2006 Program Overview
Chess SUPERB-IT

Edited by J. Sprinkle [1]

June 11–August 04, 2006

The Center for Hybrid & Embedded Software Systems

Department of Electrical Engineering and Computer Sciences
College of Engineering
University of California, Berkeley

[1]J. Sprinkle is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, and is the Executive Director of the Center for Hybrid and Embedded Software Systems (CHESS). e-mail: `sprinkle@EECS.Berkeley.Edu`

Contents

Overview

This Document

This projects description document provides an introductory look at our goals and objectives for the summer, and describes in some depth the technical topics which the students and mentors will investigate through research. As a student reading this document, please find your name and mentor's name under a project, read that chapter, and follow the instructions given in the *Preliminary Tasks* subsection, which will give you a good head start to having an excellent research achievement by the end of the summer.

As a general rule, project topics are fairly fixed—meaning that we spent a large amount of time devising topics that looked like they would intersect on some level with,

- the goals of Chess,
- student background, in terms of classes and performance,
- student aptitude, as we guessed from resumés,
- student preferences listed in the SUPERB application,
- mentor background and interests, and
- overall contribution to a cohesive project.

It should be noted, though, that these constraints were not necessarily applied in that order. When students arrive, or if need be prior to their arrival, we will refine their projects based on their feedback and any incoming research goals, to hope to align them for increased performance throughout the summer.

In the meantime, please do not hesitate to contact the project leader, Dr. Jonathan Sprinkle, or your mentor, with any questions about the program, your project, or what you can do to prepare for what we hope is the most exciting and interesting summer you have ever had.

Jonathan Sprinkle, Ph.D.
May 17, 2006
Berkeley, CA

About the programs

Chess

The Center for Hybrid and Embedded Software Systems (Chess[1]) is aimed at developing model-based and tool-supported design methodologies for real-time fault tolerant software on heterogeneous distributed platforms. We are bridging the gap between computer science and systems science by developing the foundations of a modern systems science that is simultaneously computational and physical. This represents a major departure from the current, separated structure of computer science (CS), computer engineering (CE), and electrical engineering (EE): it reintegrates information and physical sciences. The center is funded in part by an Information Technology Research (ITR) project from the National Science Foundation (NSF). It operates cooperatively with the Institute for Software Integrated Systems (ISIS[2]) at Vanderbilt University, and the Department of Mathematical Sciences at the University of Memphis.

SUPERB-IT—2006

The Center for Hybrid and Embedded Software Systems is proud to sponsor four undergraduate students from diverse backgrounds and cultures to participate in the Summer Undergraduate Program in Engineering Research at Berkeley (SUPERB-IT). These students will interact with individual mentors throughout the summer, and perform research and supporting activities in the area of hybrid and embedded systems.

Electrical Engineering & Computer Sciences (EECS)—SUPERB-IT

The Summer Undergraduate Program in Engineering Research at Berkeley–Information Technology (SUPERB–IT) in the Electrical Engineering and Computer Sciences (EECS) Department offers a group of talented undergraduate engineering students the opportunity to gain research experience. The program's objective is to provide research opportunities in engineering to students who have been historically underrepresented in the field for reasons of social, cultural, educational or economic barriers, by affirming students' motivation for graduate study and strengthening their qualifications.

From the College

The goal of the Summer Undergraduate Program in Engineering Research at Berkeley (SUPERB) is to provide research opportunities in engineering to students who have been historically underrepresented in the field for reasons of social, cultural, educational or economic barriers. The program provides students with the opportunity to gain research experience by participating in research projects with engineering faculty and graduate students. Upon completion of this program students will be better prepared and motivated to attend graduate school.

[1] http://chess.eecs.berkeley.edu/
[2] http://www.isis.vanderbilt.edu/

Part I

Goals & Objectives

Cohesive Project Vision

The SUPERB program is dedicated to providing undergraduate students with the opportunities and experiences of research. At Chess, where our research goals are at the intersection of traditional Electrical Engineering and Computer Science, this includes a wide variety of possible topics including robotics, systems theory, programming, image processing, algorithm development, simulation, and good old mathematics.

More importantly, the Chess philosophy is that new developments in traditional research areas will emerge from interdisciplinary cooperation between Electrical Engineering and Computer Science researchers. To reflect this, and the wide array of possible topics, we have chosen for the 2006 SUPERB program a set of projects that intersect in some areas, are independent in others, and will allow for inter-student cooperation to devise new solutions to unsolved problems.

