What is Giotto?

- A time-triggered, platform independent programming language + compiler + runtime library.
- For hard real-time, safety-critical control applications on distributed platforms.
- Abstract programming model for control system development.
- VxWorks runtime library.
- Ptolemy II domain.
Troubles for Control Systems Implementation

- Development is expensive.
  - System integration.
  - Temporal composability.
  - Distribution.
  - Fault tolerance.

- This is in part a programming languages problem.
  - Existing languages are either too low-level or lack useful features.
  - Little notion, in engineering practice, of platform-independence.

Control system design with Giotto

Control Engineer
Matlab → decomposes designs simulates → Application

Control Design

Software Engineer
Giotto → implements tests → Giotto Program

Platform

Functionality & Timing
Motivating Example: Fly-by-wire Flight Control

Requirements for Control Programming Language

- Talk about **time**:
  - Periodic computation and IO.
- **Operational modes**:
  - Take off, cruise, land.
  - Different control laws needed in each mode.
- **Easy to distribute**:
  - For performance...
  - ...or fault tolerance.
- **Encapsulate existing (legacy) code**.
  - Or output of tools such as Matlab.

- Deterministic, platform independent, simple.
A Single Giotto Task

Input ports → Private ports $f$ → Output ports

Private ports

Input ports → Private ports $f$ → Output ports

Frequency 10ms

Time

Read @ time $t$  Write @ time $t+10ms$

Two Communicating Giotto Tasks

Input $f$

Frequency 10ms

Input $g$

Frequency 5ms
Timing of Two Communicating Tasks

Mode Switches in Giotto
Timing of Mode Switch

Generating Schedules Automatically

- For **static priority preemptive** systems:
  - Use rate-monotonic scheduling plus “exact characterization” theorems (Lehoczky, Sha, Ding 1989)

- For **dynamic priority preemptive** systems:
  - Use earliest deadline first for scheduling tasks (Liu, Layland 1973), and earliest deadline late for scheduling communications (Chetto, Chetto 1989).

- For **optimal** scheduling of **both** tasks and communications:
  - Use recent multiprocessor scheduling algorithms (Peng, Shin, Abdelzaher, 1997; Hou, Shin 1997)
Giotto Annotations

Giotto Program
Giotto-P Program
Giotto-PS Program
Giotto-PSC Program

Hosts, Nets, Performance
Task Schedules
Communication Schedules

Giotto Compiler
Distributed Platform

Example of Giotto Annotations

```plaintext
task Command1() outputs (com_Command1) {...} [host1]
predicate ToStop1(bool sensor1, bool sensor2) {...} [host1]
predicate ToStop2(bool sensor1, bool sensor2) {...} [host2]

Mode Lead1Follow() period 400ms entryfreq1 {
  taskfreq 1 do Command1();
  taskfreq 4 do MotorCtr1(com_Command1);
  taskfreq 4 do MotorCtr2(com_Command1);
  exitfreq 2 if ToStop1(sensor1, sensor2) then Stop1(1);
  exitfreq 2 if ToStop2(sensor1, sensor2) then Stop1(2);
  [push sensor1 to host2 in (2.5, 5);
   push sensor2 to host1 in (7.5, 10);
   push mode from host1 to host2 in (17.5, 20);
   push mode from host2 to host1 in (22.5, 25)];
}
```
Comparison with Synchronous Languages

- In spirit, Giotto is similar – allowing a precise semantic description of programs...
- ...However:
  - Giotto is more of a “glue” language.
  - Giotto has more restricted scope (periodic tasks, flat mode structure).
  - But Giotto can better leverage real-time scheduling theory.

Disadvantages of synchronous languages:
- Esterel tasking mechanisms make it difficult to prove that Esterel programs meets their deadlines.
- Lustre programs often underutilize CPU (lots of activity around tick, idle otherwise).

Comparison with Architecture Description Languages (ADLs)

- ADLs shift programmer’s perspective from small- to large-grained features.
- ADLs allow the automatic generation of code for task communication and scheduling.
- Giotto is particularly similar with MetaH (Vestal 1997):
  - Periodic tasks, multi-modal control, distributed and real-time implementations.
- Mode switches in Giotto are handled more cleanly than in MetaH.
- Giotto has an abstract semantics:
  - Does not constrain choice of scheduler.
  - Allows for optimal real-time scheduling.
For More Information...

- Visit [http://www-cad.eecs.berkeley.edu/fresco/~giotto/](http://www-cad.eecs.berkeley.edu/fresco/~giotto/)

- ... Or contact:
  - Ben Horowitz ([bhorowit@cs.berkeley.edu](mailto:bhorowit@cs.berkeley.edu))
  - Tom Henzinger ([tah@eecs.berkeley.edu](mailto:tah@eecs.berkeley.edu))
  - Christoph Meyer Kirsch ([cm@eecs.berkeley.edu](mailto:cm@eecs.berkeley.edu))

- ... Or read:
  - Giotto: A Time-Triggered Language for Embedded Programming
  - Embedded Control Systems Development with Giotto.
    - (All papers are available at the above URL.)