; that is, the bounds of the mapped interval are given by the map of the bounds of the argument interval. From Algorithm 1, it follows that $h(d_i) = d_{i-1}$ for all $i \in \{2, \dots, N-1\}$. Since there is exactly one interval in $\mathcal{I}$ $[\mathcal{I}_{\text{int}}]$ with d_{i-1} as lower bound, for all $i \in \{2, \dots, N-1\}$,

$$h(I_i) = h([d_i, *)) = [d_{i-1}, *) \subseteq I_{i-1} \tag{51}$$

$$[h(\text{int}(I_i)) = h((d_i, *)) = (d_{i-1}, *) \subseteq \text{int}(I_{i-1})]. \tag{52}$$

Similarly, since $h(d_1) = h(\bar{p} + \delta) = p_1$ by the definitions of d_1 and p_1, it follows that

$$\begin{gathered} h(I_1) = h([d_1, *)) = [p_1, *) \subseteq I_N \\ [h(\text{int}(I_1)) = h((d_1, *)) = (p_1, *) \subseteq \text{int}(I_N)]. \end{gathered} \tag{53}$$

From Proposition 2, (ii), it follows that $h(p_1) \in [d_{N-1}, *) = I_{N-1}$ $[h(p_1) \in (d_{N-1}, *) = \text{int}(I_{N-1})]$. Therefore,

$$\begin{gathered} h(I_N) = h([p_1, *)) = [h(p_1), *) \subseteq I_{N-1} \\ [h(\text{int}(I_N)) = h((p_1, *)) = (h(p_1), *) \subseteq \text{int}(I_{N-1})]. \end{gathered} \tag{54}$$

□

Corollary 1 The following statements hold:

(i) $h^N(I_i) \subseteq I_i \;\; \forall I_i \in \mathcal{I}$.

(ii) $h^N(\text{int}(I_i)) \subseteq \text{int}(I_i) \;\; \forall I_i \in \mathcal{I}$.

■

Proof. (i) and (ii) follow directly from Proposition 4 and the fact: for two sets S_1, S_2 and a function f, $S_1 \subseteq S_2 \Rightarrow f(S_1) \subseteq f(S_2)$. □

4.4 Invariant Closed Subintervals

The intervals $\mathcal{I}$ cover the whole domain of interest $[p_1, p_2)$, and they are invariant under h^N. However, closed intervals are required if Theorem 1 is to be applied. The proposition below states that such subintervals $\tilde{I}_i \subseteq I_i$ exist. Another technical assumption is required for this proposition:

Assumption 2 $\quad h(p_2) \neq d_{N-1}$. ■

Notice that with $h(p_2) \leq d_{N-1}$ by Proposition 2, (ii), this implies

$$h(p_2) < d_{N-1}. \tag{55}$$

Proposition 5 There exists a collection of intervals $\tilde{\mathcal{I}} = \{\tilde{I}_1, \tilde{I}_2, \ldots, \tilde{I}_N\}$ such that for all $i \in \{1, \ldots, N\}$ the following statements hold:

(i) $\tilde{I}_i$ is closed.

(ii) $\tilde{I}_i \subseteq \operatorname{int}(I_i) \subseteq I_i$.

(iii) $h^N(\tilde{I}_i) \subseteq \tilde{I}_i$.

(iv) $h^{2N}(I_i) \subseteq \tilde{I}_i$.

■

Proof. We define N intervals $\tilde{I}_1, \ldots, \tilde{I}_N$ and prove that the properties (i)–(iv) hold for these. Let $m_1 := \bar{i} + 1$ (> 1). We define recursively

$$\tilde{I}_{N-1} := h^{m_1}([d_{\bar{i}}, p_2]), \tag{56}$$

$$\tilde{I}_{i -_N 1} := h(\tilde{I}_i) \qquad \forall i \in \{1, \ldots, N-1\}, \tag{57}$$

where '$-_N$' is defined in (1).

We first show that (i)–(iii) hold for $\tilde{I}_{N-1}$. Notice that $\bar{i} \in \{1, \ldots, N-1\}$. We have

$$\begin{aligned} h([d_{\bar{i}}, p_2]) &\underset{\text{Prop. 1 (i)}}{=} [h(d_{\bar{i}}), h(p_2)] = \begin{cases} [d_{\bar{i}-1}, h(p_2)] & \text{if } \bar{i} > 1 \\ [p_1, h(p_2)] & \text{if } \bar{i} = 1 \end{cases} \\ &\underset{(55)}{\subseteq} \begin{cases} [d_{\bar{i}-1}, d_{N-1}) & \text{if } \bar{i} > 1 \\ [p_1, d_{N-1}) & \text{if } \bar{i} = 1 \end{cases} \underset{\text{Lem. 1 (iii)}}{=} I_{\bar{i} -_N 1}. \end{aligned} \tag{58}$$

From Proposition 4, it follows that, for all $i \in \{1, \dots, N\}$ and for all $m \in \mathbb{N}$,

$$h^m(I_i) \subseteq I_{i -_N m}, \tag{59}$$

$$h^m(\mathrm{int}(I_i)) \subseteq \mathrm{int}(I_{i -_N m}). \tag{60}$$

With this and (58),

$$h^{\bar{i}}([d_{\bar{i}}, p_2]) = h^{\bar{i}-1}(h([d_{\bar{i}}, p_2])) \subseteq h^{\bar{i}-1}(I_{\bar{i} -_N 1}) \subseteq I_{(\bar{i} -_N 1) -_N (\bar{i}-1)} = I_N,$$

and

$$\begin{aligned}
\tilde{I}_{N-1} &= h^{m_1}([d_{\bar{i}}, p_2]) = h^{\bar{i}+1}([d_{\bar{i}}, p_2]) \subseteq h(I_N) \underset{(50)}{=} h([p_1, d_{\underline{i}})) \\
&\underset{\text{Prop. 1 (i)}}{=} \begin{cases} [h(p_1), d_{\underline{i}-1}) & \text{if } \underline{i} > 1 \\ [h(p_1), p_2) & \text{if } \underline{i} = 1 \end{cases} \\
&\underset{\text{Prop. 2 (ii)}}{\subseteq} \begin{cases} (d_{N-1}, d_{\underline{i}-1}) & \text{if } \underline{i} > 1 \\ (d_{N-1}, p_2) & \text{if } \underline{i} = 1 \end{cases} \underset{\text{Lem. 1 (iv)}}{=} \mathrm{int}(I_{N-1}) \subseteq I_{N-1}.
\end{aligned} \tag{61}$$

Thus, (ii) holds for $\tilde{I}_{N-1}$.

Property (i) can be seen as follows: $h([d_{\bar{i}}, p_2]) = [h(d_{\bar{i}}), h(p_2)]$ is closed. From (58) and Proposition 1, (i), it follows that h is continuous and strictly increasing on $h([d_{\bar{i}}, p_2])$. Similarly, by (59), $h^m([d_{\bar{i}}, p_2]) = h^{m-1}(h([d_{\bar{i}}, p_2])) \subseteq h^{m-1}(I_{\underline{i} -_N 1}) \subseteq I_{\underline{i} -_N m}$, $m \geq 1$; thus, h is continuous and strictly increasing on $h^m([d_{\bar{i}}, p_2])$. Since, for a continuous and strictly increasing function f and $a, b \in \mathbb{R}$, $f([a, b]) = [f(a), f(b)]$ (the image of a closed interval under f is a closed interval), $h^m([d_{\bar{i}}, p_2])$ is closed for any $m \geq 1$ and, in particular, for $m = m_1$.

To show (iii) for $\tilde{I}_{N-1}$, let $m_2 := N - m_1$ (≥ 0). We then get with (61), (59), and (49), $h^{m_2}(\tilde{I}_{N-1}) \subseteq h^{m_2}(I_{N-1}) \subseteq I_{(N-1) -_N m_2} = I_{\bar{i}} = [d_{\bar{i}}, p_2) \subseteq [d_{\bar{i}}, p_2]$. Property (iii) then follows by

$$h^N(\tilde{I}_{N-1}) = h^{m_1}(h^{m_2}(\tilde{I}_{N-1})) \subseteq h^{m_1}([d_{\bar{i}}, p_2]) \underset{(56)}{=} \tilde{I}_{N-1}.$$

Hence, we have shown that (i)–(iii) hold for $i = N - 1$. We next prove (i)–(iii) for $i \in \{1, \dots, N-2, N\}$ by induction.

Induction assumption (IA): (i)–(iii) are valid for some $i \in \{1, \dots, N-1\}$. Show that this implies that (i)–(iii) hold for $i -_N 1$.

Property (ii) holds since

$$\tilde{I}_{i -_N 1} \underset{(57)}{=} h(\tilde{I}_i) \underset{\text{IA (ii)}}{\subseteq} h(\mathrm{int}(I_i)) \underset{(60)}{\subseteq} \mathrm{int}(I_{i -_N 1}) \subseteq I_{i -_N 1}.$$

Since $\tilde{I}_i \subseteq I_i$ (IA (ii)), h is continuous and strictly increasing on $\tilde{I}_i$. Moreover, $\tilde{I}_i$ is closed (IA (i)). Together, this implies that the image under h, $\tilde{I}_{i-_N 1} = h(\tilde{I}_i)$, is also closed; hence, (i) is true.

Property (iii) can be seen to hold by

$$h^N(\tilde{I}_{i-_N 1}) \underset{(57)}{=} h^{N+1}(\tilde{I}_i) = h(h^N(\tilde{I}_i)) \underset{\text{IA (iii)}}{\subseteq} h(\tilde{I}_i) \underset{(59)}{=} \tilde{I}_{i-_N 1}.$$

This completes the proof of statements (i)–(iii).

To see statement (iv), take $I_i \in \mathcal{I}$ for any $i \in \{1, \ldots, N\}$. Let $m_3 := i -_N \bar{i}$ (≥ 1). Then, from (59) and (49), $h^{m_3}(I_i) \subseteq I_{i-_N m_3} = I_{i-_N(i-_N \bar{i})} = I_{\bar{i}} = [d_{\bar{i}}, p_2) \subseteq [d_{\bar{i}}, p_2]$, and, with this and (56), $h^{m_1+m_3}(I_i) = h^{m_1}(h^{m_3}(I_i)) \subseteq h^{m_1}([d_{\bar{i}}, p_2]) = \tilde{I}_{N-1}$. Let $m_4 := (N -_N i) - 1$ ($0 \leq m_4 \leq N-1$). Then, with the preceding expression, (56), and (57), we get

$$\begin{aligned} h^{m_1+m_3+m_4}(I_i) &= h^{m_4}(h^{m_1+m_3}(I_i)) \subseteq h^{m_4}(\tilde{I}_{N-1}) \\ &= \tilde{I}_{(N-1)-_N m_4} = \tilde{I}_{(N-1)-_N((N-_N i)-1)} = \tilde{I}_i. \end{aligned} \tag{62}$$

Now, consider three different cases for i.

First case: $i = N$. Since $m_1+m_3+m_4 = (\bar{i}+1)+(N-\bar{i})+(N-1) = 2N$, (iv) follows directly from (62).

Second case: $\bar{i} < i < N$. Since $m_1 + m_3 + m_4 = (\bar{i}+1) + (i-\bar{i}) + (N-i-1) = N$, (62) reads $h^N(I_i) \subseteq \tilde{I}_i$, which implies (iv) as follows: $h^{2N}(I_i) = h^N(h^N(I_i)) \underset{(62)}{\subseteq} h^N(\tilde{I}_i) \underset{\text{(iii)}}{\subseteq} \tilde{I}_i$.

Third case: $1 \leq i \leq \bar{i}$. Since $m_1+m_3+m_4 = (\bar{i}+1)+(i-\bar{i}+N)+(N-i-1) = 2N$, (iv) follows from (62). □

4.5 Contraction Mapping

In this section, we show that h^N is a contraction mapping (i.e. it has a Lipschitz constant strictly less than one, cf. Theorem 1) on each of the intervals $\tilde{\mathcal{I}}$. To this end, we first derive an upper bound less than one on the derivative of h^N for the interior of the intervals $\mathcal{I}$.

PROPOSITION 6 h^N is differentiable on all open intervals $\operatorname{int}(I_i)$, $I_i \in \mathcal{I}$. Furthermore, there exists an L, $0 \leq L < 1$, such that

$$\left|\frac{d(h^N)}{dp}(p)\right| < L \quad \forall p \in \operatorname{int}(I_i), \forall I_i \in \mathcal{I}.$$

■

The following Lemma is needed in the proof.

LEMMA 2 For all $p \in [p_1, \bar{p}+\delta)$, there exists an $m(p) \in \mathbb{N}^+$ such that

$$p, h(p), \dots, h^{m(p)-1}(p) < \bar{p}+\delta \quad \text{and} \quad h^{m(p)}(p) \geq \bar{p}+\delta. \tag{63}$$

Furthermore, there exists an $\bar{N} \in \mathbb{N}^+$ (independent of p) such that $m(p) \leq \bar{N}$, and

$$a^{2\bar{N}} < a^2 \frac{\bar{p}+\delta}{p_1}. \tag{64}$$

■

Proof. The proof is given in Appendix B. □

The lemma says that if $p(0)$ starts anywhere in $[p_1, \bar{p}+\delta)$, there is a maximum number $\bar{N}$ of iterations (31), for which $p(k)$ remains in $[p_1, \bar{p}+\delta)$.

Proof of Proposition 6. Take any $I_i \in \mathcal{I}$ and $\tilde{p} \in \operatorname{int}(I_i)$.

Differentiability: By Proposition 1, (ii), h is differentiable at $\tilde{p}$. We prove by induction that h^j is differentiable at $\tilde{p}$ for all $1 \leq j \leq N$.

Induction assumption (IA): h^j is differentiable at $\tilde{p}$. By the chain rule, [26], $h^{j+1}(\tilde{p}) = h(h^j(\tilde{p}))$ is differentiable at $\tilde{p}$ if h^j is differentiable at $\tilde{p}$ (IA), and h is differentiable at $h^j(\tilde{p})$. From Proposition 4, (ii), it follows that

$$h^j(\tilde{p}) \in \operatorname{int}(I_{i -_N j}). \tag{65}$$

Since h is differentiable on any interval $\operatorname{int}(I)$, $I \in \mathcal{I}$ (Proposition 1, (ii)), the differentiability of h^{j+1} at $\tilde{p}$ follows.

Contraction mapping: By the chain rule,

$$(h^N)'(\tilde{p}) = h'(h^{N-1}(\tilde{p})) \cdot (h^{N-1})'(\tilde{p}) = \prod_{j \in \{0,\dots,N-1\}} h'(h^j(\tilde{p})) = \prod_{p \in \mathcal{P}} h'(p), \tag{66}$$

with $\mathcal{P} := \{\tilde{p}, h(\tilde{p}), \dots, h^{N-1}(\tilde{p})\}$. Notice from (65) for $j \in \{0, 1, \dots, N-1\}$ that, for every point $p \in \mathcal{P}$, there is exactly one interval $I \in \mathcal{I}$ such that $p \in \operatorname{int}(I)$.

Let $\mathcal{I}_L \subset \mathcal{I}$ denote the set of all intervals $I \in \mathcal{I}$ with $I < \bar{p}+\delta$ (intervals left of the discontinuity $\bar{p}+\delta$), and let $\mathcal{I}_R \subset \mathcal{I}$ denote the set of all $I \in \mathcal{I}$ with $I \geq \bar{p}+\delta$ (intervals right of the discontinuity $\bar{p}+\delta$). Furthermore, let N_L and N_R denote the number of elements in $\mathcal{I}_L$ and $\mathcal{I}_R$, respectively. Notice that $N_L \geq 1$ and $N_R \geq 1$ by the construction of the intervals. Then,

$$h'(p) = a^2 > 0 \qquad \forall p \in \operatorname{int}(I),\, I \in \mathcal{I}_L, \tag{67}$$

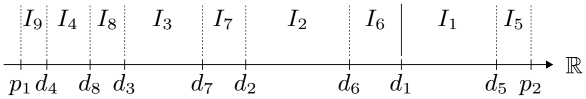


Figure 9. Illustration of the intervals $\mathcal{I}$ obtained for the parameter values $a = 1.2$, $c = 1$, and $\delta = 9.6$ (for better visibility the relative scaling of the intervals has been adapted). There are two distinct interval subsequences satisfying (70): $\underline{\mathcal{I}}_1 = \{I_4, I_3, I_2\}$ and $\underline{\mathcal{I}}_2 = \{I_9, I_8, I_7, I_6\}$.

follows directly from (32); and

$$0 < h'(p) = g'(p) < g'(\bar{p} + \delta) \qquad \forall p \in \operatorname{int}(I),\ I \in \mathcal{I}_R, \tag{68}$$

where the first inequality follows from $g'(p) = \frac{a^2}{(c^2p+1)^2} > 0$, and the second inequality follows from g' being strictly decreasing, which is seen from $g''(p) = -\frac{2a^2c^2}{(c^2p+1)^3} < 0$. With these results, it follows from (66) that

$$0 < (h^N)'(\tilde{p}) < a^{2N_L} \big(g'(\bar{p} + \delta)\big)^{N_R}. \tag{69}$$

Since $a^2 > 1$ and $g'(\bar{p} + \delta) < 1$, whether the map h^N is contractive depends on the ratio of N_R to N_L, which is investigated next.

Define a subset $\underline{\mathcal{I}} \subset \mathcal{I}$ as a maximum sequence of m intervals $I_\ell, I_{\ell -_N 1}, \dots$ all being left of $\bar{p} + \delta$:

$$\begin{aligned} \underline{\mathcal{I}} &:= \big\{I_\ell, I_{\ell -_N 1}, \dots, I_{\ell -_N (m-1)}\big\},\ m \le N_L, \\ &\quad \text{such that } I_\ell, I_{\ell -_N 1}, \dots, I_{\ell -_N (m-1)} \in \mathcal{I}_L, \text{ and } I_{\ell -_N (N-1)}, I_{\ell -_N m} \in \mathcal{I}_R, \end{aligned} \tag{70}$$

Let there be $\kappa \ge 1$ distinct interval sequences (70), which we call $\underline{\mathcal{I}}_1, \dots, \underline{\mathcal{I}}_\kappa$ with $m_1, \dots, m_\kappa$ their numbers of elements, respectively. An example with two interval sequences $\underline{\mathcal{I}}_1$, $\underline{\mathcal{I}}_2$ is provided in Fig. 9. Notice that $N_L = m_1 + \dots + m_\kappa$.

From Lemma 2, it follows that $m_j \le \bar{N}$ for all $j \le \kappa$ ($\bar{N}$ is as defined in Lemma 2), and

$$N_L = m_1 + \dots + m_\kappa \le \kappa \bar{N}. \tag{71}$$

For each sequence of intervals $\underline{\mathcal{I}}_j$, $j \le \kappa$, there is at least one distinct interval $I \in \mathcal{I}_R$ (namely, $I_{\ell -_N M}$); hence,

$$N_R \ge \kappa. \tag{72}$$

Combining (71) and (72), we obtain the sought bound on the ratio of N_L and N_R: $N_L \leq \kappa \bar{N} \leq N_R \bar{N}$. With this result, we can rewrite (69),

$$\begin{aligned} 0 &< (h^N)'(\tilde{p}) < a^{2N_L} \big(g'(\bar{p}+\delta)\big)^{N_R} \\ &\leq a^{2N_L}\, a^{2(N_R\bar{N}-N_L)} \left(g'(\bar{p}+\delta)\right)^{N_R} = \big(a^{2\bar{N}}\, g'(\bar{p}+\delta)\big)^{N_R}. \end{aligned} \tag{73}$$

Using (64) from Lemma 2, we get

$$a^{2\bar{N}} g'(\bar{p}+\delta) < a^2 \frac{\bar{p}+\delta}{p_1}\, g'(\bar{p}+\delta). \tag{74}$$

The right-hand side in (74) depends on the problem parameters, and it can be shown to be less than one (the proof is omitted due to space constraints; the symbolic calculation is provided in a supplementary Matlab script). With this, the statement of Proposition 6 follows from (73) with $L := (a^{2\bar{N}}\, g'(\bar{p}+\delta))^{N_R} < 1$. □

Corollary 2 h^N is a contraction mapping on any interval of $\tilde{\mathcal{I}}$ (defined in Proposition 5); that is, there exists an L, $0 \leq L < 1$, such that

$$|h^N(p) - h^N(\tilde{p})| \leq L|p - \tilde{p}| \quad \forall p, \tilde{p} \in \tilde{I}_i, \forall \tilde{I}_i \in \tilde{\mathcal{I}}.$$

■

Proof. Take any $p, \tilde{p} \in \tilde{I}_i$ with $\tilde{p} < p$ without loss of generality. By Proposition 2, (iii), and 5, (ii), h^N is continuous on $[\tilde{p}, p]$ and, by Proposition 6, h^N is differentiable on $(\tilde{p}, p)$. The claim then follows from the mean value theorem, [26]. □

4.6 Main Result

Equipped with the results of the previous subsections, we can now state the main result of this section:

Theorem 2 Under Assumptions 1 and 2, the solution to (31) is asymptotically N-periodic for any initial condition $p_0 \geq 0$. ■

Proof. By Proposition 1, (iv) and (v), it follows that there exists an $m_1 \in \mathbb{N}$ such that

$$h^{m_1 N}(p_0) \in [p_1, p_2). \tag{75}$$

Since the disjoint intervals $\mathcal{I}$ cover $[p_1, p_2)$, there exists a unique $i \in \{1, \ldots, N\}$ such that

$$h^{m_1 N}(p_0) \in I_i. \tag{76}$$

By Proposition 5, (iv),

$$h^{(m_1+2)N}(p_0) \in \tilde{I}_i. \tag{77}$$

From Proposition 5, (i) and (iii), Corollary 2, and Theorem 1, it follows that there exists a unique fixed point p_i^* of h^N (hence, an N-periodic point of (31)) in $\tilde{I}_i$ and that, for all $\tilde{p} \in \tilde{I}_i$,

$$\lim_{m \to \infty} h^{mN}(\tilde{p}) = p_i^*. \tag{78}$$

In particular, for $\tilde{p} = h^{(m_1+2)N}(p_0)$ and by (77),

$$\begin{aligned} \lim_{m \to \infty} h^{mN}\big(h^{(m_1+2)N}(p_0)\big) &= \lim_{m \to \infty} h^{(m_1+2+m)N}(p_0) \\ &= \lim_{m \to \infty} h^{mN}(p_0) = p_i^*. \end{aligned} \tag{79}$$

□

Theorem 2 essentially offers a sufficiency test for periodicity (if the two assumptions are satisfied, convergence to a periodic solution with a known period is guaranteed) as an alternative to simulating (28) and having to interpret the result.

5. Discussion

The proposed method for event-based state estimation is a direct extension of the classic Kalman filter to a distributed estimation problem with costly communication. Starting from the design of a discrete-time Kalman filter with access to all sensor measurements at every sampling time, the presented method allows the designer to trade off communication requirements with estimation performance by selecting suitable thresholds for each sensor (these thresholds are the only additional tuning parameters).

The estimation method can be implemented as shown in Fig. 1, where each sensor computes the variance of the estimator on-line and uses it to decide whether or not to broadcast its measurement. This way, the algorithm can adapt the sensor transmit rates in real time (for example, for non-stationary noise processes or to respond to packet drops). Alternatively, a periodic solution to the estimator variance iteration for a stationary process allows the recovery of a time-based implementation: the periodic transmit sequence can be precomputed and implemented on a sensor as a time-based schedule, which reduces the computational requirements on the sensor. The rate of the periodic transmission is, however, not a design parameter as in traditional time-sampled estimation, but obtained from an event-based approach, where the designer specifies tolerable bounds on the estimation error variance. Hence, the method can be used as a tool for designing update rates in a sensor network.

The presented event-based estimator is the optimal Bayesian estimator given the triggering policy that a measurement is transmitted if, and only if, its prediction variance exceeds a given threshold (equation (17)). The event-based estimator variance being periodic in the limit (which was proven herein for the scalar case under certain assumptions) means that a periodic transmit schedule is optimal for this problem in steady state. The result thus creates an interesting link between event-based and time-based optimal state estimation. Whether the convergence result for the scalar case generalizes to the matrix case (i.e. to equation (20)) is an open question.

Various extensions of the presented event-based estimation approach are conceivable. Instead of making the transmit decision based on a measurement's prediction variance, it could be based on its prediction error; that is, the difference between the measurement and its prediction mean (see [4, 5]). A promising approach is to combine the two methods by augmenting fixed minimum sensor communication rates to keep the variance bounded with triggering thresholds on real-time prediction errors. If improving the estimator performance locally on each agent is of interest (for example, when the estimate is used in feedback for controlling an actuator), a second estimator can be used to compute an improved estimate from using the data received from the network and, additionally, exploiting its local sensor data at every time step (see [4]).

Appendix

A. Proof of Lemma 1

For the proof of Lemma 1 (at the end of this section), we need the following lemma and corollary.

Lemma 3 Let $\mathcal{I} = \{I_1, I_2, \dots, I_N\}$ be a collection of nonempty, mutually disjoint intervals $I_i := [a_i, b_i)$ (or $I_i := (a_i, b_i)$) with $a_i, b_i \in \mathbb{R}$. A unique representation of $\mathcal{I}$ is given by the sets $\mathcal{L} = \{a_1, a_2, \dots, a_N\}$ and $\mathcal{U} = \{b_1, b_2, \dots, b_N\}$ of all lower and upper bounds, respectively, in the following sense: the collection $\bar{\mathcal{I}} := \{\bar{I}_1, \bar{I}_2, \dots, \bar{I}_N\}$ of intervals constructed such that, for all i, j with $1 \leq i, j \leq N$,

$$\bar{I}_i := [\alpha_i, \beta_i) \text{ (or } \bar{I}_i := (\alpha_i, \beta_i)), \ \alpha_i \in \mathcal{L}, \ \beta_i \in \mathcal{U}, \tag{80}$$

$$\bar{I}_i \neq \emptyset, \quad \text{and} \quad \bar{I}_i \cap \bar{I}_j = \emptyset, \tag{81}$$

exists, it is unique, and $\bar{\mathcal{I}} = \mathcal{I}$. ■

This lemma is useful, since it allows one to work with the (unordered) sets of interval bounds $\mathcal{L}$ and $\mathcal{U}$ instead of the actual intervals. The unique relationship between the bounds (which lower bound belongs to which upper bound) essentially follows from all intervals being disjoint and nonempty.

Proof. We present the proof simultaneously for the case of left-closed, right-open intervals $\bar{I}_i = [\alpha_i, \beta_i)$ and for the case of open intervals $\bar{I}_i = (\alpha_i, \beta_i)$ (where required to distinguish the two, the latter case is augmented in square brackets).

Since, for all $i \leq N$, $I_i \in \mathcal{I}$ is nonempty, $a_i < b_i$. Since the intervals $\mathcal{I}$ are mutually disjoint, there exists a permutation of indices $\tilde{\Pi} : \{1, \dots, N\} \to \{1, \dots, N\}$ such that

$$a_{\tilde{\Pi}(1)} < b_{\tilde{\Pi}(1)} \leq a_{\tilde{\Pi}(2)} < b_{\tilde{\Pi}(2)} \leq \dots \leq a_{\tilde{\Pi}(N)} < b_{\tilde{\Pi}(N)}.$$

Assume w.l.o.g. (by renaming of the intervals in $\mathcal{I}$) that

$$a_1 < b_1 \leq a_2 < b_2 \leq \dots \leq a_N < b_N. \tag{82}$$

Notice that the choice

$$\alpha_i = a_i \quad \text{and} \quad \beta_i = b_i \quad \text{for } 1 \leq i \leq N \tag{83}$$

satisfies (80), (81), and $\bar{\mathcal{I}} = \mathcal{I}$ trivially. Hence, a collection of intervals $\bar{\mathcal{I}}$ satisfying (80) and (81) exists; it remains to show that the choice (83) is unique.

We first show that, for any $a_i \in \mathcal{L}$, there is exactly one interval in $\bar{\mathcal{I}}$ that has a_i as a lower bound. We will show this by contradiction.

- Assume there is more than one interval with a_i as a lower bound; that is, there are $[a_i, b_j)$, $[a_i, b_\ell) \in \bar{\mathcal{I}}$ $[(a_i, b_j), (a_i, b_\ell) \in \bar{\mathcal{I}}]$ with $b_j, b_\ell \in \mathcal{U}$ and $b_j > a_i$, $b_\ell > a_i$ (otherwise the intervals would be empty, which contradicts with (81)). But then, $[a_i, b_j) \cap [a_i, b_\ell) = [a_i, \min(b_j, b_\ell)) \neq \emptyset$ $[(a_i, b_j) \cap (a_i, b_\ell) = (a_i, \min(b_j, b_\ell)) \neq \emptyset]$, which contradicts with (81).
- Assume there is no interval in $\bar{\mathcal{I}}$ that has a_i as a lower bound. Then, there can only be $N-1$ intervals in total, since it follows from the previous discussion that each of the remaining $a_j \in \mathcal{L} \setminus \{a_i\}$ can be chosen at most once as a lower bound. This contradicts with (80) (the collection $\bar{\mathcal{I}}$ having N elements).

Analogously, it can be shown that, for any $b_i \in \mathcal{U}$, there is exactly one interval in $\bar{\mathcal{I}}$ that has b_i as an upper bound. The detailed proof is omitted.

Now, take $\alpha_i = a_i$ for any $i \in \{1, \ldots, N\}$. From the discussion above, it follows that there is an interval $[\alpha_i, \beta_i) \in \bar{\mathcal{I}}$ $[(\alpha_i, \beta_i) \in \bar{\mathcal{I}}]$, $\beta_i \in \mathcal{U}$. We prove by contradiction that this implies $\beta_i = b_i$, and, hence, that the choice (83) is unique.

Assume $\beta_i = b_j$, $b_j \neq b_i$. Then, from the above discussion, there exists also an interval $[a_\ell, b_i) \in \bar{\mathcal{I}}$ $[(a_\ell, b_i) \in \bar{\mathcal{I}}]$, $a_\ell \in \mathcal{L}$. For $[a_i, b_j)$ $[(a_i, b_j)]$ to be nonempty, it follows that $a_i < b_j$, which implies by (82) that $b_i \leq b_j$. Similarly, for $[a_\ell, b_i)$ $[(a_\ell, b_i)]$ to be nonempty, $a_\ell < b_i$ implying $a_\ell \leq a_i$. But then, $[a_i, b_j) \cap [a_\ell, b_i) = [a_i, b_i) \neq \emptyset$ $[(a_i, b_j) \cap (a_\ell, b_i) = (a_i, b_i) \neq \emptyset]$, which contradicts (81). ☐

Corollary 3 Let $\mathcal{I}_1$, $\mathcal{I}_2$ be two collections of nonempty and mutually disjoint intervals. Let $\mathcal{L}_1$ and $\mathcal{U}_1$ be the sets of lower and upper bounds, respectively, of $\mathcal{I}_1$; and let $\mathcal{L}_2$ and $\mathcal{U}_2$ be the sets of lower and upper bounds, respectively, of $\mathcal{I}_2$. If $\mathcal{L}_1 = \mathcal{L}_2$ and $\mathcal{U}_1 = \mathcal{U}_2$, then $\mathcal{I}_1 = \mathcal{I}_2$. ■

Proof. Follows directly from Lemma 3. ☐

Proof of Lemma 1. (i), (ii): Statements (i) and (ii) are treated simultaneously ((ii) in brackets where required).

By Proposition 3, the intervals

$$\mathcal{I} = \{I_1, I_2, \ldots, I_N\} = \{[p_1, d_{\Pi(1)}), [d_{\Pi(1)}, d_{\Pi(2)}), \ldots, [d_{\Pi(N-1)}, p_2)\}$$

are mutually disjoint and nonempty. Therefore, also the intervals

$$\begin{aligned}\mathcal{I}_{\text{int}} &= \{\text{int}(I_1), \text{int}(I_2), \ldots, \text{int}(I_N)\} \\ &= \{(p_1, d_{\Pi(1)}),\ (d_{\Pi(1)}, d_{\Pi(2)}),\ \ldots,\ (d_{\Pi(N-1)}, p_2)\}\end{aligned}$$

are mutually disjoint and nonempty. Hence, by Lemma 3, $\mathcal{I}$ $[\mathcal{I}_{\text{int}}]$ is uniquely represented by the sets of lower and upper bounds

$$\mathcal{L} = \big\{p_1, d_{\Pi(1)}, \ldots, d_{\Pi(N-1)}\big\} = \big\{p_1, d_1, \ldots, d_{N-1}\big\}, \tag{84}$$

$$\mathcal{U} = \big\{d_{\Pi(1)}, \ldots, d_{\Pi(N-1)}, p_2\big\} = \big\{p_2, d_1, \ldots, d_{N-1}\big\} \tag{85}$$

and the conditions (80) and (81) (note that the sets $\mathcal{L}$ and $\mathcal{U}$ are the same for $\mathcal{I}$ and $\mathcal{I}_{\text{int}}$, but (80) is different).

Define the collection of images of h on $\mathcal{I}$ $[\mathcal{I}_{\text{int}}]$ as

$$\begin{aligned}\mathcal{I}_h &:= \big\{h\big([p_1, d_{\Pi(1)})\big), h\big([d_{\Pi(1)}, d_{\Pi(2)})\big), \ldots, h\big([d_{\Pi(N-1)}, p_2)\big)\big\} \\ [\mathcal{I}_{\text{int},h} &= \big\{h\big((p_1, d_{\Pi(1)})\big), h\big((d_{\Pi(1)}, d_{\Pi(2)})\big), \ldots, h\big((d_{\Pi(N-1)}, p_2)\big)\big\}].\end{aligned}$$

Hence, by definition,

$$\mathcal{I} \xrightarrow{h} \mathcal{I}_h \qquad\qquad [\mathcal{I}_{\text{int}} \xrightarrow{h} \mathcal{I}_{\text{int},h}]. \tag{86}$$

Since, by Proposition 1, (i), h is strictly increasing on each $I_i = [a, b) \in \mathcal{I}$ $[I_i = (a, b) \in \mathcal{I}_{\text{int}}]$, it holds that $h([a, b)) = [h(a), \lim_{p\nearrow b} h(p))$ $[h((a, b)) = (h(a), \lim_{p\nearrow b} h(p))]$. Therefore, the sets of lower and upper bounds of $\mathcal{I}_h$ $[\mathcal{I}_{\text{int},h}]$ are given by

$$\begin{aligned}\mathcal{L}_h &:= \big\{h(a) \mid a \in \mathcal{L}\big\} = \big\{h(p_1), h(d_1), h(d_2), \ldots, h(d_{N-1})\big\} \\ &= \big\{h(p_1), p_1, d_1, \ldots, d_{N-2}\big\}, \qquad (87) \\ \mathcal{U}_h &:= \big\{\lim_{p\nearrow b} h(p) \mid b \in \mathcal{U}\big\} = \big\{h(p_2), \lim_{p\nearrow d_1} h(p), h(d_2), \ldots, h(d_{N-1})\big\} \\ &= \big\{h(p_2), p_2, d_1, \ldots, d_{N-2}\big\}, \qquad (88)\end{aligned}$$

where we used the facts that h is continuous from the right at all $a \in \mathcal{L}$ and continuous from the left at all $b \in \mathcal{U} \setminus \{d_1\}$; and that

$$h(d_1) = h(\bar{p} + \delta) = p_1 \qquad\qquad \text{(def. of } p_1\text{)}, \tag{89}$$

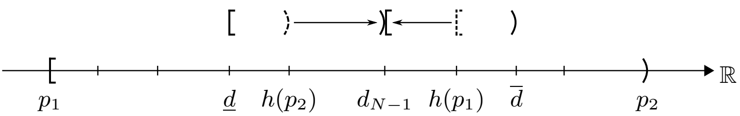


Figure 10. Illustration of the enlargement of the intervals $[\underline{d}, h(p_2))$ and $[h(p_1), \overline{d})$ to $[\underline{d}, d_{N-1})$ and $[d_{N-1}, \overline{d})$. The points unspecified are elements from $\{d_1, \ldots, d_{N-2}\}$. All intervals remain nonempty and mutually disjoint.

$$h(d_i) = d_{i-1},\ 2 \leq i \leq N-1 \qquad \text{(by Alg. 1)}, \tag{90}$$

$$\lim_{p \nearrow d_1} h(p) = a^2(\bar{p} + \delta) + 1 = p_2 \qquad \text{(def. of } p_2\text{)}. \tag{91}$$

Since h is injective (Proposition 1, (iii)), $h(I_1 \cap I_2) = h(I_1) \cap h(I_2)$ holds for any $I_1, I_2 \subseteq [p_1, p_2)$, [27]. From this, and the fact that the intervals $\mathcal{I}$ $[\mathcal{I}_{\text{int}}]$ are disjoint, it follows that the mapped intervals $\mathcal{I}_h$ $[\mathcal{I}_{\text{int},h}]$ are also disjoint. Furthermore, since h is not constant on any interval $I \in \mathcal{I}$ (it is strictly increasing by Proposition 1, (i)), the intervals $\mathcal{I}_h$ $[\mathcal{I}_{\text{int},h}]$ are all nonempty. Hence, by Lemma 3, $\mathcal{I}_h$ $[\mathcal{I}_{\text{int},h}]$ is uniquely represented by $\mathcal{L}_h$ and $\mathcal{U}_h$. Notice that $\mathcal{L}_h$ and $\mathcal{U}_h$ have the same elements as $\mathcal{L}$ and $\mathcal{U}$ except for $h(p_1)$ and $h(p_2)$ in $\mathcal{L}_h$ and $\mathcal{U}_h$, and d_{N-1} in $\mathcal{L}$ and $\mathcal{U}$. We show next that the intervals $\mathcal{I}_h$ $[\mathcal{I}_{\text{int},h}]$ are contained in the intervals of $\mathcal{I}$ $[\mathcal{I}_{\text{int}}]$.

To see this, notice first that the elements of $\mathcal{L}_h \cup \mathcal{U}_h \cup \mathcal{L} \cup \mathcal{U} = \{p_1, p_2, h(p_1), h(p_2), d_1, \ldots, d_{N-1}\}$ have the following order relation:

$$p_1 \leq \underbrace{\cdots\cdots}_{\text{other } d_i\text{'s}} < h(p_2) \leq d_{N-1} < h(p_1) < \underbrace{\cdots\cdots}_{\text{other } d_i\text{'s}} < p_2, \tag{92}$$

because

$$\begin{aligned} p_1 &< h(p_2) && \text{((34) and Prop. 1, (i))}, \\ h(p_1) &< p_2 && \text{(Prop. 1, (iv))}, \\ h(p_2) &\leq d_{N-1} < h(p_1) && \text{(Prop. 2, (ii))}, \\ d_i &\in [p_1, h(p_2)) \cup (h(p_1), p_2),\ \forall i \in \{1, \ldots, N-2\} && \text{(Prop. 2, (ii))}. \end{aligned}$$

Therefore, the upper bound of $[*, h(p_2)) \in \mathcal{I}_h$ $[(*, h(p_2)) \in \mathcal{I}_{\text{int},h}]$ can be changed to d_{N-1}, and the lower bound of $[h(p_1), *) \in \mathcal{I}_h$ $[(h(p_1), *) \in \mathcal{I}_{\text{int},h}]$ to d_{N-1}, without affecting the mutual disjointness and the nonemptiness of the intervals. This modification of the intervals is illustrated in Fig. 10, and we make it formal next.

Let $\underline{d}$ be the lower bound of $[*, h(p_2)) \in \mathcal{I}_h$ $[(*, h(p_2)) \in \mathcal{I}_{\text{int},h}]$, and let $\overline{d}$ be the upper bound of $[h(p_1), *) \in \mathcal{I}_h$ $[(h(p_1), *) \in \mathcal{I}_{\text{int},h}]$. Then, define

$$\tilde{\mathcal{I}}_h := \big\{ I \in \mathcal{I}_h \mid I \neq [\underline{d}, h(p_2)) \text{ and } I \neq [h(p_1), \overline{d}) \big\} \cup \big\{ [\underline{d}, d_{N-1}),\ [d_{N-1}, \overline{d}) \big\}, \tag{93}$$

that is, $\tilde{\mathcal{I}}_h$ has the same elements as $\mathcal{I}_h$ except for the replacements $[\underline{d}, h(p_2)) \to [\underline{d}, d_{N-1})$ and $[h(p_1), \overline{d}) \to [d_{N-1}, \overline{d})$. Similarly, define

$$\begin{aligned} \tilde{\mathcal{I}}_{\text{int},h} := \big\{ I \in \mathcal{I}_{\text{int},h} \mid I \neq (\underline{d}, h(p_2)) \text{ and } I \neq (h(p_1), \overline{d}) \big\} \\ \cup \big\{ (\underline{d}, d_{N-1}),\ (d_{N-1}, \overline{d}) \big\}. \end{aligned} \tag{94}$$

Since, from (92), $[\underline{d}, h(p_2)) \subseteq [\underline{d}, d_{N-1})$ $[(\underline{d}, h(p_2)) \subseteq (\underline{d}, d_{N-1})]$ and $[h(p_1), \overline{d}) \subseteq [d_{N-1}, \overline{d})$ $[(h(p_1), \overline{d}) \subseteq (d_{N-1}, \overline{d})]$, it follows from (86) that

$$\mathcal{I} \xrightarrow{h} \tilde{\mathcal{I}}_h \qquad\qquad [\mathcal{I}_{\text{int}} \xrightarrow{h} \tilde{\mathcal{I}}_{\text{int},h}]. \tag{95}$$

The lower and upper bounds of $\tilde{\mathcal{I}}_h$ $(\tilde{\mathcal{I}}_{\text{int},h})$ are given by

$$\tilde{\mathcal{L}}_h := \big\{ d_{N-1}, p_1, d_1, \dots, d_{N-2} \big\}, \tag{96}$$

$$\tilde{\mathcal{U}}_h := \big\{ d_{N-1}, p_2, d_1, \dots, d_{N-2} \big\}. \tag{97}$$

Since the intervals $\tilde{\mathcal{I}}_h$ $[\tilde{\mathcal{I}}_{\text{int},h}]$ are nonempty and mutually disjoint, and $\tilde{\mathcal{L}}_h = \mathcal{L}$ and $\tilde{\mathcal{U}}_h = \mathcal{U}$, it follows from Corollary 3 that $\tilde{\mathcal{I}}_h = \mathcal{I}$ $[\tilde{\mathcal{I}}_{\text{int},h} = \mathcal{I}_{\text{int}}]$. Using this result, statements (i) and (ii) are given by (95).