It is important to note that several projects fit into more than one category. We believe this is typical of research at Berkeley, and will be reflective of research everywhere in the future. Our goal is to use this as a benefit for the student over the summer, where the student will learn how to think in terms of multiple goals at once, and will achieve results that can be interesting to experts in more than one field.

0.1 Highway traffic flow analysis and control

Significant inspiration of computer network topology and communication comes from an empirical understanding of how road networks function. This project takes foundational work by CHESS researchers in hybrid systems to investigate a macroscopic switching-mode model (SMM) of traffic. The goal is to learn how hybrid systems are used for traffic modeling, experiment with different techniques for reachability analysis, implement a controller for the system, and then study the system behavior in the presence of disturbances. The end result will be MATLAB simulations which show the impact of the work.

0.2 Tool for probabilistic safety verification of stochastic hybrid systems

At the heart of research in the CHESS Center is computational hybrid systems. The purpose of this project is to develop a computational tool to study some simple stochastic hybrid systems and to implement a methodology for stochastic reachability analysis for controlled SHS. For example, safety critical systems like air traffic control involves modeling their behavior as controlled stochastic hybrid systems (SHS), the quantification of the effect of external inputs and disturbances by way of simulation, and designing controllers that guarantee a certain safety criterion can both also be posed as stochastic hybrid systems problems. The end result of this project will be MATLAB simulations which show the impact of the work, as well as possible theoretical understandings in this rich area of research.

0.3 Autopilot for ultra-light flying wing

Embedded software is most interesting when it is actually doing something, and CHESS researchers have strong ties to autonomous systems. The purpose of this project is to develop an auto-pilot for a small, light fixed-wing aircraft

named the Zagi. This aircraft is interesting because it is inexpensive, simple and fast to deploy, and is virtually indestructible since it is made of foam. Development of the autopilot will allow for seamless simulation of high-level algorithms such as pursuit-evasion games, waypoint following, and loitering. The final goal is to deploy the autopilot interface into embedded hardware.

0.4 Viptos: A graphical development and simulation environment for TinyOS-based wireless sensor networks

Wireless Sensor Networks are a burgeoning area of research and application in embedded systems. The purpose of this project is to understand and further develop Viptos (Visual Ptolemy and TinyOS), an integrated graphical development and simulation environment for TinyOS-based wireless sensor networks. TinyOS is the operating systems for the Berkeley Motes, which are small embedded systems capable of collecting audio, temperature, radio, and other kinds of sensor data. A key piece of the TinyOS simulator is the ability to simulate a network topology rapidly once it seems to behave appropriately. Viptos extends the capabilities of the TinyOS Simulator to allow simulation of heterogeneous networks. The final goal is to produce complex simulations and enable programming support that has not been previously possible.

Cross-cutting Experiences

In addition to the science of research, we will also expose students to the reporting aspects of research. These include the techniques used for writing papers, technical skills such as the use of LaTeX, and having a "canned" version of what exactly it is you are working on in speech. A brief description of some of these experiences are given below.

0.5 LaTeX template design

One student will use newfound knowledge of LaTeX paper creation to revise the existing template that each student will use for their final paper composition.

0.6 Poster design

One student will use existing poster designs, and input from mentors regarding the philosophy behind a poster, to design a template which each student will use (or choose from) for their final presentation.

0.7 Concurrent Versioning System (CVS)

The technical craft of writing software, or any collaborative project, eventually runs into the problem of versioning, and multi-user problems such as overwriting, or file locking. CVS is a technology that solves some of these problems, and students will use it for their code, reporting, and website duties while participating in SUPERB-IT.

0.8 Reading group participation

All students will participate in a reading group which will meet weekly during their tenure in SUPERB-IT. The papers and philosophy toward exploration of the body of research will enhance student understanding of why it is important to publish, and also why it is important to write well when publishing. Individual sessions will be led by some of the students, and will intersect with reading they performed for their own research.

0.9 Website update

The CHESS website is easily configurable and modifiable via CVS and other authoring techniques. Students will keep the website updated to reflect weekly presentations, reading group discussions, the resources used by students for paper-writing and poster creation, and also social interactions ongoing in the SUPERB-IT program.

