MOBIES
Project Progress Report

Engine Throttle Controller Design
Using Multiple Models of Computation

Edward Lee
Haiyang Zheng

with thanks to Ptolemy Group of UC Berkeley,
Paul Griffiths, Christoph Kirsch, Tunc Simsek, Jason Souder

UC Berkeley, March 12 2002
Engine Throttle Control (ETC)

Driver presses pedal

Throttle

Pulse-width modulated current

Physical World deals with continuous signals

Software control deals with discrete signals
Design Challenge

- Create a realistic representation of the ETC system that models ...
  - continuous physical signals
  - control modes and their transitions
  - discrete computations
  - task scheduling
  - ...

Design Overview

- **Control modes** (finite-state machine)
- **Computation** (discrete time)
- **Physical signals** (continuous time)

More models of computation …
Engine Throttle Control Model

Heterogeneous model of the UC Berkeley Vehicle Dynamics Lab
Electronic Throttle Controller.

by Paul Griffiths, Christoph Kirsch, Tunc Simsek, Jason Souder
Last updated January 15, 2002

Edward Lee, Haiyang Zheng
Last updated March 6, 2002

a new model of computation
Task 1.1: Demonstrate ability of modeling cross cutting physical constraints
Task 1.7: Demonstrate ability to compose multiple view models
Engine Throttle Control Demo

- We finished the demo of ETC model, our work is focused on controller design
- We introduced and used Giotto model to implement the controller part since we want to meet the time deadlines
- We will use Ptolemy II to study Giotto model and its interactions with other models of computation
Giotto

- What is Giotto?
  - Developed by Tom Henzinger and his group
  - A periodic time-triggered semantics
  - Deterministic and predictable behaviors
  - Details will be given by Christoph

- Why Giotto?
  - Make sure the tasks meet deadlines
  - Mobies Phase I tries to use Giotto model to implement the controller part of ETC model
Ptolemy II

- We use Ptolemy II to study the Giotto model of computation
- Ptolemy II studies heterogeneous modeling, simulation and design of concurrent systems
- Emphasis on building a framework supporting experimentation with models of computation and their interactions
Giotto Model
Implemented in Ptolemy II

Task 1.2: Demonstrate ability to customize generic modeling tools
Task 1.3: Demonstrate ability to model domain specific model semantics
Models Interactions I

- Giotto model embedded in Discrete Events (DE) model
Task 1.5: Demonstrate ability to integrate different models of concurrency
Task 1.6: Demonstrate ability to integrate domain specific modeling tools
Software Control Design Flow

- CodeGenerator
  - Double click to generate code.
- Executable Controller Code
- Giotto code
- Giotto
  - E-Compiler
  - E-Machine
  - Runtime Library

Processes:
- Simulate
- Refine
Summary

- Complex ETC model is designed and simulated in Ptolemy
- Multiple models of computation involved as necessary
- Hierarchically heterogeneous structure
- Ptolemy II as a framework supports experimentation with models of computation and their interactions
Software Control Design Flow

Executable Controller Code

Giotto
E-Compiler
E-Machine
Runtime Library

Simulate

Refine

CodeGen

Double click to generate code.

Giotto code
Tool Integration with Giotto

- Visual block diagram design
- Simulation for design refinement
- Giotto code generated from Giotto model of Ptolemy II
- Giotto code schedulability analysis by E-Compiler
- Task code manually generated for E-Machine

Task 2.2: Demonstrate ability to customize frameworks with generators
Task 2.4: Demonstrate ability to generate embedded software from models
Task 4.2: MIDTERM DEMONSTRATION: Generate embedded software for avionics/vetronics systems using model-based environment
Tool Integration with Charon

- Visual block diagram design of Hybrid Systems
- Simulation to refine design
- Generated Charon code from Hybrid Systems models in Ptolemy II for verification tools of Univ. of Penn

Task 2.7. Demonstrate ability to guarantee properties of generated systems