(iii): First, notice that $1 \leq \overline{i} \leq N-1$ and

$$h(I_{\overline{i}}) \underset{(49)}{=} h([d_{\overline{i}}, p_2)) \underset{(89),(90)}{=} \begin{cases} [d_{\overline{i}-1}, h(p_2)) & \text{if } \overline{i} > 1 \\ [p_1, h(p_2)) & \text{if } \overline{i} = 1. \end{cases} \tag{98}$$

Since $h(I_{\overline{i}}) \in \mathcal{I}_h$, it follows that

$$\underline{d} = \begin{cases} d_{\overline{i}-1} & \text{if } \overline{i} > 1 \\ p_1 & \text{if } \overline{i} = 1 \end{cases} \tag{99}$$

($\underline{d}$ has been defined above as the lower bound of the (unique) interval in $\mathcal{I}_h$ that has $h(p_2)$ as an upper bound). Notice that in $\mathcal{I}$, there is exactly one interval with $d_{\overline{i}-1}$ as lower bound (for $\overline{i} > 1$) and exactly one interval with

p_1 as lower bound. Therefore, it follows from $[\underline{d}, d_{N-1}) \in \tilde{\mathcal{I}}_h = \mathcal{I}$ (see (93)), and (48)–(50) that

$$[\underline{d}, d_{N-1}) = \begin{cases} [d_{\bar{i}-1}, d_{N-1}) & \text{if } \bar{i} > 1 \\ [p_1, d_{N-1}) & \text{if } \bar{i} = 1 \end{cases} = \begin{cases} I_{\bar{i}-1} & \text{if } \bar{i} > 1 \\ I_N & \text{if } \bar{i} = 1 \end{cases} = I_{\bar{i} -_N 1}. \tag{100}$$

(iv): Notice that $1 \leq \underline{i} \leq N-1$ and

$$h(\text{int}(I_N)) \underset{(50)}{=} h((p_1, d_{\underline{i}})) \underset{(90),(91)}{=} \begin{cases} (h(p_1), d_{\underline{i}-1}) & \text{if } \underline{i} > 1 \\ (h(p_1), p_2) & \text{if } \underline{i} = 1\ . \end{cases} \tag{101}$$

Since $h(\text{int}(I_N)) \in \mathcal{I}_{\text{int},h}$, it follows that

$$\overline{d} = \begin{cases} d_{\underline{i}-1} & \text{if } \underline{i} > 1 \\ p_2 & \text{if } \underline{i} = 1\ , \end{cases} \tag{102}$$

and, from $(d_{N-1}, \overline{d}) \in \tilde{\mathcal{I}}_{\text{int},h} = \mathcal{I}_{\text{int}}$ (see (94)) and (48)–(49),

$$(d_{N-1}, \overline{d}) = \begin{cases} (d_{N-1}, d_{\underline{i}-1}) & \text{if } \underline{i} > 1 \\ (d_{N-1}, p_2) & \text{if } \underline{i} = 1 \end{cases} = I_{N-1}. \tag{103}$$

□

B. Proof of Lemma 2

Take $p \in [p_1, \bar{p} + \delta)$. Let $k \in \mathbb{N}^+$ such that $p, h(p), \ldots, h^{k-1}(p) < \bar{p} + \delta$ (such a k exists since $p < \bar{p} + \delta$). Then, from (32),

$$h^k(p) = a^2\, h^{k-1}(p) + 1 > a^2\, h^{k-1}(p), \tag{104}$$

and, therefore,

$$h^k(p) > a^{2k}\, p. \tag{105}$$

Since $|a| > 1$, $\lim_{k\to\infty} a^{2k} p = \infty$. Hence, there exists an $m(p) \in \mathbb{N}^+$ such that (63) holds.

Now, we seek the largest possible integer $m(p)$ over all $p \in [p_1, \bar{p} + \delta)$, for which (63) holds. Since $h^k(p_1) \leq h^k(p)$ for all $p \in [p_1, \bar{p} + \delta)$ and $k \leq m(p)$, the greatest $m(p)$ such that (63) holds is $\bar{N} \in \mathbb{N}^+$ defined by

$$p_1, h(p_1), \ldots, h^{\bar{N}-1}(p_1) < \bar{p} + \delta \quad \text{and} \quad h^{\bar{N}}(p_1) \geq \bar{p} + \delta. \tag{106}$$

Hence, $\bar{N}$ is independent of p, and $m(p) \leq \bar{N}$. From (105) and (106), it follows that

$$a^{2(\bar{N}-1)}p_1 < h^{\bar{N}-1}(p_1) < \bar{p} + \delta \underset{(p_1>0,\, a^2>0)}{\Rightarrow} a^{2\bar{N}} < a^2 \frac{\bar{p}+\delta}{p_1}.$$

References

[1] K.-D. Kim and P. Kumar, “Cyber-physical systems: A perspective at the centennial,” *Proceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1287–1308, May 2012.

[2] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, “A survey of recent results in networked control systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[4] S. Trimpe and R. D’Andrea, “An experimental demonstration of a distributed and event-based state estimation algorithm,” in *Proc. of the 18th IFAC World Congress*, Milano, Italy, Aug. 2011, pp. 8811–8818.

[5] S. Trimpe, “Event-based state estimation with switching static-gain observers,” in *Proc. of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, Santa Barbara, CA, USA, Sep. 2012, pp. 91–96.

[6] S. Trimpe and R. D’Andrea, “The Balancing Cube: A dynamic sculpture as test bed for distributed estimation and control,” *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 48–75, Dec. 2012.

[7] ——, “Reduced communication state estimation for control of an unstable networked control system,” in *Proc. of the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, 2011, pp. 2361–2368.

[8] ——, “Event-based state estimation with variance-based triggering,” in *Proc. of the 51st IEEE Conference on Decision and Control*, Maui, HI, USA, Dec. 2012, pp. 6583–6590.

[9] M. Lemmon, “Event-triggered feedback in control, estimation, and optimization,” in *Networked Control Systems*, ser. Lecture Notes in

Control and Information Sciences, A. Bemporad, M. Heemels, and M. Johansson, Eds. Springer Berlin / Heidelberg, 2011, vol. 406, pp. 293–358.

[10] O. C. Imer and T. Basar, "Optimal estimation with limited measurements," in *Proc. of the 44th IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, Dec. 2005, pp. 1029–1034.

[11] L. Li, M. Lemmon, and X. Wang, "Event-triggered state estimation in vector linear processes," in *Proc. of the American Control Conference*, Baltimore, MD, USA, Jul. 2010, pp. 2138–2143.

[12] M. Rabi, G. V. Moustakides, and J. S. Baras, "Multiple sampling for estimation on a finite horizon," in *Proc. of the 45th IEEE Conference on Decision and Control*, San Diego, CA, USA, Dec. 2006, pp. 1351–1357.

[13] Y. Xu and J. P. Hespanha, "Estimation under uncontrolled and controlled communications in networked control systems," in *Proc. of the 44th IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, Dec. 2005, pp. 842–847.

[14] R. Cogill, S. Lall, and J. P. Hespanha, "A constant factor approximation algorithm for event-based sampling," in *Proc. of the American Control Conference*, New York, NY, USA, Jul. 2007, pp. 305–311.

[15] J. Sijs and M. Lazar, "On event based state estimation," in *Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, R. Majumdar and P. Tabuada, Eds. Springer Berlin / Heidelberg, 2009, vol. 5469, pp. 336–350.

[16] J. Sijs, M. Lazar, and W. P. M. H. Heemels, "On integration of event-based estimation and robust MPC in a feedback loop," in *Proc. of the 13th ACM international conference on hybrid systems: computation and control*, New York, NY, USA, 2010, pp. 31–40.

[17] J. Weimer, J. Araujo, and K. H. Johansson, "Distributed event-triggered estimation in networked systems," in *Proc. of the 4th IFAC Conference on Analysis and Design of Hybrid Systems*, Eindhoven, Netherlands, Jun. 2012, pp. 178–185.

[18] J. K. Yook, D. M. Tilbury, and N. R. Soparkar, "Trading computation for bandwidth: reducing communication in distributed control systems using state estimators," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 4, pp. 503–518, Jul. 2002.

[19] Y. Xu and J. P. Hespanha, "Optimal communication logics in networked control systems," in *Proc. of the 43rd IEEE Conference on Decision and Control*, Atlantis, Bahamas, Dec. 2004, pp. 3527–3532.

[20] H. Sandberg, M. Rabi, M. Skoglund, and K. H. Johansson, "Estimation over heterogeneous sensor networks," in *Proc. of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, Dec. 2008, pp. 4898–4903.

[21] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* Mineola, New York: Dover Publications, 2005.

[22] S. Bittanti, P. Colaneri, and G. De Nicolao, "The difference periodic Riccati equation for the periodic prediction problem," *IEEE Transactions on Automatic Control*, vol. 33, no. 8, pp. 706–712, Aug. 1988.

[23] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, "Kalman filtering with intermittent observations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1453–1464, Sep. 2004.

[24] S. Elaydi, *An Introduction to Difference Equations*, 3rd ed., ser. Undergraduate Texts in Mathematics. Springer New York, 2005.

[25] W. G. Kelley and A. C. Peterson, *Difference equations: an introduction with applications.* San Diego, USA: Academic Press, Inc., 1991.

[26] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. McGraw-Hill, 1976.

[27] E. T. Copson, *Metric spaces.* Cambridge University Press, 1968.

Paper II

An Experimental Demonstration of a Distributed and Event-Based State Estimation Algorithm

Sebastian Trimpe · Raffaello D'Andrea

Abstract

A distributed state estimation algorithm that makes use of model-based predictions to reduce communication requirements in a networked control architecture is tested on an unstable system. A cube balancing on one of its edges serves as the test platform, and six rotating bodies on the cube's inner faces constitute the agents in the control network. Each agent carries a computational unit, which runs estimation and control algorithms, and is associated with local sensors and an actuator. Measurement data is shared among the agents over a broadcast network. Each agent maintains two estimates of the system state: the first reflecting the common knowledge in the network, and the second additionally including all local sensor information. An agent's sensor measurement is only broadcast if it deviates from the common estimate of that measurement by more than a specified threshold. Experimental results show that the number of communicated measurements required for stabilizing the system can be significantly reduced with this event-based communication protocol.

Published in *Proc. of the 18th IFAC World Congress*, 2011, pp. 8811-8818.
DOI: 10.3182/20110828-6-IT-1002.00564.

©2011 IFAC. NOTICE: This is the author's version of a work that was accepted for publication in *Proc. of the 18th IFAC World Congress, 2011.* Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in *Proc. of the 18th IFAC World Congress, 2011*, [Volume 18, Part 1], DOI: 10.3182/20110828-6-IT-1002.00564.

1. Introduction

In most traditional control systems, a communication medium is dedicated exclusively to the exchange of data between the plant and the controller. However, recent developments in the area of networked control systems (see [1] for an overview) suggest a different view and regard communication as a shared resource. Multiple agents share a network and may exchange data of various content, in addition to sensor data and control commands of typical feedback control systems.

Viewing communication as a resource may also be useful from an architectural point of view. On a system level, one may distinguish between time-critical tasks (e.g. real-time feedback control) and non-time-critical tasks (e.g. adaptation). Clearly, sufficient resources (such as communication bandwidth) must be allocated to the critical tasks in order to maintain the operation of the system in times of need. On average, however, the full capacity of the resource is rarely exploited. In such cases, unused resources may be reallocated to non-critical tasks, resulting in improved system performance in the long run.

From control theory it is known that sensor feedback is critical, for ex-

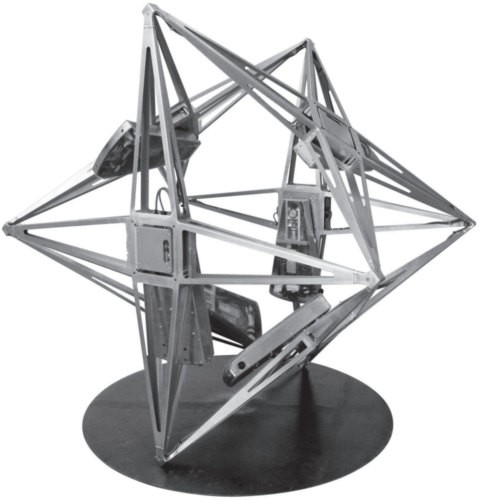


Figure 1. The Balancing Cube is an example of a networked control system: six rotating modules, each having sensors, actuation, and computational unit, share information over a network to balance the cube on its edge.

ample, for stabilization of unstable systems, disturbance rejection, and when dealing with uncertainties. In other situations, a system may operate satisfactorily open-loop, at least for limited time periods. In event-based control, sensor feedback caused by certain *events* (such as a measurement exceeding a threshold) is used as an alternative to time-triggered sending (see [2] and references therein). Lunze and Lehmann [3] use the deviation of the actual state from the trajectory of the closed-loop system with continuous sensor feedback to decide if an event should be triggered. Hence, the feedback loop is closed only when required.

At a high level, a similar idea can be applied to reduce the communication in networked control systems with multiple agents, each associated with local sensors and actuators. The agents may use a process model to make predictions about all other agents' sensor measurements. Using the same model, each agent also makes predictions about its own state or measurements. If the prediction of its local data deviates significantly from the true data, the local data is broadcast to all other agents, which can then update their estimates. Communication schemes like these (where, in order to reduce network traffic, sensor data is not sent at every time step) are referred to as *controlled communication*, cf. [1]. The distributed estimation algorithm of this paper falls into this class of algorithms, which have previously been proposed in [4]–[6].

The implementation of the method for distributed state estimation in this work makes use of two independent discrete-time Kalman filters: one uses the measurement data that has been broadcast and is thus globally known and identical for all agents, and the other additionally exploits all local measurements. Each agent bases its decision to broadcast its local measurements on a comparison of the actual measurement to the common estimate of the system. Using this communication protocol, the Kalman filters receive a varying number of measurements (including none) at every time step.

A related problem for state estimation in networked control systems is considered in [7]. The authors analyze Kalman filtering for the scenario where non-delivery of measurement data results from packet drops rather than a decision made by some agent on the network. They model the arrival of a measurement as a binary random variable that is independent of the process data (states and measurements). In contrast, the delivery of a measurement in the presented approach is purposefully chosen to depend on past measurements.

The distributed estimators exploit the prediction capabilities of the Kalman filter to compensate for the deliberate non-communication of measurements. Zhang et al. [8] use a similar idea for a different purpose in that they

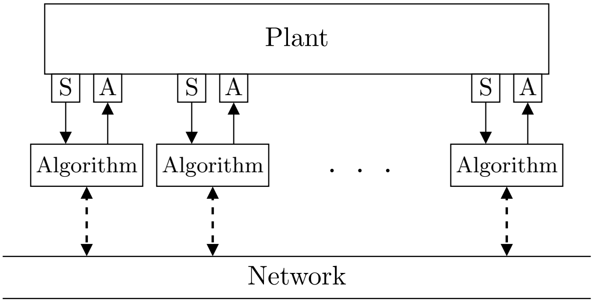


Figure 2. The considered networked control architecture: the blocks A and S denote (possibly heterogeneous) actuator and sensor units; the Algorithm block includes estimation and control algorithms as well as the communication logic.

exploit model-based predictions to compensate for network-induced delays.

The Balancing Cube, a 1.2 m cube that can balance autonomously on any of its edges or corners[1] (see Fig. 1), serves as the platform for testing the distributed estimation algorithm presented in this paper. Six rotating *modules* located on each inner face of the cube enable the cube to balance. The modules are self-contained units that carry sensors, an actuator, a battery, and a computer. The modules share information over a communication network. Experiments demonstrate that the distributed estimation algorithm combined with the event-based communication protocol allows a significant reduction of the average number of communicated sensor measurements while keeping the cube balanced.

This paper is organized as follows: The distributed estimation algorithm with event-based communication is presented in Sec. 2. An overview of the physical test bed – the Balancing Cube – is given in Sec. 3, where the experimental results are also presented. The paper concludes with remarks in Sec. 4.

2. State Estimation Algorithm

Figure 2 shows the network control architecture considered in this work. The block *Algorithm* represents a computational unit that runs the estimation and control algorithm and also handles the communication with the other units. Each unit is associated with local sensors (S) and actuators (A). Hence, the algorithm determines the appropriate commands for its actuator

---

[1] Videos may be found online at `http://www.cube.ethz.ch`.

based on local sensor data, as well as data possibly received from the network. The combination of computational unit, sensor, and actuator will be referred to as *agent* for the remainder of this paper. The total number of agents is denoted by N.

The data unit that is sent over the network is a scalar sensor measurement. We assume that data is broadcast over the network; that is, if an agent sends a measurement, then all other agents receive this measurement. Furthermore, we take an abstract view of the network and assume the communication to be ideal; that is, we assume that the communication of measurements is instantaneous and no data is lost. This may be partly ensured by low level communication protocols. It is recognized however that sending only scalar measurements may not always be the best approach. For network protocols that require a minimum data length, it may be more efficient to send the full measurement vector. The method presented in the following can readily be adapted to this scenario.

We consider the common setup of a linear stochastic process given by a discrete-time state space model,

$$x(k) = A\,x(k-1) + B\,u(k-1) + v(k-1) \tag{1}$$

$$y(k) = C\,x(k) + w(k), \tag{2}$$

where k is the time index; $x(k),\ v(k) \in \mathbb{R}^n$; $u(k) \in \mathbb{R}^m$; $y(k),\ w(k) \in \mathbb{R}^p$; and all matrices are of corresponding dimensions. The process and measurement noise, $v(k)$ and $w(k)$, are random variables with $v(k) \sim \mathcal{N}(0, Q)$ and $w(k) \sim \mathcal{N}(0, R)$, where $\mathcal{N}(m, V)$ denotes a normally distributed random variable with mean m and covariance matrix V. All noise sources are assumed temporally independent; and R is assumed diagonal. The initial state $x(0)$ is also assumed to be normally distributed with known mean x_0 and covariance P_0.

In the proposed setup, each agent maintains an estimate of the full system state $x(k)$, which is input to a state feedback controller. Hence, the question of effective data communication rests with the state estimator: the objective is to maintain an estimate of the full system state $x(k)$ on each agent with a limited exchange of data between the agents. Two key issues will be addressed: (1) how to make use of the varying sets of measurements that arrive at an agent (*receiver algorithm*), and (2) how to decide if local sensor data should be broadcast over the network (*sender algorithm*).

Before addressing the distributed estimation problem with these two points in Sec. 2.2 and 2.3, the standard results for centralized state estimation (i.e. with access to the full measurement vector $y(k)$) are presented in Sec. 2.1. The centralized case will serve as a baseline for the distributed estimation method.

2.1 Centralized estimation

It is well known that the *optimal* state estimator of the system given by (1) and (2) is the time-varying Kalman filter. It is optimal in the sense that it keeps track of the entire conditional probability density of the system state $x(k)$ conditioned on all measurements and control inputs up to time k (cf. [9]). The Kalman filter can be given in the following recursive form,

$$\begin{aligned} \hat{x}(k|k-1) &= A\hat{x}(k-1|k-1) + Bu(k-1) && (3) \\ P(k|k-1) &= AP(k-1|k-1)A^T + Q && (4) \end{aligned}$$

$$\begin{aligned} K(k) &= P(k|k-1)\, C^T \big(CP(k|k-1)C^T + R\big)^{-1} && (5) \\ \hat{x}(k|k) &= \hat{x}(k|k-1) + K(k)\big(y(k) - C\hat{x}(k|k-1)\big) && (6) \\ P(k|k) &= \big(I - K(k)C\big)P(k|k-1) && (7) \end{aligned}$$

where $\hat{x}(k|k-1)$ denotes the expected value of the state $x(k)$ given all measurements and inputs up to time $k-1$, $\hat{x}(k|k)$ is the expected value of $x(k)$ given all data up to time k, and $P(k|k–1)$ and $P(k|k)$, respectively, are their covariance matrices. The filter is initialized by $\hat{x}(0|0) = x_0$ and $P(0|0) = P_0$. There are many different variants of the Kalman filter; which implementation of the Kalman filter is used does not matter for the method presented below.

For simplicity, static state feedback is considered,

$$u(k-1) = F\,\hat{x}(k-1|k-1), \tag{8}$$

where F denotes the matrix feedback gain. The exposition of the proposed method can, however, be adapted to controllers with states. Each agent is responsible for a subset of the control input vector $u(k)$. To ease the exposition, an index denoting the elements of the vector is not used unless otherwise noted.

It is assumed that both the Kalman state observer (3)–(7) and the feedback controller (8) are designed such that the given control objective is satisfied.

In a network with sufficient communication bandwidth, a conceptually equivalent implementation of the centralized Kalman filter (3)–(7) on the distributed control network in Fig. 2 would be to communicate at every time step all measurement data to all agents. Each agent can then simply run a copy of the estimator (3)–(7) and the controller (8). With this design as a starting point, the objective is to develop a distributed and event-driven estimation scheme that utilizes less communication bandwidth on average, but that may revert to the centralized design if required.

2.2 Distributed estimation: receiver algorithm

The *receiver* of agent i denotes the algorithm that uses all information available at time k to compute an estimate of the system state $x(k)$. The counterpart in the event-driven estimation scheme, i.e. the *sender algorithm*, is derived in the next section. It should be noted, however, that both the receiver and sender algorithms run on each agent in parallel.

The following notation is used to distinguish the different types of measurements that are available to agent i at time k:

$\bar{y}_i(k) \in \mathbb{R}^{\bar{p}_i}$	local sensor data
$\tilde{y}_i(k) \in \mathbb{R}^{\tilde{p}_i(k)}$	data received over the network
$y_i(k) \in \mathbb{R}^{p_i(k)}$	all available data, $y_i(k) = (\tilde{y}_i(k), \bar{y}_i(k))$.

The dimension of $\tilde{y}_i(k)$ and hence $y_i(k)$ is time-varying because of the varying and a-priori unknown number of measurements received. In particular, $\tilde{p}_i(k) = 0$ in the case where no measurement is received at time k by agent i.

The elements of the vectors $\bar{y}_i(k)$, $\tilde{y}_i(k)$, and $y_i(k)$ are subsets of the elements of the full measurement vector $y(k)$ in (2). Analogous notation is used to denote the output matrices and the measurement noise,

$$\bar{y}_i(k) = \bar{C}_i\, x(k) + \bar{w}_i(k) \tag{9}$$

$$\tilde{y}_i(k) = \tilde{C}_i(k)\, x(k) + \tilde{w}_i(k) \tag{10}$$

$$y_i(k) = C_i(k)\, x(k) + w_i(k), \tag{11}$$

where $\bar{C}_i$, $\tilde{C}_i(k)$, $C_i(k)$ are of appropriate dimensions, and $\bar{w}_i(k) \sim \mathcal{N}(0, \bar{R}_i)$, $\tilde{w}_i(k) \sim \mathcal{N}(0, \tilde{R}_i(k))$, and $w_i(k) \sim \mathcal{N}(0, R_i(k))$. Note that the dimension of the matrices $\tilde{C}_i(k)$, $C_i(k)$, $\tilde{R}_i(k)$, and $R_i(k)$ are time-varying due to the varying dimension of the corresponding measurement vector. This includes the case where $\tilde{C}_i(k)$ and $\tilde{R}_i(k)$ have zero rows when no measurement is received. In order to avoid special treatment of this case, the Kalman filter equations below should be understood such that the measurement update step is skipped if no measurement is available.

A time-varying Kalman filter that determines an estimate $\hat{x}_i(k|k)$ of the system state $x(k)$ on agent i, taking into account all available measurements up to and including time k, is designed analogously to the filter (3)–(7) for the centralized case:

$$\hat{x}_i(k|k-1) = A\hat{x}_i(k-1|k-1) + B\hat{u}_i(k-1) \tag{12}$$

$$P_i(k|k-1) = AP_i(k-1|k-1)A^T + Q \tag{13}$$

$$\begin{aligned} K_i(k) &= P_i(k|k-1)\, C_i^T(k) \\ &\quad \cdot \big(C_i(k)P_i(k|k-1)C_i^T(k) + R_i(k)\big)^{-1} \end{aligned} \tag{14}$$

$$\hat{x}_i(k|k) = \hat{x}_i(k|k-1) + K_i(k)\big(y_i(k) - C_i(k)\hat{x}_i(k|k-1)\big) \tag{15}$$

$$P_i(k|k) = \big(I - K_i(k)C_i(k)\big)P_i(k|k-1). \tag{16}$$

To avoid misunderstanding, it should be reemphasized here that the index i in $\hat{x}_i(k|k)$ does not denote the i-th element of $\hat{x}(k|k)$, but agent i's state estimate. Similarly, $\hat{u}_i$, given by

$$\hat{u}_i(k-1) = F\,\hat{x}_i(k-1|k-1), \tag{17}$$

is agent i's estimate of what the control input to the whole system should be; in particular, agent i uses the components corresponding to its actuator as actual control commands.

We note that the distributed estimator (12)–(16) is the same as its centralized counterpart (3)–(7) if all sensor data is communicated.

2.3 Distributed estimation: sender algorithm

In the previous section, we assumed that each agent receives a varying number of sensor measurements from the other agents in the network. This section addresses the sending decision: agent i's sender algorithm determines if the local sensor data $\bar{y}_i(k)$ should be sent to all other agents on the network.

Following the key idea discussed in the introduction, agent i only sends its local measurements if it determines it is necessary; that is, if the other agents' expectation of the measurement is significantly different. One way for agent i to estimate the other agents knowledge is to construct another state estimate $\check{x}_i(k|k)$ that uses only measurements that have been broadcast to all agents. Hence, this estimator reflects the knowledge that is common to all agents. This requires that the local sensor data $\bar{y}_i(k)$ is only included in $\check{x}_i(k|k)$ if it is also broadcast to the network. The estimate $\check{x}_i(k|k)$ is again obtained from a Kalman filter,

$$\check{x}_i(k|k-1) = A\check{x}_i(k-1|k-1) + B\check{u}_i(k-1) \tag{18}$$

$$\check{P}_i(k|k-1) = A\check{P}_i(k-1|k-1)A^T + Q \tag{19}$$

$$\begin{aligned}\check{K}_i(k) &= \check{P}_i(k|k-1)\,\tilde{C}_i^T(k)\\ &\qquad\cdot\big(\tilde{C}_i(k)\check{P}_i(k|k-1)\tilde{C}_i^T(k)+\tilde{R}_i(k)\big)^{-1} \qquad (20)\\ \check{x}_i(k|k) &= \check{x}_i(k|k-1)+\check{K}_i(k)\big(\tilde{y}_i(k)-\tilde{C}_i(k)\check{x}_i(k|k-1)\big) \qquad (21)\\ \check{P}_i(k|k) &= \big(I-\check{K}_i(k)\tilde{C}_i(k)\big)\check{P}_i(k|k-1) \qquad (22)\end{aligned}$$

with the corresponding estimate of the control input

$$\check{u}_i(k-1) = F\,\check{x}_i(k-1|k-1). \qquad (23)$$

The estimate $\check{x}_i(k|k)$ ensures consistency in the network, since it is the same for all agents (whereas the estimate from (12)–(16) is generally different).

With the common estimate $\check{x}_i(k|k)$, agent i can now estimate what all other agents assume its measurement $\bar{y}_i(k)$ is: simply $\bar{C}_i\check{x}_i(k|k-1)$. This estimate can be used with some communication logic to decide if $\bar{y}_i(k)$ is broadcast. Here, a simple threshold logic, applied for each element l of the measurement vector individually, is used:

$$\text{send } \bar{y}_{i,l}(k) \;\Leftrightarrow\; |\bar{y}_{i,l}(k)-\bar{C}_{i,l}\,\check{x}_i(k|k-1)| \geq \delta_{i,l}, \qquad (24)$$

where $\bar{y}_{i,l}(k)$ denotes the l-th element of $\bar{y}_i(k)$, $\bar{C}_{i,l}$ the l-th row of $\bar{C}_i$, and $\delta_{i,l} \in [0,\infty)$ is a design parameter capturing the tolerated deviation. Because of its impact on the network communication, we hereafter refer to it as the *communication threshold*.

Obviously, the choice $\delta_{i,l} = 0$ means that the measurement $\bar{y}_{i,l}(k)$ is always sent; on the contrary, $\delta_{i,l} \to \infty$ corresponds to never sending the measurement. Again, if all sensor data is communicated, the distributed estimators (12)–(16) and (18)–(22) yield the same estimates as the centralized Kalman filter in (3)–(7). Thus, the performance of the optimal state estimation can be recovered by choosing all $\delta_{i,l}$ to zero. This turns out to be very handy for practical implementation and tuning of the algorithm.

We remark that other communication logics are possible: a condition involving a bound on $\check{P}_i(k|k-1)$ or a combination of a condition on $\check{x}_i(k|k-1)$ and on $\check{P}_i(k|k-1)$ may be used instead.

2.4 The complete algorithm

Both Kalman filters (12)–(16) and (18)–(22) run in parallel on each agent. Using the condition (24) and the estimate $\check{x}_i(k|k-1)$, agent i decides if its associated sensor data is sent. For the purpose of computing the control input for its actuator, the estimate $\hat{x}_i(k|k)$ is used, since it makes use of all information locally available.

A control loop step using the distributed estimation method of Sec. 2.2 and 2.3 is summarized in Algorithm 1.

Algorithm 1 Control step on agent i, executed every time step k.

```
apply component of $\hat{u}_i(k-1)$ to local actuator
acquire local measurement $\bar{y}_i(k)$
receive $\tilde{y}_i(k)$ from network (possibly empty)
compute estimate $\hat{x}_i(k|k)$ from (12)–(16), (17)
compute estimate $\check{x}_i(k|k)$ from (18)–(22), (23)
for $l = 1$ to $\bar{p}_i$
    if $|\bar{y}_{i,l}(k) - \bar{C}_{i,l}\,\check{x}_i(k|k-1)| \geq \delta_{i,l}$
        send $\bar{y}_{i,l}(k)$
    end if
end for
compute control $\hat{u}_i(k) = F\,\hat{x}_i(k|k)$
```

3. Application to the Balancing Cube

In this section, the effectiveness of the distributed estimation algorithm is demonstrated on the Balancing Cube, which represents an unstable system. Six agents are used to stabilize the system. A brief system description is provided in Sec. 3.1. Even though a complete treatment is beyond the scope of this paper, a brief explanation of the modeling technique and the control design is provided. Remarks on the implementation of the state estimation method are given in Sec. 3.2 and experimental results are presented in Sec. 3.3.

3.1 System description

The Balancing Cube is a multi-body system consisting of a rigid body in the shape of a cube and six rotating bodies (called *modules*) on the inner faces of the cube, see Fig. 3. Each of the six faces of the cube is made of an X-shaped aluminum structure (cf. Fig. 1 and 3); the edge length of the cube is 1.2 m. The total mass of the cube is 21.4 kg and the modules have a base mass of 3.7 kg. In the setup presented here, three of the modules carry extra weights adding up to a total mass of 5.8 kg.

Though the cube can balance on a corner (presented in [10]), for the purpose of this study it balances on its edge as shown in Fig. 1 and 3. In this configuration, the cube body has only one rotational degree of freedom (the rotational axis is the edge the cube is standing on), which results in a simpler dynamic model and eases the exposition presented below.

The modules are actuated by a DC motor and rotate relative to the cube structure. A drawing of a module with its functional parts is shown in Fig. 4. When the modules rotate, they exert reactional and gravitational forces (by shifting the center of mass) on the cube structure. Each module carries a single-board computer (SBC) that receives data from the sensors and sends commands to the motor. The computers themselves are connected over a Controller Area Network (CAN), whose wires run through a slip ring and along the cube structure. The low level CAN protocols allow each module to broadcast its local measurements to all other modules on the network. All components on a module are powered by a battery, which allows for a normal balancing operation of around 4 hours.

The local sensors associated with each module are an absolute encoder and an inertial measurement unit (IMU). The absolute encoder measures the module's angle relative to its mounting. The IMU is rigidly mounted to the cube structure (also connected to the SBC through the slip-ring). It has a tri-axis accelerometer and tri-axis rate gyro. For the demonstration of the estimation algorithm of Sec. 2, only the absolute encoder and rate gyro measurements were used. In fact, only one axis of the rate gyro is relevant, namely the axis parallel to the axis of rotation. Hence, each module has access to two local measurements ($\bar{p}_i = 2$).

Since each computational unit is connected locally to sensors and the actuator, and the computers share data over a network, the cube architecture falls into the class of systems considered in Sec. 2. The self-contained

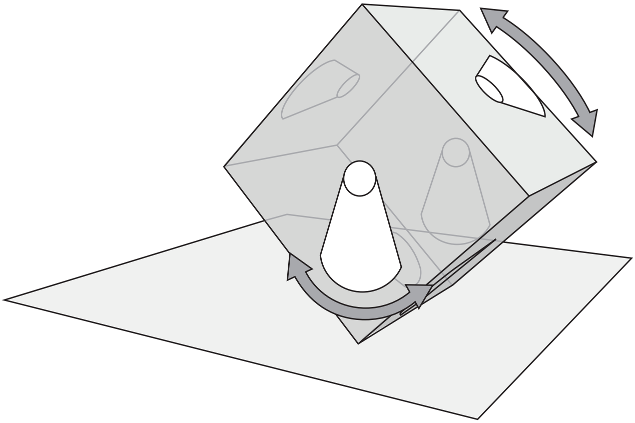


Figure 3. Rendering of the Balancing Cube, shown in the same orientation as in Fig. 1. The cube has six rotating modules, one on each face.

modules on the cube play the role of the agents.

In order to stabilize the system about an equilibrium, a cascaded control architecture is applied. On each motor, a local velocity feedback loop operates at 1 kHz. This inner loop tracks module angular velocity commands at a faster rate than the natural dynamics of the cube. With this architecture, nonlinear effects such as friction and backlash in the actuation mechanism are mitigated from the perspective of an outer loop (the full state feedback controller in the form of (8)) which stabilizes the system. The outer control loop is implemented at a frequency of 60 Hz.

Linear model and feedback controller. To obtain a model of the Balancing Cube for the design of the state feedback controller and the state estimator, the time-scale separation technique described in [11] is applied, where the inner velocity feedback loops are considered as (ideal) high gain feedback loops. The resulting linear discrete-time model with sampling frequency of 60 Hz reads

$$\begin{bmatrix} x_\mathrm{s}(k) \\ x_\mathrm{f}(k) \end{bmatrix} = \begin{bmatrix} A_\mathrm{ss} & A_\mathrm{sf} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_\mathrm{s}(k-1) \\ x_\mathrm{f}(k-1) \end{bmatrix} + \begin{bmatrix} B_\mathrm{s} \\ I \end{bmatrix} u(k-1) \tag{25}$$

$$y(k) = \begin{bmatrix} C_\mathrm{s} & 0 \end{bmatrix} \begin{bmatrix} x_\mathrm{s}(k) \\ x_\mathrm{f}(k) \end{bmatrix}. \tag{26}$$

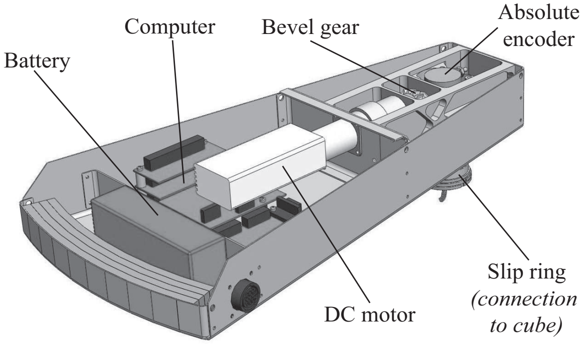


Figure 4. Rendering of a module with its functional parts. Not shown in this drawing is the inertial measurement unit that sits in the part of the module that is attached rigidly to the cube body.

Table 1. The states and utilized measurements of the Balancing Cube. (Note that in this table the indices denote elements of a vector; for example, y_1 is the first component of y.)

state	physical meaning
$x_{\mathrm{s},1}$	angle module 1
$x_{\mathrm{s},2}$	angle module 2
$x_{\mathrm{s},3}$	angle module 3
$x_{\mathrm{s},4}$	angle module 4
$x_{\mathrm{s},5}$	angle module 5
$x_{\mathrm{s},6}$	angle module 6
$x_{\mathrm{s},7}$	cube angle
$x_{\mathrm{s},8}$	cube ang. vel.
$x_{\mathrm{f},1}$	ang. vel. module 1
$x_{\mathrm{f},2}$	ang. vel. module 2
$x_{\mathrm{f},3}$	ang. vel. module 3
$x_{\mathrm{f},4}$	ang. vel. module 4
$x_{\mathrm{f},5}$	ang. vel. module 5
$x_{\mathrm{f},6}$	ang. vel. module 6

meas.	sensor
y_1	encoder module 1
y_2	rate gyro module 1
y_3	encoder module 2
y_4	rate gyro module 2
y_5	encoder module 3
y_6	rate gyro module 3
y_7	encoder module 4
y_8	rate gyro module 4
y_9	encoder module 5
y_{10}	rate gyro module 5
y_{11}	encoder module 6
y_{12}	rate gyro module 6

The states[2] $x_{\mathrm{f}}(k)$ are the angular velocities of the modules, $u(k)$ their reference values, and $x_{\mathrm{s}}(k)$ combines all other states. Notice that the approximation of the inner velocity loops as high gain feedback results in tracking of the reference inputs in (25) in one time step, i.e. $x_{\mathrm{f}}(k) = u(k-1)$. It has been verified experimentally that this is a valid approximation. All states and measurements of the linear model are listed in Table 1. The matrices of the state space model may be found in Appendix A.

Each module has access to an encoder and a gyro measurement. For module i, $\bar{y}_{i,1}(k)$ denotes the encoder and $\bar{y}_{i,2}(k)$ the gyro measurement at time k.

The state equation (25) is used to design a stabilizing LQR feedback controller; the feedback law is

$$u(k-1) = \begin{bmatrix} F_{\mathrm{s}} & F_{\mathrm{f}} \end{bmatrix} \begin{bmatrix} x_{\mathrm{s}}(k-1) \\ x_{\mathrm{f}}(k-1) \end{bmatrix}. \tag{27}$$

The feedback gain matrices F_{s} and F_{f} may also be found in Appendix A.

---

[2] The index f corresponds to “fast” and s to “slow.”

For state estimation, the following reduced state-space representation is used, which follows from (25), (26) with added process and measurement noise,

$$x_{\mathrm{s}}(k) = A_{\mathrm{ss}}\, x_{\mathrm{s}}(k-1) + B_{\mathrm{s}}\, u(k-1) + A_{\mathrm{sf}}\, u(k-2) + v(k-1) \tag{28}$$
$$y(k) = C_{\mathrm{s}}\, x_{\mathrm{s}}(k) + w(k). \tag{29}$$

The update equations for $\hat{x}(k|k–1)$, $\hat{x}_i(k|k–1)$, and $\check{x}_i(k|k–1)$ in (3), (12), and (18), respectively, are adapted accordingly. The feedback law (27) becomes

$$u(k-1) = [\, F_{\mathrm{s}} \quad F_{\mathrm{f}}\,] \begin{bmatrix} x_{\mathrm{s}}(k-1) \\ u(k-2) \end{bmatrix}. \tag{30}$$

Truth model. The truth model that is used for experimental comparison of the different state estimators is based on the nonlinear state estimation method for the Balancing Cube presented in [10], which is augmented with further non-causal post-processing. To obtain the “true” state denoted by $x^{\mathrm{truth}}(k)$, all sensor data (including, in particular, the accelerometer data) is recorded and the state is obtained in post-processing. The estimate of the cube tilt obtained from this method has been verified with a camera-based motion capture system (cf. results in [10]) and has proven to work well on the cube.

3.2 Implementation of the state estimation algorithm

The noise parameters Q and R of the Kalman filter (3)–(7) applied to the system given by (28) and (29) were treated as tuning parameters to obtain satisfactory centralized closed loop performance on the Balancing Cube. The following parameters were chosen:

$$Q = \mathrm{diag}\left([1\ 1\ 1\ 1\ 1\ 1\ 0.01\ 1]\right) \tag{31}$$
$$R = \mathrm{diag}\left([0.1\ 1\ 0.1\ 1\ 0.1\ 1\ 0.1\ 1\ 0.1\ 1\ 0.1\ 1]\right). \tag{32}$$

The Kalman filter was initialized with $x_0 = 0$ and $P_0 = Q/10$.

The same parameters were used for the distributed Kalman filters (12)–(16) and (18)–(22). The only additional parameters that had to be chosen for the distributed implementation are the communication thresholds: for all agents i, they were set to $\delta_{i,1} = 0.02\,\mathrm{rad}$ for the absolute encoder measurements and, for the gyro measurements, to about 2.5 times the standard deviation of the measurement noise, that is, $\delta_{i,2} = 0.01\,\mathrm{rad/s}$.

3.3 Experimental results

Generally, it is expected that communicating less than all data will affect the performance of the feedback control system. If the communication thresholds $\delta_{i,l}$ are all zero, then the performance is equivalent to implementing the centralized state estimator (3)–(7). Below we define the performance and communication measures used in this work.