Part II

Projects

Project 1

Highway traffic flow analysis and control

Researcher: Dominique Duncan[1]
Mentor: Alex Kurzhanskiy[2]

1.1 Overview

Based on how the traffic is treated, traffic models can be classified into two broad categories: macroscopic and microscopic. In a macroscopic model, traffic is treated as a continuum and modeled by aggregated, fluid-like quantities such as density and flow; while in a microscopic model, the dynamics of individual vehicle-driver units and the interactions between these units and their surroundings are explicitly modeled.

This project investigates macroscopic switching-mode model (SMM) of traffic [1]. The goal is to learn how hybrid systems are used for traffic modeling, experiment with different techniques for reachability analysis, implement a controller for the system, and then, study the system behavior in the presence of disturbances.

1.2 Technical Tasks

- The first step is to get familiar with the switching-mode traffic model [1]: write the simulation, test it against different inputs, tune the parameters.
- Once the model is in place, perform reachability analysis using available methods and tools for reach set computation.
- Program a controller that steers the system to a given target.
- Introduce bounded disturbance to the model and do the control synthesis for an uncertain system.

The suggested programming environment for the listed tasks is MATLAB/Simulink.

1.3 Preparatory Tasks

Before arriving, the following tasks should be done by the student:

1. Do the reading:
    - chapters 1-5 of [2] - to get familiar with hybrid systems and related terminology;

[1] Dominique Duncan is a rising Senior at the University of Chicago, majoring in Math and Polish Literature.
[2] Alex Kurzhanskiy is a Ph.D. student at the EECS department of University of California, Berkeley.

- chapters 1, 3 of [3] - to find out about reachability computation;
- [1] - to learn about switching-mode model of highway traffic.

2. Refresh your knowledge of linear algebra.
3. Refresh your knowledge of MATLAB and Simulink. If you haven't used Simulink, take a look at it and play with simple examples.
4. Download and install Multi-Parametric Toolbox (MPT) [4] from `http://control.ee.ethz.ch/~mpt`.

1.4 Contact

For questions, suggestions, problems, etc. related to this project, email Alex Kurzhanskiy at `akurzhan@eecs.berkeley.edu`.

Bibliography

[1] L.Muñoz, X.Sun, R.Horowitz, L.Alvarez (2003). *Traffic density estimation with the cell transmission model*, In: *Proceedings of the American Control Conference*, Denver, Colorado, USA, pp.3750-3755.

[2] J.Lygeros (2004). *Lecture notes on hybrid systems*, available online: `http://robotics.eecs.berkeley.edu/~sastry/ee291e/lygeros.pdf`

[3] A.A.Kurzhanskiy, P.Varaiya (2006). *Ellipsoidal Toolbox - technical report*, available online: `http://www.eecs.berkeley.edu/~akurzhan/ellipsoids/ET_TechReport_v1.pdf`

[4] M.Kvasnica, P.Grieder, M.Baotić, M.Morari (2004). *Multi-Parametric Toolbox (MPT)*. In: R.Alur and G.J.Pappas, editors, *Hybrid Systems: Computation and Control*, Vol.2993 of *Lecture Notes in Computer Science*, pp.448-462. Springer-Verlag.

Project 2

Tool for probabilistic safety verification of stochastic hybrid systems

Researcher: Nandita Mitra[1]
Mentor: Saurabh Amin[2] *and Alessandro Abate*[3]

2.1 Overview

Study of safety critical systems like air traffic control involves modeling their behavior as controlled stochastic hybrid systems (SHS) [3], quantifying the effect of external inputs and disturbances by way of simulation, and designing controllers that guarantee a certain safety criterion [1]. The purpose of this project is to develop a computational tool to study some simple stochastic hybrid systems and to implement a methodology for stochastic reachability analysis for controlled SHS.

2.2 Technical Tasks

- Get familiar with controlled stochastic hybrid models by way of simple examples;
- Choose an interesting toy system and implement numerical simulation for the stochastic hybrid model of the system and study the effect of noise parameters on the simulated trajectories;
- Implement stochastic reachability algorithm to determine the maximal safe sets and optimal control law. Study the effect of noise parameters on the safety probability;
- Extend the implementation to multidimensional cases and address the issues related to speed and accuracy.

The suggested programming environment for the listed tasks is MATLAB.

2.3 Preparatory Tasks

Before arriving, the following tasks should be taken care of by the student:

[1]Nandita Mitra is a rising Senior at Rutgers University, majoring in Electrical Engineering.

[2]Saurabh Amin is a Ph.D. student in the systems program at the CEE department of University of California, Berkeley. Saurabh's general area of interest include control and estimation of hybrid, stochastic and distributed systems.

[3]Alessandro Abate is a Ph.D. student in EECS and is interested in Systems Theory

1. Do the reading and studying of:
   - chapters 1-5 of [4] - to get familiar with hybrid systems and related terminology;
   - introduction and example section of [1];
   - [5] - to get familiar with dynamic programming algorithm. In particular, focus on the stochastic dynamic programming section.
2. Refresh your knowledge of probability;
3. Refresh your knowledge of MATLAB;
4. Code a couple of MATLAB programs from [2] to get a feel of numerical methods for simulating stochastic differential equations.

2.4 Contact

For questions, suggestions, problems, etc. related to this project, email Saurabh Amin at `saurabh@eecs.berkeley.edu`.

Bibliography

[1] S.Amin, A.Abate, M.Prandini, J.Lygeros, S.Sastry (2006). *Reachability analysis for controlled discrete time stochastic hybrid systems*, In: *Hybrid Systems: Computation and Control*, Vol.3927 of *Lecture Notes in Computer Science* pp.49-63, Springer-Verlag.

[2] D.J.Higham. (2001) *An algorithmic introduction to numerical simulation of stochastic differential equations*, In: *SIAM Review*, 43(3), pp. 525-546.

[3] J.Lygeros (2004). *Columbus: Design of Embedded Controllers for Safety Critical Systems*, available online: `http://www.columbus.gr/finalreport/index.htm`

[4] J.Lygeros (2004). *Lecture notes on hybrid systems*, available online: `http://robotics.eecs.berkeley.edu/~sastry/ee291e/lygeros.pdf`

[5] M.Trick *A tutorial on dynamic programming*, available online: `http://mat.gsia.cmu.edu/classes/dynamic/dynamic.html`

Project 3

Autopilot for ultra-light flying wing

Researcher: Nashlie Sephus[1]
Mentor: Todd Templeton[2]

3.1 Overview

The purpose of this project is to develop an auto-pilot for a small, light fixed-wing aircraft named the Zagi. This aircraft is interesting because it is inexpensive, simple and fast to deploy, and is virtually indestructible since it is made of foam. It is also challenging from a control perspective because it is vulnerable to wind and can only carry a minimal payload.

Some good online references are listed below:

1. Trick R/C, home of the Zagi. The URL is `http://www.zagi.com`. The movies on this site are particularly interesting.
2. CRRCsim, a cross-platform model airplane simulation program that includes a Zagi model. The URL is `http://crrcsim.sourceforge.net`.

3.2 Technical Tasks

The first step is to develop optimal local trajectories given current wind conditions. These local trajectories will be used to determine the path that the vehicle should try to maintain between widely-spaced waypoints, and should trade off between overshooting corners and maintaining the desired trajectory.

The second step is to implement these local trajectories in the Zagi software, modifying the existing controller as necessary, to enable it to follow an incrementally-specified global trajectory. Initial tests will be performed using the CRRCsim simulator interfaced to the Zagi hardware, with final testing performed on a real Zagi at Richmond Field Station.

If both of the above steps are completed with time to spare, the final step is to implement a general wind-compensation scheme in the low-level controller.

[1] Nashlie Sephus is a rising Senior at Mississippi State University, majoring in Computer Engineering.

[2] Todd Templeton is a Ph.D. student at the EECS department of University of California, Berkeley. Todd's research interests include computer vision, particularly terrain reconstruction and vehicle motion estimation; artificial intelligence; control; embedded systems; and robotics, particularly unmanned aerial vehicles.