For evaluating the closed loop performance, a performance index $\mathcal{P}$ is defined as the root-mean square (RMS) value of the truth model state x^{truth},

$$\mathcal{P} := \sqrt{\frac{1}{K}\sum_{k=1}^{K}(x^{\text{truth}}(k))^T x^{\text{truth}}(k)}, \tag{33}$$

for data of length K. For a system with state output that is driven by white noise with unit variance, the RMS value of the system state is equivalent to the $\mathcal{H}_2$ system norm (see e.g. [12]), which is a standard performance measure for stochastic control systems.

In network control systems the communication rate is commonly measured as the number of packets sent per time interval, (cf. [6]). Similarly, we consider the number of measurements sent per M steps as a measure for the amount of communication. The *communication rate* is computed as a moving average over M steps, that is, for the measurement $\bar{y}_{i,l}(k)$ (l denoting the element of the vector, i the agent), we define

$$\mathcal{R}_{i,l}(k) := \frac{\text{number of } \bar{y}_{i,l}(k) \text{ sent in } [(k-M+1)T_s, kT_s]}{MT_s}, \tag{34}$$

with the sampling time $T_s = 1/60$ s. The horizon is chosen as $M = 100$. Furthermore, the time average $\bar{\mathcal{R}}_{i,l}$ of $\mathcal{R}_{i,l}(k)$ and the *average total rate* $\mathcal{R}$ are defined by

$$\bar{\mathcal{R}}_{i,l} := \frac{1}{K}\sum_{k=1}^{K}\mathcal{R}_{i,l}(k),\ \mathcal{R} := \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{\bar{p}_i}\sum_{l=1}^{\bar{p}_i}\bar{\mathcal{R}}_{i,l}\right). \tag{35}$$

The communication rates $\mathcal{R}_{i,l}(k)$, $\bar{\mathcal{R}}_{i,l}$, and $\mathcal{R}$ all lie in the interval $[0,1]$ by definition. In particular, $\mathcal{R} = 1$ corresponds to the case where at each time step all data is exchanged between the agents, while $\mathcal{R} = 0$ means no data is exchanged.

Experiment: steady-state balancing. The distributed estimation method from Sec. 2.2 and 2.3 was implemented on the Balancing Cube in order to stabilize the cube about the equilibrium configuration shown in Fig. 3.

Table 2. Communication and performance measures for centralized (eq. (3)–(7)) and distributed state estimation (Algorithm 1).

	$\mathcal{R}$	$\mathcal{P}$
centralized estimation	1.000	0.192
distributed estimation	0.060	0.285

Table 3. Average communication rates for the encoder measurements (top row) and for the gyro measurements (bottom row).

(i, l)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(6, 1)
$\bar{\mathcal{R}}_{i,l}$	0.0048	0.0137	0.0028	0.0014	0.0052	0.0011
(i, l)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)
$\bar{\mathcal{R}}_{i,l}$	0.1059	0.1076	0.0913	0.1374	0.1157	0.1276

Data was recorded over a period of five minutes of balancing. The obtained measures of performance and communication, $\mathcal{P}$ and $\mathcal{R}$, are given in Table 2; they are compared to a run with the centralized Kalman filter of Sec. 2.1. The average communication rates for the distributed implementation are listed in Table 3.

The expected trade-off between communication rate and performance can be observed from these results: tolerating a decrease of performance roughly by a factor of 1.5 compared to the centralized case allows a reduction in communication events roughly by a factor of 16.

The communication rates for some of the absolute encoder measurements are particularly low (significantly less than 1 %, cf. Table 3). Communicating the positions at every time step is obviously not necessary, since this part of the system can apparently be predicted very well from the model. Still, this prediction needs to be updated occasionally with an actual measurement.

For a 30-second sequence, module 1's estimates of the module angles 1 and 3, the cube angle, and the cube angular velocity obtained by (12)–(16) and (18)–(22) are shown in Fig. 5. They are compared to the truth model state x^{truth}. The same module's communication rates are shown in Fig. 6.

4. Concluding Remarks

Experimental results demonstrate that the algorithm for distributed state estimation presented in this paper is an effective tool for reducing the av-

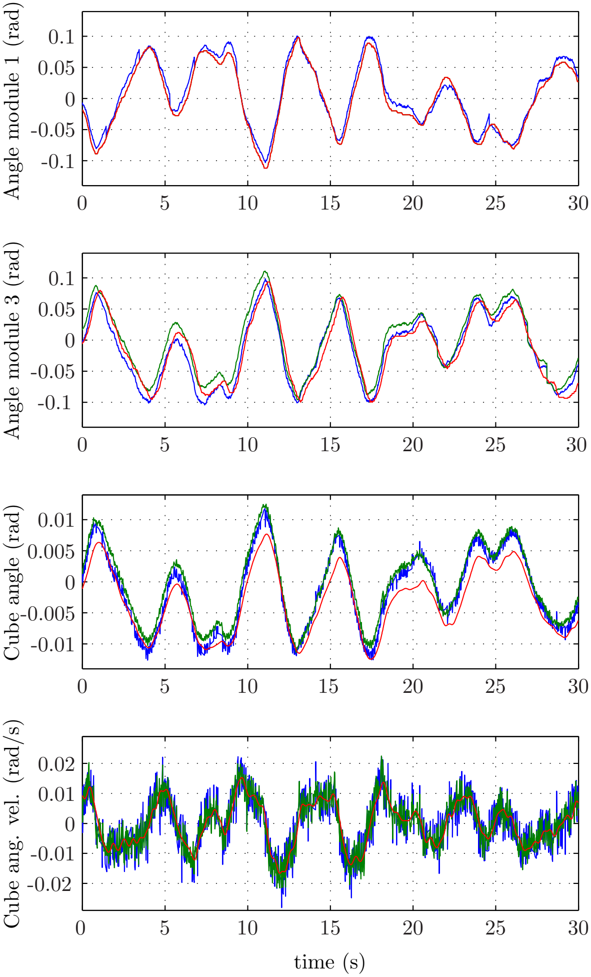


Figure 5. Agent 1's state estimates $\check{x}_1$ (blue) and $\hat{x}_1$ (green), compared to the truth model state x^{truth} (red) for its own module angle (top), agent 3's module angle, the cube angle, and the cube angular velocity (bottom). The graphs $\hat{x}_1$ and x^{truth} are practically identical in the top diagram.

erage communication rate in a networked control system. Moreover, it is a straight-forward tool to implement. First, it is based on the centralized design of the commonly used discrete-time Kalman filter. Second, the communication threshold parameters provide a practical handle for the de-

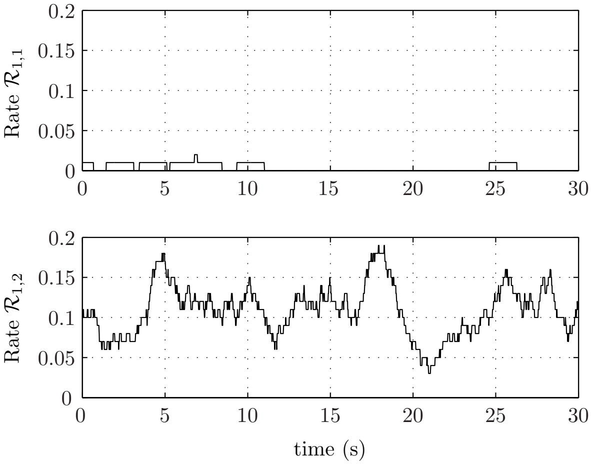


Figure 6. Agent 1's communication rates for its encoder (top) and rate gyro measurement (bottom).

signer to parametrize the trade-off between communication and estimator performance. In particular, the performance of the centralized design can be recovered and hence used as a starting point for fine-tuning the system performance.

A particularly useful feature of the algorithm presented herein is that the bandwidth required for sensor feedback is determined autonomously by the system, and need not be known beforehand. This encompasses the possibility that the average communication rates may vary for different types of sensors in the system. Furthermore, the system can easily adapt to change in communication requirements, using only the resources needed at any given moment. One may, in fact, view centralized estimation as the "fallback" system, in that it is simply the case where all measurements are communicated.

A theoretical analysis of the presented distributed estimation algorithm is beyond the scope this paper. Likewise, further study of variants of the employed algorithm on the Balancing Cube, such as making the sending decision of a measurement also based on its associated estimation variance, remain for future research.

Acknowledgements

The authors would like to thank Jan Lunze for stimulating discussions leading to the initial idea of this work.

Appendix

A. State Space Model and Feedback Gains of the Balancing Cube

The matrices of the state space model of the Balancing Cube in (25) and (26) and the static feedback gain matrices in (27) are given here for completeness:

$$A_{\mathrm{ss}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ a_{7,1} & a_{7,2} & a_{7,3} & a_{7,4} & a_{7,5} & a_{7,6} & a_{7,7} & a_{7,8} \\ a_{8,1} & a_{8,2} & a_{8,3} & a_{8,4} & a_{8,5} & a_{8,6} & a_{8,7} & a_{8,8} \end{bmatrix}$$

$$\begin{array}{llll} a_{7,1} = \text{2.8e-5} & a_{7,2} = \text{-5.6e-5} & a_{7,3} = \text{-2.8e-5} & a_{7,4} = \text{-2e-5} \\ a_{7,5} = \text{2.8e-5} & a_{7,6} = \text{2e-5} & a_{7,7} = 1 & a_{7,8} = 0.017 \\ a_{8,1} = 0.0033 & a_{8,2} = \text{-0.0067} & a_{8,3} = \text{-0.0033} & a_{8,4} = \text{-0.0024} \\ a_{8,5} = 0.0033 & a_{8,6} = 0.0024 & a_{8,7} = 0.15 & a_{8,8} = 1 \end{array}$$

$$A_{\mathrm{sf}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \tilde{a}_{7,1} & \tilde{a}_{7,2} & \tilde{a}_{7,3} & \tilde{a}_{7,4} & \tilde{a}_{7,5} & \tilde{a}_{7,6} \\ \tilde{a}_{8,1} & \tilde{a}_{8,2} & \tilde{a}_{8,3} & \tilde{a}_{8,4} & \tilde{a}_{8,5} & \tilde{a}_{8,6} \end{bmatrix}$$

$\tilde{a}_{7,1} = \text{6.3e-5}$ $\tilde{a}_{7,2} = \text{-0.00032}$ $\tilde{a}_{7,3} = \text{-6.3e-5}$ $\tilde{a}_{7,4} = \text{-0.00036}$
$\tilde{a}_{7,5} = 0.00018$ $\tilde{a}_{7,6} = 0.00036$ $\tilde{a}_{8,1} = 0.0038$ $\tilde{a}_{8,2} = \text{-0.019}$
$\tilde{a}_{8,3} = \text{-0.0038}$ $\tilde{a}_{8,4} = \text{-0.022}$ $\tilde{a}_{8,5} = 0.011$ $\tilde{a}_{8,6} = 0.022$

$$B_{\mathrm{s}} = \begin{bmatrix} 0.0167 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0167 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0167 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0167 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0167 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0167 \\ b_{7,1} & b_{7,2} & b_{7,3} & b_{7,4} & b_{7,5} & b_{7,6} \\ b_{8,1} & b_{8,2} & b_{8,3} & b_{8,4} & b_{8,5} & b_{8,6} \end{bmatrix}$$

$b_{7,1} = \text{-6.3e-5}$ $b_{7,2} = 0.00032$ $b_{7,3} = \text{6.3e-5}$ $b_{7,4} = 0.00036$
$b_{7,5} = \text{-0.00018}$ $b_{7,6} = \text{-0.00036}$ $b_{8,1} = \text{-0.0038}$ $b_{8,2} = 0.019$
$b_{8,3} = 0.0038$ $b_{8,4} = 0.022$ $b_{8,5} = \text{-0.011}$ $b_{8,6} = \text{-0.022}$

$$C_{\mathrm{s}}^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

$$F_{\mathrm{s}} = \begin{bmatrix} -0.175 & 0.246 & 0.121 & 0.106 & -0.121 & -0.106 & -6.31 & -2.07 \\ 0.295 & -0.676 & -0.295 & -0.258 & 0.295 & 0.258 & 15.4 & 5.05 \\ 0.121 & -0.246 & -0.175 & -0.106 & 0.121 & 0.106 & 6.31 & 2.07 \\ -0.142 & 0.288 & 0.142 & 0.0165 & -0.142 & -0.124 & -7.39 & -2.43 \\ -0.231 & 0.471 & 0.231 & 0.202 & -0.339 & -0.202 & -12.1 & -3.97 \\ 0.142 & -0.288 & -0.142 & -0.124 & 0.142 & 0.0165 & 7.39 & 2.43 \end{bmatrix}$$

$$F_{\mathrm{f}} = \begin{bmatrix} 0.916 & 0.0567 & 0.0215 & 0.0412 & -0.0292 & -0.0412 \\ 0.0524 & 0.794 & -0.0524 & -0.1 & 0.0713 & 0.1 \\ 0.0215 & -0.0567 & 0.916 & -0.0412 & 0.0292 & 0.0412 \\ -0.0252 & 0.0664 & 0.0252 & 0.974 & -0.0343 & -0.0483 \\ -0.0412 & 0.108 & 0.0412 & 0.0788 & 0.87 & -0.0788 \\ 0.0252 & -0.0664 & -0.0252 & -0.0483 & 0.0343 & 0.974 \end{bmatrix}$$

References

[1] J. Hespanha, P. Naghshtabrizi, and Y. Xu, “A survey of recent results in networked control systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[2] K. Åström, “Event based control,” in *Analysis and Design of Nonlinear Control Systems*, A. Astolfi and L. Marconi, Eds. Springer Berlin Heidelberg, 2008, pp. 127–147.

[3] J. Lunze and D. Lehmann, “A state-feedback approach to event-based control,” *Automatica*, vol. 46, no. 1, pp. 211–215, Jan. 2010.

[4] J. Yook, D. Tilbury, and N. Soparkar, “Trading computation for bandwidth: reducing communication in distributed control systems using state estimators,” *IEEE Transactions on Control Systems Technology*, vol. 10, no. 4, pp. 503 –518, Jul. 2002.

[5] Y. Xu and J. Hespanha, “Optimal communication logics in networked control systems,” in *Proc. of the 43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, Dec. 2004, pp. 3527–3532.

[6] ——, “Estimation under uncontrolled and controlled communications in networked control systems,” in *Proc. of the 44th IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, Dec. 2005, pp. 842–847.

[7] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, “Kalman filtering with intermittent observations,” *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1453–1464, Sep. 2004.

[8] W. Zhang, M. Branicky, and S. Phillips, “Stability of networked control systems,” *IEEE Control Systems Magazine*, vol. 21, no. 1, pp. 84–99, Feb. 2001.

[9] B. Anderson and J. Moore, *Optimal filtering.* Englewood Cliffs, NJ: Prentice-Hall, 1979.

[10] S. Trimpe and R. D'Andrea, "Accelerometer-based tilt estimation of a rigid body with only rotational degrees of freedom," in *Proc. of the IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, USA, May 2010, pp. 2630–2636.

[11] ——, "A limiting property of the matrix exponential with application to multi-loop control," in *Proc. of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, P.R. China, Dec. 2009, pp. 6419–6425.

[12] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*, 2nd ed. Wiley-Interscience, November 2005.

Paper III

Event-Based State Estimation with Switching Static-Gain Observers

Sebastian Trimpe

Abstract

An event-based state estimation problem is considered where the state of a dynamic system is observed from multiple distributed sensors that sporadically transmit their measurements to a remote estimator over a common bus. The common bus allows each sensor to run a copy of the remote estimator and to make the triggering decision based on this estimate: a measurement is transmitted only if its prediction by the estimator deviates by more than a tunable threshold. The event-based estimator is a switching observer that mimics a Luenberger observer with full communication of all measurements. It is proven that the difference between the event-based estimator and its full communication counterpart is bounded. The reduction of average sensor communication rates achieved by using the event-based state estimator for feedback control is demonstrated in experiments on a balancing cube.

Published in *3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, 2012, pp. 91–96.
DOI: 10.3182/20120914-2-US-4030.00022.

©2012 IFAC. NOTICE: This is the author's version of a work that was accepted for publication in *3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (2012).* Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in *3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (2012)*, [Volume 3, Part 1], DOI: 10.3182/20120914-2-US-4030.00022.

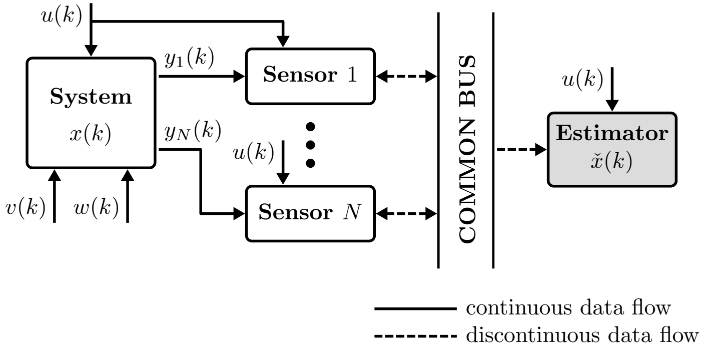


Figure 1. Distributed state estimation problem. The state $x(k)$ of a dynamical system is observed by measurements $y_i(k)$ of N sensors. The system is driven by a known input $u(k)$ and unknown process and measurement disturbances $v(k)$ and $w(k)$. The sensor nodes are connected to each other and to a remote estimator via a common bus. Each sensor decides when to broadcast its local measurement over the bus. The remote estimator generates an estimate $\check{x}(k)$ of the state $x(k)$ based on the received data.

1. Introduction

Figure 1 illustrates the state estimation problem that is considered in this paper. The state of a dynamic system is estimated from measurements of multiple sensors distributed along the system. The sensors have computing capability, and decide when to broadcast their measurement over a common bus network that all sensors and the estimator are connected to. Since the system state is not necessarily observable from a single sensor alone, communication from different sensors is required. The objective is to maintain an estimate of the full system state while, at the same time, reducing the load on the communication network.

An event-based strategy to addressing this problem is explained in Fig. 2, which depicts a single sensor node of Fig. 1. The key idea is to implement, on each sensor, a copy of the remote estimator and an appropriate transmit decision rule (event generator): a measurement is transmitted *only when it is required to meet a certain estimation performance.* The event-based state estimator consists of the state estimator itself and the event generator.

The common bus (or broadcast network) is a key enabling feature of the architecture in Fig. 1. It allows all sensors to also listen to the other sensors' data and, hence, to generate the estimate $\check{x}(k)$ locally at no additional com-

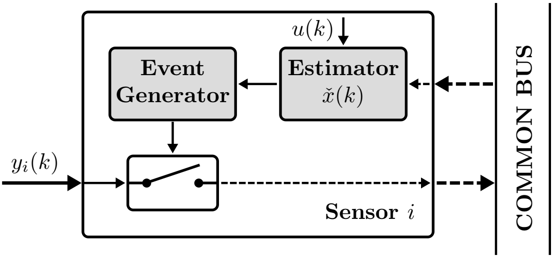


Figure 2. A single sensor node. Each sensor node listens to all data communicated from other sensors on the common bus. Therefore, it has access to the same data set as the remote estimator of Fig. 1 and can implement a copy of the same. Based on this estimate, the event generator decides when to broadcast the local measurement.

munication cost. This architecture allows for an effective implementation of a *distributed event triggering* scheme (i.e. one where the triggering depends only on local data), since the quantity of interest (the estimate $\check{x}(k)$) is actually available locally.

The input signal $u(k)$ is assumed to be accessible at the sensor nodes for making the estimator updates. For example, $u(k)$ may be an a-priori known reference input, or $u(k) = 0$ for an autonomous system. When $u(k)$ is computed from a feedback control law, it needs to be communicated to the sensor nodes. In that case, continuously communicating $u(k)$, but only sporadically transmitting the measurements $y_i(k)$ may be beneficial if there are fewer inputs than measurements. Furthermore, with knowledge of the inputs, the state estimation problem (which is the focus of this paper) can be treated without consideration of the feedback control law. This reduces the complexity of the design problem.

Different measures are conceivable for the event generation (i.e. for the decision to transmit a measurement). Trimpe and D'Andrea [1] use a constant threshold on the difference of an actual measurement and its prediction by the estimator as condition to trigger an event. We refer to such a triggering rule, which depends on (realtime) measurement data, as *measurement-based triggering*. This is also the approach taken herein. In contrast, a triggering condition that is based on the estimator variance is referred to as *variance-based triggering* (considered in [2]). Measurement-based triggering allows for the estimator to react to events in real time, whereas transmit schedules can typically be computed off-line with variance-based triggering.

In contrast to the previously mentioned work, where the state estimator is a time-varying Kalman filter, we use a switching Luenberger-type observer herein. The design is based on a Luenberger observer for the case of synchronous communication of all measurements (referred to as the *full communication state estimator* (FCSE)). The *event-based state estimator* (EBSE) updates its estimate with the available measurements using the corresponding sub-blocks of the static FCSE gain matrix. Hence, the estimator switches between different pre-computed static gains. Compared to a time-varying Kalman filter, where the filter gain is recomputed at every time step from a few matrix multiplications, the approach taken herein is less computationally demanding.

The main contributions of this paper are: the proposal of the EBSE as a switching static-gain observer combined with an appropriate event trigger; proof of an upper bound on the difference between the EBSE and the FCSE; and demonstration of the EBSE performance in experiments on an unstable networked control system (the Balancing Cube; see www.cube.ethz.ch for a video).

This paper is organized as follows: after a brief review of related work and an introduction of notation below, the estimation problem is stated in Sec. 2. The event-based state estimator is derived in Sec. 3, and its stability is analyzed in Sec. 4. Experimental results on the Balancing Cube testbed are given in Sec. 5, and the paper concludes with remarks in Sec. 6.

Related Work. The approach to event-based *state estimation* taken herein is conceptually related to the approach for event-based *control* by Lunze and Lehmann [3] for a centralized design, and by Stöcker et al. [4] for a decentralized problem. Therein, the authors design an event triggering rule such that the difference between the state of a reference system with continuous feedback and the state of the event-based control system is bounded. Here, the FCSE is the reference estimator, and event triggers are designed such that the difference of the EBSE to the FCSE is bounded.

Event-based strategies are a popular means of ensuring an efficient use of the communication resource in control, estimation, and optimization problems in networked control systems (see [5] for an overview). For a single sensor and a single estimator node, event-based state estimation problems have been studied by several researchers (see [5] and references therein). A distributed estimation problem related to the one herein is addressed by Weimer et al. [6]. The authors design communication policies for wireless sensor nodes that may either transmit information to a central estimator, listen to information from the central estimator, or be turned off. When either sensor or estimator transmit data, all data since the last update is sent; hence, the load per packet is variable whereas it is fixed for the approach herein.

Notation. For a vector $v \in \mathbb{R}^n$ and $q \in [1, \infty]$, $\|v\|_q$ (or simply $\|v\|$) denotes the vector Hölder norm of v (see [7])

$$\|v\| = \|v\|_q = \begin{cases} \left(\sum_{j=1}^{n} |v_j|^q \right)^{1/q} & \text{for } 1 \le q < \infty \\ \max\limits_{j \in \{1,\dots,n\}} |v_j| & \text{for } q = \infty. \end{cases} \tag{1}$$

For a matrix A, $\|A\|_q$ (or simply $\|A\|$) denotes the matrix norm of A induced by the chosen vector norm. For a vector-valued sequence $v = \{v(0), v(1), v(2), \dots\}$, $\|v\|_\infty$ denotes the ℓ^∞ norm of v (see [8])

$$\|v\|_\infty := \sup_{k \ge 0} \|v(k)\|,$$

where $\|v(k)\|$ is the chosen vector norm.

2. Estimation Problem Formulation

Consider the linear time-invariant (LTI) system

$$x(k) = A\,x(k-1) + B\,u(k-1) + v(k-1) \tag{2}$$

$$y_1(k) = C_1\,x(k) + w_1(k) \tag{3}$$

$$\vdots$$

$$y_N(k) = C_N\,x(k) + w_N(k), \tag{4}$$

where $k \ge 1$ is the discrete-time index, $x(k) \in \mathbb{R}^n$ is the system state, $u(k) \in \mathbb{R}^{n_u}$ the control input, $y_i(k) \in \mathbb{R}^{p_i}$, $i \in \{1, \dots, N\}$, are measurements by N sensors, $v(k) \in \mathbb{R}^n$, $w_i(k) \in \mathbb{R}^{p_i}$, $i \in \{1, \dots, N\}$, are disturbances, and all matrices are of corresponding dimensions. We use $y(k)$ to denote the vector that combines all measurements; that is,

$$y(k) := \begin{bmatrix} y_1(k) \\ \vdots \\ y_N(k) \end{bmatrix} = \underbrace{\begin{bmatrix} C_1 \\ \vdots \\ C_N \end{bmatrix}}_{=:C} x(k) + \underbrace{\begin{bmatrix} w_1(k) \\ \vdots \\ w_N(k) \end{bmatrix}}_{=:w(k)}$$

$$= Cx(k) + w(k).$$

Hence, $y(k), w(k) \in \mathbb{R}^p$ with $p := \sum_{i=1}^{N} p_i$. We assume that (A, C) is detectable. Notice that (A, C_i) is not assumed to be detectable; that is, the system state is not necessarily detectable from any individual sensor alone.

Notice that no assumption on the characteristics of the disturbances $v(k)$ and $w(k)$ is made. For example, $v(k)$ and $w(k)$ may be random variables with known statistics in a stochastic setting; or they may be bounded disturbances in a deterministic setting.

2.1 Full Communication State Estimator

Next, we introduce the FCSE, which uses the full measurement vector $y(k)$ at every time step and serves as a reference to the EBSE design later:

$$\hat{x}(k|k-1) = A\,\hat{x}(k-1|k-1) + B\,u(k-1) \tag{5}$$

$$\hat{x}(k|k) = \hat{x}(k|k-1) + L\,\big(y(k) - C\,\hat{x}(k|k-1)\big), \tag{6}$$

with the static estimator gain L. The estimator is initialized with some $\hat{x}(0) \in \mathbb{R}^n$. It generates an estimate $\hat{x}(k|k)$ of the state $x(k)$ based on all past measurements up to, and including, $y(k)$. For ease of notation, we write $\hat{x}(k) := \hat{x}(k|k)$.

The estimator gain L is designed such that $(I - LC)A$ is stable (i.e. all eigenvalues have magnitude less than one). If (A, C) is detectable, such an L is guaranteed to exist (see [9]). It can be designed, for instance, via pole placement (see [9]) or as the steady-state solution of the Kalman filter (see [10]).

Let $\hat{e}(k) := x(k) - \hat{x}(k)$ denote the estimation error of the FCSE. The error evolves according to

$$\hat{e}(k) = (I - LC)A\hat{e}(k-1) + (I - LC)v(k-1) - Lw(k). \tag{7}$$

In a state estimation scenario where $v(k)$ and $w(k)$ are bounded disturbances, the stability of $(I - LC)A$ implies that the estimation error is also bounded. If $v(k)$ and $w(k)$ are independent random variables with finite variance, (7) ensures that the estimation error variance is bounded.

2.2 Problem Statement

An EBSE is sought that approximates the estimate $\hat{x}(k)$ of the FCSE (5)–(6) up to a guaranteed bound, but uses fewer measurements. The EBSE consists of an event generator and a state estimator (with state $\check{x}(k)$) as symbolized by the blocks in Fig. 2.

The sensor nodes and the remote estimator are assumed to be synchronized in time, and transmission via the communication network is assumed to be instantaneous and without data loss. Hence, the state estimates $\check{x}(k)$ on all nodes are assumed identical.

3. Event-Based State Estimator

In this section we present the EBSE, which addresses the problem stated above.

Event Generator. The event generator on sensor i decides at every time step k whether or not to transmit the local measurement $y_i(k)$. The measurement $y_i(k)$ is transmitted whenever a prediction of that measurement based on the previous estimate $\check{x}(k-1)$ is off by more than a tolerable threshold.

Without any information on $v(k)$ and $w_i(k)$, a prediction $\check{y}_i(k)$ of the measurement $y_i(k)$ may be obtained from (2)–(4) by setting $v(k) = 0$ and $w_i(k) = 0$; that is,

$$\check{y}_i(k) := C_i\big(A\check{x}(k-1) + Bu(k-1)\big).$$

Using $\check{x}(k|k-1) := A\check{x}(k-1) + Bu(k-1)$, the employed event triggering rule is

$$\text{transmit } y_i(k) \quad \Leftrightarrow \quad \|y_i(k) - C_i\check{x}(k|k-1)\| \geq \delta_i, \tag{8}$$

with threshold parameters $\delta_i \geq 0$, $i \in \{1, \ldots, N\}$. Tuning δ_i allows the designer to trade off each sensor's frequency of events (and, hence, the communication rate) for estimator performance. For notational convenience, we denote $\delta = (\delta_1, \ldots, \delta_N) \in \mathbb{R}^N$ the vector of threshold parameters δ_i.

State Estimator. Let $I(k)$ denote the tuple of indices of those sensors that transmitted their measurement at time k; that is,

$$I(k) := \big(i \mid 1 \leq i \leq N,\ \|y_i(k) - C_i\check{x}(k|k-1)\| \geq \delta_i\big). \tag{9}$$

The filter that is used to generate an estimate $\check{x}(k) := \check{x}(k|k)$ of the state $x(k)$ based on the measurements broadcast up to time k is given by:

$$\check{x}(k|k-1) = A\,\check{x}(k-1|k-1) + B\,u(k-1) \tag{10}$$

$$\check{x}(k|k) = \check{x}(k|k-1) + \sum_{i\in I(k)} L_i\big(y_i(k) - C_i\check{x}(k|k-1)\big), \tag{11}$$

where $L = [L_1, L_2, \dots, L_N]$ with $L_i \in \mathbb{R}^{n \times p_i}$ is the decomposition of the estimator gain matrix according to the dimensions of the individual measurements. By rewriting (6) as

$$\hat{x}(k) = \hat{x}(k|k-1) + \sum_{i \in \{1,\dots,N\}} L_i \big(y_i(k) - C_i \hat{x}(k|k-1) \big), \tag{12}$$

one can see that (11) is obtained from (6) by including only those elements in the summation where measurements are available. If, at time k, no measurement is transmitted (i.e. $I(k) = \emptyset$), (11) is to be understood such that the summation vanishes; that is, $\check{x}(k|k) = \check{x}(k|k-1)$. In order to ease the presentation, this case is not explicitly mentioned hereafter. The filter (10)–(11) and the triggering rule (8) constitute the EBSE.

We assume henceforth that the EBSE is initialized with the same value as the FCSE, i.e. $\check{x}(0) = \hat{x}(0)$. This is a reasonable assumption since we seek to mimic the FCSE with the EBSE.

Notice that the estimator gains L_i in (11) are blocks of the constant matrix L; that is, the entries can be computed off-line. This is different from the approach in [1], where a time-varying Kalman filter is used, and the entries of the estimator gain are recomputed at every step k. The approach herein thus has lower computational complexity.

For a fixed sequence $\{I(1), \dots, I(k)\}$, the filter (10)–(11) is a linear filter. The index tuple $I(k)$ does, however, depend on $y(k)$ by (9); hence, (10)–(11) represent a nonlinear filter. It is a switching observer, whose switching modes correspond to the available measurements at time k.

Notice that any individual mode of the filter may be unstable ((A, C_i) is not necessarily detectable). Moreover, even if the individual modes were stable, this would not imply the stability of the switching observer, as discussed in [11] and [12] (Sec. 6.2). The stability of the presented EBSE follows from the combination of (10)–(11) with the triggering condition (8). This is discussed in detail in the next section.

4. Analysis

We introduce the error measures

$$e(k) := \hat{x}(k) - \check{x}(k) \quad \text{and} \tag{13}$$

$$\check{e}(k) := x(k) - \check{x}(k), \tag{14}$$

which are analyzed below. The error $e(k)$ is the difference between the state estimate of the FCSE and the EBSE. It is required to be bounded according to the problem statement in Sec. 2.2. The error $\check{e}(k)$ is the estimation error of the EBSE (defined analogously to $\hat{e}(k)$ for the FCSE).

Difference of the EBSE to the FCSE. From (5), (10), (11), (12), and (13), we get

$$\begin{aligned} e(k) &= \hat{x}(k) - \check{x}(k) \\ &= A\hat{x}(k-1) + Bu(k-1) - A\check{x}(k-1) - Bu(k-1) \\ &\quad + \sum_{i\in\{1,\dots,N\}} L_i\big(y_i(k) - C_iA\hat{x}(k-1) - C_iBu(k-1)\big) \\ &\quad - \sum_{i\in\{1,\dots,N\}} L_i\big(y_i(k) - C_iA\check{x}(k-1) - C_iBu(k-1)\big) \\ &\quad + \sum_{i\in\bar{I}(k)} L_i\big(y_i(k) - C_iA\check{x}(k-1) - C_iBu(k-1)\big), \end{aligned}$$

where

$$\begin{aligned} \bar{I}(k) &:= (1,\dots,N) \setminus I(k) \\ &= \big(i \mid 1 \le i \le N,\, \|y_i(k) - C_i\check{x}(k|k-1)\| < \delta_i\big). \end{aligned} \tag{15}$$

Straightforward manipulation then yields

$$\begin{aligned} e(k) &= \Big(A - \sum_{i\in\{1,\dots,N\}} L_iC_iA\Big)\, e(k-1) \\ &\qquad + \sum_{i\in\bar{I}(k)} L_i\big(y_i(k) - C_i\check{x}(k|k-1)\big) \\ &= (I-LC)Ae(k-1) + \sum_{i\in\bar{I}(k)} L_i\big(y_i(k) - C_i\check{x}(k|k-1)\big). \end{aligned} \tag{16}$$

The dynamics of $e(k)$ are those of a stable LTI system ($(I - LC)A$ is stable) with an input which is bounded according to (15). Hence, we have the following theorem.

THEOREM 1 Let all eigenvalues of $(I - LC)A$ have magnitude less than one (i.e. the error dynamics of the FCSE are stable). Then, the difference $e(k)$ between the FCSE and the EBSE is bounded for all k. In particular, there exist constants $m > 0$ and $\rho \in [0, 1)$ such that

$$\|e\|_\infty \le \frac{m}{1-\rho}\, \|L\|\, \|\delta\| =: e_{\max}. \tag{17}$$

■

Proof. Rewrite (16): for $k \geq 1$,

$$\bar{e}(k{+}1) = (I - LC)A\bar{e}(k) + L_{\bar{I}(k)}\Delta_{\bar{I}(k)}(k), \tag{18}$$

where $\bar{e}(k{+}1) := e(k)$; $\Delta_i(k) := y_i(k) - C_i\check{x}(k|k{-}1)$; $\Delta_{\bar{I}(k)}(k)$ denotes the matrix obtained from consecutively stacking the vectors $\Delta_i(k)$, $i \in \bar{I}(k)$, from top to bottom; and $L_{\bar{I}(k)}$ denotes the matrix from consecutively stacking L_i, $i \in \bar{I}(k)$, from left to right.

First, notice that $\bar{e}(k{+}1) = (I - LC)A\bar{e}(k)$ is exponentially stable by assumption. Therefore, there exist constants $m > 0$ and $\rho \in [0, 1)$ such that for all $k_0 \in \mathbb{N}$ and $k \geq k_0$,

$$\|\big((I - LC)A\big)^{k-k_0}\| \leq m\,\rho^{k-k_0}, \tag{19}$$

[8, p. 212–213, Def. 17, Thm. 33].

Recalling the definition of the vector norm (1), one can see that, for $1 \leq q < \infty$,

$$\|\Delta_{\bar{I}(k)}(k)\|_q^q = \sum_{i\in\bar{I}(k)} \|\Delta_i(k)\|_q^q \underset{(15)}{<} \sum_{i\in\bar{I}(k)} \delta_i^q \leq \sum_{i=1}^{N} \delta_i^q = \|\delta\|_q^q,$$

and, for $q = \infty$,

$$\|\Delta_{\bar{I}(k)}(k)\|_q = \max_{i\in\bar{I}(k)} \|\Delta_i(k)\|_q \underset{(15)}{<} \max_{i\in\bar{I}(k)} \delta_i \leq \|\delta\|_q.$$

Hence, for $1 \leq q \leq \infty$, $\|\Delta_{\bar{I}(k)}(k)\| < \|\delta\|$, and

$$\sup_{k\geq 1} \|\Delta_{\bar{I}(k)}(k)\| \leq \sup_{k\geq 1} \|\delta\| = \|\delta\|.$$

Since also $\left\|L_{\bar{I}(k)}\right\| \leq \|L\|$, it follows that the input term $L_{\bar{I}(k)}\Delta_{\bar{I}(k)}(k)$ in (18) is bounded. Using these results and applying the bounded trajectories theorem [8, p. 218, Thm. 75] then yields

$$\sup_{k\geq 1} \|\bar{e}(k)\| \leq m\|\bar{e}(1)\| + \frac{m}{1-\rho}\,\|L\|\,\|\delta\|.$$

Equation (17) follows by $\bar{e}(1) = e(0) = \hat{x}(0) - \check{x}(0) = 0$. □

Notice that the bound (17) holds irrespective of the representation of the disturbances $v(k)$ and $w_i(k)$ in (2)–(4). In particular, it holds for the case where the disturbances are unbounded, such as for Gaussian noise.

Estimator error. The estimation error $\check{e}(k)$ of the EBSE can be written as

$$\check{e}(k) = x(k) - \hat{x}(k) + \hat{x}(k) - \check{x}(k) = \hat{e}(k) + e(k). \tag{20}$$

Therefore, Theorem 1 can be used to deduce properties of the EBSE from properties of the FCSE. For example, if the estimation error $\hat{e}(k)$ of the FCSE is bounded, then by (20) and Theorem 1, the error $\check{e}(k)$ of the EBSE is also bounded. In general, Theorem 1 shows that for $\delta_i \to 0$, $e(k)$ becomes arbitrarily small; that is, the performance of the FCSE is recovered.

5. Experiments

The Balancing Cube shown in Fig. 3 serves as the testbed for demonstrating the event-based state estimation method. Six rotating arms on the inner faces of the cube allow the cube to balance on any of its edges or corners. The arms (called *modules*) constitute the agents of the networked control system: each one is equipped with sensors, actuation, and a single-board computer. The computers share data over a Controller Area Network (CAN) bus. For the purpose of this paper, the cube balances on one of its edges. The experimental setup is the same as in [1], where more detailed descriptions of the system, the linear model, and the controller can be found.

The eight states of the system model (2) are the angles of the six modules (rotation relative to the cube structure), and the angle and angular rate of the cube about its axis of rotation. Since each module regulates its angular velocity locally with a fast feedback controller, the angular velocities of the six modules are treated as plant inputs. Two types of sensors are used on each module: an absolute encoder measuring the module angle and a rate gyroscope measuring the angular velocity of the cube. Hence, $N = 12$ sensors are used in total. Each module is able to observe its own angle and the cube states with its local sensors, but the complete system state is not locally observable.

The gain L of the linear observer (5)–(6) is designed as the gain of a steady-state Kalman filter. The resulting eigenvalues of $(I - LC)A$ are 0.936, 0.427, and (with algebraic multiplicity 6) 0.382. For the rate gyro sensors, the transmit threshold $\delta_{\text{gyro}} = 0.004\,\text{rad/s}$ is used, which corresponds to roughly one standard deviation of the sensor noise. For the encoders, $\delta_{\text{enc}} = 0.008\,\text{rad}$ is chosen.

Every module implements a copy of the EBSE (10)–(11) as shown in Fig. 2. In addition to using the estimate $\check{x}(k)$ for the transmit decision in

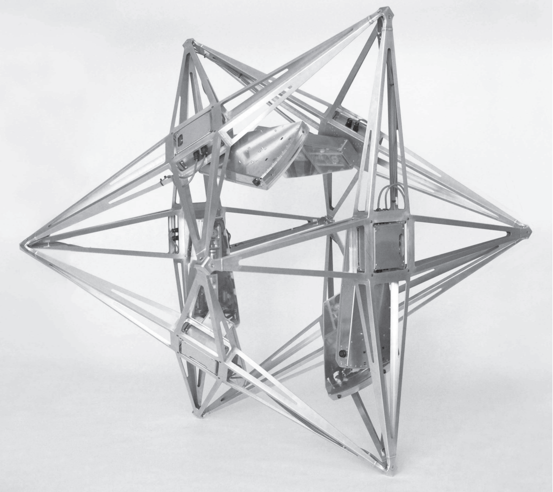


Figure 3. The experimental testbed: six rotating modules balance a cubic structure on one of its edges. (See www.cube.ethz.ch for a video.)

the event generator, the estimator feeds an LQR controller that computes the input to the local actuator (i.e. one of the elements of $u(k)$). Hence, there is feedback from the estimators on the sensor nodes to the system input $u(k)$ (not shown in Fig. 1). A detailed discussion and analysis of the distributed feedback control system is beyond the scope of this paper and a subject for future work.

The sensors are sampled, and the controller commands are updated every 16.6 ms. The control inputs are shared over the CAN bus at every time step, so that $u(k)$ is available to compute (10) on each module. The network bandwidth is sufficient to broadcast all input and measurement data within the duration of one time step. Hence, data transmission is assumed synchronous for all practical purposes.

A truth model state is computed in post-processing from all recorded sensor data, which also includes measurements from multiple accelerometers on the cube (see [1] for details). The performance $\mathcal{P}$ of the feedback control system is measured as the root-mean square (RMS) value of the truth model state. By $\mathcal{R}_i(k)$ we denote the average communication rate of measurement y_i. It is computed at time k as the moving average over the last 100 steps. Furthermore, $\bar{\mathcal{R}}_i$ denotes the time average of $\mathcal{R}_i(k)$ over the duration of the experiment, and $\mathcal{R}$ denotes the *average total rate* (the average of $\bar{\mathcal{R}}_i$ over all sensors i). Hence, $\mathcal{R}$ is a measure of the total communication in the network

Table 1. Experimental results.

	$\mathcal{R}$	$\mathcal{P}$
FCSE	1.000	0.183
EBSE	0.221	0.206

Table 2. Average communication rates $\bar{\mathcal{R}}_i$.

Module #	1	2	3	4	5	6
Encoder	0.0001	0.024	0	0	0.022	0
Gyro	0.429	0.406	0.384	0.472	0.453	0.459

($\mathcal{R} = 1$ means full communication, $\mathcal{R} = 0$ means no communication).

Table 1 shows the communication and performance measures $\mathcal{R}$ and $\mathcal{P}$ for two three-minute balancing experiments: one experiment using the EBSE and another with the FCSE. The results illustrate the expected trade-off between control performance and average communication.

In Table 2, the average communication rates $\bar{\mathcal{R}}_i$ of the individual sensors are given for the same EBSE experiment. They can be interpreted as follows: due to local regulation of the module angular velocities, the module angles can be predicted very accurately from the known velocity input; hence, little communication is required. The gyro sensors, on the other hand, observe the unstable mode of the cube; hence, sufficiently high rates are required for stabilizing the cube. The encoder communication rates of Modules 2 and 5 (on the front and back face of the cube in Fig. 3) are greater than those of the other modules, because Modules 2 and 5 move more during balancing, and their motion is affected more severely by gear backlash in the actuation. Gear backlash is not captured by the linear model (2), and its effect can therefore not be predicted by (10).

In another experiment, Module 1 was manually displaced during balancing (a clutch in the actuation mechanism allows the module to slip when pushed). The communication rates over time are shown in Fig. 4. Clearly, Module 1's encoder rate adapts to the external disturbance. While the module is being pushed, (10) cannot accurately predict the module angle, hence its communication rate goes up.

For the same experiment, Fig. 5 illustrates the performance of the EBSE exemplarily for Module 1's angle and the cube angle. For $q = \infty$ in (1), the bound in (17) is $e_{\max} = 0.1162$. Clearly, the error signals $e(k)$ in Fig. 5

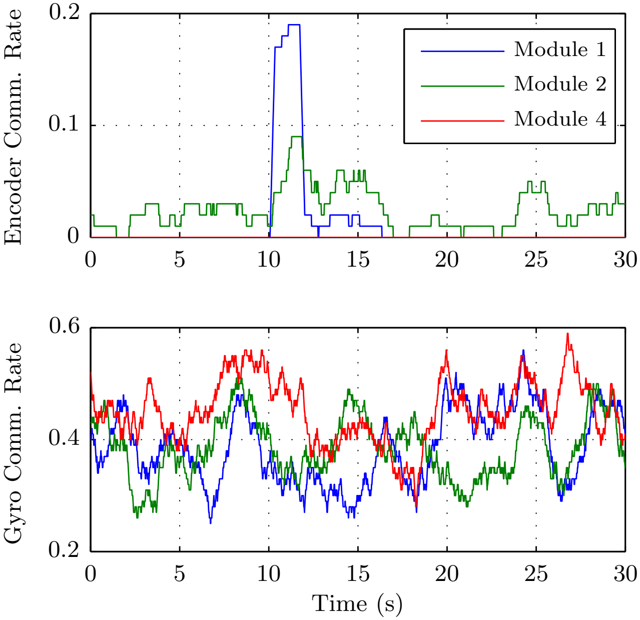


Figure 4. Experimental communication rates. Roughly at 10 s, Module 1 was displaced by pushing it.

are well below. The conservatism of the bound (17) stems from the upper bound (19) on the state-transition matrix of the multivariate system (16).

6. Concluding Remarks

The proposed event-based state estimator for a distributed arrangement of sensors is a direct extension of well-known methods for linear state estimation with synchronous (time-sampled) measurement feedback (such as the Luenberger observer or the steady-state Kalman filter). The approach allows one to trade off estimator performance achievable with the full communication design for communication bandwidth. Experiments on the Balancing Cube illustrated the ability of the event-based control system to discriminate different sensor types and adapt the sensor communication rates to the need for feedback control.

The setup for the experiments on the Balancing Cube is essentially the

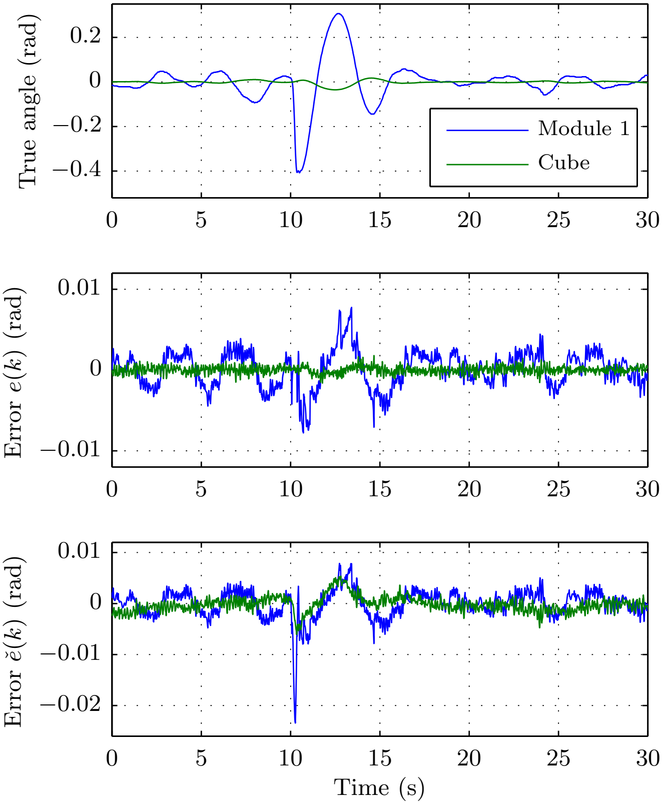


Figure 5. Experimental estimator performance (same data sequence as in Fig. 4). The estimates of Module 1's angle and the cube angle are shown. Top graph: truth model states; middle: difference between FCSE and EBSE (the error $e(k)$); bottom: EBSE error $\check{e}(k)$.

same as in [1], but the event-based state estimator implementation herein is different: a switching Luenberger observer is used instead of a time-varying Kalman filter. Whereas a time-varying filter is computationally more expensive, it potentially has a better performance, since the filter gains adapt on-line to the set of received measurements at a time step. The experimental results (e.g. in Table 1) should, however, not directly be compared with those in [1], because different design parameters were chosen. An experimental comparison of the different methods is planned for future work.

Each agent on the Balancing Cube implements a copy of the event-based state estimator and uses the estimate to compute its local control input. To satisfy the assumption that the input vector $u(k)$ is available at all sensor nodes (see Fig. 1), the inputs are shared between the agents over the network. The experiments demonstrated the performance of the event-based control system under realistic conditions, where the individual estimates are not perfectly identical. The stability of the distributed feedback system with non-identical estimators is, however, not analyzed herein. Strategies for removing the requirement of the continuous exchange of the control inputs, as well as the stability analysis of the distributed feedback control system shall be addressed in future work.

Acknowledgements

The author would like to thank Raffaello D'Andrea, Jan Lunze, and Christian Stöcker for insightful discussions.

References

[1] S. Trimpe and R. D'Andrea, "An experimental demonstration of a distributed and event-based state estimation algorithm," in *Proc. 18th IFAC World Congress*, Milan, Italy, Aug. 2011, pp. 8811–8818.

[2] ——, "Reduced communication state estimation for control of an unstable networked control system," in *Proc. 50th IEEE Conf. on Decision and Control and European Control Conf.*, Orlando, Florida, USA, 2011, pp. 2361–2368.

[3] J. Lunze and D. Lehmann, "A state-feedback approach to event-based control," *Automatica*, vol. 46, no. 1, pp. 211–215, 2010.

[4] C. Stöcker, J. Lunze, and D. Vey, "Stability analysis of interconnected event-based control loops," in *Proc. 4th IFAC Conf. on Analysis and Design of Hybrid Systems*, Eindhoven, Netherlands, 2012, pp. 58–63.

[5] M. Lemmon, "Event-triggered feedback in control, estimation, and optimization," in *Networked Control Systems*, A. Bemporad, M. Heemels, and M. Johansson, Eds. Springer-Verlag, 2011, vol. 406, pp. 293–358.

[6] J. Weimer, J. Araujo, and K. H. Johansson, "Distributed event-triggered estimation in networked systems," in *Proc. 4th IFAC Conf. on*

Analysis and Design of Hybrid Systems, Eindhoven, Netherlands, 2012, pp. 178–185.

[7] D. S. Bernstein, *Matrix mathematics: theory, facts, and formulas with applications to linear system theory.* New Jersey: Princeton University Press, 2005.

[8] F. M. Callier and C. A. Desoer, *Linear System Theory.* Springer-Verlag, 1991.

[9] K. J. Åström and B. Wittenmark, *Computer-controlled systems: theory and design.* Prentice Hall, 1997.

[10] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* Mineola, New York: Dover Publications, 2005.

[11] J. Lunze, "Ein Beispiel für den Entwurf schaltender Beobachter," *Automatisierungstechnik*, vol. 48, pp. 556–562, 2000.

[12] G. Böker and J. Lunze, "Stability and performance of switching Kalman filters," *International Journal of Control*, vol. 75, no. 16/17, pp. 1269–1281, 2002.

Part B

The Balancing Cube: A Test Bed for Distributed Estimation and Control

Paper IV

The Balancing Cube: A Dynamic Sculpture as Test Bed for Distributed Estimation and Control

Sebastian Trimpe · Raffaello D'Andrea

Published in *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 48–75, Dec. 2012.
DOI: 10.1109/MCS.2012.2214135.

©2012 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Note about this reprint: the original article in the *IEEE Control Systems Magazine* includes sidebars, which are given herein as separate sections (marked by Sidebar 1 to Sidebar 7) at the end of this paper.

1. Introduction

The Balancing Cube is a dynamic sculpture that can balance autonomously on any of its edges or corners (see Fig. 1 to 4). When standing on a corner, the cube represents a three dimensional inverted pendulum with multiple actuation, sensing, and control units that are interconnected over a communication network. The main structural components are the *cube body* (a rigid aluminum structure with a cubic shape), and six identical rotating arms located on each of the cube's inner faces. The rotating arms are self-contained units carrying sensors, actuation, a computer, and a battery. Due to their modular design, these units are referred to as *modules.* As they rotate, they shift the overall center of mass of the system, exert forces on the cube structure, and can, as a result, influence the cube's motion. The modules constitute the agents in the distributed and networked control system; their joint objective is the stabilization of the cube. A video of the cube can be found on the project website [1].

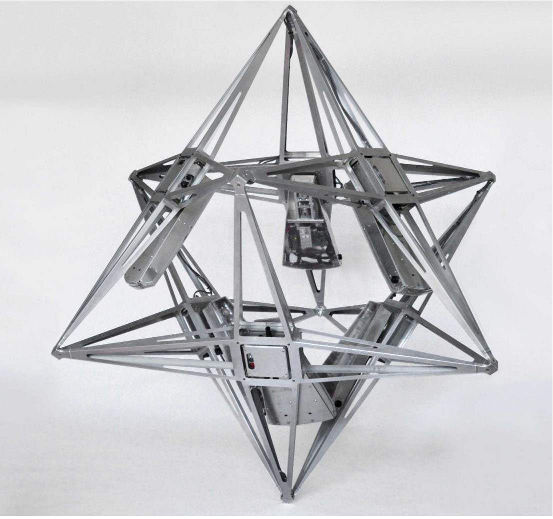


Figure 1. The cube balancing on one of its corners. The cube balances through the action of six rotating arms on the cube's inner faces. The diagram in Fig. 2 is helpful for visualizing why the sculpture is called a cube: its tips are simply the corners of a cube. Alternative terms for this shape are the star tetrahedron and, less commonly used, the stellated octahedron. Self-contained with onboard sensing, actuation, computation, and communication, the rotating arms are called *modules* due to their modular design. The cube's height from tip to tip is 2.08 m.

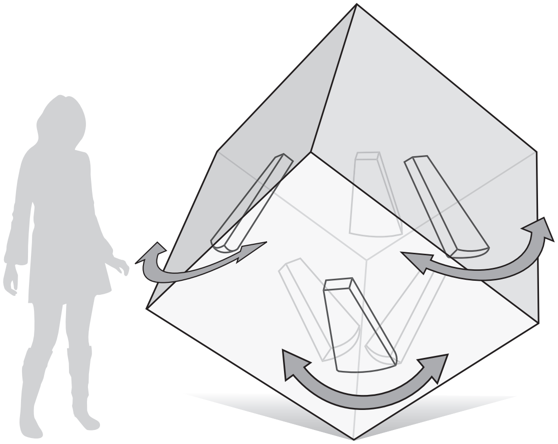


Figure 2. Corner balancing. The diagram visualizes the cubic shape and the six modules. It shows the cube in the same orientation as the photo in Fig. 1: the cube stands on a corner, and all modules are pointing down. Due to their position on the cube body, two types of modules are distinguished: the *bottom modules* and the *top modules*. The top modules are less effective for balancing the sculpture, as explained later in the sidebar "Why Are the Top Modules Used Less?"

Inverted pendulum systems are popular in controls education and research; see for example [2]–[4] and references therein. In the most basic version, an inverted pendulum is a point mass on a massless link that is connected to a base (ground or moving platform) through a revolute joint with one degree of freedom (DOF, see Table 1 for acronyms). More generally, if the pendulum is considered as a general rigid body with three rotational degrees of freedom at its pivot, the pendulum is referred to as a 3D pendulum, [2]. A defining characteristic of all *inverted* pendulums is that the pendulum is pointing upward as seen from the pivot (that is, in opposite direction to the gravity vector). The corresponding equilibrium is hence unstable, which makes the system interesting for controls researchers and educators: in order to balance, the pendulum needs active stabilization by some actuation mechanism, such as actuated masses or a moving base.

Used as demonstrators for controls research since the 1950s (see [4] and references therein), inverted pendulum systems have remained popular experiments (simulation or physical) in various current research areas, such as learning control [5]–[7], networked control [8]–[12], adaptive control [13]–[15], model predictive control [8, 16, 17], decentralized control [18]–[20], and

Table 1. Acronyms used in this article.

Acronym	**Meaning**
CAN	controller area network
CG	center of gravity
DC	direct current
DOF	degree of freedom
IMC	inter-module communication
IMU	inertial measurement unit
LED	light-emitting diode
LQR	linear-quadratic regulator
MEMS	micro-electro-mechanical system
RMS	root mean square
SBC	single-board computer
SPI	serial peripheral interface
WLAN	wireless local area network

different branches of nonlinear control [21]–[28]. Common to most inverted pendulum systems is that the control algorithms are implemented on a single, central processing unit. In contrast, the Balancing Cube is stabilized by the joint action of six agents (the modules) with independent processing units, and the implementation of the feedback control system is distributed among the agents. The dynamics of the individual agents are coupled through the cube's rigid body.

Because data is exchanged between the agents over a digital communication network, the Balancing Cube qualifies as a *networked control system*, [29]. Other experimental testbeds that have been developed to study distributed control and/or control of multiple agents over networks include modular robot systems [30, 31], a two-axis contouring system [32], a system for handling materials [33], a formation flight experiment [34], and multi-vehicle systems of various types [31, 35]–[37].

The cube is a 3D inverted pendulum when balancing on one of its corners (denoted as *corner balancing*). When balancing on one of its edges (*edge balancing*) as shown in Fig. 3 and 4, it becomes a 1D inverted pendulum. Moreover, with the six modules on its inner faces, the cube is a multi-body system and may therefore be qualified as a 1D/3D multi-body inverted pendulum, [2]. For both edge and corner balancing, different equilibrium configurations and, hence, different dynamics can be obtained by

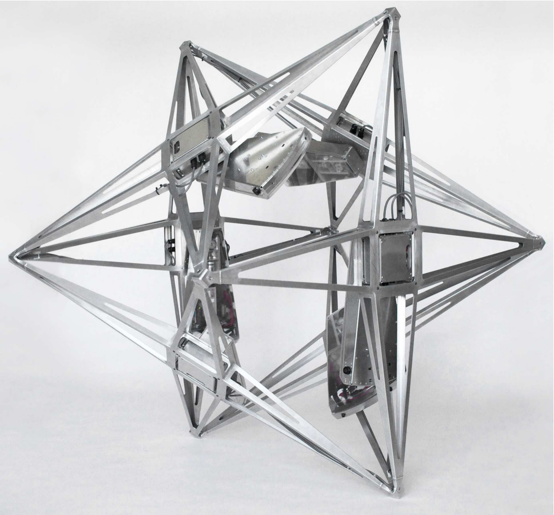


Figure 3. The cube balancing on one of its edges. When the cube body is placed on two of its tips, it has only one rotational degree of freedom left. This is called edge balancing (see Fig. 4).

varying the nominal angle of the modules. Since the multi-body system has fewer inputs than DOFs (each of the module DOFs is actuated, the cube body DOFs are not), it is an *underactuated mechanical system*, [38]. Overall, the system combines the challenges of nonlinear unstable dynamics with distributed control and networked communication, making it a rich platform for research in dynamics and control.

Enabled by its control system, the Balancing Cube is a dynamic sculpture: the cube body is kept in balance through the slight corrective movements of the six modules. No external system is required for balancing. Set on one of its corners, the cube can balance as long as its batteries last (four or more hours) or until someone pushes it over. With many peoples' understanding of balancing, it makes an ideal device for communicating key concepts of control engineering such as stability, feedback control, and cooperation to the general public.

The Balancing Cube was built at the Institute for Dynamic Systems and Control (IDSC) at ETH Zurich, and was completed in 2009. Since then, it has been demonstrated at public exhibitions and at an international control conference (see the sidebar "Balancing Cube on Tour"). To the best of the authors' knowledge, the cube presented in this article is the only cube to

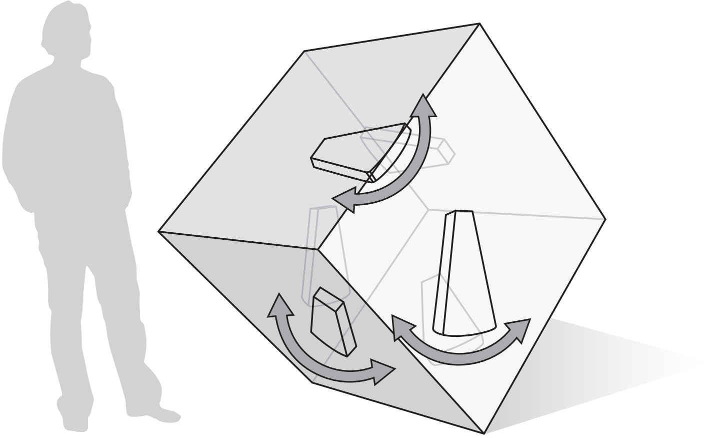


Figure 4. Edge balancing. The cube's orientation is the same as in the photo in Fig. 3: the cube stands on an edge, the modules on the front and back face are pointing down, and the bottom and top modules are rotated away from the downward position.

date that can balance autonomously on a corner. Another cube, which can balance on a fixed edge, is sold by Quanser Inc., [39]. Quanser's cube uses a single actuation mechanism to stabilize the cube on its edge.

This article explains the design, modeling, and control of the Balancing Cube, and demonstrates its balancing performance with experimental data. For the purpose of this article, all sensor data is exchanged between the agents. This way, the design of the control system can be posed as a centralized problem, which is addressed by separately designing a state estimator and a state-feedback controller. The resulting control and estimation algorithms are implemented in a distributed fashion; that is, they are running on all six agents in parallel with no hierarchical distinction among the agents. The full communication case presented herein serves as the baseline for studies with constrained or reduced communication. In [40, 41], two approaches are presented for the problem of state estimation and stabilization of the cube with reduced communication.

Whereas the state-feedback controller is designed using standard linear-quadratic regulator (LQR) design techniques, the state estimator is tailored to the specific problem. It exploits the facts that the cube has only rotational DOFs, and that measurements from multiple inertial sensors are available, to generate an estimate of the cube's tilt that is independent of the rigid body dynamics. In particular, the estimator provides a tilt estimate for whatever motion of the cube (slow or fast), and no assumption on near equilibrium configuration is made. The estimation algorithm only requires geometric system knowledge, namely the sensor locations on the cube. Since the algorithm does not rely on a dynamic system model, it is inherently robust to modeling errors or changes in the system (for example, in the mass or module configuration). The developed tilt estimation algorithm is applicable to any rigid body with only rotational degrees of freedom that is equipped with multiple inertial sensors.

This article focuses on the concepts and tools that were used to build and control the cube. The models that are presented herein include sensor models capturing the nonlinear dependency of the measurements on the system states (used for the state estimator design) and a linear model of the system dynamics (used for the design of the linear state-feedback controller). Nonlinear dynamic models and nonlinear controllers for similar 3D pendulum systems (but with different actuation mechanisms) can be found in [21]–[24], for example.

2. Design

In this section, the hardware design of the Balancing Cube's two key components is discussed: the cube body and the modules.

2.1 Cube body

The cube body is formed by six sheet metal constructions (one for each face), and eight corner parts (to connect the faces), see Fig. 5. Each of the cube's faces is an X-shaped welded construction fabricated from 1.5 mm sheet metal. The particular shape provides support for the modules, which are mounted in a square slot in the center of each face. The cube's corners are CNC-machined aluminum parts. On each corner part, the three adjacent faces are attached at right angles. The result is a single rigid body in the shape of a cube with an edge length of 1.2 m. The modular construction of the cube body allows for easy disassembly and transportation.

The total mass of the cube body (without the modules) is 14 kg, and represents a trade-off between weight and structural integrity. On the one hand, the structure must be light enough for the modules to manipulate the cube's overall center of gravity (CG). On the other hand, the structure must be strong enough to support the modules and to withstand repeated falls. The three principal moments of inertia of the cube body are roughly 5 $\mathrm{kg\,m^2}$ (each of the principal axes goes through the center of two opposing cube faces).

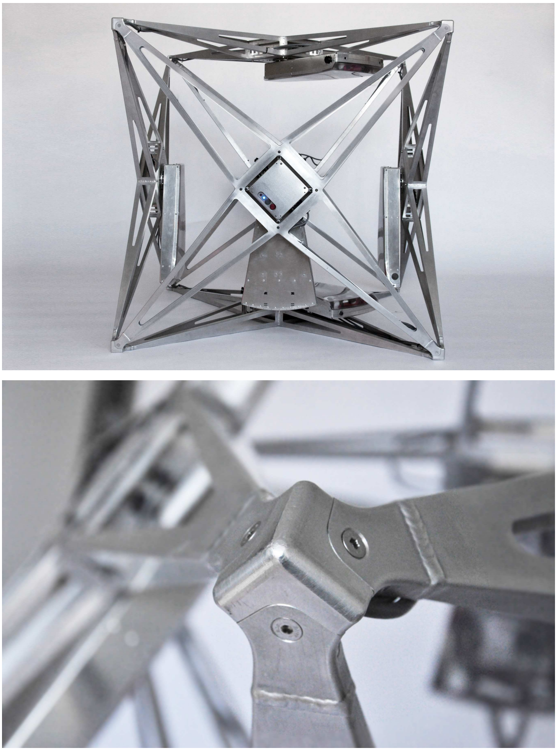


Figure 5. Detailed views of the cube's rigid body. The cube body has six faces and eight corners. The faces are formed by an X-shaped welded aluminum construction (top image). Three adjacent faces join at right angles at one of the eight identical corner parts (bottom).

2.2 Modules

The six modules carry the system's mechanical and electrical system components. At the same time, they constitute the actuation mechanisms that allow the cube body to balance. One of the cube's modules is shown in Fig. 6. Each module is composed of two parts: 1) a square-shaped component that is fixed to the cube's rigid body and, 2) a pie-shaped, eccentrically mounted "arm" that rotates relative to the cube body. Because of its motion relative to the rigid body, the former is called the *nonmoving part* and the latter the *moving part* of the module.

The cube is actuated by the rotating arms through: 1) gravitational moments caused by their displacement; and 2) reaction moments caused by their

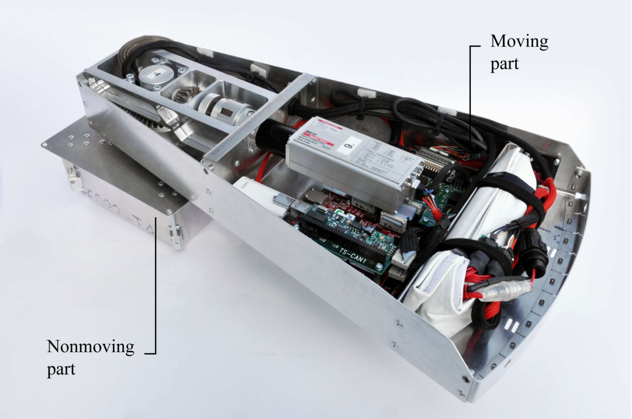


Figure 6. A module. Each module consists of a nonmoving part, which is rigidly mounted to the cube body, and a moving part (the pie-shaped arm), which rotates relative to the nonmoving part.

acceleration or deceleration. Other actuation mechanisms are conceivable: in [2, 21] for example, reaction wheels, proof masses, and fans are discussed as actuators to control a 3D pendulum. Rotating arms were chosen for the Balancing Cube mostly for aesthetic reasons.

Nonmoving part The nonmoving part connects the module's moving part to the cube body. It also houses inertial sensors and a user interface, see Fig. 7. The nonmoving part is rigidly mounted to the center of the cube's face so that the user interface faces outward. The nonmoving part has a mass of about 1.2 kg. The six nonmoving parts, together with the cube body, constitute one rigid body that must be balanced through the action of the moving parts.

The nonmoving part houses an inertial measurement unit (IMU) *(Analog Devices, ADIS16350)* with tri-axis accelerometer and tri-axis rate gyroscope. A lowpass filter onboard the IMU results in accelerometer and gyro noise standard deviations of $\sigma_{\text{acc}} = 0.04\,\text{m/s}^2$ and $\sigma_{\text{gyro}} = 0.0042\,\text{rad/s}$, respectively. From the IMU measurements, the tilt of the rigid body and its rate of change are estimated as described in the section "State Estimation."

The user interface consists of two LEDs and two push-buttons. Located on each face of the cube, the user interfaces are used to set the state of the system (for example, calibration mode or balance mode), and to turn the system on and off.

Connectors on two sides of the nonmoving part (see Fig. 7) connect all

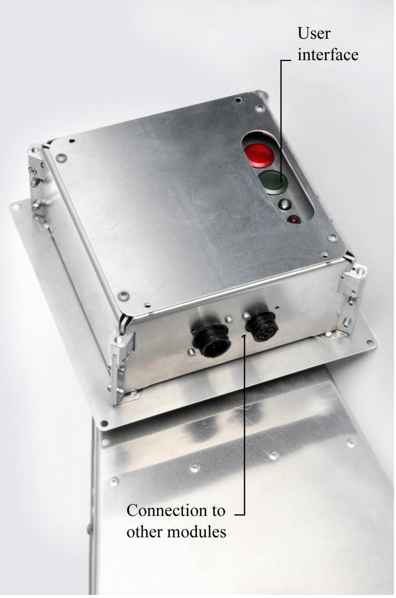


Figure 7. The module's nonmoving part. The user interface has two buttons and two LEDs indicating the status of the system. Cables running along the cube structure (not shown) connect the nonmoving parts and allow both the exchange of data and the synchronization of the power circuits. The inertial measurement unit (IMU) observing the cube motion sits inside the nonmoving part (not shown).

modules through wires running along the cube structure. One set of wires connects the modules' power circuits so that they can all be turned on/off at once by pressing a button on any module. A second set of wires forms the data network, allowing communication between the modules.

Moving part The motion of the cube body is influenced by actuating the moving part of the modules. In addition to the actuation mechanism, the moving part carries an absolute encoder, a computer, and a battery. All components are shown in Fig. 8. The moving part has a total mass of roughly 3.7 kg, however additional weights of up to 1.9 kg can be added to increase control authority. For the results presented in this article, the moving parts of the bottom modules (see Fig. 2) are equipped with an extra mass of 1.9 kg each.

The frame of the moving part is made of sheet metal. A removable cover of semi-transparent plastic protects the system components from dirt.

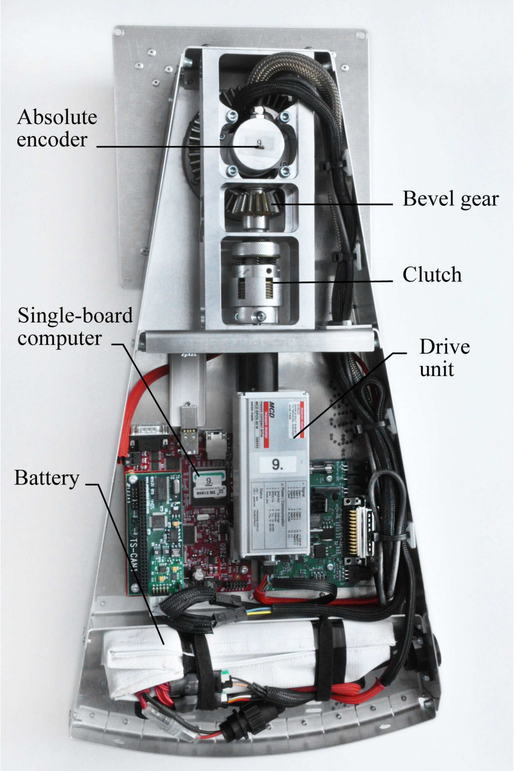


Figure 8. System components on the moving part of a module. The torque produced by the DC motor of the drive unit is applied between moving part and nonmoving part through the bevel gear. The clutch in the drive train protects the motor from mechanical damage. The absolute encoder measures the angle of the moving part relative to its mounting. All sensors are read by the single-board computer, which issues commands to the drive unit. The electronic connection to the components on the nonmoving part is through a slip ring (not shown). The module is powered by the lithium polymer battery.

Machined aluminum parts attach to the sheet metal frame and hold the actuation mechanism, which consists of a 60 W brushless DC motor with a planetary gear head (reduction 1:103) to drive the pinion of a bevel gear (reduction 1:3). The bevel gear is connected to the nonmoving part of the module and hence the cube body. The motor therefore rotates the moving part relative to the cube. A clutch is used in the drive train to protect the motor from mechanical damage (such as when the cube falls) and to protect users if they are unintentionally hit by a rotating module.

The DC motor and the gear head are part of a compact drive unit *(Maxon, MCD EPOS 60W)* that also includes a motor shaft encoder and a digital position and velocity control unit. In normal operation, the control unit is used in velocity mode to control the motor shaft velocity and, hence the angular velocity of the module.

The motor receives commands from a single-board computer (SBC) over a dedicated controller area network (CAN). Through the same interface, the SBC can also read out motor data, such as the shaft velocity or the motor current. The SBC *(embeddedARM, TS-7260)* has a 200 MHz ARM9 processor and consumes less than 1 W of power. In addition to the motor, it interfaces with all local sensors, the local user interface, and all other SBCs.

An absolute encoder *(Hengstler, AC 36)* with 12 bit resolution is attached to the module's axis of rotation and hence allows for absolute positioning of the module's moving part relative to its nonmoving part. It is connected to the SBC's serial peripheral interface (SPI).

The wires connecting the components on the nonmoving part to the SBC are routed through a slip ring *(Moog, AC6846)*. The IMU is connected to the SBC over SPI. The SBCs are connected with each other over a CAN separate from the CAN connecting the motor. The CAN used for the *inter-module communication* (IMC) is a broadcast network operating at a data rate of 500 kBit/s. The network allows the reliable exchange of all absolute encoder and IMU measurements between all modules every 10 ms.

The SBC handles the IMC and runs the estimation and control algorithms that enable the cube to balance. For programming and monitoring purposes, each SBC also has a wireless local area network (WLAN) module that allows for a connection to an external computer. A Linux operating system on the SBC allows easy handling of data and external access.

All components on the module are powered by a 6 cell lithium polymer battery *(FlightPower, EVO 20 3700 mAh 22.2 V)*. Customized electronics manage each module's power locally, including safety shut-off. Each module's battery voltage is additionally monitored by the SBC through its analog input. When fully charged, the cube can balance on one of its corners for more than four hours. The batteries can be charged through a connector on one side of the moving part.

3. Operation

The usual operation of the system is illustrated in Fig. 9. First, a human operator lifts the cube and brings it near the desired equilibrium. The cube recognizes its balancing mode, that is, which edge or corner it is standing on, and rotates its modules to the appropriate starting configuration. After a short calibration phase, the cube starts to balance autonomously. If the cube falls, for example, after it was pushed too hard, the operation cycle – lifting the cube, holding near equilibrium, balancing until disturbed – may be repeated.

The Balancing Cube is a standalone device (without external systems) that can operate anywhere there is solid ground. In order to lower the impact on the structure when the cube falls (such as when the batteries run low or a viewer pushes the structure too hard) the cube is usually balanced inside a foam ring. The complete setup is shown in Fig. 9.

In normal operation mode, the operator controls the cube through the dedicated user interfaces on the cube's faces, or by guiding or pushing the cube body (such as during the setup phase). The system uses its inertial sensors to respond. An external computer can be used for monitoring real-time data sent over WLAN.

4. Modeling

This section presents the models that are used in later sections for the design of the state estimation and control algorithms.

Linear dynamic models have proven sufficient for designing controllers that can stabilize the cube about an equilibrium. The system's nonlinear equations of motion are therefore omitted. Nonlinear dynamic models for similar 3D pendulum systems (with more straightforward actuation mechanisms than the rotating arms on the cube) are discussed in [2, 21]–[24]. The multi-body system of the cube is complicated enough so that it is arguable how to best represent it in terms of being comprehensible, manageable, and not error-prone – as a system of nonlinear differential equations or as a computer-based symbolic model, for example. Herein, a 3D multi-body model in Matlab/Simulink is presented, which is used for nonlinear simulations and to extract linearized dynamic models. For a one dimensional abstraction of the cube (an inverted pendulum being balanced by a single module; presented later in the sidebar "Why Are the Top Modules Used Less?"), the modeling procedure was verified by comparing the computer-based models to the analytically derived equations of motion.

The design of the global and nonlinear state estimator does not rely on the linear dynamics model; in fact, it does not rely on any dynamics model. The sensor models that are the basis of the state estimator design are algebraic equations expressing the sensor measurements as a (nonlinear) function of

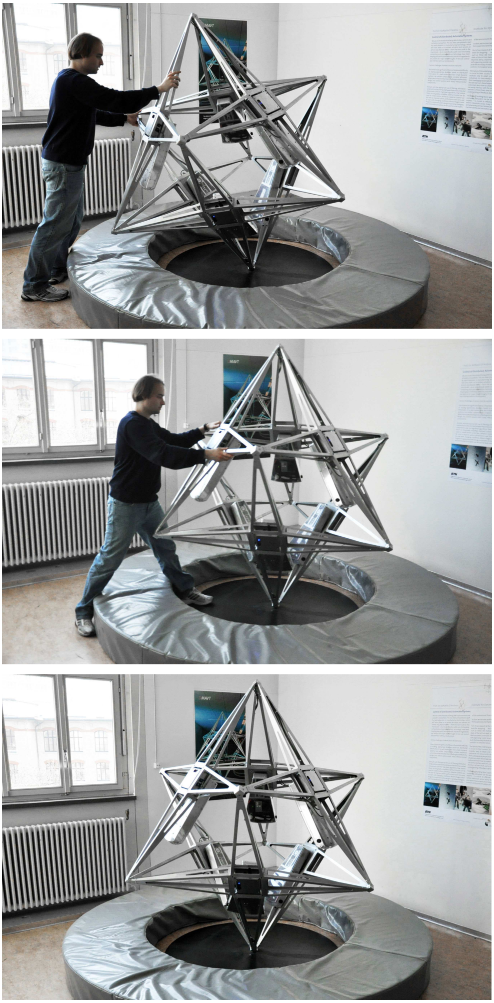


Figure 9. Operation of the Balancing Cube. This photo sequence, read from top to bottom, illustrates a typical operation sequence: lifting the cube, holding near equilibrium, and finally, balancing autonomously.

the system states. To state these sensor models, only geometric information about the system (namely the sensor locations on the cube body) is required.

4.1 Multi-body system

The Balancing Cube is a multi-body system, with the cube body and the modules as rigid bodies. The cube body and the nonmoving parts of the modules are treated as a single rigid body, which stands with either one or two of its tips in contact with the ground. The modules' moving parts are connected to the cube body through revolute joints. Because of its large mass, it can be assumed that the supporting points of the cube body do not slip, and that the cube, therefore, does not experience any translational motion. The support of the rigid body is hence modeled as a ball joint with three rotational DOFs for the case of corner balancing. When the cube balances on its edge, the second ground contact constrains two rotational DOFs. Hence the cube support is modeled as a revolute joint with one rotational DOF.

The cube's body is subject to gravity and moments generated by the actuation mechanisms (the rotating arms). The latter include gravitational moments due to displacement of the eccentric moving parts and reaction moments from accelerating or decelerating the arms. Centripetal forces from the rotation of the arms are negligible because of low angular rates in typical operation.

The coordinate frames shown in Fig. 10 are used to describe the orientation of the cube: $\hat{O}$ denotes the inertial frame of reference, and $\hat{B}$ denotes the cube body-fixed coordinate frame. The origin of $\hat{B}$ lies on the cube's balancing corner and its axes are along the cube's edges as depicted in Fig. 10. The origins of frames $\hat{O}$ and $\hat{B}$ coincide.

The rotation between the inertial frame $\hat{O}$ and the cube frame $\hat{B}$ is expressed by the rotation matrix ${}^{O}_{B}S$. For all rotation matrices, the notation from [42] is adopted, where the matrix ${}^{O}_{B}S$ describes the rotation of $\hat{B}$ relative to $\hat{O}$, and a vector quantity v given in frame $\hat{B}$, ${}^{B}v \in \mathbb{R}^3$, is expressed in frame $\hat{O}$ by ${}^{O}v = {}^{O}_{B}S\,{}^{B}v$.

In this article, the attitude of the rigid body is represented by Z-Y-X-Euler angles (yaw, pitch, roll) such that (see [42])

$$ {}^{O}_{B}S = S_{\mathrm{Z}}(\alpha)\, S_{\mathrm{Y}}(\beta)\, S_{\mathrm{X}}(\gamma), \tag{1} $$

with

$$ S_{\mathrm{Z}}(\alpha) := \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix},\ S_{\mathrm{Y}}(\beta) := \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}, $$

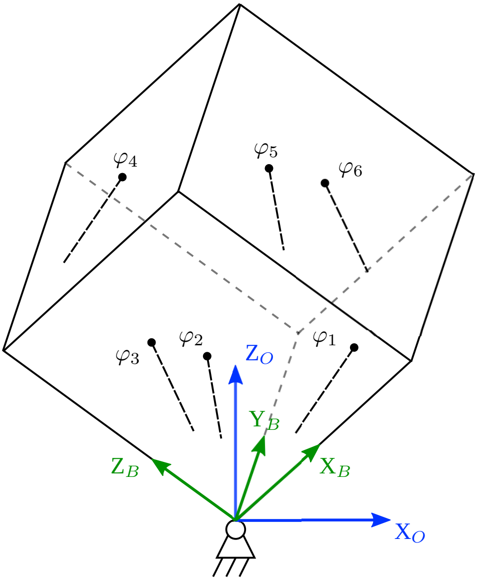


Figure 10. Coordinate frame definitions and zero module angles. Frame $\hat{O}$ (in blue with axes X_O–Y_O–Z_O, Y_O axis not shown and pointing inside the drawing plane) is the inertial frame of reference, and $\hat{B}$ (green, X_B–Y_B–Z_B) is the body-fixed frame. The cube is shown standing on its corner, where the corner coincides with the inertial frame origin. For edge balancing, the body frame Y-axis Y_B is the balancing axis. The zero directions of the module angles $\varphi_1, \varphi_2, \ldots, \varphi_6$ are indicated as the dashed black lines (pointing downward, toward the balancing corner), and the direction of positive rotation is such that the rotation vector points to the cube's center.

$$S_{\mathrm{X}}(\gamma) := \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix},$$

where α, β, and γ are the yaw, pitch, and roll Euler angles, respectively. Notice that the yaw angle α (capturing rotations about the gravity axis) is irrelevant for balancing since gravity does not excite any yaw motion. Therefore, the yaw angle is not part of the system state that is considered later for the controller and estimator design. Because yaw represents a DOF of the rigid body, it is introduced here nonetheless. The particular choice of the Euler angle order (Z-Y-X) will result in the yaw angle naturally dropping out in the derivation of the state estimator later. The pair of pitch and roll angle (β, γ) is denoted the *tilt* of the rigid body. When the cube balances on its edge, the body Y-axis is the axis of rotation; that is, the rotation angle is β, and $\gamma = 0$.

The modules' rotation vectors are orthogonal to the corresponding cube face and point to the cube's center. The angle φ_i is used to denote the rotation angle of module i relative to the cube body. The zero angles of φ_i are shown in Fig. 10. The full configuration of the multi-body system is described by the generalized coordinates

$$q = (\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \varphi_6, \alpha, \beta, \gamma). \tag{2}$$

The acceleration that is sensed by an IMU depends on the IMU mounting position and orientation on the rigid body. Because the IMU is a component of the nonmoving part of the module, its position and orientation are constant in the body frame $\hat{B}$. The positions of the six IMUs on the cube body are denoted by ${}^{B}p_i$. Each sensor measures accelerations and rotations in its local sensor frame $\hat{A}_i$. The IMU orientation is captured by the rotation matrix ${}^{A_i}_{B}S$. Since the mounting of the IMUs is known, ${}^{B}p_i$ and ${}^{A_i}_{B}S$ are known for all sensors.