3.2.1 Optimal Local Trajectories

Consider the following situation: The vehicle is currently located at position x_0 (a three-vector) at orientation α_0 (roll, pitch, and yaw) with velocities $\dot{x}_0$ and $\dot{\alpha}_0$, and we are given waypoints x_1, x_2, and x_3, and a desired speed s, where we would like to follow the vector from x_1 to x_2 and then the vector from x_2 to x_3 (note that x_0 is not necessarily on either of these vectors). The exact location of x_1 is unimportant; it exists only to define the arrival vector for x_2. We would, however, like to be following the vector between x_2 and x_3 before we arrive at x_3. Given x_0, $\alpha_0, \dot{x}_0, \dot{\alpha}_0, x_1, x_2, x_3, s$, constant wind velocity u (a three-vector), and the maximum angular velocity of the vehicle ϕ (maximum rate of change in roll, pitch, and yaw), what is the optimal trajectory? Note that a significant portion of this problem is to formulate it rigorously in such a way that it is solvable.

This portion of the project may be done in any suitable environment, presumably Matlab and/or Mathematica.

3.2.2 Implementation on the Zagi Hardware

The existing Zagi autopilot (`http://sourceforge.net/projects/micronav`) accepts a single waypoint and a nominal speed. We want to use this autopilot to follow higher-level three-point trajectories as described in the previous section (where the three points will be changed sometime between x_2 and x_3, or possibly before x_2 if the high-level planner changes its mind; this is not a major concern, since the current position, orientation, and velocities just become the new x_0, α_0, $\dot{x}_0$, and $\dot{\alpha}_0$). This higher-level controller will be implemented either on the same microprocessor as the low-level controller or on a separate microprocessor in C/C++.

Implementation of this scheme may involve modification of the existing autopilot to accept inputs roll, pitch, and yaw rate, and nominal speed, instead of a single waypoint and a nominal speed.

3.2.3 General Wind Compensation

This portion of the project involves wind compensation at the vehicle servo-command level, and is a lower priority than the previous two portions. Note that this should not be used when the optimal local trajectories are compensating for wind.

3.3 Preparatory Tasks

Before arriving, the following tasks should be done by the student:

1. Go through the above two online references. Watch the Zagi movies and fly the vehicle using the simulator. Be sure to understand the vehicle's control inputs and how they effect its flight.
2. Ponder a formulation for the optimal local trajectories problem.
3. Practice plotting parametric functions.
4. Time permitting, become familiar with the existing autopilot functionality.

3.4 Contact

For questions, suggestions, problems, etc. related to this project, email Todd Templeton at `ttemplet@eecs.berkeley.edu`.

Bibliography

[1] J. M. Eklund, J. Sprinkle, and S. S. Sastry, "Implementing and testing a nonlinear model predictive tracking controller for aerial pursuit evasion games on a fixed wing aircraft," in *Proceedings of American Control Conference (ACC) 2005*, June 2005, pp. 1509–1514.

[2] J. Sprinkle, J. M. Eklund, H. J. Kim, and S. S. Sastry, "Encoding aerial pursuit/evasion games with fixed wing aircraft into a nonlinear model predictive tracking controller," in *Proceedings of the 43rd IEEE Conference on Decision and Control, vol. 3*, Dec. 2004, pp. 2609–2614.

Project 4

Viptos: A graphical development and simulation environment for TinyOS-based wireless sensor networks

Researcher: Heather Taylor[1]
Mentor: Elaine Cheong[2]

4.1 Overview

The purpose of this project is to understand and further develop Viptos (Visual Ptolemy and TinyOS) [1], an integrated graphical development and simulation environment for TinyOS-based wireless sensor networks.

TinyOS [2] is a component-based, event-driven runtime environment designed for wireless sensor networks. Viptos allows networked embedded systems developers to construct block and arrow diagrams to create TinyOS programs from any standard library of TinyOS components written in nesC [3], a C-based programming language. Viptos automatically transforms the diagram into a nesC program that can be compiled and downloaded from within the graphical environment onto any TinyOS-supported target platform.

Viptos is built on Ptolemy II [4], a modeling and simulation environment for embedded systems, and TOSSIM [5], an interrupt-level discrete event simulator for homogeneous TinyOS networks. In particular, Viptos includes the full capabilities of VisualSense [6], a Ptolemy II environment that can model communication channels, networks, and non-TinyOS nodes. Viptos extends the capabilities of TOSSIM to allow simulation of heterogeneous networks.

Viptos provides a bridge between VisualSense and TOSSIM by providing interrupt-level simulation of actual TinyOS programs, with packet-level simulation of the network, while allowing the developer to use other models of computation available in Ptolemy II for modeling the physical environment and other parts of the system. This framework allows application developers to easily transition between high-level simulation of algorithms to low-level implementation and simulation.