4.2 3D simulation model

A multi-body model capturing the nonlinear system dynamics in 3D was built in SimMechanics, which is an extension of Matlab/Simulink for physics-oriented symbolic modeling of mechanical systems, [43]. A library was created that includes models of all system components, in particular, the cube body and the modules with their actuation mechanism and sensors. The library facilitates the assembly of models for different simulation scenarios (for example, corner or edge balancing) and allows for easy modification of the complete system or its subcomponents. A screenshot of a part of the model is shown in Fig. 11.

The SimMechanics model provides a convenient simulation platform for design and verification of the control and estimation algorithms. Furthermore, it is used to automatically generate linearized models about different equilibrium configurations, as described below. An earlier version of this model was used for feasibility studies and to support design decisions early in the project's development.

4.3 Linear dynamics model

Different static equilibrium configurations can be obtained for both edge and corner balancing by varying the nominal angles of the modules. Herein, the two basic configurations in Fig. 1 and 2 (for corner balancing), and in Fig. 3 and 4 (for edge balancing) are considered. The modeling procedure does not,

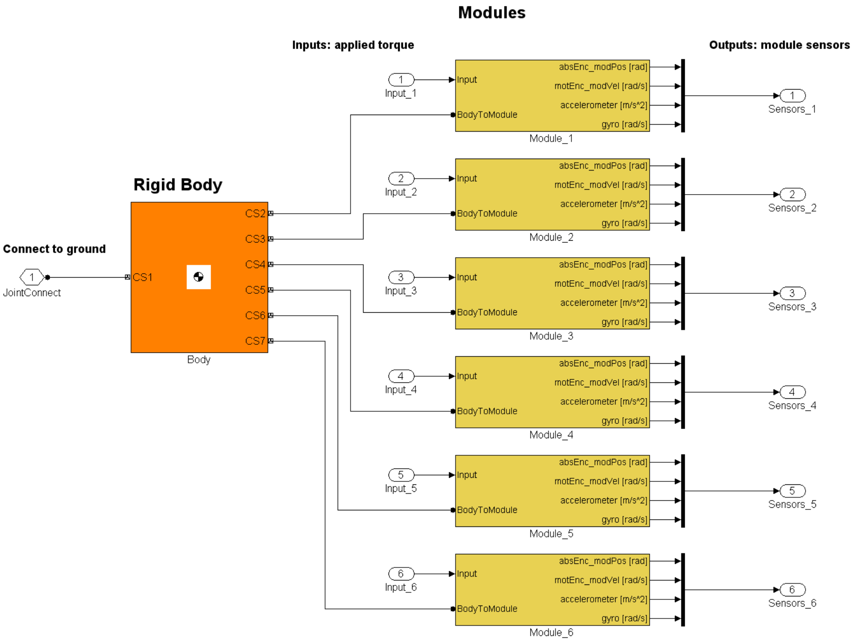


Figure 11. SimMechanics model. Shown is a screenshot of the highest model level with the cube body block (orange) and the six modules (yellow). Inputs to the modules are the motor torques, and the outputs are the sensor measurements. The cube's ground connection is either to a ball joint (for corner balancing) or a revolute joint (for edge balancing).

however, depend on the specific equilibrium, and can readily be applied in the same way for other equilibria. Due to the particular mass configuration of the cube body and the modules, the range of equilibria is limited. The maximum that the cube can tilt and still remain in equilibrium is discussed in the sidebar "What Is the Cube's Maximal Balancing Range?"

Expressed in generalized coordinates (2), the considered equilibrium for corner balancing is

$$q_0^{\mathrm{c}} = \Big(0, 0, 0, 0, 0, 0, \alpha, -\mathrm{atan}(1/\sqrt{2}), \frac{\pi}{4}\Big). \tag{3}$$

It corresponds to the cube standing upright (its body diagonal being parallel to the gravity axis) and all modules pointing downward as shown in Fig. 1 and 2. The yaw angle α is left unspecified and can be arbitrary. The edge balancing equilibrium configuration shown in Fig. 3 and 4 is given by

$$q_0^{\mathrm{e}} = \Big(0, 0, \frac{\pi}{2}, -\frac{\pi}{4}, 0, -\frac{3\pi}{4}, \alpha, -\frac{\pi}{4}, 0\Big). \tag{4}$$

The dynamics of the multi-body system about an equilibrium configuration (3) or (4) are described by the state-space model

$$\dot{x}(t) = A\,x(t) + B\,u(t) \tag{5}$$

with the system inputs $u(t) \in \mathbb{R}^6$ being torques applied to the modules, and $x(t) \in \mathbb{R}^n$ the state vector. The system states are the generalized coordinates of the multi-body system and their time derivatives; that is, the angles and angular velocities of the modules and the cube body. When the cube stands on an edge, it has one rotational DOF (the pitch angle β); when it is balancing on a corner, it has three rotational DOFs. The rotation about the gravitational axis can, however, be neglected for balancing and is removed from the state-space equations. The relevant state of the cube body is therefore fully characterized by the pitch and roll angle, β and γ, and their derivatives. Hence, $n = 14$ for edge balancing, and $n = 16$ for corner balancing. The physical meaning of states and inputs is summarized in tables 2 and 3.

Built-in functions in Matlab/Simulink are used to compute numerical values for the state space model (A, B) from the SimMechanics model. Since for the controller design procedure it makes no difference whether edge or corner balancing is considered, the general state space description (A, B) is used throughout this article. The open-loop poles of the system are listed in Table 4. The numerical values for the state space matrices (A, B) can be found in [44].

4.4 Nonlinear sensor model

Functional relations between measurements y and the system states x, which form the basis for developing the state estimation algorithms, are sought in this section. The obtained sensor model is linear for the absolute encoder and nonlinear for the accelerometer and rate gyroscope.

Accelerometer model The previous definitions of the inertial frame $\hat{O}$, the body frame $\hat{B}$, and the IMU frame $\hat{A}_i$ are used to express the acceleration ${}^{A_i}y_i^{\mathrm{acc}}$ measured by the tri-axis accelerometer on module i. The accelerometer located at the position ${}^{O}p_i$ measures the acceleration ${}^{O}\ddot{p}_i$ of the cube

Table 2. The states of the cube model (5). Roll angle and roll rate, x_{15} and x_{16}, are relevant only for corner balancing.

State variable	**Physical meaning**
x_1	φ_1, angle module 1
x_2	φ_2, angle module 2
$\vdots$	$\vdots$
x_6	φ_6, angle module 6
x_7	$\dot{\varphi}_1$, angular velocity module 1
x_8	$\dot{\varphi}_2$, angular velocity module 2
$\vdots$	$\vdots$
x_{12}	$\dot{\varphi}_6$, angular velocity module 6
x_{13}	β, cube pitch angle
x_{14}	$\dot{\beta}$, cube pitch rate
x_{15}	γ, cube roll angle (corner balancing only)
x_{16}	$\dot{\gamma}$, cube roll rate (corner balancing only)

Table 3. The inputs of the cube model (5).

Input variable	**Physical meaning**
u_1	torque at module 1
u_2	torque at module 2
$\vdots$	$\vdots$
u_6	torque at module 6

body at this position plus the gravity vector ${}^{O}g$ plus sensor noise – all in its local frame $\hat{A}_i$. That is,

$$ {}^{A_i}y_i^{\mathrm{acc}} = {}^{A_i}_{B}S\, {}^{B}_{O}S \left({}^{O}\ddot{p}_i + {}^{O}g \right) + {}^{A_i}w_i^{\mathrm{acc}}, \tag{6} $$

where ${}^{A_i}y_i^{\mathrm{acc}}$ is module i's accelerometer measurement (in m/s^2), and ${}^{A_i}w_i^{\mathrm{acc}}$ is measurement noise. The noise is assumed to be zero-mean, band-limited white noise with standard deviation σ_{acc}; that is, $\mathrm{E}\left[{}^{A_i}w_i^{\mathrm{acc}}\right] = 0$, $\mathrm{E}\left[{}^{A_i}w_i^{\mathrm{acc}} ({}^{A_i}w_i^{\mathrm{acc}})^T\right] = \sigma_{\mathrm{acc}}^2 I$, where $\mathrm{E}\left[\cdot\right]$ denotes the expected value and I denotes the identity matrix of appropriate dimensions. This noise model is reasonable for

Table 4. Open-loop poles. The linear cube model (5) has 16 poles for corner balancing and 14 poles for edge balancing. The poles were computed numerically from the SimMechanics models. The two unstable poles for corner balancing correspond to the two dimensions of the cube that need to be stabilized. The pole roughly at 0 for edge balancing is due to the fact that two modules are horizontal (the top modules in Fig. 4). It corresponds to an eigenmode with equally directed displacement of these two modules.

Corner balancing	**Edge balancing**
2.92	3.09
2.92	0.0
−2.92	−3.11
−2.92	−1.04
−0.505 ± 3.8i	−0.294 ± 3.09i
−0.505 ± 3.8i	−0.294 ± 3.14i
−0.54 ± 3.97i	−0.516 ± 4.15i
−0.362 ± 4i	−0.315 ± 4.53i
−0.341 ± 4.14i	−0.559 ± 0.207i
−0.341 ± 4.14i	–

many MEMS accelerometers once the bias has been removed, and if scaling and axes cross coupling errors are neglected, [45]. Biases in the state estimates resulting from sensor biases are compensated by using integral action in the control algorithm. This is explained later in the section “Control.”

From the identity ${}^{O}p_i = {}^{O}_{B}S\,{}^{B}p_i$ and the fact that ${}^{B}p_i$ is constant with time, it follows that

$$ {}^{O}\ddot{p}_i = {}^{O}_{B}\ddot{S}\ {}^{B}p_i, \tag{7} $$

where ${}^{O}_{B}\ddot{S}$ denotes the second derivative of the rotation matrix ${}^{O}_{B}S$ with respect to time. The matrix ${}^{O}_{B}\ddot{S}$ captures the rotational and centripetal acceleration terms of the cube rigid body motion. Using (7) and multiplying (6) with ${}^{B}_{A_i}S$ from the left yields

$$ {}^{B}y_i^{\text{acc}} = \tilde{S}\ {}^{B}p_i + {}^{B}g + {}^{B}w_i^{\text{acc}}, \tag{8} $$

where

$$ \tilde{S} := {}^{B}_{O}S\ {}^{O}_{B}\ddot{S} \tag{9} $$

combines the rotation and acceleration of the rigid body,

$$ {}^{B}g = {}^{B}_{O}S\ {}^{O}g \tag{10} $$

is the gravity vector in body coordinates, and

$$^{B}w_i^{\mathrm{acc}} = {}^{B}_{A_i}S \; {}^{A_i}w_i^{\mathrm{acc}} \tag{11}$$

is the noise vector rotated to the body frame. The mean and variance of the noise still satisfy $\mathrm{E}\,[{}^{B}w_i^{\mathrm{acc}}] = 0$ and $\mathrm{E}\,[{}^{B}w_i^{\mathrm{acc}}({}^{B}w_i^{\mathrm{acc}})^T] = \sigma_{\mathrm{acc}}^2 I$.

Equation (8) expresses an accelerometer measurement as a function of the rigid body dynamics (captured in $\tilde{S}$), the gravity vector in the body frame ^{B}g, and sensor noise. This relation is used in the section "State Estimation" to obtain an estimate of ^{B}g from multiple accelerometer measurements. By means of (10) and the representation (1) of ${}^{B}_{O}S$, an estimate of the cube tilt is then obtained.

Rate gyroscope model The six rate gyros measure the angular rate vector of the cube body: expressed in the body frame of reference $\hat{B}$,

$$^{B}y_i^{\mathrm{gyro}} = {}^{B}\omega + {}^{B}w_i^{\mathrm{gyro}}, \tag{12}$$

where $^{B}y_i^{\mathrm{gyro}}$ is the i-th rate gyro measurement (in rad/s) rotated to the body frame, $^{B}\omega$ is the angular rate vector in the body frame, and $^{B}w_i^{\mathrm{gyro}}$ is sensor noise with zero mean and variance $\mathrm{E}\,[{}^{B}w_i^{\mathrm{gyro}}({}^{B}w_i^{\mathrm{gyro}})^T] = \sigma_{\mathrm{gyro}}^2 I$.

The body rotation vector relates to the Euler angle rates $\dot{\alpha}$, $\dot{\beta}$, $\dot{\gamma}$ by (see [46], for example)

$$\begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix} = T(\beta, \gamma)\, {}^{B}\omega, \tag{13}$$

with the nonlinear transformation

$$T(\beta,\gamma) := \begin{bmatrix} 0 & \sin(\gamma)/\cos(\beta) & \cos(\gamma)/\cos(\beta) \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 1 & \sin(\gamma)\tan(\beta) & \cos(\gamma)\tan(\beta) \end{bmatrix}. \tag{14}$$

The transformation matrix (14) is nonsingular for the considered equilibria (3) and (4). The equations (12) and (13) are used later to obtain estimates of the pitch rate $\dot{\beta}$ and the roll rate $\dot{\gamma}$.

Absolute encoder model The sensor model of the absolute encoder is straightforward: each absolute encoder measures the corresponding module angle. Hence, the measurement y_i^{enc} (in rad) of module i's absolute encoder is given by

$$y_i^{\mathrm{enc}} = \varphi_i + w_i^{\mathrm{enc}}, \tag{15}$$

where w_i^{enc} is a random variable modeling the quantization error due to finite encoder resolution.

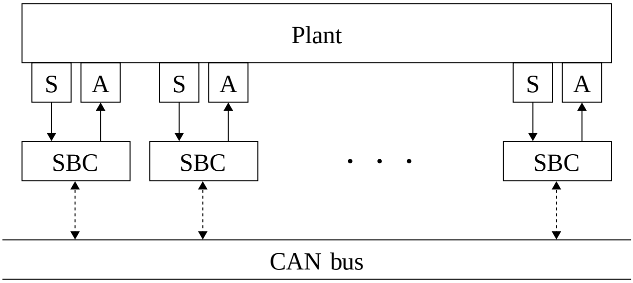


Figure 12. Abstraction of the networked control system. The blocks A and S denote actuator and sensor units. The single-board computer (SBC) runs estimation and control algorithms, and manages the communication with the other modules over the controller area network (CAN) bus.

5. Control System Architecture

The SBCs on the modules run the estimation and control algorithms, which enable the cube to balance. Based on its local sensor data and data communicated from the other modules over CAN, each module's SBC computes commands for its local actuator. The components of the networked control system are depicted in Fig. 12.

The algorithms implemented on each module comprise a state estimator and a state-feedback controller. Each module maintains an estimate $\hat{x}$ of the full system state x, which it computes from all available sensor data (local and IMC data). The controller uses this estimate to compute the actuator command. The block diagram in Fig. 13 represents the implementation of the feedback-control system.

A crucial question for the design of the estimation and control algorithms is what data is shared between the modules over CAN. Since CAN is a broadcast network, if one module sends data, the data can be received by all the others. For the results presented in this article, all modules share all their local sensor data (IMU and absolute encoder) over CAN. The capacity of the network is such that all modules can broadcast their sensor measurements every 10 ms, which is the time step of the feedback controllers K_i. Delays and losses in the transmission of sensor data are not taken into account in the design of the algorithms in this article.

With this communication scheme, the available sensor information on each module is identical. Therefore, the modules can run a copy of the same state estimator (see Fig. 13). Consequently, the inputs to the controllers K_i are also identical. Hence, state estimation and control design are treated

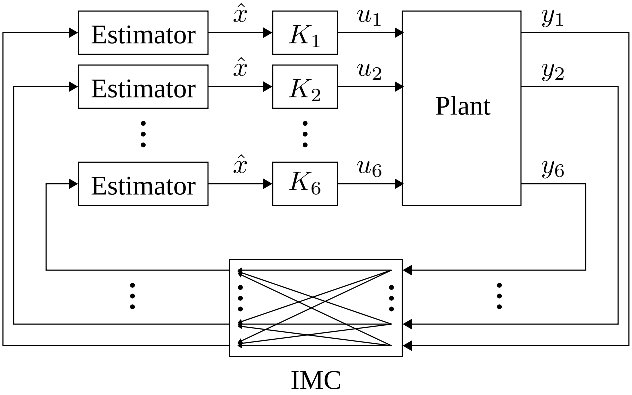


Figure 13. Distributed implementation of the control system. Each module i runs a state estimator and a controller K_i. The signal y_i combines all of module i's sensor measurements, and u_i denotes its control input. Different communication protocols can be implemented for the intermodule communication (IMC): for the design and results presented in this article, each module broadcasts its sensor measurements to all other modules.

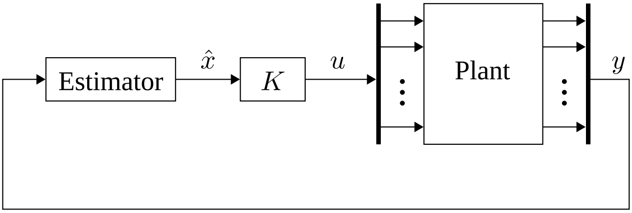


Figure 14. Centralized design problems. Since each module shares its sensor data with all other modules at every time step, the designs of the state estimator and the state-feedback controller K can be addressed as centralized problems. Therein, the state estimator has access to all sensor measurements $y = (y_1, \ldots, y_6)$, and the state-feedback controller K computes all system inputs $u = (u_1, \ldots, u_6)$.

as the centralized problem given in Fig. 14: the state estimator has access to *all* measurements and the controller K computes *all* system inputs. The distributed implementation of the feedback system in Fig. 13 is then straightforward: a copy of the estimator runs on each module, and the local controllers K_i are obtained from K by selecting the output corresponding to the local actuator.

The communication protocol considered herein represents the case of maximal information in terms of sensor data and serves as a baseline for distributed or event-based algorithms that cope with a reduced set of sensor data. Depending on what aspect of a distributed and networked control system is to be studied, different protocols and topologies can be implemented on the Balancing Cube by constraining the IMC (for example, reducing the average communication rate such as in [40, 41]).

6. State Estimation

This section addresses the design of the centralized state estimator shown in Fig. 14. Since the estimator is ultimately implemented on a digital computer, the estimator equations are expressed in discrete time. For this purpose, the discrete-time index $k \in \mathbb{N}$ is used: for a continuous-time signal $s(t)$, $s[k]$ denotes its value at time $t = kT_\mathrm{s}$, where T_s is the sampling time; thus, $s[k] = s(kT_\mathrm{s})$. Measurements are assumed to be acquired at the discrete-time instants k. The objective of this section is to develop an algorithm that computes an estimate $\hat{x}[k]$ of the system state $x[k]$ based on all sensor measurements at time k.

In contrast to many standard methods for state estimation, such as the Luenberger observer [47] or the Kalman filter [48], the approach to state estimation presented herein requires neither the knowledge of a dynamic system model (for instance, in the form of (5)) nor knowledge of the system inputs $u[k]$. Instead, estimates of all states are computed from the sensor measurements only. In addition to reducing the modeling effort, an immediate favorable consequence of this approach is that the state estimator is robust to modeling errors in the system dynamics or their intentional modification (for example, when weights are added to the modules). Furthermore, the estimator works both for slow and fast motion, and irrespective of the operation mode (such as corner balancing, edge balancing, or the cube being moved into starting position by an operator).

The design of the state estimator is addressed below separately for the module and the cube states.

6.1 Module states

The module angle φ_i is measured by the absolute encoder on module i according to the sensor model (15). Since the quantization error is negligible for the balancing application presented herein, filtering of the encoder measurement is not necessary. The encoder measurement y_i^enc is hence directly used as the module angle estimate; that is,

$$\hat{x}_i[k] = y_i^\mathrm{enc}[k], \quad i = 1, \ldots, 6. \tag{16}$$

An estimate of the module angular velocities $\dot{\varphi}_i$ may be readily obtained from the encoder measurements (15); for example, by numerical differentiation combined with appropriate lowpass filtering. Since the motors are operated with local velocity feedback ensuring fast tracking of velocity commands, actual estimates of the module velocities are not required for the state-feedback controller. This aspect is discussed in detail in the section "Control."

6.2 Cube states

Estimates of the cube's tilt (β, γ) and the tilt rates $(\dot{\beta}, \dot{\gamma})$ are obtained from the measurements of the six IMUs on the cube body, each of which includes a tri-axis accelerometer and rate gyro. In a first step, the accelerometer measurements are used to generate an estimate of the tilt angles that is independent of the rigid body motion. This estimate is fused in a second step with an estimate of the tilt rates, which itself is obtained from the rate gyro measurements.

Tilt estimate from accelerometers An estimate of the cube's tilt is obtained by the following approach: first, an unbiased estimate ${}^{B}\hat{g}$ of the gravity vector ${}^{B}g$ in the body frame is derived as a linear combination of all accelerometer measurements based on the model (8). Second, the *accelerometer-based tilt estimate* $(\hat{\beta}^{\mathrm{acc}}[k], \hat{\gamma}^{\mathrm{acc}}[k])$ is constructed from the gravity vector estimate using (10) and the representation (1) for the rotation matrix ${}^{B}_{O}S$.

In static conditions, a single tri-axis accelerometer mounted on a rigid body is enough to measure the gravity vector ${}^{B}g$ in body frame. In fact, from (8) it can be seen that, for static conditions where ${}^{O}_{B}\ddot{S} = 0$ and hence $\tilde{S} = 0$, each individual accelerometer measurement ${}^{B}y_i^{\mathrm{acc}}$ is an unbiased estimate of the gravity vector ${}^{B}g$. The difficulty in the context of the Balancing Cube arises from the fact that the rigid body moves during balancing, and therefore $\tilde{S} \neq 0$ in general.

The tilt estimation method presented herein makes use of multiple accelerometers mounted on the same rigid body. Exploiting the kinematics of the rigid body with only rotational DOFs and the knowledge of the different sensor locations allows one to compensate for the dynamic terms of the rigid body motion in the accelerometer measurements. The algorithm works for any rigid body that has only rotational DOFs and measurements from multiple tri-axis accelerometers. To state the method for the general case, the constant N is used below to denote the number of sensors (for the cube, $N = 6$). The algorithm was first presented in [49].

Gravity vector estimate. The first objective is to obtain an estimate ${}^B\hat{g}$ of the gravity vector from all accelerometer measurements ${}^By_i^{\mathrm{acc}}, i = 1, \dots, N$ that is optimal in a least-squares sense. For notational convenience, all N measurements (8) are combined into one matrix equation,

$$Y = XP + W, \tag{17}$$

$$Y := \begin{bmatrix} {}^By_1^{\mathrm{acc}} & {}^By_2^{\mathrm{acc}} & \dots & {}^By_N^{\mathrm{acc}} \end{bmatrix} \qquad \in \mathbb{R}^{3\times N}, \tag{18}$$

$$X := \begin{bmatrix} {}^Bg & \tilde{S} \end{bmatrix} \qquad \in \mathbb{R}^{3\times 4}, \tag{19}$$

$$P := \begin{bmatrix} 1 & 1 & \dots & 1 \\ {}^Bp_1 & {}^Bp_2 & \dots & {}^Bp_N \end{bmatrix} \qquad \in \mathbb{R}^{4\times N}, \tag{20}$$

$$W := \begin{bmatrix} {}^Bw_1^{\mathrm{acc}} & {}^Bw_2^{\mathrm{acc}} & \dots & {}^Bw_N^{\mathrm{acc}} \end{bmatrix} \qquad \in \mathbb{R}^{3\times N}, \tag{21}$$

where Y combines all accelerometer measurements, X is the matrix of unknown parameters, P is the matrix of known parameters (the sensor locations), and W combines all accelerometer noise vectors, with $\mathrm{E}\,[W] = 0$ and $\mathrm{E}\,[W^TW] = 3\sigma_{\mathrm{acc}}^2 I$. Notice that Y, X, and W in (17) are time varying, whereas P is constant. The time index k is omitted for ease of notation.

In addition to the sought gravity vector ${}^Bg$, the unknown matrix X also contains the matrix $\tilde{S}$, which captures the second derivatives of the rigid body motion according to (9). In [49], a method is presented for optimally estimating the entire matrix X. To shorten the exposition, only the results for optimally estimating the gravity vector ${}^Bg$ are presented here.

An unbiased estimate ${}^B\hat{g}$ of the gravity vector ${}^Bg$ is sought such that the 2-norm of the estimation error is minimized; that is,

$${}^B\hat{g} = \arg\min_{{}^B\hat{g}} \mathrm{E}\,\left[\|{}^B\hat{g} - {}^Bg\|_2^2\right] \quad \text{subject to} \quad \mathrm{E}\,\left[{}^B\hat{g}\right] = {}^Bg, \tag{22}$$

where $\|\cdot\|_2$ denotes the vector 2-norm. The estimate ${}^B\hat{g}$ is restricted to linear combinations of the measurements Y; that is, a vector $\lambda \in \mathbb{R}^N$ is sought for ${}^B\hat{g} = Y\lambda$. This approach yields a straightforward implementation: at each time step, the estimate ${}^B\hat{g}$ is obtained by a single matrix-vector multiplication. The following proposition, the proof for which can be found in [49, Lemma 2.2], states the optimal unbiased linear estimate of the gravity vector.

PROPOSITION 1 Let the matrices $P \in \mathbb{R}^{4\times N}$ and $Y \in \mathbb{R}^{3\times N}$ be given and satisfy $Y = XP + W$ with unknown matrix $X = [{}^Bg\ \tilde{S}] \in \mathbb{R}^{3\times 4}$ and the matrix random variable $W \in \mathbb{R}^{3\times N}$ with $\mathrm{E}\,[W] = 0$, $\mathrm{E}\,[W^TW] = \sigma_W^2 I$.

Assuming P has full row rank, the (unique) minimizer $\lambda^* \in \mathbb{R}^{N\times 1}$ of

$$\min_{\lambda} \mathrm{E}\left[\|Y\lambda - {}^{B}g\|_2^2\right] \quad \text{subject to} \quad \mathrm{E}\left[Y\lambda\right] = {}^{B}g \tag{23}$$

is given by $\Lambda^* = [\,\lambda^* \; \Lambda_2^*\,] = P^T(PP^T)^{-1}$. ■

Applying Proposition 1 and reintroducing time index k for all time-variant quantities yields the gravity vector estimate ${}^{B}\hat{g}[k]$,

$${}^{B}\hat{g}[k] = Y[k]\,\lambda^*. \tag{24}$$

The *optimal fusion vector* λ^* is constant and entirely defined by the geometry of the problem (through P).

In Proposition 1 it is assumed that P has full row rank. In order to obtain a physical interpretation of this assumption, consider the case where P does not have full row rank. Then, there exists a nontrivial linear combination of the rows of P; that is,

$$\text{there exists } \; \xi \neq 0 \in \mathbb{R}^4 \; \text{ such that } \; \xi_1\, p_{\mathrm{X}} + \xi_2\, p_{\mathrm{Y}} + \xi_3\, p_{\mathrm{Z}} + \xi_4\, \mathbf{1} = 0, \tag{25}$$

where $p_{\mathrm{X}}^T, p_{\mathrm{Y}}^T, p_{\mathrm{Z}}^T \in \mathbb{R}^{1\times N}$ are the last three rows of P (the vectors of X, Y, and Z-coordinates of all sensor locations) and $\mathbf{1}^T \in \mathbb{R}^{1\times N}$ is the vector of all ones. Expression (25) is equivalent to the statement

$$\begin{aligned} &\text{there exists } \; \xi \neq 0 \in \mathbb{R}^4 \\ &\text{such that } \; \xi_1\, {}^{B}p_{\mathrm{X},i} + \xi_2\, {}^{B}p_{\mathrm{Y},i} + \xi_3\, {}^{B}p_{\mathrm{Z},i} = -\xi_4 \; \text{ for all } i = 1, \ldots, N \end{aligned} \tag{26}$$

where ${}^{B}p_{\mathrm{X},i}, {}^{B}p_{\mathrm{Y},i}, {}^{B}p_{\mathrm{Z},i}$ denote the X, Y, and Z-coordinate of the i-th sensor location in the body frame. Since the equation $\xi_1 x + \xi_2 y + \xi_3 z = -\xi_4$ defines a plane in (x, y, z)-space, condition (26) is equivalent to *all* N sensors lying on the same plane. Therefore, the full row rank condition on P is satisfied if and only if *not* all sensors lie on the same plane. Moreover, since three points always lie on a plane, this result implies that at least four tri-axis accelerometers are required for the method presented herein. For the cube, the full row rank assumption of P is satisfied.

The gravity vector estimate (24) can be shown to be independent of the rigid body dynamics (captured in $\tilde{S}$) as follows: from the singular value decomposition of the parameter matrix P,

$$P = U\,[\,\Sigma \quad 0\,] \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U\Sigma V_1^T, \tag{27}$$

with $U \in \mathbb{R}^{4\times 4}$ unitary, $\Sigma \in \mathbb{R}^{4\times 4}$ diagonal, $V_1 \in \mathbb{R}^{N\times 4}$, $V_2 \in \mathbb{R}^{N\times(N-4)}$, and $V = [V_1\, V_2]$ unitary, it can be verified that $\Lambda^* = V_1\Sigma^{-1}U^T$. With this result and using the partition $U^T = [U_1^T\, U_2^T]$, $U_1^T \in \mathbb{R}^{4\times 1}$, $U_2^T \in \mathbb{R}^{4\times 3}$, the gravity vector estimate can be written as

$$\begin{aligned}
{}^B\hat{g} &= Y\lambda^* = XP\lambda^* + W\lambda^* \\
&= [\,{}^Bg \quad \tilde{S}\,]\, \underbrace{U\Sigma V_1^T}_{P}\, \underbrace{V_1\Sigma^{-1}U_1^T}_{\lambda^*} + W\lambda^* \\
&= [\,{}^Bg \quad \tilde{S}\,] \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} U_1^T + W\lambda^* \\
&= [\,{}^Bg \quad \tilde{S}\,] \begin{bmatrix} 1 \\ 0 \end{bmatrix} + W\lambda^* = {}^Bg + W\lambda^*.
\end{aligned} \tag{28}$$

Obviously, the matrix $\tilde{S}$ does not appear in the estimate; that is, the gravity vector observation is not affected by any motion of the rigid body. As expected, the sensor noise W does enter the estimation equation.

Tilt (pitch and roll) estimate. With the estimate ${}^B\hat{g}$ of the gravity vector in the body frame, (10) can be used to compute an estimate of the rigid body tilt (β, γ), since the direction of the gravity vector in the inertial frame is known. Inserting the representation (1) for ${}^O_BS$, and ${}^Og = [0\ 0\ g_0]^T$, with gravity constant g_0, (10) can be rewritten as

$${}^Bg = S_{\mathrm{X}}^T(\gamma)\, S_{\mathrm{Y}}^T(\beta)\, S_{\mathrm{Z}}^T(\alpha)\ {}^Og = g_0 \begin{bmatrix} -\sin\beta \\ \sin\gamma\cos\beta \\ \cos\gamma\cos\beta \end{bmatrix}. \tag{29}$$

Given the estimate of the gravity vector (24), the *accelerometer-based tilt estimate* at time k is

$$\hat{\beta}^{\mathrm{acc}}[k] = \mathrm{atan2}\left(-{}^B\hat{g}_x[k], \sqrt{{}^B\hat{g}_y^2[k] + {}^B\hat{g}_z^2[k]}\right), \tag{30}$$

$$\hat{\gamma}^{\mathrm{acc}}[k] = \mathrm{atan2}\left({}^B\hat{g}_y[k], {}^B\hat{g}_z[k]\right), \tag{31}$$

where atan2 is the four-quadrant inverse tangent. Note that the gravity constant g_0 does not need to be known and, hence, the estimator does not to be calibrated for it. The estimator requires only knowledge of the accelerometer locations on the cube body.

Tilt rate estimate from rate gyros Estimates of the tilt rates are obtained from the rate gyro measurements (12) and the transformation (13). First, an estimate ${}^{B}\hat{\omega}[k]$ of the rotation vector ${}^{B}\omega[k]$ at time k is computed by averaging the available rate gyro measurements,

$$
{}^{B}\hat{\omega}[k] = \frac{1}{6}\sum_{i=1}^{6} {}^{B}y_i^{\text{gyro}}[k]. \tag{32}
$$

Given that the rate gyro sensors have identical noise variance, the average represents the best unbiased linear estimate of ${}^{B}\omega[k]$ minimizing the mean squared error. Using the transformation (13), an estimate of yaw, pitch and roll rates is given by

$$
\begin{bmatrix} \hat{\dot{\alpha}}[k] \\ \hat{x}_{14}[k] \\ \hat{x}_{16}[k] \end{bmatrix} = \begin{bmatrix} \hat{\dot{\alpha}}[k] \\ \hat{\dot{\beta}}[k] \\ \hat{\dot{\gamma}}[k] \end{bmatrix} = T\big(\hat{\beta}[k-1], \hat{\gamma}[k-1]\big)\, {}^{B}\hat{\omega}[k], \tag{33}
$$

where the transformation matrix $T(\cdot,\cdot)$ is evaluated at the yet undefined estimates $\hat{\beta}[k-1]$ and $\hat{\gamma}[k-1]$ of pitch and roll angle at the previous time step. They are made precise in the next subsection. The estimate of the yaw rate $\hat{\dot{\alpha}}[k]$ is not required for the balancing application presented in this article.

Sensor fusion In order to further reduce the noise level of the tilt estimate, the accelerometer-based tilt estimate (30), (31) is fused with the integrated estimate of the tilt rates (33). The *tilt estimate* $(\hat{\beta}[k], \hat{\gamma}[k])$ is obtained from a linear combination of accelerometer- and gyro-based estimates according to

$$
\hat{x}_{13}[k] = \hat{\beta}[k] = \kappa_1 \hat{\beta}^{\text{acc}}[k] + (1-\kappa_1)\big(\hat{\beta}[k-1] + T_{\text{s}}\hat{\dot{\beta}}[k]\big), \tag{34}
$$

$$
\hat{x}_{15}[k] = \hat{\gamma}[k] = \kappa_2 \hat{\gamma}^{\text{acc}}[k] + (1-\kappa_2)\big(\hat{\gamma}[k-1] + T_{\text{s}}\hat{\dot{\gamma}}[k]\big), \tag{35}
$$

where T_{s} is the sampling time and κ_1 and κ_2 are tuning parameters, which, given the noise specifications of accelerometers and rate gyros, can be chosen to minimize the estimation error variance.

Experimental validation The estimator for the cube states is validated in [49] using a camera-based global positioning system, which tracks the position and orientation of the cube body with sub-millimeter precision. The main experimental results from [49] are restated here. For further details, the reader is referred to the original publication.

Figure 15 shows the accelerometer-based tilt estimate (30), (31) in comparison to the camera-based reference measurement. For this experiment, the cube was moved manually about the nominal corner balancing equilibrium (3). Additionally, the tilt estimate is included that would result if only a single tri-axis accelerometer was used to observe the gravity vector (instead of (24), the accelerometer measurement (8) is used directly as an estimate of the gravity vector). The data shows that, when the cube is relatively static (from 45s to 50s), the single-accelerometer-based estimate is satisfactory. However, when the cube body is moving fast, the estimate suffers from the dynamic terms that act as disturbances to the single-accelerometer-based estimator. The experimental data illustrates the result discussed in (28), namely that the presented (multi) accelerometer-based tilt estimator compensates for the rigid body dynamics.

Figure 16 shows the comparison of the improved tilt estimates (34) and (35), which are based on accelerometer and rate gyro data, to the camera-based reference during autonomous balancing of the cube.

6.3 Summary: The complete state estimator

The complete state estimator is given by the equations (16), (24), and (30)–(35). It is summarized in the block diagram of Fig. 17. Memory is present only in the sensor fusion in (34) and (35); hence, the estimator has only two states. The estimator is nonlinear because of the nonlinear transformations in (30), (31), and (33).

None of the estimator equations in Fig. 17 depend on the system dynamics. The only assumption on the rigid body is that it is pivoting (it has only rotational DOFs). Hence, the estimator works irrespective of the system dynamics and, in particular, for slow and fast motion, for different mass configurations, and for both edge and corner balancing.

7. Control

The design of the centralized controller K shown in Fig. 14 is described in this section. The controller is designed as a linear-quadratic regulator (LQR) with additional integrator feedback on the module angles. The controller design is based on the model (5). The main control objective is the stabilization of (5); that is, to balance the cube about the corresponding equilibrium (3) or (4).

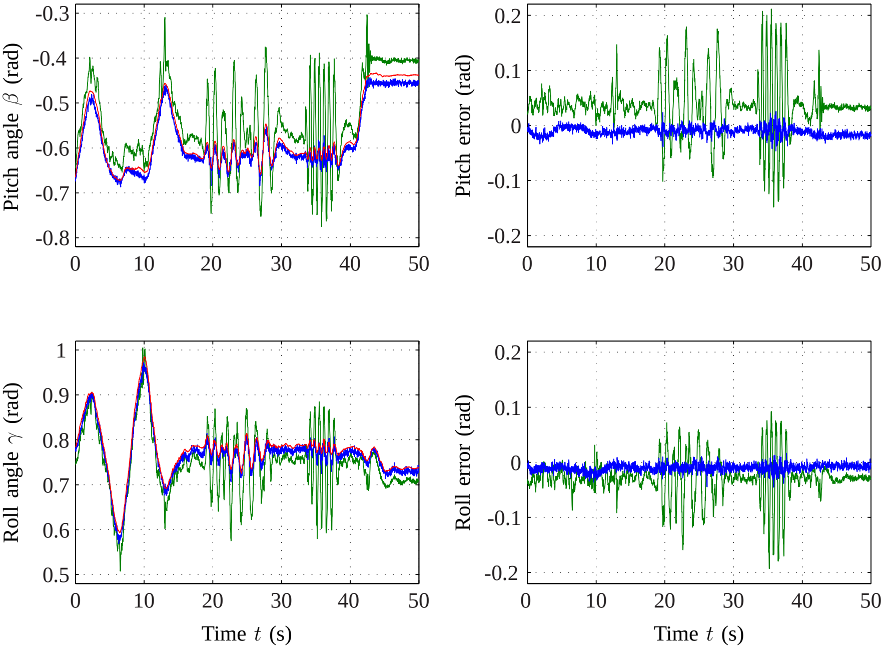


Figure 15. Verification of the accelerometer-based tilt estimator. For this experiment, the cube was moved manually around the nominal equilibrium (3) ($\beta_0^{\mathrm{c}} = -0.615$ and $\gamma_0^{\mathrm{c}} = 0.785$). The pitch and roll estimates $\hat{\beta}^{\mathrm{acc}}$ and $\hat{\gamma}^{\mathrm{acc}}$ are shown in the graphs on the left (in blue) with their camera-based reference (red). For comparison, the graph in green is the tilt estimate that results if a single tri-axis accelerometer measurement is used directly as an estimate of the gravity vector (instead of the fusion equation (24)). The corresponding error signals are shown on the right. Clearly, the multi-accelerometer-based estimator outperforms the single-accelerometer one, especially when the cube is being moved fast. The static biases visible in the estimation error signals result from biases in the accelerometers and/or in the camera system. The biases are, however, irrelevant for the balancing application, since biases in the state estimates are compensated by integral action in the controller. This effect is analyzed in the sidebar "What Is the Effect of Integral Action in the Controller?" (The experimental data is taken from [49].)

The system (5) captures the linearized pitch and roll dynamics of the cube; the controller thus locally stabilizes pitch and roll. Control of the yaw angle is not considered herein (the yaw motion is negligible in typical operation of the cube). The problem of simultaneously stabilizing yaw, pitch, and roll angles of a 3D pendulum is addressed in [22]–[24].

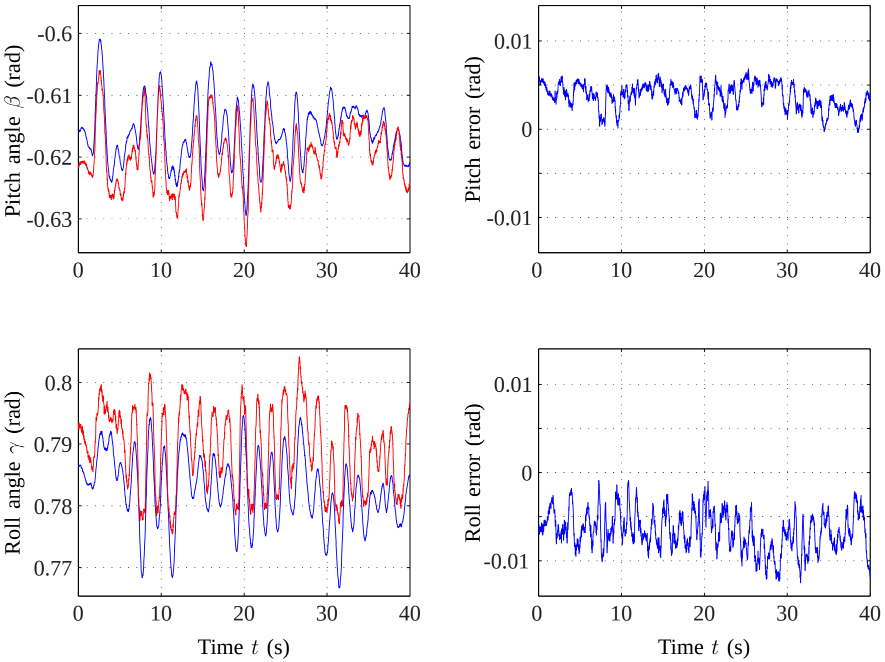


Figure 16. Verification of the tilt estimator. The data was taken while the cube was balancing about the nominal equilibrium (3) ($\beta_0^c = -0.615$ and $\gamma_0^c = 0.785$). The pitch and roll estimates $\hat{\beta}$ and $\hat{\gamma}$, resulting from fusing accelerometer and rate gyro measurements, are shown on the left (blue) with the camera-based reference (red). The graphs on the right show the corresponding error signals. Constant biases in the state estimates are compensated by integral action in the control algorithm. (The experimental data is taken from [49].)

The drive units are operated in velocity mode; that is, the motor shaft angular velocity is controlled locally on each drive unit. When neglecting gear backlash, the shaft velocity is proportional to the module velocity by the gear ratio; hence the feedback on the drive units is treated as feedback on the states $(x_7, \ldots, x_{12})$. The corresponding controllers are denoted by K_{loc}. Their reference input v is computed by the controller K. This cascaded control architecture is shown in Fig. 18. It consists of the *inner-loop controllers* K_{loc} and the *outer-loop controller* K. The inner-loop controllers K_{loc} operate at an update rate of 1 kHz, whereas the outer loop runs at rate of 100 Hz. The parameters of K_{loc} are tuned using a software tool provided by the manufacturer of the drive unit.

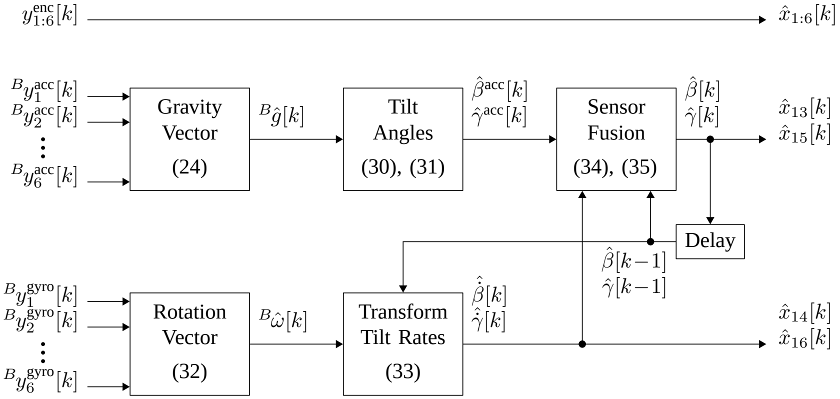


Figure 17. The state estimator. All encoder, accelerometer, and rate gyro measurements are input to the estimator; its outputs are the estimates of all states that are required for feedback control (estimates of the module velocities $x_{7:12}[k]$ are not required as is discussed in the section "Control").

In contrast to designing a controller for the system (5) directly (that is, a controller that computes the torque inputs u), the cascaded architecture with fast local velocity feedback reduces the complexity of the controller design problem. In model (5), it is assumed that the torque at the module can be controlled directly. In reality, however, only the torque at the motor can be controlled. This is translated to the torque at the module in a nontrivial way through a transmission system, which involves nonlinearities such as kinetic and static friction and backlash. The application of high-gain velocity feedback mitigates the effect of the nonidealities in the actuation mechanism and allows one to abstract them away for the design of the controller K.

7.1 Simplified model incorporating local feedback loops from time scale separation

The design of the outer-loop controller K requires a system model that incorporates the effect of the inner loops (represented by the dashed block $(\tilde{A}, \tilde{B})$ in Fig. 18). To avoid modeling the details of the local controllers K_{loc}, the motor and its local controller are abstracted as a system that achieves a commanded module velocity sufficiently fast. In fact, the ideal case of infinitely fast feedback is considered. This approximation is legitimate as long as the inner control loop operates sufficiently faster than the outer loop (the tracking performance of the inner loop is discussed in the section "Experiments"). The method described in the sidebar "Time Scale Separation Algorithm" is used to compute a model of the system (5) that incorporates the feedback

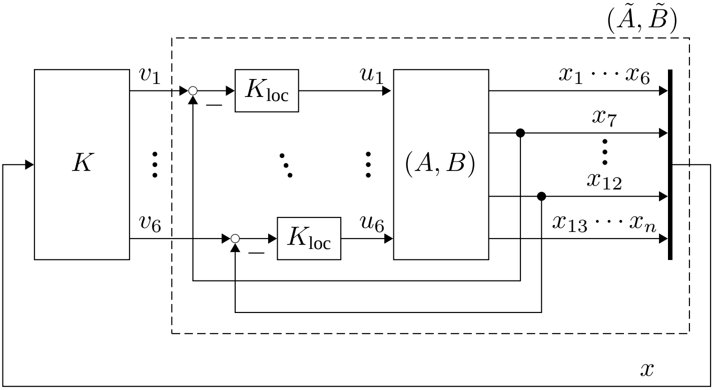


Figure 18. The cascaded control architecture. The block (A, B) denotes the cube model (5) and the blocks K_{loc} represent the local velocity controllers on each drive unit. The dashed block combines the plant with the local feedback loops. A simplified model $(\tilde{A}, \tilde{B})$ for this block is derived in the sidebar "Time Scale Separation Algorithm" with the assumption of high-gain controllers K_{loc}. The controller K is the centralized state-feedback controller that is also shown in Fig. 14. When comparing Fig. 14 and 18, notice that the inner feedback loops are not shown in Fig. 14; they can be thought of as included in the block *Plant*. Furthermore, the *Estimator* block is omitted here, since a plant with state output is assumed for the purpose of controller design.

on the module velocities. The obtained model is a discrete-time model with the sampling time of the outer-loop controller, $T_{\mathrm{s}} = 0.01\,\mathrm{s}$,

$$\underbrace{\begin{bmatrix} x_{1:6}[k+1] \\ x_{7:12}[k+1] \\ x_{13:n}[k+1] \end{bmatrix}}_{x[k+1]} = \underbrace{\begin{bmatrix} I & 0 & 0 \\ 0 & 0 & 0 \\ \tilde{A}_{31} & \tilde{A}_{32} & \tilde{A}_{33} \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} x_{1:6}[k] \\ x_{7:12}[k] \\ x_{13:n}[k] \end{bmatrix}}_{x[k]} + \underbrace{\begin{bmatrix} T_{\mathrm{s}}\, I \\ I \\ \tilde{B}_3 \end{bmatrix}}_{\tilde{B}} v[k] \tag{36}$$

where $x_{i:j} := (x_i, \ldots, x_j)$. For easier reference below, the notation from sidebar "Time Scale Separation Algorithm" is adopted; that is, the module velocity states $x_{7:12}$ are denoted by x_{f} (f for "fast") and the remaining states by x_{s} (s for "slow"). Notice from (36) that $x_{\mathrm{f}}[k+1] = v[k]$ (the module velocities are equal to the previously commanded reference), which corresponds to the assumption of ideal feedback loops. By using this approximation in the controller, no measurements or estimates of the module velocities are required. Estimates for the states x_{s} are available from the state estimator shown in Fig. 17.

7.2 State-feedback controller design

The controller K is obtained from a discrete-time LQR design. In LQR design (see [50], for example), a quadratic cost function involving weights on system states x and system inputs v is minimized, which allows one to trade off control performance with control effort. The resulting optimal controller is a static feedback gain.

To ensure zero steady-state error of the module angles, the plant (36) is first augmented with integrator states on the module angles; that is, the system model used for the LQR design is

$$\begin{bmatrix} x[k+1] \\ x_{\text{int}}[k+1] \end{bmatrix} = \begin{bmatrix} \tilde{A} & 0 \\ [T_{\text{s}} I_{6\times 6}\ 0] & I \end{bmatrix} \begin{bmatrix} x[k] \\ x_{\text{int}}[k] \end{bmatrix} + \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} v[k], \tag{37}$$

where $x_{\text{int}}[k]$ are the augmented integrator states, and $I_{6\times 6}$ is the 6-by-6 identity matrix. The augmented integrator states are eventually implemented in the controller. In addition to ensuring that the module angles are, on average, at their set points, any offset in the cube tilt estimates (for example, due to sensor bias or imperfect calibration) is compensated by the integrators. This property of the control system is analyzed in detail in the sidebar “What Is the Effect of Integral Action in the Controller?”

In addition to the usual weights on states and system inputs in LQR control, the difference in the control commands $v[k]-v[k–1]$ is also penalized. This is motivated by the special structure of the model (36): penalizing the difference in velocity commands corresponds to imposing a penalty on the applied module torque (the original system input in (5)), since change in velocity is proportional to acceleration, which itself relates to torque at a module. Therefore, the cost function

$$\begin{aligned} J = \sum_{k=0}^{\infty} [\, x_{\text{s}}^T[k] \quad x_{\text{int}}^T[k] \,]\, Q \begin{bmatrix} x_{\text{s}}[k] \\ x_{\text{int}}[k] \end{bmatrix} \\ + v^T[k]\, R\, v[k] + \big(v[k] - v[k-1]\big)^T\, \Theta\, \big(v[k] - v[k-1]\big), \end{aligned} \tag{38}$$

is used with suitable weighting matrices Q, R, and Θ (numerical values may be found in [44]). Since $v[k-1] = x_{\text{f}}[k]$, (38) can be reformulated as a standard LQR cost with nonzero weights on state and input cross terms,

$$J = \sum_{k=0}^{\infty} \tilde{x}^T[k] \begin{bmatrix} \Theta & 0 \\ 0 & Q \end{bmatrix} \tilde{x}[k] + v^T[k]\,(R+\Theta)\,v[k] + 2\tilde{x}^T[k] \begin{bmatrix} -\Theta \\ 0 \end{bmatrix} v[k], \tag{39}$$

where $\tilde{x}[k] := (x_{\text{f}}[k], x_{\text{s}}[k], x_{\text{int}}[k])$. The discrete-time LQR design problem can be solved using standard tools, [50] (such as the Matlab implementation `dlqr`). Let $F = [F_1\, F_2]$ be the resulting gain matrix with $F_1 \in \mathbb{R}^{6\times n}$ corresponding to the gains on $(x_\mathrm{f}[k], x_\mathrm{s}[k])$, and $F_2 \in \mathbb{R}^{6\times 6}$ corresponding to the gains on the integral states $x_\mathrm{int}[k]$. Using the approximation $\hat{x}_\mathrm{f}[k] = v[k-1]$ in addition, a representation of controller K then is

$$\zeta[k+1] = \zeta[k] + [\,T_\mathrm{s}\, I_{6\times 6} \quad 0\,]\, \hat{x}_s[k] \tag{40}$$

$$v[k] = F_2\, \zeta[k] + F_1 \begin{bmatrix} v[k-1] \\ \hat{x}_s[k] \end{bmatrix}, \tag{41}$$

which has the state estimates $\hat{x}_\mathrm{s}[k]$ from Fig. 17 as input.

8. Experiments

To compare the balancing performance in experiments, the root mean square (RMS) value of the state estimates,

$$x_i^\mathrm{RMS} := \sqrt{\frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \hat{x}_i^2[k]} \tag{42}$$

for data of length $\bar{k}$, is used (for the experimental evaluation, the motor shaft velocity measurement from the motor encoder scaled by the gear ratio is used as an estimate for the module velocity; estimates of all other states are as in Fig. 17). The RMS values for a sequence of five minutes of undisturbed balancing on a corner are given in Table 5 (middle column). The data shows that the RMS values for angular position and velocity differ by more than one order of magnitude for the bottom modules (1 to 3) versus the top ones (4 to 6) for corner balancing. The top modules are used significantly less because the corresponding inputs are much less effective than those of the bottom ones. This fact is explained by considering the analogy of a one dimensional abstraction of the Balancing Cube in the sidebar “Why Are the Top Modules Used Less?”

In a different experiment, the cube was disturbed by pushing one of its corners. A 60-second balancing sequence with the disturbance applied at 15 seconds is shown in Fig. 19. For a shorter sequence of the same experiment, the angular velocity command and the actual angular velocity of module 2 are shown in Fig. 20. The data demonstrates the tracking capability of the inner velocity controllers, which was assumed for the derivation of model (36) in the sidebar “Time Scale Separation Algorithm.”

Table 5. Balancing performance. The root mean square (RMS) value x_i^{RMS} according to (42) measures the average squared deviation of the state estimates from the equilibrium $x = 0$. It is used as a measure for the control performance. The data for corner balancing shows that the motion of the bottom modules (1 to 3) is an order of magnitude greater than the motion of the top ones (4 to 6). The reason for this is explained in the sidebar "Why Are the Top Modules Used Less?" Also for edge balancing, the motion of modules 1 to 3 is greater than the motion of the modules 4 to 6. The difference is, however, not as pronounced as for corner balancing. Due to the symmetrical arrangement of modules 1 and 3, and modules 4 and 6, their RMS values are almost the same. As expected, the balancing performance on edge is better than on corner since only one dimension needs to be stabilized (instead of two).

State	**RMS value x_i^{RMS} Corner balancing**	**RMS value x_i^{RMS} Edge balancing**
angle module 1	0.1036	0.0434
angle module 2	0.1214	0.0581
angle module 3	0.1225	0.0433
angle module 4	0.0080	0.0158
angle module 5	0.0092	0.0255
angle module 6	0.0098	0.0159
angular velocity module 1	0.2688	0.1034
angular velocity module 2	0.3266	0.1446
angular velocity module 3	0.3227	0.1029
angular velocity module 4	0.0275	0.0382
angular velocity module 5	0.0316	0.0658
angular velocity module 6	0.0281	0.0391
cube pitch angle	0.0048	0.0034
cube pitch rate	0.0103	0.0061
cube roll angle	0.0066	–
cube roll rate	0.0144	–

From the data in Table 5 and Fig. 19 and 20, it can be seen that the cube exhibits slight motion during balancing; that is, it is not perfectly steady. Some motion of the cube is inevitable due to excitation of the feedback system by sensor noise. Other effects such as network delays or gear backlash may be partially compensated for with a more sophisticated controller design. A discussion on the achievable balancing performance of the cube

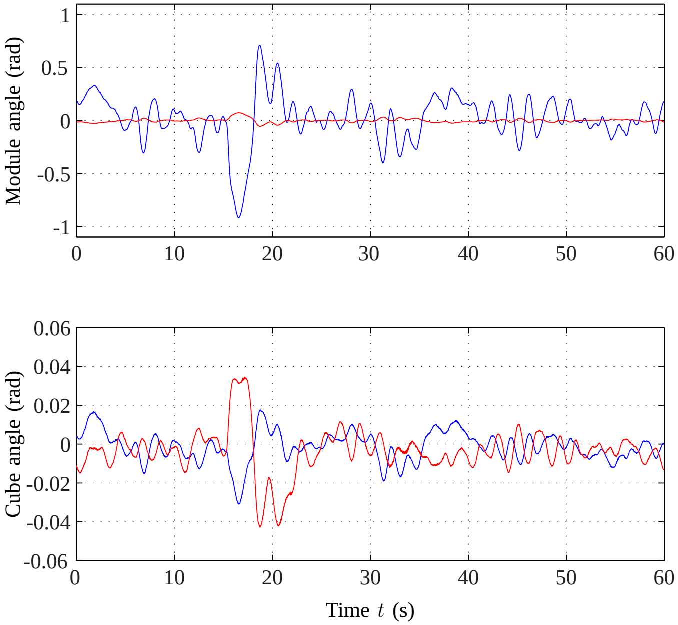


Figure 19. Corner balancing experiment. The top graph shows module 2's angle $\hat{x}_2$ (blue) and module 5's angle $\hat{x}_5$ (red); the bottom graph shows the cube pitch $\hat{x}_{13}$ (blue) and roll $\hat{x}_{15}$ (red). All angles are about the nominal equilibrium (3). At roughly 15 s, the cube was disturbed by pushing one of its corners. From the data, it is obvious that module 2 moves more than module 5. The reason is that the bottom modules are more effective than the top modules and, hence, used more by the optimal state-feedback controller. This fact is investigated in the sidebar "Why Are the Top Modules Used Less?"

based on the $\mathcal{H}_2$ system norm is presented in the sidebar "How Steady Can the Cube Balance?" The slight oscillations during balancing do, however, enable viewers to perceive the cube as a dynamic sculpture.

Experimental data for balancing the cube on one of its edges is shown in Table 5 (right column) and Fig. 21. It can be seen that the cube exhibits less motion in edge balancing compared to corner balancing. This behavior is expected, since only one DOF needs to be stabilized when on edge (compared to two for corner balancing) with the same number of actuators.

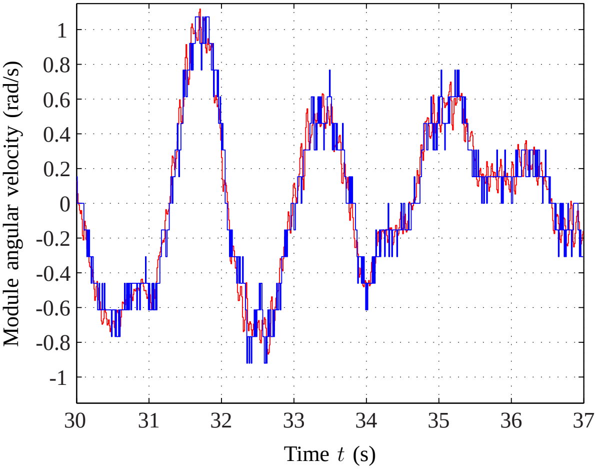


Figure 20. Command tracking for corner balancing experiment. The inner feedback loop on each module with controller K_{loc} tracks velocity commands from the outer-loop state-feedback controller K (compare Fig. 18). As an example for the tracking performance of the inner loop, the angular velocity (blue) and the corresponding command (red) are shown for module 2. The module velocity is computed as the change of the absolute encoder angle signal per time step, $(y_2^{\mathrm{enc}}[k] - y_2^{\mathrm{enc}}[k-1])/T_s$. The module velocity signal shows a significant quantization error, which is caused by the quantization of the absolute encoder. The signal is, however, not used in feedback, but only to demonstrate the tracking capabilities of the inner feedback loop here. The data is from the same experiment as in Fig. 19.

9. Concluding Remarks

This article presents the Balancing Cube and, in particular, the control system that enables the cube to live up to its name. The cube is a multi-agent 3D inverted pendulum system: stability is achieved through the coordination of six rotating bodies on the cube – each equipped with sensors, actuation, and a computer, and all communicating with each other over a digital network. With this architecture, the Balancing Cube combines the challenges of an unstable nonlinear system, a distributed control system, and a networked control system.

The concept of balancing a rigid body with multiple modules can also be used to balance other three dimensional shapes. The modeling techniques as well as the developed state estimation and control algorithms are generalized from the concrete representation of the cube, and can be applied to other

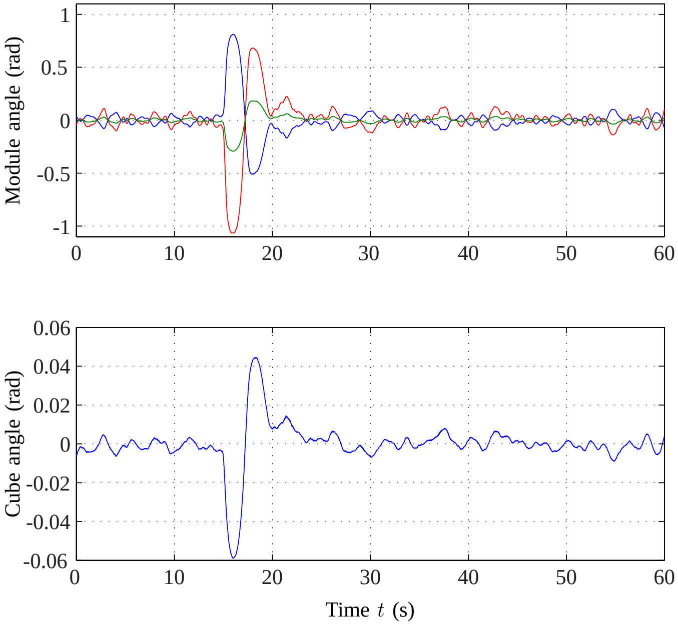


Figure 21. Edge balancing experiment. Shown in the top graph are the angles of modules 1, 2, and 6, $\hat{x}_1$ (blue), $\hat{x}_2$ (red), and $\hat{x}_6$ (green), respectively, and, in the bottom graph, the cube pitch angle $\hat{x}_{13}$. All angles are about the nominal equilibrium (4). At roughly 15 s, the system was disturbed by pushing the cube. The motion of module 6, which is one of the lighter modules, is less than the motion of modules 1 and 2.

shapes with slight modification. See the sidebar "Other Balancing Shapes" for a discussion and some conceptual ideas.

Problems addressed during the design and construction phase of the project have triggered unanticipated research results that are applicable beyond the cube itself. The "Time Scale Separation Algorithm," first presented in [51], is an example. Without knowing the details of the feedback controllers, and yet taking their effects into account, this algorithm yields a simplified discrete-time model of a continuous-time process under high-gain feedback on some of its states. A limiting property of the matrix exponential was also derived in the process. Another example is the algorithm for estimating the tilt of the cube (presented both in this article and in [49]). The algorithm can be used to estimate the tilt of any rigid body with only rotational degrees of freedom from measurements of multiple inertial sensors without requiring a dynamic system model.

For the control and estimation algorithms presented herein, full communication between the agents is assumed: each agent shares its sensory data with all its peers at the closed-loop rate. The state estimation and control design problems can hence be treated in a centralized fashion. In [40,41], the problem of reduced communication state estimation is addressed. Therein, event-based communication protocols are used to reduce the amount of sensor data shared over the network while maintaining a certain estimation performance.

Acknowledgments

The authors' special thanks goes to Matthew Donovan for his contributions to the concept, the mechanical design, and the realization of the Balancing Cube. Furthermore, the authors thank their colleagues at ETH Zurich – Daniel Burch, Sergei Lupashin, Hans Ulrich Honegger, and Gajamohan Mohanarajah – as well as all students who have participated in the project for making the Balancing Cube happen. Carolina Flores is gratefully acknowledged for drawings and photos of the cube in this article. The authors would finally like to thank Philipp Reist and Raymond Oung for their valuable comments and suggestions.

This work was supported by the Swiss National Science Foundation (SNSF).

References

[1] "Balancing Cube website," [accessed 13.08.2012]. [Online]. Available: http://www.cube.ethz.ch

[2] J. Shen, A. K. Sanyal, N. A. Chaturvedi, D. S. Bernstein, and N. H. McClamroch, "Dynamics and control of a 3D pendulum," in *Proc. of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004, pp. 323–328.

[3] P. Horáček, "Laboratory experiments for control theory courses: A survey," *Annual Reviews in Control*, vol. 24, pp. 151–162, 2000.

[4] K. Åström and K. Furuta, "Swinging up a pendulum by energy control," *Automatica*, vol. 36, no. 2, pp. 287–295, Feb. 2000.

[5] L. Buşoniu, D. Ernst, B. De Schutter, and R. Babuška, "Online least-squares policy iteration for reinforcement learning control," in *Proc. of*

the American Control Conference, Baltimore, MD, USA, Jul. 2010, pp. 486–491.

[6] A. P. Schoellig and R. D'Andrea, "Optimization-based iterative learning control for trajectory tracking," in *Proc. of the European Control Conference*, Budapest, Hungary, Aug. 2009, pp. 1505–1510.

[7] S. Jung and S. S. Kim, "Control experiment of a wheel-driven mobile inverted pendulum using neural network," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 2, pp. 297–303, Mar. 2008.

[8] R. Findeisen and P. Varutti, "Stabilizing nonlinear predictive control over nondeterministic communication networks," in *Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences, L. Magni, D. Raimondo, and F. Allgöwer, Eds. Springer Berlin / Heidelberg, 2009, vol. 384, pp. 167–179.

[9] H. Gao, X. Meng, and T. Chen, "Stabilization of networked control systems with a new delay characterization," *IEEE Transactions on Automatic Control*, vol. 53, no. 9, pp. 2142–2148, Oct. 2008.

[10] J. Colandairaj, G. W. Irwin, and W. G. Scanlon, "Wireless networked control systems with QoS-based sampling," *IET Control Theory Applications*, vol. 1, no. 1, pp. 430–438, Jan. 2007.

[11] L. Zhang, Y. Shi, T. Chen, and B. Huang, "A new method for stabilization of networked control systems with random delays," *IEEE Transactions on Automatic Control*, vol. 50, no. 8, pp. 1177–1181, Aug. 2005.

[12] X. Liu and A. Goldsmith, "Wireless network design for distributed control," in *Proc. of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004, pp. 2823–2829.

[13] D. V. Efimov and A. L. Fradkov, "Robust and adaptive observer-based partial stabilization for a class of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 54, no. 7, pp. 1591–1595, Jul. 2009.

[14] R.-J. Wai, M.-A. Kuo, and J.-D. Lee, "Design of cascade adaptive fuzzy sliding-mode control for nonlinear two-axis inverted-pendulum servomechanism," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 5, pp. 1232–1244, Oct. 2008.

[15] R.-J. Wai and L.-J. Chang, "Adaptive stabilizing and tracking control for a nonlinear inverted-pendulum system via sliding-mode technique,"

IEEE Transactions on Industrial Electronics, vol. 53, no. 2, pp. 674–692, Apr. 2006.

[16] A. Mills, A. Wills, and B. Ninness, "Nonlinear model predictive control of an inverted pendulum," in *Proc. of the American Control Conference*, St. Louis, MO, USA, Jun. 2009, pp. 2335–2340.

[17] S. Jung and J. T. Wen, "Nonlinear model predictive control for the swing-up of a rotary inverted pendulum," *Journal of Dynamic Systems, Measurement, and Control*, vol. 126, no. 3, pp. 666–673, Sep. 2004.

[18] C. R. Magers and A. N. Gündeş, "Low order decentralized stabilizing controller design for a mobile inverted pendulum robot," in *Proc. of the American Control Conference*, St. Louis, MO, USA, Jun. 2009, pp. 4233–4234.

[19] W. Chen and J. Li, "Decentralized output-feedback neural control for systems with unknown interconnections," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 1, pp. 258–266, Feb. 2008.

[20] G. Liu, I. Mareels, and D. Nešić, "Decentralized control design of interconnected chains of integrators: A case study," *Automatica*, vol. 44, no. 8, pp. 2171–2178, 2008.

[21] S. Cho, J. Shen, and N. McClamroch, "Mathematical models for the triaxial attitude control testbed," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 9, no. 2, pp. 165–192, 2003.

[22] N. A. Chaturvedi, N. H. McClamroch, and D. S. Bernstein, "Stabilization of a 3D axially symmetric pendulum," *Automatica*, vol. 44, no. 9, pp. 2258–2265, 2008.

[23] ——, "Asymptotic smooth stabilization of the inverted 3-D pendulum," *IEEE Transactions on Automatic Control*, vol. 54, no. 6, pp. 1204–1215, June 2009.

[24] N. Chaturvedi and H. McClamroch, "Asymptotic stabilization of the inverted equilibrium manifold of the 3-D pendulum using non-smooth feedback," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2658–2662, Nov. 2009.

[25] B. Srinivasan, P. Huguenin, and D. Bonvin, "Global stabilization of an inverted pendulum – control strategy and experimental verification," *Automatica*, vol. 45, no. 1, pp. 265–269, 2009.

[26] S. Riachy, Y. Orlov, T. Floquet, R. Santiesteban, and J.-P. Richard, "Second-order sliding mode control of underactuated mechanical systems I: Local stabilization with application to an inverted pendulum," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 4–5, pp. 529–543, 2008.

[27] K. Pathak, J. Franch, and S. K. Agrawal, "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 505–513, Jun. 2005.

[28] P. Reist and R. Tedrake, "Simulation-based LQR-trees with input and state constraints," in *Proc. of the IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, USA, May 2010, pp. 5504–5510.

[29] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[30] K. Gilpin and D. Rus, "Modular robot systems," *IEEE Robotics & Automation Magazine*, vol. 17, no. 3, pp. 38–55, Sep. 2010.

[31] R. Oung and R. D'Andrea, "The distributed flight array," *Mechatronics*, vol. 21, no. 6, pp. 908–917, Sep. 2011.

[32] J. K. Yook, D. M. Tilbury, and N. R. Soparkar, "Trading computation for bandwidth: reducing communication in distributed control systems using state estimators," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 4, pp. 503–518, Jul. 2002.

[33] J. E. Luntz and W. Messner, "A distributed control system for flexible materials handling," *IEEE Control Systems Magazine*, vol. 17, no. 1, pp. 22–28, Feb. 1997.

[34] J. M. Fowler and R. D'Andrea, "A formation flight experiment," *IEEE Control Systems Magazine*, vol. 23, no. 5, pp. 35–43, Oct. 2003.

[35] A. Stubbs, V. Vladimerou, A. T. Fulford, D. King, J. Strick, and G. E. Dullerud, "Multivehicle systems control over networks: a hovercraft testbed for networked and decentralized control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 56–69, Jun. 2006.

[36] Z. Jin, S. Waydo, E. B. Wildanger, M. Lammers, H. Scholze, P. Foley, D. Held, and R. M. Murray, "MVWT-II: the second generation Caltech multi-vehicle wireless testbed," in *Prof. of the American Control Conference*, Boston, MA, USA, Jul. 2004, pp. 5321–5326.

[37] N. Michael, J. Fink, and V. Kumar, “Experimental testbed for large multirobot teams,” *IEEE Robotics & Automation Magazine*, vol. 15, no. 1, pp. 53–61, Mar. 2008.

[38] M. W. Spong, “Underactuated mechanical systems,” in *Control Problems in Robotics and Automation*, ser. Lecture Notes in Control and Information Sciences, B. Siciliano and K. Valavanis, Eds. Springer Berlin / Heidelberg, 1998, vol. 230, pp. 135–150.

[39] Quanser Inc., “Cube,” Online, [accessed 13.08.2012]. [Online]. Available: http://www.quanser.com

[40] S. Trimpe and R. D’Andrea, “An experimental demonstration of a distributed and event-based state estimation algorithm,” in *Proc. of the 18th IFAC World Congress*, Milano, Italy, Aug. 2011, pp. 8811–8818.

[41] ——, “Reduced communication state estimation for control of an unstable networked control system,” in *Proc. of the 50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, USA, 2011, pp. 2361–2368.

[42] J. J. Craig, *Introduction to robotics: mechanics and control*, 3rd ed. Prentice Hall, 2005.

[43] MathWorks Inc., “SimMechanics$^{\text{TM}}$user’s guide,” Sep. 2009, [accessed 30.04.2011]. [Online]. Available: http://www.mathworks.com

[44] S. Trimpe and R. D’Andrea, “Numerical models and controller design parameters for the Balancing Cube,” IDSC, ETH Zürich, Tech. Rep., 2012. [Online]. Available: http://e-collection.library.ethz.ch/

[45] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology*, 2nd ed. Institution of Engineering and Technology, 2004.

[46] J. Farrell and M. Barth, *The global positioning system and inertial navigation.* McGraw-Hill Professional, 1999.

[47] D. Luenberger, “An introduction to observers,” *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 596–602, Dec. 1971.

[48] B. D. O. Anderson and J. B. Moore, *Optimal Filtering.* Mineola, New York: Dover Publications, 2005, originally published by Prentice-Hall 1979.

[49] S. Trimpe and R. D’Andrea, “Accelerometer-based tilt estimation of a rigid body with only rotational degrees of freedom,” in *Proc. of the IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, USA, May 2010, pp. 2630–2636.

[50] B. D. O. Anderson and J. B. Moore, *Optimal Control: Linear Quadratic Methods*. Mineola, New York: Dover Publications, 2007, originally published by Prentice-Hall 1990.

[51] S. Trimpe and R. D'Andrea, "A limiting property of the matrix exponential with application to multi-loop control," in *Proc. of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, P.R. China, Dec. 2009, pp. 6419–6425.

Sidebar 1: Balancing Cube on Tour

Since its completion in fall 2009, the cube has been exhibited at public events such as the *European researchers' night* in Zurich, Switzerland, [S1], and the *Festival della Scienza* in Genoa, Italy, [S2]. More recently, it made an appearance at the triennial *IFAC World Congress* in Milan, Italy, [S3], as part of the interactive presentation of [40] (see Fig. S1).

A large cube balancing on a corner in a public place is an unusual sight, and tends to attract a good deal of interest. As passers-by stop to push the cube and test its ability to maintain equilibrium, they engage in a unique opportunity to learn about control engineering and its limitations.

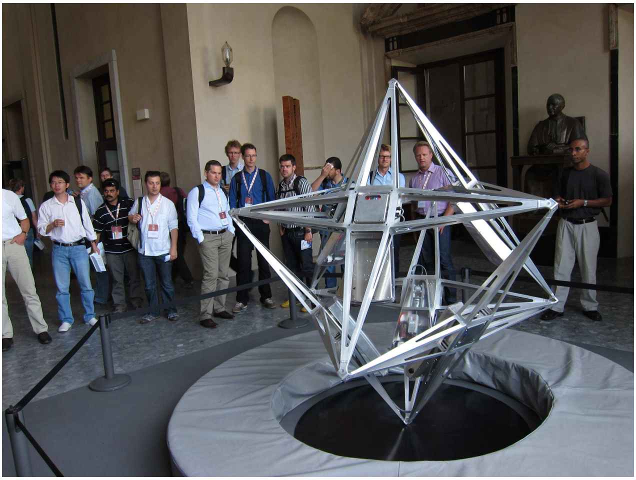


Figure S1. The Balancing Cube at the 2011 IFAC World Congress in Milan, Italy. The cube was shown in one of the interactive sessions.

References

[S1] "European researchers' night," Zurich, Switzerland, Sep. 2009, [accessed 13.08.2012]. [Online].
Available: http://www.nachtderforschung.ethz.ch/en

[S2] "Festival della scienza," Genoa, Italy, Oct. 2009, [accessed 13.08.2012]. [Online]. Available: http://www.festivalscienza.eu

[S3] "18th World Congress of the International Federation of Automatic Control (IFAC)," Milan, Italy, Aug. 2011, [accessed 13.08.2012]. [Online]. Available: http://www.ifac2011.org

Sidebar 2: What Is the Cube's Maximal Balancing Range?

This study investigates how much the cube body can tilt in static equilibrium by changing the module angles, as compared to the upright configurations (3) and (4). The maximal tilt calculated below is a theoretical upper bound on the feasible balancing range (for greater tilt, the multi-body system has no static equilibrium for any configuration of the modules). The calculations are based on the SimMechanics multi-body model.

To illustrate the dependency of the cube tilt on the module angles in static equilibrium for corner balancing, the equilibrium tilt is computed for the module angles parameterized by

$$(\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \varphi_6) = (\phi, -\phi, 0, -\phi, \phi, 0), \tag{S1}$$

with parameter ϕ ranging from 0° to 180°. For $\phi = 0^\circ$, the module configuration (S1) is the same as for the nominal "upright" equilibrium (3). The chosen parameterization affects the equilibrium only in pitch direction; the roll equilibrium angle is identical to the nominal γ_0 in (3) for all ϕ. The difference of the resulting equilibrium pitch (denoted by $\bar{\beta}$) from the nominal pitch β_0 in (3) is shown in Fig. S2. Additionally, the resulting horizontal displacement d_{xy} of the cube tip is shown in Fig. S2; it is given by

$$\begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} := \left({}^{O}_{B}S(\beta_0, \gamma_0) - {}^{O}_{B}S(\bar{\beta}, \bar{\gamma}) \right) \begin{bmatrix} 1.2\,\mathrm{m} \\ 1.2\,\mathrm{m} \\ 1.2\,\mathrm{m} \end{bmatrix}, \tag{S2}$$

$$d_{xy} := \sqrt{d_x^2 + d_y^2}, \tag{S3}$$

where 1.2 m is the edge length of the cube. The maximal pitch angle difference $\bar{\beta} - \beta_0 = -6.74^\circ$ corresponds to the maximal tip displacement of 24.4 cm; it is attained for $\phi = 102.5^\circ$. As verified by an optimization over the module angles without the constraint (S1), the maximal displacement is, in fact, the global maximum for all possible module angles. The module configuration (S1) with $\phi = 102.5^\circ$ is, however, not the only configuration that maximizes the cube tip displacement.

A similar analysis for edge balancing yields a maximal tilt from the upright standing cube (β_0 in (4)) of 8.15°, which corresponds to a tip displacement of $d_{xy} = 24.1\,\mathrm{cm}$.

The obtained maximal tip displacements represent the maximal range of static equilibria. In practice, the maximal range of equilibria that can be stabilized is smaller due to actuation limitations and sensor noise.

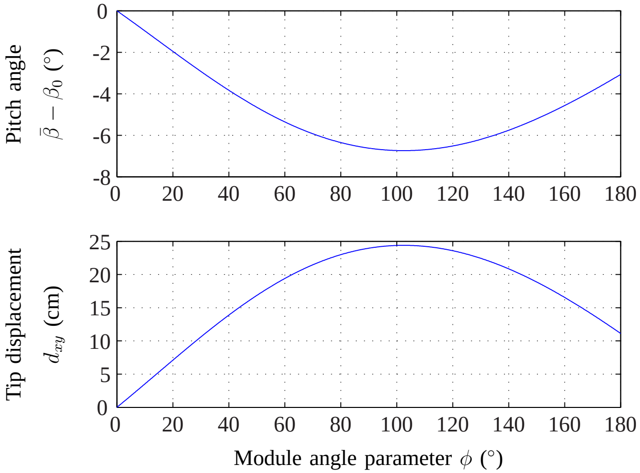


Figure S2. Cube tilt and tip displacement for different corner balancing equilibria. The pitch angle $\bar{\beta}$ in static equilibrium is shown in the top graph as a function of the module angle parameter ϕ (the module angles are parametrized according to (S1)). For better comparability, the nominal pitch β_0 is subtracted; that is, an angle of 0° corresponds to the nominal equilibrium (3), where the cube is standing upright. For the chosen parameterizations, the roll angle is equal to the nominal ($\bar{\gamma} = \gamma_0$) and therefore omitted here. The corresponding horizontal displacement d_{xy} of the cube tip from the upright configuration is shown in the bottom graph.

Sidebar 3: Time Scale Separation Algorithm

The time scale separation algorithm is a transformation of a continuous-time state-space model (A, B) to a discrete-time model $(\tilde{A}, \tilde{B})$ that represents (A, B) under high-gain feedback on some of its states through a controller K_{loc} (see Fig. S3). The method allows one to obtain a simplified model of the feedback system without detailed knowledge of the controller K_{loc}, by approximating it as an ideal controller with infinite proportional gain. This approximation is legitimate as long as the time scales of the feedback loop dynamics are sufficiently smaller than those of the remaining system. The resulting model can be used, for example, to design an outer-loop controller that feeds references to the inner-loop controller K_{loc}. The algorithm was first presented in [51].

Consider the continuous-time state-space representation

$$\begin{bmatrix} \dot{x}_{\text{f}}(t) \\ \dot{x}_{\text{s}}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} A_{\text{ff}} & A_{\text{fs}} \\ A_{\text{sf}} & A_{\text{ss}} \end{bmatrix}}_{A} \begin{bmatrix} x_{\text{f}}(t) \\ x_{\text{s}}(t) \end{bmatrix} + \underbrace{\begin{bmatrix} B_{\text{f}} \\ B_{\text{s}} \end{bmatrix}}_{B} u(t), \tag{S4}$$

where the states $x(t)$ are separated into those on which local feedback is applied ($x_{\text{f}}(t) \in \mathbb{R}^{n_{\text{f}}}$, index f for “fast”) and the remaining ones ($x_{\text{s}}(t)$, index s for “slow”). To model the controller K_{loc}, proportional feedback on the states x_{f} is assumed to be of the form

$$u(t) = B_{\text{f}}^{-1} F \left(v(t) - x_{\text{f}}(t)\right), \tag{S5}$$

where $v(t) \in \mathbb{R}^{n_{\text{f}}}$ is a piecewise constant reference signal changing at a rate T_{s}, $F = \operatorname{diag}(f_1, f_2, \ldots, f_{n_{\text{f}}})$ is a diagonal matrix with entries f_j, and B_{f}

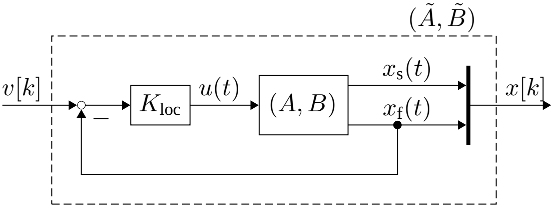


Figure S3. System with partial state feedback. The system is described by the continuous-time state-space model (A, B). Feedback loops are closed on some states $x_{\text{f}}(t)$ through controller K_{loc} with reference input $v[k]$ changing at a rate T_{s}. The time scale separation algorithm computes a discrete-time model $(\tilde{A}, \tilde{B})$ with sampling time T_{s} under the assumption of an ideal feedback controller K_{loc} with infinitely high proportional gains.

is assumed invertible. The feedback (S5) means that individual loops are closed on the states $x_\mathrm{f}(t)$.

To obtain the discrete-time representation $(\tilde{A}, \tilde{B})$, the feedback system given by (S4) and (S5) is first discretized at the sampling rate T_s. To represent ideal feedback loops, the controller gains f_j in (S5) are chosen to approach infinity in the limit. The model $(\tilde{A}, \tilde{B})$ is then obtained using a limiting property of the matrix exponential, which is derived in [51]. The resulting model has the structure

$$\begin{bmatrix} x_\mathrm{f}[k{+}1] \\ x_\mathrm{s}[k{+}1] \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 \\ \tilde{A}_\mathrm{sf} & \tilde{A}_\mathrm{ss} \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} x_\mathrm{f}[k] \\ x_\mathrm{s}[k] \end{bmatrix} + \underbrace{\begin{bmatrix} I \\ \tilde{B}_\mathrm{s} \end{bmatrix}}_{\tilde{B}} v[k]. \tag{S6}$$

The details of the derivation are omitted here, but can be found in [51]. Notice from (S6) that $x_\mathrm{f}[k{+}1] = v[k]$; that is, the reference is achieved in one time step, which corresponds to the infinite gain assumption.