A few good online references are listed below:

1. The TinyOS website: `http://tinyos.net/`.
2. The Ptolemy Project: `http://ptolemy.eecs.berkeley.edu/`.

[1] Heather Taylor is a rising Senior at the University of Vermont, majoring in Electrical Engineering.

[2] Elaine Cheong is a Ph.D. student in the EECS department at the University of California, Berkeley. Elaine's research interests include software for embedded systems, especially operating system and programming language support. Her current research focuses on programming, modeling, and simulation tools for wireless sensor networks.

3. The Viptos website: http://ptolemy.eecs.berkeley.edu/viptos/.

4.2 Technical Tasks

The first step is to understand the software architecture of Viptos and learn how to run simple Viptos demos.

The second step is to implement and add a small feature to Viptos, such as support for conditional compilation, simulation, and/or code generation.

The last step is to develop and contribute a more significant feature to Viptos, such as enabling programming support for nesC components that are not already in the TinyOS library. A different approach would be to create a more complex demonstration application and modify Viptos to support this application.

4.2.1 Understanding Viptos

Viptos is a conglomeration of several large programming tools/systems. Understanding its software architecture requires knowledge of several different programming models and scheduling strategies. The student will need to learn the basics of the TinyOS/nesC programming model (event-based programming), the TOSSIM scheduler (discrete-event scheduling), the Ptolemy II actor/director programming model (models of computation and code generation), and JNI (Java Native Interface for combining Java and C code) programming techniques that are used to interface Ptolemy II and TOSSIM.

4.2.2 Contributing to Viptos

The student will work with the mentor to determine the specific feature(s) to add to Viptos. Depending on the student's interests, the student can choose to focus on either creating applications for Viptos or implementing additional features for Viptos. However, both approaches will involve contributing code and documentation to the Viptos code base.

4.3 Preparatory Tasks

Before arriving, the following tasks should be done by the student:

1. Download and install tinyos-1.x from the TinyOS website (url above).

2. Try to go through as much of the TinyOS tutorial as possible. The tutorial can be found online at http://www.tinyos.net/tinyos-1.x/doc/tutorial/. Note that some parts of the tutorial are not possible without sensor networking hardware, but the student should still read these steps to get an idea of the capabilities of TinyOS. The student should focus on Lesson 5 (Using the TOSSIM Simulator to Develop TinyOS Components) and try to run both TinyOS and TinyViz.

   Note that http://tinyos.net/scoop/special/support states that the tutorial is outdated. The student can try to find alternate tutorials online.

   There are errors in the tinyos-1.x tree that prevent TinyViz from being run immediately without modifying a necessary Makefile. The student should consult the tinyos-help mailing list (http://tinyos.net/scoop/special/support#mailing-lists) or the mentor for support.

3. Time permitting, try to walk through a basic JNI tutorial and run some code examples, such as those at http://java.sun.com/j2se/1.5.0/docs/guide/jni/.

4.4 Contact

For questions, suggestions, problems, etc. related to this project, email Elaine Cheong at celaine@eecs.berkeley.edu.

Bibliography

[1] E. Cheong, E. A. Lee, and Y. Zhao, "Viptos: A graphical development and simulation environment for tinyos-based wireless sensor networks," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2006-15, February 15 2006. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-15.html

[2] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM Press, 2000, pp. 93–104.

[3] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to networked embedded systems," in *Proceedings of Programming Language Design and Implementation (PLDI) 2003*, June 2003.

[4] C. Brooks, E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, and H. Z. (eds.), "Heterogeneous concurrent modeling and design in Java: Volume 1: Introduction to Ptolemy II," University of California, Berkeley, Tech. Rep. Technical Memorandum UCB/ERL M05/21, July 15 2005. [Online]. Available: http://ptolemy.eecs.berkeley.edu/publications/papers/05/ptIIdesign1-intro/

[5] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys 2003)*. ACM Press, 2003, pp. 126–137.

[6] P. Baldwin, S. Kohli, E. A. Lee, X. Liu, and Y. Zhao, "Modeling of sensor nets in Ptolemy II," in *IPSN'04: Proceedings of the Third International Aymposium on Information Processing in Sensor Networks*. New York, NY, USA: ACM Press, 2004, pp. 359–368.