Application to the cube model

The time scale separation algorithm is used to compute a model for the block $(\tilde{A}, \tilde{B})$ in Fig. 18. For this purpose, the state-space model (5) is partitioned in blocks corresponding to module angles, module velocities, and cube states,

$$\begin{bmatrix} \dot{x}_{1:6}(t) \\ \dot{x}_{7:12}(t) \\ \dot{x}_{13:n}(t) \end{bmatrix} = \begin{bmatrix} 0 & I & 0 \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_{1:6}(t) \\ x_{7:12}(t) \\ x_{13:n}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ B_2 \\ B_3 \end{bmatrix} u(t). \tag{S7}$$

Through the controllers on each drive unit, local feedback is applied on the module velocities $x_{7:12}(t)$ (see Fig. 18). Hence, the time scale separation technique above is applied with $x_\mathrm{f}(t) = x_{7:12}(t)$ and $x_\mathrm{s}(t) = (x_{1:6}(t), x_{13:n}(t))$. Since each module has an individual torque input, $B_\mathrm{f} = B_2$ is invertible. The reference signals $v[k]$ are the velocity commands sent to the motors at an update rate of $T_\mathrm{s} = 0.01\,\mathrm{s}$. The resulting model corresponding to (S6) is

$$\underbrace{\begin{bmatrix} x_{1:6}[k{+}1] \\ x_{7:12}[k{+}1] \\ x_{13:n}[k{+}1] \end{bmatrix}}_{x[k+1]} = \underbrace{\begin{bmatrix} I & 0 & 0 \\ 0 & 0 & 0 \\ \tilde{A}_{31} & \tilde{A}_{32} & \tilde{A}_{33} \end{bmatrix}}_{\tilde{A}} \underbrace{\begin{bmatrix} x_{1:6}[k] \\ x_{7:12}[k] \\ x_{13:n}[k] \end{bmatrix}}_{x[k]} + \underbrace{\begin{bmatrix} T_\mathrm{s}\, I \\ I \\ \tilde{B}_3 \end{bmatrix}}_{\tilde{B}} v[k]. \tag{S8}$$

The discrete-time model (S8) captures the dynamics of the cube including the local velocity feedback. The numerical values for the matrices $(\tilde{A}, \tilde{B})$ may be found in [44].

Sidebar 4: What Is the Effect of Integral Action in the Controller?

Integral control is typically used to improve the steady-state behavior of a feedback control system. In particular, the quantity whose integral is used in a feedback controller (for example, a system state) is forced to zero when the system reaches steady state.*

The control algorithm described in the section “Control” includes integral feedback on the module angles. The analysis below shows that the integrators in the controller ensure average zero steady-state error not only for the module angles, but for *all* states – despite a possible bias on the estimates of the cube states (pitch, roll, and their rates). First, the analysis is presented for a general linear time-invariant system, and then the results are interpreted for the Balancing Cube.

Consider the discrete-time system

$$x[k+1] = \begin{bmatrix} x_1[k+1] \\ x_2[k+1] \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} x[k] + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u[k]. \tag{S9}$$

The state measurements

$$y[k] = \begin{bmatrix} x_1[k] + w_1[k] \\ x_2[k] + w_2[k] + d_2 \end{bmatrix} \tag{S10}$$

are corrupted by zero-mean sensor noise $w[k] = (w_1[k], w_2[k])$ and a static bias d_2. Notice that the measurement of x_1 is bias-free. A feedback controller with integral action on the measurement of state x_1 is used,

$$x_{\text{int}}[k+1] = x_{\text{int}}[k] + T_{\text{s}}(x_1[k] + w_1[k]) \tag{S11}$$

$$u[k] = F \begin{bmatrix} y[k] \\ x_{\text{int}}[k] \end{bmatrix} = \begin{bmatrix} F_1 & F_2 & F_{\text{int}} \end{bmatrix} \begin{bmatrix} x_1[k] + w_1[k] \\ x_2[k] + w_2[k] + d_2 \\ x_{\text{int}}[k] \end{bmatrix}, \tag{S12}$$

where F is a static gain matrix, and T_{s} is the sampling time. The following assumptions are made on the system and the controller:

---

*For a general discussion of integral action in feedback controllers, see, for example: K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*, Princeton University Press, 2008.
[This footnote was added in this reprint; it is not contained in the original publication in the *IEEE Control Systems Magazine*.]

(A1) B_1 has full column rank.

(A2) $I - A_{22} + B_2(B_1^T B_1)^{-1} B_1 A_{12}$ is invertible.

(A3) The closed-loop system given by (S9), (S11), and (S12) is stable.

It is argued below that, under these assumptions, all states $x[k]$ of (S9) have zero steady-state mean,

$$\lim_{k\to\infty} \mathrm{E}\left[x[k]\right] = 0, \tag{S13}$$

for any static disturbance d_2.

The closed loop system given by (S9), (S11), and (S12) reads

$$\begin{bmatrix} x_1[k{+}1] \\ x_2[k{+}1] \\ x_{\mathrm{int}}[k{+}1] \end{bmatrix} = \underbrace{\begin{bmatrix} A_{11} + B_1F_1 & A_{12} + B_1F_2 & B_1F_{\mathrm{int}} \\ A_{21} + B_2F_1 & A_{22} + B_2F_2 & B_2F_{\mathrm{int}} \\ T_{\mathrm{s}}I & 0 & I \end{bmatrix}}_{=:\bar{A}} \underbrace{\begin{bmatrix} x_1[k] \\ x_2[k] \\ x_{\mathrm{int}}[k] \end{bmatrix}}_{=:\tilde{x}[k]}$$
$$+ \underbrace{\begin{bmatrix} B_1F_2 \\ B_2F_2 \\ 0 \end{bmatrix}}_{=:\bar{B}} d_2 + \begin{bmatrix} B_1F_1 & B_1F_2 \\ B_2F_1 & B_2F_2 \\ T_{\mathrm{s}}I & 0 \end{bmatrix} w[k]. \tag{S14}$$

Since $w[k]$ has zero mean, taking expected values yields

$$\mathrm{E}\left[\tilde{x}[k{+}1]\right] = \bar{A}\,\mathrm{E}\left[\tilde{x}[k]\right] + \bar{B}\,d_2. \tag{S15}$$

From (A3), it follows that $\mathrm{E}\left[\tilde{x}[k]\right]$ converges to a constant $\bar{x}$,

$$\lim_{k\to\infty} \mathrm{E}\left[\tilde{x}[k]\right] = \bar{x}, \tag{S16}$$

which is the solution to

$$\bar{x} = \bar{A}\,\bar{x} + \bar{B}\,d_2. \tag{S17}$$

From the equation for $\bar{x}_{\mathrm{int}}$ in (S17) (with $\bar{A}$ and $\bar{B}$ as defined in (S14)), it follows that

$$\bar{x}_{\mathrm{int}} = T_{\mathrm{s}}\bar{x}_1 + \bar{x}_{\mathrm{int}} \quad\Leftrightarrow\quad \bar{x}_1 = 0, \tag{S18}$$

which is the aforementioned typical effect of integral action on x_1. Using this result, the top rows of (S17) read

$$0 = (A_{12} + B_1F_2)\bar{x}_2 + B_1F_{\mathrm{int}}\,\bar{x}_{\mathrm{int}} + B_1F_2\,d_2, \tag{S19}$$

which, with assumption (A1), yields

$$F_{\text{int}}\,\bar{x}_{\text{int}} = -(B_1^T B_1)^{-1} B_1^T (A_{12} + B_1 F_2)\bar{x}_2 - F_2\, d_2. \tag{S20}$$

Using this result, (S17) can be solved for $\bar{x}_2$,

$$\begin{aligned}\bar{x}_2 &= (A_{22} + B_2 F_2)\bar{x}_2 - B_2 (B_1^T B_1)^{-1} B_1^T (A_{12} + B_1 F_2)\bar{x}_2 \\ &\quad - B_2 F_2\, d_2 + B_2 F_2\, d_2 && \text{(S21)} \\ \Leftrightarrow&\ \ (I - A_{22} + B_2 (B_1^T B_1)^{-1} B_1^T A_{12})\bar{x}_2 = 0, && \text{(S22)}\end{aligned}$$

which, by assumption (A2), has the unique solution $\bar{x}_2 = 0$. Hence, (S13) holds as claimed. Furthermore, from (S20), it can be seen that the bias d_2 maps to steady-state offsets in the integrator states given by the relation

$$F_{\text{int}}\,\bar{x}_{\text{int}} = -F_2\, d_2. \tag{S23}$$

Interpretation for the Balancing Cube

The cube model (36) corresponds to (S9) with x_1 being the module angles and x_2 the remaining states. The absolute encoder measurements of the module angles can be considered bias-free. The controller (40)–(41) is of the form (S11)–(S12), and it can be shown that the assumptions (A1) to (A3) hold.

Hence, the control system ensures that the deviation from the equilibrium is zero for all states – on average and in the long run – despite any static offset in the estimates of the cube states. Such offsets result from IMU sensor biases and the fact that the cube's physical equilibrium is not exactly at the model-based pitch and roll equilibrium angles in (3) and (4).

An intuitive explanation for this property of the feedback system is as follows: enforced by the integral feedback, the module angles are on average at the module equilibrium angles given in (3) or (4). These module angles define a unique equilibrium for the cube tilt parameterized by pitch and roll angles. Since the feedback control system is stable, the average pitch and roll angles must be equal to the equilibrium angles – otherwise the cube would not be in equilibrium on average and, hence, it would fall.

Sidebar 5: Why Are the Top Modules Used Less?

When the cube balances on one of its corners, the top modules move considerably less than the bottom ones (see the section "Experiments"). In order to understand the impact of the mounting position on a module's ability to balance the cube, a one dimensional abstraction of the balancing problem is considered first. The controllability of an inverted pendulum that is balanced by a single module is analyzed as a function of the module's mounting height on the pendulum. The insights of the 1D analysis are then interpreted for the cube.

An inverted pendulum of the above type is shown in Fig. S4. The rotating arm that balances the planar inverted pendulum about the single rotational DOF is identical to the moving part of the modules on the cube. The system is essentially a double pendulum with torque applied at the joint linking the two bodies (see Fig. S5). This type of double pendulum has also been termed an *acrobot*, and it is a typical example of an underactuated system [38, S4]. For the analysis herein, the two bodies are approximated as point masses.

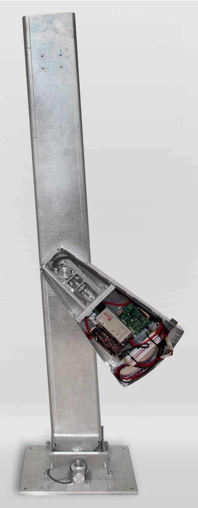


Figure S4. Inverted pendulum. The pendulum is a one dimensional abstraction of the Balancing Cube: one module balances the pendulum body about one rotational degree of freedom. This system was built as a prototype prior to the cube.

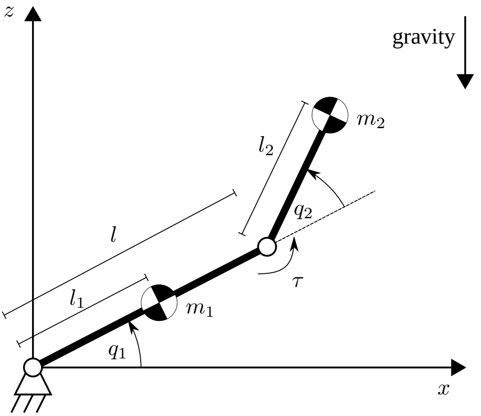


Figure S5. Acrobot. The acrobot is an abstraction of the pendulum in Fig. S4. The module is represented by mass m_2, which is linked through a revolute joint to mass m_1 representing the pendulum body, which itself is linked to the ground. The acrobot is a typical example of an underactuated system: it has two degrees of freedom (described by the generalized coordinates $q = (q_1, q_2)$) with only one actuator (torque τ applied at the link between m_1 and m_2). The controllability of the system about the equilibrium given by $\bar{q} = (\pi/2, \pi)$ and $\bar{\tau} = 0$ is studied in this section for different location l of the module link on the pendulum.

The equations of motion of the point-mass acrobot are, [S4],

$$H(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + G(q) = \begin{bmatrix} 0 \\ \tau \end{bmatrix}, \tag{S24}$$

where q are the generalized coordinates (angles of the two links, defined in Fig. S5), τ is the torque applied at the second link, and the matrices are given by

$$H(q) = \begin{bmatrix} m_1 l_1^2 + m_2 l^2 + m_2 l_2^2 + 2 m_2 l_2 l \cos(q_2) & m_2 l_2^2 + m_2 l_2 l \cos(q_2) \\ m_2 l_2^2 + m_2 l_2 l \cos(q_2) & m_2 l_2^2 \end{bmatrix}, \tag{S25}$$

$$C(q,\dot{q}) = \begin{bmatrix} -2 m_2 l_2 l \sin(q_2)\dot{q}_2 & -m_2 l_2 l \sin(q_2)\dot{q}_2 \\ m_2 l_2 l \sin(q_2)\dot{q}_1 & 0 \end{bmatrix}, \tag{S26}$$

$$G(q) = \begin{bmatrix} (m_1 l_1 + m_2 l) g_0 \cos(q_1) + m_2 l_2 g_0 \cos(q_1 + q_2) \\ m_2 l_2 g_0 \cos(q_1 + q_2) \end{bmatrix}, \tag{S27}$$

with g_0 the gravity constant and all other parameters as given in Fig. S5.

Next, the local controllability of the acrobot about the equilibrium configuration of an upright pendulum body and a downward pointing module is analyzed for different values of the link location l. For this purpose, the system (S24) is linearized about the equilibrium given by $\bar{q} = (\pi/2, \pi)$ and $\bar{\tau} = 0$. The obtained state space representation with the state $x = (q, \dot{q}) - (\bar{q}, 0)$ reads

$$
\begin{aligned}
\dot{x} &= \begin{bmatrix} 0 & I \\ -H^{-1}(\bar{q})\,\frac{\partial G}{\partial q}(\bar{q}) & -H^{-1}(\bar{q})\,C(\bar{q},0) \end{bmatrix} x + \begin{bmatrix} 0 \\ H^{-1}(\bar{q}) \begin{bmatrix} 0 \\ \tau \end{bmatrix} \end{bmatrix} \\
&= \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{g_0}{l_1} & -\frac{g_0 l m_2}{m_1 l_1^2} & 0 & 0 \\ \frac{g_0(l-l_1-l_2)}{l_1 l_2} & -\frac{g_0(m_1 l_1^2 + m_2 l(l-l_2))}{m_1 l_1^2 l_2} & 0 & 0 \end{bmatrix}}_{=:\bar{A}} x + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \frac{l-l_2}{m_1 l_1^2 l_2} \\ \frac{m_1 l_1^2 + m_2 (l-l_2)^2}{m_1 m_2 l_1^2 l_2^2} \end{bmatrix}}_{=:\bar{B}} \tau .
\end{aligned}
\tag{S28}
$$

For varying l and fixed parameter values $g_0 = 9.81\,\mathrm{m/s^2}$, $l_1 = 0.66\,\mathrm{m}$, $l_2 = 0.2\,\mathrm{m}$, $m_1 = 5.3\,\mathrm{kg}$, $m_2 = 3.7\,\mathrm{kg}$, which are representative for the pendulum in Fig. S4, the two smallest singular values of the controllability matrix $\mathcal{C} = [\bar{B}, \bar{A}\bar{B}, \bar{A}^2\bar{B}, \bar{A}^3\bar{B}]$ are shown in Fig. S6. It can be seen that the system becomes uncontrollable for $l = 0.66\,\mathrm{m} + 0.2\,\mathrm{m} = 0.86\,\mathrm{m}$. More generally, it can be shown from (S28) that for $l = l_1 + l_2$ and arbitrary l_1, l_2, m_1, and m_2, the rank of $\mathcal{C}$ drops from four to two. That is, for the configuration where the center of gravity (CG) of the module is at the same height as the CG of the pendulum, the system becomes locally uncontrollable.

How can this insight be interpreted for the cube? It can be expected that the location of a module relative to the CG of the cube body is a crucial factor in how effectively the module can act on it. The top modules' CG is above the CG of the cube body, whereas the bottom modules' CG is well below (especially since the lower modules are heavier and their CG is lower). In analogy to the 1D analysis, it can therefore be expected that the top modules are less effective than the bottom ones. This is confirmed by the analysis of the singular values of the controllability matrix of the cube shown in Fig. S7.

In conclusion, the top modules are used less, since they are less effective for control. The controller resulting from the LQR design seeks to minimize the total control effort and hence tends to use mainly the more effective bottom modules (equal weighting provided).

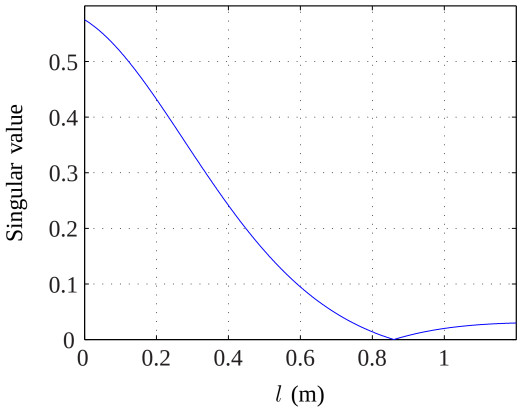


Figure S6. Controllability of the acrobot about the equilibrium $\bar{q} = (\pi/2, \pi)$. Shown are the two smallest (identical) singular values of the controllability matrix of the linearized model as a function of the link location l. For $l = l_1 + l_2 = 0.86\,\mathrm{m}$ the system is uncontrollable. The pendulum in Fig. S4 with $l = 0.69\,\mathrm{m}$ is controllable.

References

[S4] M. W. Spong, "The swing up control problem for the acrobot," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 49–55, Feb. 1995.

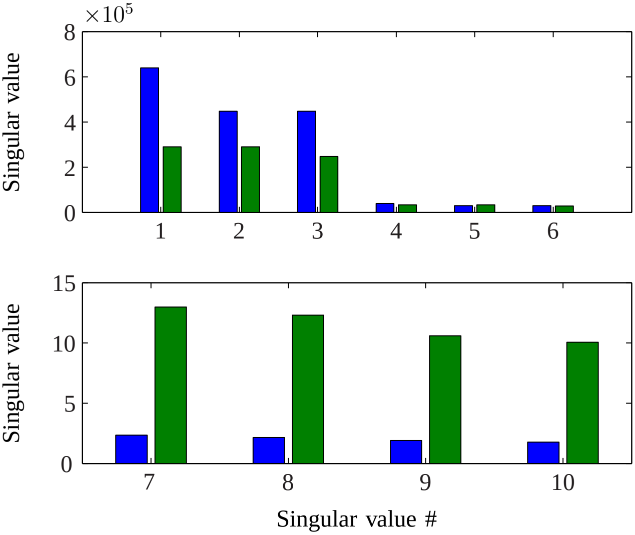


Figure S7. Singular values of the controllability matrix for corner balancing. The controllability of the cube is analyzed for the case that *only the top modules* (blue) and the case that *only the bottom modules* (green) are used. The controllability matrix $\mathcal{C}$ is computed from the model (5) after removing the states and inputs corresponding to the unused modules. The first six singular values (top diagram) correspond mainly to motion of the modules, whereas the last four (bottom diagram) correspond mainly to cube states. The fact that the last four singular values are smaller than the first six means that the cube states are "harder" to control than the module states (which is expected since the cube's degrees of freedom are not directly actuated). Furthermore, stabilizing the cube is harder when only the top modules are used compared to using only the bottom ones. This is indicated by the corresponding singular values being almost one order of magnitude smaller. Since the top modules are less effective in influencing the cube motion, they are used less by the optimal LQR controller, which is designed in the section "Control."

Sidebar 6: How Steady Can the Cube Balance?

The cube is not perfectly steady when it balances (see the section "Experiments"). Since the control system relies on sensory feedback for stabilization, sensor noise (mainly from the IMUs) inevitably excites the closed-loop system. In addition, unmodeled effects such as backlash or network-induced delays not taken into account in the control design can further deteriorate the balancing performance.

The question of how steady the cube can balance in principle with the current sensors is analyzed by employing tools from optimal $\mathcal{H}_2$ controller design. The $\mathcal{H}_2$ system norm is a measure of the overall gain of a dynamic system driven by noise: it equals the root mean square (RMS) value of the system output when the input is white noise of unit intensity, [S5]. Hence, by appropriately scaling the input and output channels, the $\mathcal{H}_2$ norm is used to analyze the response of the cube (measured as the RMS of its state) to sensor noise excitation. It corresponds to the achievable balancing performance if no other nonidealities in the control system are present. For an introduction to $\mathcal{H}_2$ optimal controller design, the interested reader is referred to [S5].

For the $\mathcal{H}_2$ analysis and synthesis below, the discrete-time model (36) is assumed with noisy state measurements; that is,

$$x[k+1] = \tilde{A}\,x[k] + \tilde{B}\,v[k] \tag{S29}$$

$$\tilde{y}[k] = x[k] + w_1[k], \tag{S30}$$

where the artificial measurement noise $w_1[k]$ is assumed to have zero mean and variance R_n. The variance R_n is chosen below to match the variance of the state estimates resulting from the state estimation algorithms employed on the cube. Hence, the state estimation problem is abstracted away for the purpose of this analysis.

The $\mathcal{H}_2$ norm is computed for the closed-loop system given by (S29), (S30), and, firstly, the actual balancing controller K from (40), (41), and, secondly, an $\mathcal{H}_2$ optimal controller. The analysis allows one to estimate how well the control system would perform under ideal circumstances (that is, if the linear model captured the dynamics perfectly). It also allows one to determine how much could be gained from a more sophisticated controller design.

$\mathcal{H}_2$ analysis of balancing controller

The generalized plant shown in Fig. S8 combines the cube model (36) with exogenous weighted inputs and outputs that are used to express the analysis objective. The $\mathcal{H}_2$ norm $\|G_{w_1 \to z_1}\|_2$ from measurement noise input $w_1[k]$ to

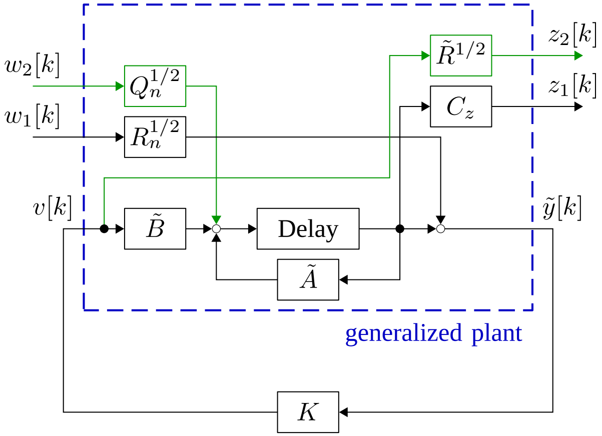


Figure S8. Generalized plant used to analyze the cube's balancing performance. The blocks $\tilde{A}$ and $\tilde{B}$ represent the cube model (36). The output $\tilde{y}[k]$ feeds to the controller K, which computes the system input $v[k]$. The exogenous input $w_1[k]$ and exogenous output $z_1[k]$ are used to express the design objective: input $w_1[k]$ is scaled with $R_\mathrm{n}^{1/2}$ (the square root of the measurement noise variance), and $z_1[k]$ are the system states of interest, which are selected from $x[k]$ by means of the output matrix C_z. Hence, the $\mathcal{H}_2$ norm from input $w_1[k]$ to output $z_1[k]$, $\|G_{w_1 \to z_1}\|_2$, is the RMS of the selected state signals when the system is excited by noise of variance R_n. It is therefore a measure of the cube's balancing performance. The additional green signals and blocks (input $w_2[k]$ with weighting $Q_\mathrm{n}^{1/2}$ and output $z_2[k]$ with weighting $\tilde{R}^{1/2}$) are required for the $\mathcal{H}_2$ controller synthesis problem to be well defined.

the cube states $z_1[k] = C_z x[k]$ is a measure of the balancing performance (C_z is chosen to select a subset of the state vector $x[k]$). The norm corresponds to the RMS of the output $z_1[k]$ under noise excitation representative of the sensor noise on the Balancing Cube. The additional green blocks in Fig. S8 are required for the $\mathcal{H}_2$ optimal controller design presented later.

To have a noise excitation representative of the actual cube hardware, the measurement noise variance R_n is chosen to be equal to the variance of the state estimates presented in the section "State Estimation." Hence, the noise variance on the cube states is set to be

$$R^\mathrm{c}_{\mathrm{n,cube}} = \mathrm{diag}\big(\mathrm{Var}\,[\hat{\beta}[k]], \mathrm{Var}\,[\hat{\dot{\beta}}[k]], \mathrm{Var}\,[\hat{\gamma}[k]], \mathrm{Var}\,[\hat{\dot{\gamma}}[k]]\big) \tag{S31}$$

for corner balancing, and

$$R^\mathrm{e}_{\mathrm{n,cube}} = \mathrm{diag}\big(\mathrm{Var}\,[\hat{\beta}[k]], \mathrm{Var}\,[\hat{\dot{\beta}}[k]]\big) \tag{S32}$$

for edge balancing, where $\hat{\beta}[k]$, $\hat{\dot{\beta}}[k]$, $\hat{\gamma}[k]$, and $\hat{\dot{\gamma}}[k]$ are the cube state estimates in (33), (34), (35). Given the noise variance of the IMU sensors, first order approximations of the variances in (S31) and (S32) can readily be computed from the involved estimator equations. Since the quantization error on the module encoders can be neglected, the noise variances for the modules states are chosen to be substantially lower than those for the cube states; hence, the overall noise variance is set to

$$R_{\mathrm{n}} = \begin{bmatrix} 10^{-6}\,\bar{\sigma}(R_{\mathrm{n,cube}})\,I & 0 \\ 0 & R_{\mathrm{n,cube}} \end{bmatrix}, \tag{S33}$$

where $\bar{\sigma}(\cdot)$ denotes the largest singular value.

With the weighting matrix (S33) and output matrix C_z chosen to select the cube state(s) of interest, the $\mathcal{H}_2$ norm of the generalized plant in Fig. S8 from $w_1[k]$ to $z_1[k]$ can be evaluated for the balancing controller (40), (41). The results are shown in Fig. S9 in green. The comparison to the experimental data (blue) reveals that the practically achieved balancing performance with the current LQR controller is lower than the theoretically achievable performance.

$\mathcal{H}_2$ optimal design

In order to design an $\mathcal{H}_2$ optimal controller for the generalized plant in Fig. S8, some augmentations are necessary to make the $\mathcal{H}_2$ synthesis problem well defined. For this purpose, the generalized plant is augmented with the process noise input $w_2[k]$ (weight Q_{n}) and weighted control signal $z_2[k]$ (weight $\tilde{R}$). The process noise intensity is chosen to be considerably smaller than the measurement noise, in order to have a negligible effect on the controller design, $Q_{\mathrm{n}} = 10^{-6}\,\bar{\sigma}(R_{\mathrm{n}})\,I$. For simplicity and to allow for a fair comparison in terms of control effort, diagonal weights $\tilde{R} = \tilde{\rho} I$ with parameter $\tilde{\rho}$ are used for the control input, where $\tilde{\rho} = 10^{-7}$ has been tuned such that the $\mathcal{H}_2$ gain from input $(w_1[k], w_2[k])$ to the output $z_2[k]$ is comparable to the respective gain when using the actual controller (40), (41). The output matrix C_z is chosen as $C_z = \mathrm{diag}(0.01 I_{12\times 12}, I)$ to express the primary design objective of minimizing the cube state variance.

The $\mathcal{H}_2$ optimal controller that minimizes the $\mathcal{H}_2$ norm from exogenous input $w[k] = (w_1[k], w_2[k])$ to exogenous output $z[k] = (z_1[k], z_2[k])$ can be obtained using standard $\mathcal{H}_2$ synthesis tools (for example, the Matlab implementation `h2syn`). The resulting $\mathcal{H}_2$ gains are shown in Fig. S9 (red). The resulting smaller gains compared to the LQR design (green) are at the expense of a higher order controller. The $\mathcal{H}_2$ design presented herein is mainly of interest as a theoretical bound on the balancing performance.

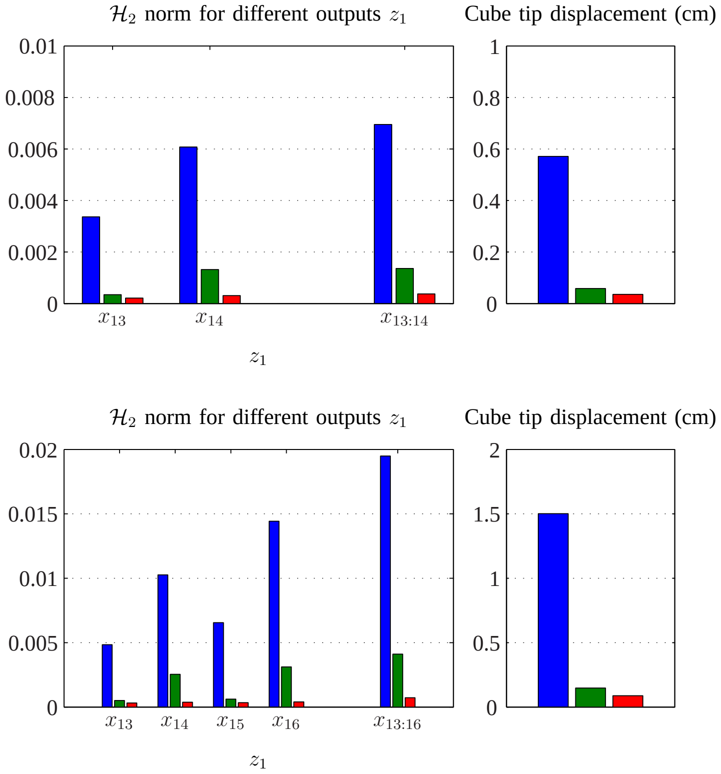


Figure S9. $\mathcal{H}_2$ analysis of balancing performance on edge (top diagrams) and corner (bottom). The bar diagram on the left shows the $\mathcal{H}_2$ norm from state noise input w_1 to the cube states x_{13} through x_n, both individually and combined. These norms are obtained from experimental data (blue), from the linear model with the actual balancing controller (green), and with the $\mathcal{H}_2$ optimal controller (red). The $\mathcal{H}_2$ norm corresponds to the RMS value of the output signals under noise excitation of the system. The experimental data is the same as in Table 5. The diagram on the right shows the RMS displacement of the cube tip that is equivalent to the RMS of the cube tilt angles x_{13} and x_{15} (it is computed analogously to (S2), (S3) in the sidebar "What Is the Cube's Maximal Balancing Range?").

It does not take into account other design objectives that are of practical importance, such as steady-state behavior or actuator limitations.

In conclusion, the results of this section point to a potential improvement in the balancing performance of the cube. This may partly be achieved by taking currently unmodeled effects (such as gear backlash or communication delays) into account in the controller design.

References

[S5] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*, 2nd ed. Wiley-Interscience, Nov. 2005.

Sidebar 7: Other Balancing Shapes

Because the modules are self-contained, with onboard sensing, actuation, computation, and communication, they can be arranged into other "balancing" shapes. Conceptual drawings of such shapes are depicted in Fig. S10.

The tilt estimation method presented in the section "State Estimation" requires at least four tri-axis accelerometers. If fewer than four modules are used (such as for the wedge in Fig. S10), additional sensors may be placed on the balanced body. Alternatively, a model-based state estimation technique (such as a Kalman filter) can be used to observe the tilt state from rate gyro sensors only. As far as the modeling process and the design of the control algorithms are concerned, the same strategy as presented in this article for the cube may be used for other shapes.

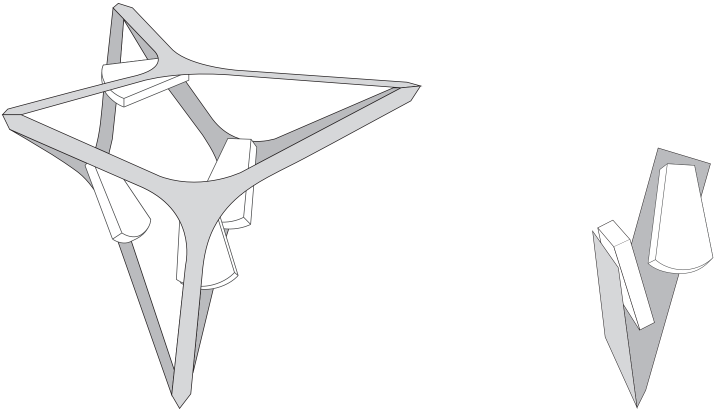


Figure S10. Other balancing shapes. Instead of balancing a cube, the modules could be used to balance a tetrahedron or a wedge (standing on a tip), for example.

Paper V

A Limiting Property of the Matrix Exponential

Sebastian Trimpe · Raffaello D'Andrea

Abstract

A limiting property of the matrix exponential is proven: if the (1,1)-block of a 2-by-2 block matrix becomes "arbitrarily small" in a limiting process, the matrix exponential of that matrix tends to zero in the (1,1), (1,2), and (2,1)-blocks. The limiting process is such that either the log norm of the (1,1)-block goes to negative infinity, or, for a certain polynomial dependency, the matrix associated with the largest power of the variable that tends to infinity is stable. The limiting property is useful for simplification of dynamic systems that exhibit modes with sufficiently different time scales. The obtained limit then implies the decoupling of the corresponding dynamics.

Submitted to *IEEE Transactions on Automatic Control*, 2012.

1. Introduction

The subject of study in this paper is the matrix exponential

$$\exp\left(\begin{bmatrix} A_{11} - K(\alpha) & A_{12} \\ A_{21} & A_{22} \end{bmatrix} t\right), \quad t > 0 \tag{1}$$

in the limit as $K(\alpha)$ grows large for $\alpha \to \infty$ in some sense to be made precise later. All matrices are complex, and α is a real parameter. For different classes of $K(\alpha)$, we derive sufficient (and in one case also necessary) conditions on $K(\alpha)$ such that, for all $t > 0$,

$$\lim_{\alpha\to\infty} \exp\left(\begin{bmatrix} A_{11} - K(\alpha) & A_{12} \\ A_{21} & A_{22} \end{bmatrix} t\right) = \begin{bmatrix} 0 & 0 \\ 0 & e^{A_{22}t} \end{bmatrix}. \tag{2}$$

That is, we are interested in conditions that guarantee that the coupling blocks (1,2) and (2,1) will vanish (in addition to the (1,1)-block).

In addition to being an interesting matrix problem, the result can be applied to control systems that exhibit significantly different time scales, such as systems with high-gain feedback on some states. For example, consider the system

$$\dot{x}_{\mathrm{f}}(t) = A_{11}x_{\mathrm{f}}(t) + A_{12}x_{\mathrm{s}}(t) + u(t) \tag{3}$$

$$\dot{x}_{\mathrm{s}}(t) = A_{21}x_{\mathrm{f}}(t) + A_{22}x_{\mathrm{s}}(t), \tag{4}$$

with state feedback on the states $x_{\mathrm{f}}(t)$ (index f for “fast” and s for “slow”),

$$u(t) = -K(\alpha)x_{\mathrm{f}}(t). \tag{5}$$

The matrix function $K(\alpha)$ then represents the feedback gain parametrized by α. The feedback system is depicted in Fig. 1. A more general multi-loop feedback system with additional reference inputs is considered in [1].

The matrix exponential (1) is the fundamental matrix (see e.g. [2]) of the feedback system (3)–(5). The limit (2) means that the dynamics of $x_{\mathrm{f}}(t)$ and $x_{\mathrm{s}}(t)$ are decoupled in the limit as $K(\alpha)$ grows large. In this context, we seek to determine what type of feedback yields a decoupling of the states in feedback from the remaining ones in the limit as the feedback gains become arbitrarily large.

This question is of interest, for example, when designing multi-loop control systems with high-gain inner loops, since a decoupling of the states allows for a simplified system description and, hence, a simplified control

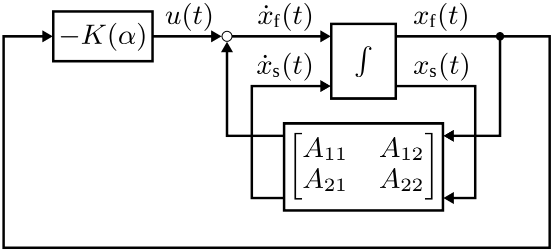

$-K(\alpha)$
$u(t)$
$\dot{x}_{\mathrm{f}}(t)$
$x_{\mathrm{f}}(t)$
$\dot{x}_{\mathrm{s}}(t)$
$\int$
$x_{\mathrm{s}}(t)$
$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$

Figure 1. Linear system with feedback on the first part of the state vector, the "fast" states x_{f}.

design. The matrix result herein is applied in [1] to derive a time-scale separation algorithm for a cascaded control system with high-gain inner feedback loops. The algorithm yields a system description that includes the plant dynamics and the effect of the inner feedback loops. The obtained representation is useful, for example, for designing an outer-loop controller. This methodology is applied in the design of a cascaded feedback control system for an inverted pendulum in [1] and for a balancing cube (a multi-body 3-D inverted pendulum) in [3].

Related to the problem studied herein is the work by Campbell et al., [4, 5]. The authors consider the matrix exponential with its argument being a polynomial in $1/\varepsilon$ and derive conditions for its convergence in the limit as $\varepsilon \to 0^+$. In [4], for example, Campbell et al. present a necessary and sufficient condition for pointwise convergence of

$$\exp((A + B/\varepsilon)t), \quad t > 0 \tag{6}$$

as $\varepsilon \to 0^+$. While they are interested in general convergence to *some* limit, we seek conditions that yield the *particular* limit (2); that is, where the cross coupling blocks (1,2) and (2,1) vanish.

Before deriving the technical results, this article continues with notation and preliminaries in Sec. 2. In Sec. 3, we present a sufficient condition for (2) that is based on the log norm of $K(\alpha)$ (to be defined in equation (8) of Sec. 2) and that makes no prior assumption on the function type of $K(\alpha)$ (Theorem 1). In Sec. 4, we present a necessary and sufficient convergence condition for the case when $K(\alpha)$ is affine (Theorem 4), and we give another sufficient condition for the case when $K(\alpha)$ has an affine term and an additional term in α^r, $r \geq 2$ (Theorem 5). The latter two results are based on [4, 5]. Numerical examples illustrating the applicability of the different theorems are given throughout in Sec. 3 and 4. The article concludes with remarks in Sec. 5.

A preliminary version of the result in Sec. 3 (Theorem 1) was first published in [1].

2. Notation and Preliminaries

We use $\mathbb{R}$, $\mathbb{C}$, and $\mathbb{R}^+$ to denote real numbers, complex numbers, and nonnegative real numbers, respectively. For the derivations in the paper, we work exclusively with the vector 2-norm and its induced matrix norm; that is, for $x \in \mathbb{C}^n$ and $A \in \mathbb{C}^{p\times n}$,

$$\|x\| = \left(\sum\nolimits_{i=1}^{n} |x_i|^2\right)^{1/2}, \quad \|A\| = \max_{\|x\|=1} \|Ax\|. \tag{7}$$

For $A \in \mathbb{C}^{n\times n}$, $\mu(A)$ denotes the *log norm* of A (associated with the 2-norm), [6],

$$\mu(A) := \max\{\mu | \mu \,\text{an eigenvalue of}\, (A + A^*)/2\}, \tag{8}$$

where A^* is the conjugate transpose of A. We shall exploit the following properties of $\mu(A)$, [6]: for $A, B \in \mathbb{C}^{n\times n}$ and $t \in \mathbb{R}^+$,

$$\|e^{At}\| \le e^{\mu(A)t} \tag{9}$$
$$\mu(A) \le \|A\| \tag{10}$$
$$\mu(A + B) \le \mu(A) + \|B\|. \tag{11}$$

Let spec(A) denote the *spectrum of* $A \in \mathbb{C}^{n\times n}$ (the set of all eigenvalues of A ignoring algebraic multiplicity), and let OLHP denote the open left half plane in $\mathbb{C}$ (i.e. OLHP $:= \{x \in \mathbb{C} : \operatorname{Re} x < 0\}$), [7]. The matrix A is called *stable* if spec$(A) \subset$ OLHP, and it is called *semistable* if spec$(A) \subset$ OLHP $\cup \{0\}$ and, if $0 \in$ spec(A), then 0 is semisimple (i.e. its algebraic and geometric multiplicity are identical), [7]. The *index* of A, denoted Index A, is the smallest nonnegative integer j such that $\operatorname{rank} A^j = \operatorname{rank} A^{j+1}$, [7]. The *Darzin inverse* of A is the unique matrix A^{D} satisfying $AA^{\mathrm{D}} = A^{\mathrm{D}}A$, $A^{\mathrm{D}}AA^{\mathrm{D}} = A^{\mathrm{D}}$, and $A^{j+1}A^{\mathrm{D}} = A^j$ with $j = \operatorname{Index} A$, [7]. For $A, B \in \mathbb{C}^{n\times n}$, define $[A; B] := (I - B^{\mathrm{D}}B)A(I - B^{\mathrm{D}}B)$, [5], where I is the identity matrix.

The following two facts are useful in later derivations; their proofs are given in the appendix.

Fact 1 Consider the matrix differential equation

$$\dot{Z}(t) = A\,Z(t) + B\,U(t),\ t \geq 0,\ Z(0) = Z_0, \tag{12}$$

where $Z : \mathbb{R}^+ \to \mathbb{C}^{n\times p}$ continuously differentiable, $U : \mathbb{R}^+ \to \mathbb{C}^{m\times p}$ continuous, $A \in \mathbb{C}^{n\times n}$, $B \in \mathbb{C}^{n\times m}$, and $Z_0 \in \mathbb{C}^{n\times p}$. The unique solution of (12) is, for $t \geq 0$,

$$Z(t) = e^{At} Z_0 + \int_0^t e^{A(t-\tau)}\, B\,U(\tau)\,d\tau. \tag{13}$$

■

Fact 2 Let $A : [a, b] \to \mathbb{C}^{n\times m}$ be continuous. Then

$$\Big\| \int_a^b A(t)\,dt \Big\| \leq \int_a^b \|A(t)\|\,dt. \tag{14}$$

■

3. Condition Based on the Log-norm of $K(\alpha)$

A sufficient condition for (2) is the log norm (8) of $-K(\alpha)$ becoming arbitrarily small.

Theorem 1 Let $A = \left[\begin{smallmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right] \in \mathbb{C}^{(n+m)\times(n+m)}$, and let $K : \mathbb{R} \to \mathbb{C}^{n\times n}$ be a matrix function of the real parameter α. If $\lim_{\alpha\to\infty} \mu(-K(\alpha)) = -\infty$, then (2) holds for all $t > 0$; that is,

$$\lim_{\alpha\to\infty} \exp\left(\begin{bmatrix} A_{11} - K(\alpha) & A_{12} \\ A_{21} & A_{22} \end{bmatrix} t\right) = \begin{bmatrix} 0 & 0 \\ 0 & e^{A_{22}t} \end{bmatrix}.$$

■

The proof of this result is deferred to Sec. 3.2. It is based on the matrix differential equation that is solved by the matrix exponential (1), and bounding its solution using a Gronwall-type inequality. The required lemmas are presented in Sec. 3.1. Numerical examples and remarks for Theorem 1 are presented in Sec. 3.3.

3.1 Lemmas

The following Gronwall-type inequality is adapted from [8]:

Lemma 1 Let $v(t)$, $a(t)$, $b(t)$ be real-valued, nonnegative, continuous functions on $J = [t_0, t_1]$. Let $\kappa(t,s)$ be a real-valued, nonnegative, continuous function for $t_0 \le s \le t \le t_1$, and suppose

$$v(t) \le a(t) + b(t) \int_{t_0}^{t} \kappa(t,s) v(s)\, ds, \quad t \in J.$$

Then

$$v(t) \le \bar{a}(t) \exp\left(\bar{b}(t) \int_{t_0}^{t} \bar{\kappa}(t,s)\, ds \right), \quad t \in J,$$

where $\bar{a}(t) := \sup_{\tau \in [t_0,t]} a(\tau)$, $\bar{b}(t) := \sup_{\tau \in [t_0,t]} b(\tau)$, and $\bar{\kappa}(t,s) := \sup_{\tau \in [s,t]}$ $\kappa(\tau, s)$. ■

Proof. The proof is given in [8] (Theorem 1.9). □

Consider the matrix differential equation

$$\dot{X}(t) = (A_{11} - K(\alpha))X(t) + A_{12}Y(t), \qquad X(0) = X_0 \qquad (15)$$

$$\dot{Y}(t) = A_{21}X(t) + A_{22}Y(t), \qquad Y(0) = Y_0 \qquad (16)$$

with $X : \mathbb{R}^+ \to \mathbb{C}^{n \times p}$ and $Y : \mathbb{R}^+ \to \mathbb{C}^{m \times p}$ continuously differentiable, and complex matrices A_{11}, A_{12}, A_{21}, A_{22}, $K(\alpha)$, X_0, and Y_0 of appropriate dimensions. Note that by Fact 1, the unique solutions to (15) and (16) are, for all $t \ge 0$,

$$X(t) = e^{(A_{11} - K(\alpha))t} X_0 + \int_0^t e^{(A_{11} - K(\alpha))(t-\tau)} A_{12} Y(\tau)\, d\tau \qquad (17)$$

$$Y(t) = e^{A_{22}t} Y_0 + \int_0^t e^{A_{22}(t-\tau)} A_{21} X(\tau)\, d\tau. \qquad (18)$$

The key in the proof of Theorem 1 is to consider the solutions (17) and (18), and to compute their limit as $\alpha \to \infty$. This is made precise in the following two lemmas:

Lemma 2 Consider the solutions (17) and (18) with the initial conditions $X_0 = I$ and $Y_0 = 0$. If $\lim_{\alpha \to \infty} \mu(-K(\alpha)) = -\infty$, then for $t > 0$,

$$\lim_{\alpha \to \infty} X(t) = 0 \quad \text{and} \quad \lim_{\alpha \to \infty} Y(t) = 0. \qquad (19)$$

■

LEMMA 3 Consider the solutions (17) and (18) with the initial conditions $X_0 = 0$ and $Y_0 = I$. If $\lim_{\alpha\to\infty} \mu(-K(\alpha)) = -\infty$, then for $t > 0$,

$$\lim_{\alpha\to\infty} X(t) = 0 \quad \text{and} \quad \lim_{\alpha\to\infty} Y(t) = e^{A_{22}t}. \tag{20}$$

■

Proof of Lemma 2. Since $\lim_{\alpha\to\infty} \mu(-K(\alpha)) = -\infty$, there exists an $\alpha_0 \in \mathbb{R}$ such that for all $\alpha \geq \alpha_0$

$$\mu(A_{11} - K(\alpha)) \leq \|A_{11}\| + \mu(-K(\alpha)) < 0, \tag{21}$$

$$\mu(A_{11} - K(\alpha)) - \|A_{22}\| < -1. \tag{22}$$

In the following, we consider sufficiently large α such that $\alpha \geq \alpha_0$.

Substituting (17) into (18) and using the initial conditions $X_0 = I$ and $Y_0 = 0$ yields

$$\begin{aligned} Y(t) &= \int_0^t e^{A_{22}(t-\tau)} A_{21} e^{(A_{11}-K(\alpha))\tau}\, d\tau \\ &\quad + \int_0^t\!\!\int_0^\tau e^{A_{22}(t-\tau)} A_{21} e^{(A_{11}-K(\alpha))(\tau-s)} A_{12} Y(s)\, ds\, d\tau \\ &= \int_0^t e^{A_{22}(t-\tau)} A_{21} e^{(A_{11}-K(\alpha))\tau}\, d\tau \\ &\quad + \int_0^t\!\!\int_s^t e^{A_{22}(t-\tau)} A_{21} e^{(A_{11}-K(\alpha))(\tau-s)} A_{12} Y(s)\, d\tau\, ds, \end{aligned} \tag{23}$$

where the order of integration in the last term was interchanged. This is valid by Fubini's theorem, [9, Prop. 5.36], and the facts that the integrand is continuous, and the integration region can be expressed in either of the two ways: $\{(\tau, s) : 0 \leq \tau \leq t, 0 \leq s \leq \tau\}$ or $\{(\tau, s) : 0 \leq s \leq t, s \leq \tau \leq t\}$.

Using (9), (10), Fact 2, and submultiplicativity of the induced matrix norm, we obtain the inequality

$$\begin{aligned} \|Y(t)\| &\leq \|A_{21}\| \int_0^t \|e^{A_{22}(t-\tau)}\| \|e^{(A_{11}-K(\alpha))\tau}\|\, d\tau \\ &\quad + \|A_{21}\|\|A_{12}\| \int_0^t\!\!\int_s^t \|e^{A_{22}(t-\tau)}\| \|e^{(A_{11}-K(\alpha))(\tau-s)}\|\, d\tau\, \|Y(s)\|\, ds \\ &\leq \|A_{21}\| \int_0^t e^{\|A_{22}\|(t-\tau)} e^{\mu(A_{11}-K(\alpha))\tau}\, d\tau \end{aligned}$$

$$+ \|A_{21}\|\|A_{12}\| \int_0^t \int_s^t e^{\|A_{22}\|(t-\tau)} e^{\mu(A_{11}-K(\alpha))(\tau-s)}\, d\tau\, \|Y(s)\|\, ds \tag{24}$$

$$= a(t) + \int_0^t \kappa(t,s)\|Y(s)\|\, ds, \tag{25}$$

where $a(t) := \|A_{21}\| \int_0^t e^{\|A_{22}\|(t-\tau)} e^{\mu(A_{11}-K(\alpha))\tau}\, d\tau$ and $\kappa(t,s) := \|A_{21}\|$ $\|A_{12}\| \int_s^t e^{\|A_{22}\|(t-\tau)} e^{\mu(A_{11}-K(\alpha))(\tau-s)}\, d\tau$. Applying Lemma 1 to (25) yields, for all $t \geq 0$,

$$\|Y(t)\| \leq \bar{a}(t) \exp\left(\int_0^t \bar{\kappa}(t,s)\, ds\right), \tag{26}$$

where $\bar{a}(t) = \sup_{\tau\in[0,t]} a(\tau)$ and $\bar{\kappa}(t,s) = \sup_{\tau\in[s,t]} \kappa(\tau,s)$.

Next, we derive bounds for $a(t)$, $\bar{a}(t)$ and $\kappa(t,s)$, $\bar{\kappa}(t,s)$ using the properties (21), (22). First,

$$\begin{aligned} a(t) &= \|A_{21}\| e^{\|A_{22}\|t} \int_0^t e^{(\mu(A_{11}-K(\alpha))-\|A_{22}\|)\tau}\, d\tau \\ &= \frac{\|A_{21}\|}{\xi(\alpha)} \big(e^{\|A_{22}\|t} - \underbrace{e^{\mu(A_{11}-K(\alpha))t}}_{\in(0,1] \text{ by } (21)}\big) \\ &\leq \frac{\|A_{21}\|}{\xi(\alpha)} e^{\|A_{22}\|t} = \frac{M_1(t)}{\xi(\alpha)}, \end{aligned}$$

where $\xi(\alpha) := \|A_{22}\| - \mu(A_{11}-K(\alpha)) > 1$ by (22), and $M_1(t) := \|A_{21}\| e^{\|A_{22}\|t}$ ≥ 0 is a continuous function in t. Therefore,

$$\bar{a}(t) = \sup_{\tau\in[0,t]} a(\tau) \leq \sup_{\tau\in[0,t]} \frac{\|A_{21}\|}{\xi(\alpha)} e^{\|A_{22}\|\tau} = \frac{\|A_{21}\|}{\xi(\alpha)} e^{\|A_{22}\|t} = \frac{M_1(t)}{\xi(\alpha)}. \tag{27}$$

Similarly, we obtain a bound for $\kappa(t,s)$. With $s \leq t$,

$$\begin{aligned} \kappa(t,s) &= \|A_{21}\|\|A_{12}\| e^{\|A_{22}\|t} e^{-\mu(A_{11}-K(\alpha))s} \int_s^t e^{(\mu(A_{11}-K(\alpha))-\|A_{22}\|)\tau}\, d\tau \\ &= \frac{\|A_{21}\|\|A_{12}\|}{\xi(\alpha)} \big(e^{\|A_{22}\|(t-s)} - \underbrace{e^{\mu(A_{11}-K(\alpha))(t-s)}}_{\in(0,1] \text{ by } (21)}\big) \\ &\leq \frac{\|A_{21}\|\|A_{12}\|}{\xi(\alpha)} e^{\|A_{22}\|t} = \frac{M_2(t)}{\xi(\alpha)}, \end{aligned}$$

where $M_2(t) := \|A_{21}\|\|A_{12}\|e^{\|A_{22}\|t} \geq 0$ is a continuous function in t. Therefore,

$$\begin{aligned}\bar{\kappa}(t,s) &= \sup_{\tau\in[s,t]} \kappa(\tau,s) \leq \sup_{\tau\in[s,t]} \frac{\|A_{21}\|\|A_{12}\|}{\xi(\alpha)} e^{\|A_{22}\|\tau} = \frac{\|A_{21}\|\|A_{12}\|}{\xi(\alpha)} e^{\|A_{22}\|t} \\ &= \frac{M_2(t)}{\xi(\alpha)}. \end{aligned} \tag{28}$$

With (27) and (28), we can now bound (26),

$$\begin{aligned}\|Y(t)\| &\leq \frac{M_1(t)}{\xi(\alpha)} \exp\left(\int_0^t \frac{M_2(t)}{\xi(\alpha)}\, ds\right) = \frac{M_1(t)}{\xi(\alpha)} \exp\left(\frac{M_2(t)}{\xi(\alpha)} t\right) \\ &\leq \frac{M_1(t)}{\xi(\alpha)} e^{tM_2(t)} = \frac{M(t)}{\xi(\alpha)}, \end{aligned} \tag{29}$$

where $M(t) := M_1(t)e^{tM_2(t)} \geq 0$ is continuous. Since $\lim_{\alpha\to\infty} \xi(\alpha) = \infty$, $\lim_{\alpha\to\infty} Y(t) = 0$ follows directly from (29). Furthermore, with (17) and $X_0 = I$,

$$\begin{aligned}\|X(t)\| &\leq e^{\mu(A_{11}-K(\alpha))t} + \|A_{12}\| \int_0^t \underbrace{e^{\mu(A_{11}-K(\alpha))(t-\tau)}}_{\in(0,1]} \|Y(\tau)\|\, d\tau \\ &\leq e^{\mu(A_{11}-K(\alpha))t} + \frac{\|A_{12}\|}{\xi(\alpha)} \int_0^t M(\tau)\, d\tau \end{aligned}$$

Therefore, $\lim_{\alpha\to\infty} X(t) = 0$ for $t > 0$. □

Proof of Lemma 3. The proof is essentially analogous to the proof of Lemma 2.

Let $\alpha \geq \alpha_0$ such that (21) and (22) hold. Substituting (18) into (17) and using the initial conditions $X_0 = 0$ and $Y_0 = I$ yields, after interchange of integration in the second term,

$$\begin{aligned}X(t) = &\int_0^t e^{(A_{11}-K(\alpha))(t-\tau)} A_{12} e^{A_{22}\tau}\, d\tau \\ &+ \int_0^t\int_s^t e^{(A_{11}-K(\alpha))(t-\tau)} A_{12} e^{A_{22}(\tau-s)} A_{21} X(s)\, d\tau\, ds, \end{aligned}$$

and, therefore,

$$\|X(t)\| \leq \|A_{12}\| \int_0^t e^{\mu(A_{11}-K(\alpha))(t-\tau)} e^{\|A_{22}\|\tau}\, d\tau$$

$$+ \|A_{12}\|\|A_{21}\| \int_0^t \int_s^t e^{\mu(A_{11}-K(\alpha))(t-\tau)} e^{\|A_{22}\|(\tau-s)} \, d\tau \, \|X(s)\| \, ds. \tag{30}$$

Now, consider the substitutions $\tau \to t - \tau$ for the first term in (30) and $\tau \to t + s - \tau$ for the inner integral of the second term, which yields

$$\begin{aligned} \|X(t)\| \leq \|A_{12}\| \int_0^t e^{\|A_{22}\|(t-\tau)} e^{\mu(A_{11}-K(\alpha))\tau} \, d\tau \\ + \|A_{12}\|\|A_{21}\| \int_0^t \int_s^t e^{\|A_{22}\|(t-\tau)} e^{\mu(A_{11}-K(\alpha))(\tau-s)} \, d\tau \, \|X(s)\| \, ds. \end{aligned} \tag{31}$$

Comparing this inequality to (24), we find that (31) is obtained from (24) by the substitutions $\|Y(\cdot)\| \to \|X(\cdot)\|$, $\|A_{12}\| \to \|A_{21}\|$, and $\|A_{21}\| \to \|A_{12}\|$. Therefore, we can derive an upper bound on $\|X(t)\|$ the same way as in the proof of Lemma 2. Corresponding to (29) we get, for all $t \geq 0$, $\|X(t)\| \leq \frac{L(t)}{\xi(\alpha)}$, where the continuous $L(t) \geq 0$ is obtained from $M(t)$ by substituting $\|A_{12}\| \to \|A_{21}\|$ and $\|A_{21}\| \to \|A_{12}\|$. Thus, $\lim_{\alpha\to\infty} X(t) = 0$. Furthermore, with (18) and $Y_0 = I$,

$$\begin{aligned} \|Y(t) - e^{A_{22}t}\| &\leq \|A_{21}\| \int_0^t e^{\|A_{22}\|(t-\tau)} \|X(\tau)\| \, d\tau \\ &\leq \frac{\|A_{21}\|}{\xi(\alpha)} \int_0^t e^{\|A_{22}\|(t-\tau)} L(\tau) \, d\tau. \end{aligned}$$

Therefore, $\lim_{\alpha\to\infty}\|Y(t) - e^{A_{22}t}\| = 0$, and thus $\lim_{\alpha\to\infty} Y(t) = e^{A_{22}t}$. □

3.2 Proof of Theorem 1

Using the Lemmas 2 and 3, we now prove Theorem 1.

Proof of Theorem 1. By Fact 1, the matrix exponential $\mathcal{X}(t) := \exp\left(\begin{bmatrix} A_{11}-K(\alpha) & A_{12} \\ A_{21} & A_{22} \end{bmatrix} t\right)$ is the unique solution to the matrix differential equation

$$\dot{\mathcal{X}}(t) = \begin{bmatrix} A_{11} - K(\alpha) & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \mathcal{X}(t), \ t \geq 0, \ \mathcal{X}(0) = I. \tag{32}$$

Note that $\mathcal{X} : \mathbb{R}^+ \to \mathbb{C}^{(n+m)\times(n+m)}$ is continuously differentiable. By subdividing $\mathcal{X}(t) = \begin{bmatrix} \mathcal{X}_{11}(t) & \mathcal{X}_{12}(t) \\ \mathcal{X}_{21}(t) & \mathcal{X}_{22}(t) \end{bmatrix}$ into block matrices of appropriate dimensions, we can write (32) equivalently as

$$\begin{bmatrix}\dot{\mathcal{X}}_{11}(t)\\ \dot{\mathcal{X}}_{21}(t)\end{bmatrix}=\begin{bmatrix}A_{11}-K(\alpha) & A_{12}\\ A_{21} & A_{22}\end{bmatrix}\begin{bmatrix}\mathcal{X}_{11}(t)\\ \mathcal{X}_{21}(t)\end{bmatrix},\quad \begin{bmatrix}\mathcal{X}_{11}(0)\\ \mathcal{X}_{21}(0)\end{bmatrix}=\begin{bmatrix}I\\ 0\end{bmatrix}, \tag{33}$$

$$\begin{bmatrix}\dot{\mathcal{X}}_{12}(t)\\ \dot{\mathcal{X}}_{22}(t)\end{bmatrix}=\begin{bmatrix}A_{11}-K(\alpha) & A_{12}\\ A_{21} & A_{22}\end{bmatrix}\begin{bmatrix}\mathcal{X}_{12}(t)\\ \mathcal{X}_{22}(t)\end{bmatrix},\quad \begin{bmatrix}\mathcal{X}_{12}(0)\\ \mathcal{X}_{22}(0)\end{bmatrix}=\begin{bmatrix}0\\ I\end{bmatrix}. \tag{34}$$

Note that (33) and (34) represent the matrix ODEs considered in Lemmas 2 and 3, respectively. Using these two lemmas, we therefore conclude, for $t > 0$,

$$\lim_{\alpha\to\infty}\exp\left(\begin{bmatrix}A_{11}-K(\alpha) & A_{12}\\ A_{21} & A_{22}\end{bmatrix}t\right)=\lim_{\alpha\to\infty}\begin{bmatrix}\mathcal{X}_{11}(t) & \mathcal{X}_{12}(t)\\ \mathcal{X}_{21}(t) & \mathcal{X}_{22}(t)\end{bmatrix}=\begin{bmatrix}0 & 0\\ 0 & e^{A_{22}t}\end{bmatrix}.$$

□

3.3 Examples and Remarks

We apply Theorem 1 to the introductory example.

Example 1—Discretization and high-gain feedback Consider the system (3), (4) with feedback (5) and $K(\alpha)$ diagonal with diagonal elements $k_i(\alpha)$. Assume we are interested in a discrete-time description of the closed-loop system (3)–(5) at a rate $T > 0$. The discretized system reads

$$\begin{bmatrix}x_{\mathrm{f}}(t+T)\\ x_{\mathrm{s}}(t+T)\end{bmatrix}=\exp\left(\begin{bmatrix}A_{11}-K(\alpha) & A_{12}\\ A_{21} & A_{22}\end{bmatrix}T\right)\begin{bmatrix}x_{\mathrm{f}}(t)\\ x_{\mathrm{s}}(t)\end{bmatrix}. \tag{35}$$

Now, assume $k_i(\alpha) \geq \alpha$, that is, the individual controller gains are at least as large as α. Then,

$$\lim_{\alpha\to\infty}\mu(-K(\alpha))=\lim_{\alpha\to\infty}\max_i\big(-k_i(\alpha)\big)\leq\lim_{\alpha\to\infty}-\alpha=-\infty, \tag{36}$$

and, by Theorem 1, (35) becomes

$$x_{\mathrm{f}}(t+T)=0 \tag{37}$$

$$x_{\mathrm{s}}(t+T)=e^{A_{22}T}x_{\mathrm{s}}(t) \tag{38}$$

in the limit as $\alpha \to \infty$; that is, the slow and fast dynamics are decoupled. ■

In [1], Theorem 1 is applied to a linear system more general than (3), (4), with additional reference inputs changing at the rate T. A discrete-time system at the rate T is obtained, which is then used to design an outer-loop controller that commands the reference inputs.

Remark 1 Note that the function $K(\alpha)$ is not given explicitly in Example 1. The estimate $k_i(\alpha) \geq \alpha$ with the diagonal structure of $K(\alpha)$ is enough to check the condition of Theorem 1. In contrast, the convergence results in Sec. 4 require an explicit description of $K(\alpha)$. ■

Example 2 Consider (1) with

$$A = \left[\begin{array}{cc|c} 0 & 0 & 0 \\ 0 & 0 & 1 \\ \hline 0 & 0 & 0 \end{array}\right], \; K(\alpha) = \begin{bmatrix} \alpha & \alpha^2 \\ 0 & \alpha \end{bmatrix}, \text{ and } \; t = 1.$$

Notice that $-K(\alpha)$ is stable for all $\alpha > 0$ (both eigenvalues are $-\alpha$), and that both eigenvalues go to negative infinity as $\alpha \to \infty$. But $\lim_{\alpha\to\infty} \mu(-K(\alpha)) = \lim_{\alpha\to\infty} \max\{-\alpha + \frac{1}{2}\alpha^2, -\alpha - \frac{1}{2}\alpha^2\} = \infty$, and the limit of (1) for $\alpha \to \infty$ is (can be computed using [7, Fact 11.14.2])

$$\left[\begin{array}{cc|c} 0 & 0 & -1 \\ 0 & 0 & 0 \\ \hline 0 & 0 & 1 \end{array}\right],$$

which is clearly different from (2) in the (1,2)-block. ■

Remark 2 The preceding example shows that it does not suffice for (2) to hold that the eigenvalues of $-K(\alpha)$ tend to negative infinity. ■

4. $K(\alpha)$ with Special Functional Dependency

Theorem 1 gives a sufficient condition for (2), which is based on the log norm (8). In this section, results are derived when $K(\alpha)$ has a particular polynomial structure; namely

$$K(\alpha) = K_0 + \alpha K_1, \quad \text{or} \tag{39}$$

$$K(\alpha) = K_0 + \alpha K_1 + \alpha^r K_2, \; r \geq 2. \tag{40}$$

For the affine case (39), a necessary and sufficient condition is derived; for (40), we present a sufficient condition.

The results of this section are based on [4, 5], and, in particular on:

THEOREM 2—(THM. 1 IN [4]) Let $A, B \in \mathbb{C}^{n\times n}$. Then $e^{(A+B/\varepsilon)t}$ converges pointwise as $\varepsilon \to 0^+$ for $t > 0$ if and only if B is semistable. If B is semistable, then

$$\lim_{\varepsilon\to 0^+} e^{(A+B/\varepsilon)t} = e^{(I-BB^{\mathrm{D}})At}(I - BB^{\mathrm{D}}). \tag{41}$$

■

THEOREM 3—(THM. 1 IN [5]) Suppose $\operatorname{Index} C = 1$ and C is semistable. Then

$$e^{(A+B/\varepsilon+C/\varepsilon^r)t}, \quad r > 1, \tag{42}$$

converges as $\varepsilon \to 0^+$ for an $r \geq 2$, for all $t > 0$, if and only if $[B; C]$ is semistable. Suppose $[B; C]$ is semistable. If $r > 2$, then (42) converges to

$$e^{[[A;C];[B;C]]t}(I - [B; C]^{\mathrm{D}}[B; C])(I - C^{\mathrm{D}}C); \tag{43}$$

if $r = 2$, then the limit of (42) is the same as (43) except a term

$$-[[BC^D B; C]; [B; C]]t \tag{44}$$

is added into the exponential. ■

4.1 $K(\alpha)$ Affine

We study the limit of (1) with affine $K(\alpha)$ as in (39). The following result provides a necessary and sufficient condition for (2). It is obtained using Theorem 2.

THEOREM 4 Let $A = \left[\begin{smallmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right] \in \mathbb{C}^{(m+n)\times(m+n)}$, and let $K(\alpha) = K_0 + \alpha K_1$ with $K_0, K_1 \in \mathbb{C}^{n\times n}$ and $\alpha \in \mathbb{R}$. Then, (2) holds for $t > 0$ if and only if $-K_1$ is stable. ■

Proof. We first prove sufficiency. Let

$$\tilde{A} := \begin{bmatrix} A_{11} - K_0 & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad \tilde{B} := \begin{bmatrix} -K_1 & 0 \\ 0 & 0 \end{bmatrix}. \tag{45}$$

Since $-K_1$ is stable, $\tilde{B}$ is semistable and, from Theorem 2 it follows (by substituting $1/\varepsilon$ with α) that

$$\lim_{\alpha\to\infty} e^{(\tilde{A}+\alpha\tilde{B})t} = \lim_{\varepsilon\to 0^+} e^{(\tilde{A}+\tilde{B}/\varepsilon)t} = e^{(I-\tilde{B}\tilde{B}^{\mathrm{D}})\tilde{A}t}(I - \tilde{B}\tilde{B}^{\mathrm{D}}). \tag{46}$$

Since $-K_1$ is stable, it is invertible, and $\tilde{B}^{\mathrm{D}} = \left[\begin{smallmatrix} -K_1^{-1} & 0 \\ 0 & 0 \end{smallmatrix}\right]$. Hence, we have

$$\begin{aligned} e^{(I-\tilde{B}\tilde{B}^{\mathrm{D}})\tilde{A}t} &= \exp\left(\begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} A_{11}-K_0 & A_{12} \\ A_{21} & A_{22} \end{bmatrix} t\right) \\ &= \exp\left(\begin{bmatrix} 0 & 0 \\ A_{21} & A_{22} \end{bmatrix} t\right) = \begin{bmatrix} I & 0 \\ * & e^{A_{22}t} \end{bmatrix}, \end{aligned}$$

where the last equality follows from [7, Fact 11.14.2], and $*$ is a placeholder left unspecified. Therefore, we get from (46)

$$\begin{aligned} \lim_{\alpha\to\infty} e^{(\tilde{A}+\alpha\tilde{B})t} &= e^{(I-\tilde{B}\tilde{B}^{\mathrm{D}})\tilde{A}t}(I-\tilde{B}\tilde{B}^{\mathrm{D}}) \\ &= \begin{bmatrix} I & 0 \\ * & e^{A_{22}t} \end{bmatrix}\begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & e^{A_{22}t} \end{bmatrix}, \end{aligned} \tag{47}$$

which completes the sufficiency part of the proof.

For the proof of necessity, assume (2) holds. First notice that, for the limit $\lim_{\alpha\to\infty} e^{(\tilde{A}+\alpha\tilde{B})t} = \lim_{\varepsilon\to 0^+} e^{(\tilde{A}+\tilde{B}/\varepsilon)t}$ to exist, it follows from Theorem 2 that $\tilde{B}$ is semistable. From the definition of $\tilde{B}$ in (45), it can be seen that this implies that $-K_1$ is semistable, which further implies that

$$\operatorname{spec}(-K_1) \subset \mathrm{OLHP} \cup \{0\}. \tag{48}$$

From $\tilde{B}$ semistable and Theorem 2, it follows that (46) holds. Hence, the limit in (46) is equal to the limit in (2), i.e.

$$e^{(I-\tilde{B}\tilde{B}^{\mathrm{D}})\tilde{A}t}(I-\tilde{B}\tilde{B}^{\mathrm{D}}) = \begin{bmatrix} 0 & 0 \\ 0 & e^{A_{22}t} \end{bmatrix}. \tag{49}$$

Now, let

$$\begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix} := e^{(I-\tilde{B}\tilde{B}^{\mathrm{D}})\tilde{A}t}. \tag{50}$$

Using $\tilde{B}^{\mathrm{D}} = \left[\begin{smallmatrix} -K_1^{\mathrm{D}} & 0 \\ 0 & 0 \end{smallmatrix}\right]$ and (50), it follows from (49) (by considering the first block column) that

$$\begin{bmatrix} E_{11} \\ E_{21} \end{bmatrix}(I - K_1K_1^{\mathrm{D}}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{51}$$

Since the matrix exponential is nonsingular [7, Prop. 11.2.8], E in (50) is nonsingular, and $\left[\begin{smallmatrix} E_{11} \\ E_{21} \end{smallmatrix}\right]$ has full column rank. Therefore, (51) implies $(I -$

$K_1K_1^{\mathrm{D}}) = 0 \Leftrightarrow K_1K_1^{\mathrm{D}} = I$. From this and the rank formula [7, Lemma 2.5.2] $n = \operatorname{rank}(I) = \operatorname{rank}(K_1K_1^{\mathrm{D}}) \leq \min\{\operatorname{rank}(K_1), \operatorname{rank}(K_1^{\mathrm{D}})\} \leq n$, it follows that K_1 has full rank. Thus, also $-K_1$ has full rank, which implies $0 \notin \operatorname{spec}(-K_1)$, [7, Cor. 2.6.6, Prop. 5.5.20]. This and (48) imply $\operatorname{spec}(-K_1) \in$ OLHP, i.e. $-K_1$ is stable. □

EXAMPLE 3 Consider

$$K(\alpha) = \alpha K_1 \quad \text{with} \quad K_1 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}.$$

Then $-K_1$ is stable, and, by Theorem 4, (2) holds. ■

REMARK 3 For $K(\alpha)$ as in Example 3, we compute $\lim_{\alpha\to\infty} \mu(-K(\alpha)) = \lim_{\alpha\to\infty} \max\{0, -2\alpha\} = 0$. Therefore, Example 3 shows that the condition in Theorem 1 is not a necessary condition. ■

4.2 $K(\alpha)$ with Additional Power of α

We study the limit of (1) with $K(\alpha)$ as in (40); that is, compared to (39), $K(\alpha)$ possesses an additional power α^r with $r \geq 2$. A sufficient condition for convergence is derived using Theorem 3. The condition is different from the sufficient condition in Theorem 1 (one does not imply the other) as shall be pointed out later.

THEOREM 5 Let $A = \left[\begin{smallmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right] \in \mathbb{C}^{(m+n)\times(m+n)}$, and let $K(\alpha) = K_0 + \alpha K_1 + \alpha^r K_2$ with $K_0, K_1, K_2 \in \mathbb{C}^{n\times n}$ and $\alpha, r \in \mathbb{R}$, $r \geq 2$. If $-K_2$ is stable, then (2) holds for $t > 0$. ■

Proof. Let $\tilde{A}$, $\tilde{B}$ be as in (45), and let $\tilde{C} := \left[\begin{smallmatrix} -K_2 & 0 \\ 0 & 0 \end{smallmatrix}\right]$. Since $-K_2$ is stable, $\tilde{C}$ is semistable. Furthermore, $\operatorname{Index}\tilde{C} = 1$ since $\operatorname{rank}(K_2^2) = \operatorname{rank}(-K_2)$ ($-K_2$ has full rank). Thus, the assumptions of Theorem 3 are satisfied.

With $\tilde{C}^{\mathrm{D}} = \begin{bmatrix} -K_2^{-1} & 0 \\ 0 & 0 \end{bmatrix}$, we get $(I - \tilde{C}^{\mathrm{D}}\tilde{C}) = \left[\begin{smallmatrix} 0 & 0 \\ 0 & I \end{smallmatrix}\right]$ and $[\tilde{B}; \tilde{C}] = (I - \tilde{C}^{\mathrm{D}}\tilde{C})\tilde{B}(I - \tilde{C}^{\mathrm{D}}\tilde{C}) = 0$, which is semistable. Therefore, by Theorem 3, $\lim_{\alpha\to\infty} e^{(\tilde{A}+\alpha\tilde{B}+\alpha^r\tilde{C})t} = \lim_{\varepsilon\to 0^+} e^{(\tilde{A}+\tilde{B}/\varepsilon+\tilde{C}/\varepsilon^r)t}$ converges to the limit specified by (43) and (44) where A, B, C are replaced by $\tilde{A}$, $\tilde{B}$, $\tilde{C}$. We next compute the expressions (43) and (44).

From $[\tilde{B}; \tilde{C}] = 0$, we get $(I - [\tilde{B}; \tilde{C}]^{\mathrm{D}}[\tilde{B}; \tilde{C}]) = I - 0^{\mathrm{D}}0 = I$. Furthermore,

$$[\tilde{A}; \tilde{C}] = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} A_{11} - K_0 & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & A_{22} \end{bmatrix},$$

and, hence,

$$[[\tilde{A};\tilde{C}];[\tilde{B};\tilde{C}]] = [\tilde{A};\tilde{C}] = \begin{bmatrix} 0 & 0 \\ 0 & A_{22} \end{bmatrix}. \tag{52}$$

Using these results, expression (43) yields the desired limit in (2)

$$e^{[[\tilde{A};\tilde{C}];[\tilde{B};\tilde{C}]]t}(I - [\tilde{B};\tilde{C}]^{\mathrm{D}}[\tilde{B};\tilde{C}])(I - \tilde{C}^{\mathrm{D}}\tilde{C}) = \begin{bmatrix} 0 & 0 \\ 0 & e^{A_{22}t} \end{bmatrix}.$$

Since

$$[\tilde{B}\tilde{C}^{\mathrm{D}}\tilde{B};\tilde{C}] = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} -K_1K_2^{-1}K_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} = 0,$$

expression (44) is 0. □

EXAMPLE 4 Consider

$$K(\alpha) = -\alpha I + \alpha^2 K_2 \quad \text{with} \quad K_2 = \begin{bmatrix} 0.1 & 1 \\ 0 & 0.1 \end{bmatrix}.$$

Then $-K_2$ is stable, and, by Theorem 5, (2) holds for any A. The instability of matrix $-K_1 = I$ is irrelevant for the limit. Notice that $\lim_{\alpha\to\infty} \mu(-K(\alpha)) = \lim_{\alpha\to\infty} \max\{\alpha + \frac{2\alpha^2}{5}, \alpha - \frac{3\alpha^2}{5}\} = \infty$. Hence, Theorem 1 is not helpful here. ■

EXAMPLE 5 Consider

$$K(\alpha) = \alpha K_1 + \alpha^r K_2 \ \text{ with } \ 1 < r < 2 \ \text{ and } \ K_1 = \begin{bmatrix} -2 & 0 \\ 0 & 1 \end{bmatrix}, \ K_2 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

Theorem 5 is not helpful here, since $r < 2$ (neither is Theorem 3). But $\mu(-K(\alpha)) = \max\{-\alpha, 2\alpha - \alpha^r\} \to -\infty$ as $\alpha \to \infty$; hence, (2) follows from Theorem 1. ■

REMARK 4 Examples 4 and 5 show that there are problems with $K(\alpha) = K_0 + \alpha K_1 + \alpha^r K_2$ that are covered by Theorem 1, but not by Theorem 5; and vice versa. In general, both theorems provide sufficient conditions for different problem classes. ■

5. Concluding Remarks

The three theorems obtained in this paper guarantee the convergence of the matrix exponential (1) to the limit as given in (2); essentially, a "large enough" $K(\alpha)$ in the (1,1)-block forces all but the (2,2)-block of the matrix exponential to tend to zero in the limit. Theorem 1 states a sufficient condition for (2) based on the log norm of $-K(\alpha)$; and Theorems 4 and 5 provide sufficient conditions for $K(\alpha)$ having a particular polynomial form. For the affine case (Theorem 4), the condition is also necessary.

The Theorems 4 and 5 herein are obtained using the results by Campbell et al. [4,5]. Theorem 1, however, is obtained independently of those results. Its method of proof is based on the matrix differential equation that is solved uniquely by the matrix exponential (1), and on bounding its solution using a Gronwall-type inequality. In contrast, Campbell et al. make use of Cauchy's integral formula to prove their result in [4], for example.

Appendix

A. Proof of Fact 1

Let $Z(t) = [z_1(t)\ z_2(t)\ \cdots\ z_p(t)]$, $Z_0 = [z_{0,1}\ z_{0,2}\ \cdots\ z_{0,p}]$, and $U(t) = [u_1(t)\ u_2(t)\ \cdots\ u_p(t)]$. Then, the i-th column of (12) is

$$\dot{z}_i(t) = A\, z_i(t) + B\, u_i(t),\ t \geq 0,\ z_i(0) = z_{0,i}. \tag{53}$$

Since $u_i(t)$ is continuous, the (vector-valued) ODE in (53) has the unique solution (see e.g. [2])

$$z_i(t) = e^{At} z_{0,i} + \int_0^t e^{A(t-\tau)}\, B\, u_i(\tau)\, d\tau, \tag{54}$$

for $t \geq 0$. The claim (13) then follows by stacking (54) as columns of $Z(t)$ for $i = 1, \ldots, p$.

B. Proof of Fact 2

First notice that for a vector-valued, integrable function $f : [a,b] \to \mathbb{C}^n$,

$$\left\| \int_a^b f(t)\, dt \right\| \leq \int_a^b \| f(t) \|\, dt \tag{55}$$

(see proof in [10, Thm. 6.25]; the argumentation given in [10] for a real-valued function f applies analogously for complex-valued f). Then,

$$\begin{aligned}\Big\| \int_a^b A(t)\, dt \Big\| &\underset{(7)}{=} \max_{\|x\|=1} \Big\| \int_a^b A(t)\, dt\ x \Big\| = \max_{\|x\|=1} \Big\| \int_a^b A(t) x\, dt \Big\| \\ &\underset{(55)}{\leq} \max_{\|x\|=1} \int_a^b \|A(t)x\|\, dt. \qquad (56)\end{aligned}$$

Since, for all $t \in \mathbb{R}^+$ and $x \in \mathbb{C}^n$ with $\|x\| = 1$, $\|A(t)x\| \leq \max_{\|x\|=1} \|A(t)x\| = \|A(t)\|$, and $\|A(t)\|$ is integrable ($A(\cdot)$ and $\|\cdot\|$ are continuous), it follows from monotonicity of the integral that

$$\max_{\|x\|=1} \int_a^b \|A(t)x\|\, dt \leq \max_{\|x\|=1} \int_a^b \|A(t)\|\, dt = \int_a^b \|A(t)\|\, dt. \qquad (57)$$

References

[1] S. Trimpe and R. D'Andrea, "A limiting property of the matrix exponential with application to multi-loop control," in *Proc. of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, P.R. China, Dec. 2009, pp. 6419–6425.

[2] P. Hartman, *Ordinary Differential Equations*, 2nd ed. Philadelphia: Society for Industrial and Applied Mathematics, 2002.

[3] S. Trimpe and R. D'Andrea, "The Balancing Cube – a dynamic scuplture as testbed for distributed estimation and control," *IEEE Control Systems Magazine*, Dec. 2012, in press.

[4] S. L. Campbell and N. J. Rose, "Singular perturbation of autonomous linear systems," *SIAM Journal on Mathematical Analysis*, vol. 10, no. 3, pp. 542–551, 1979.

[5] S. L. Campbell, "Singular perturbation of autonomous linear systems II," *Journal of Differential Equations*, vol. 29, no. 3, pp. 362 – 373, 1978.

[6] C. Moler and C. van Loan, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM Review*, vol. 45, no. 1, pp. 3–49, 2003.

[7] D. S. Bernstein, *Matrix mathematics: theory, facts, and formulas*, 2nd ed. Princeton, New Jersey: Princeton University Press, 2009.

[8] D. Bainov and P. Simeonov, *Integral inequalities and applications*, ser. Mathematics and its applications. East European Series. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1992, vol. 57.

[9] S. Ghorpade and B. Limaye, *A course in multivariable calculus and analysis*, ser. Undergraduate texts in mathematics. Springer New York, 2010.

[10] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. McGraw-Hill, 1976.

Curriculum Vitae

Sebastian Trimpe

born July 20, 1981 in Georgsmarienhütte, Germany

2008 – 2013	*ETH Zurich, Switzerland* Doctoral studies at the Institute for Dynamic Systems and Control (advisor: Prof. Raffaello D'Andrea), Department of Mechanical and Process Engineering; graduated with Dr. sc. ETH Zürich.
2005 – 2007	*Hamburg University of Technology, Germany* Graduate studies; graduated with Dipl.-Ing. in Electrical Engineering.
2005 – 2007	*Hamburg University of Technology / Northern Institute of Technology, Germany* Graduate studies (part-time); graduated with MBA in Technology Management.
2007	*University of California at Berkeley, USA* Visiting student researcher (8 months) at the Department of Mechanical Engineering (Prof. Tarek I. Zohdi).
2006 – 2007	*Airbus Germany, Bremen* Internship (6 months) at the Department of Aeroelastics.
2002 – 2005	*Hamburg University of Technology, Germany* Undergraduate studies; graduated with B.Sc. in General Engineering Science.
2002	*IBM Germany, Stuttgart* Internship (2 months) at the European Technical Center.
2001 – 2002	*Johanniter Unfall-Hilfe, Osnabrück, Germany* Civilian service.
2001	*Angelaschule Osnabrück, Germany* Abitur (high-school diploma).