

## High-Performance Control with Slow Computing

Current techniques for the design of software-enabled control systems rely on the existence of high performance sensing, actuation and computational devices that can be embedded within a physical system at modest cost. Driven by Moore's law, the success of this paradigm can be seen through the broad usage of feedback controllers in modern application areas. The goal of this challenge lies at the other end of the computational spectrum: Can we develop new principles and tools for the design of closed loop control systems using highly distributed, but slow, computational elements?

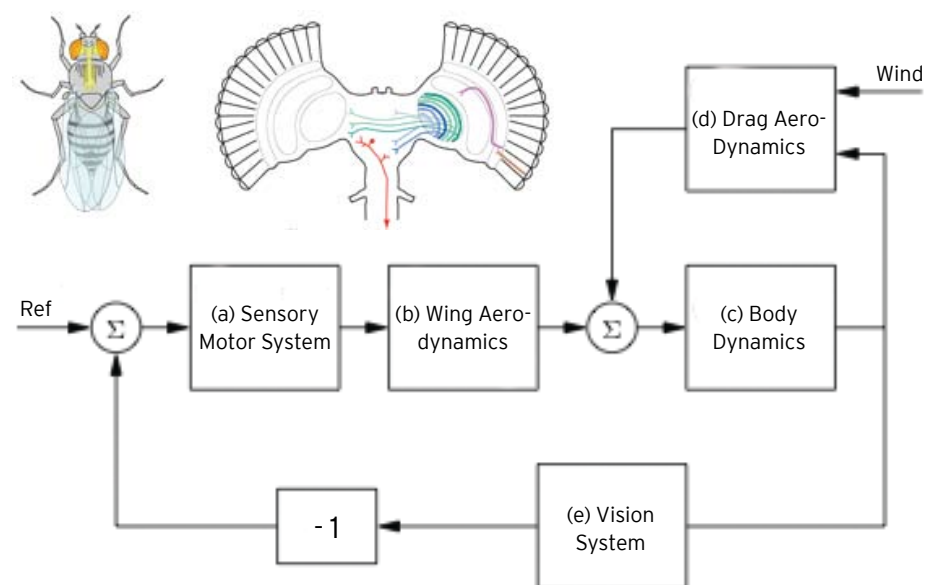
The motivation for control design using slow computing is to develop control system architectures for applications where computational power is extremely limited. One important class of such systems is that for which the energy usage of the system must remain small, either due to the source of power available (e.g., batteries or solar cells) or the physical size of the device (e.g., microscale and nanoscale robots). A longer term application area is the design of control systems using novel computing substrates, such as biological circuits. A critical feature is the fact that the speed of the basic computing elements is similar to the underlying dynamics, leading to tight coupling between dynamics and computing.

Design of feedback systems using slow computing is particularly challenging because of the performance limitations associated with computational delays that are comparable to the underlying plant dynamics. Highly parallel, non-deterministic architectures are likely to be needed to achieve what is normally accomplished through the tightly synchronized, serial interconnections of sensing, filtering, estimation, planning, regulation, and actuation that are common in traditional control systems. Unfortunately, current techniques for systematic design of control systems assume a mostly serial processing architecture and techniques that make use of parallel architectures (such as neural networks) do not provide sufficiently systematic design methods. New research is needed to develop the architectures, theory, and tools required to design controllers where computational delay does not allow current techniques to be used.

### Context

Current approaches to the design of software-controlled systems make use of a combination of abstractions and design techniques that are often implicitly based on the assumption that significant computational capacity is available to implement computations and communications. This is a good assumption for many application areas where substantial amounts of computing can be embedded within a physical system to control the dynamical behavior of the underlying process. As a consequence, many of the approaches that are available for designing complex, "cyberphysical" systems rely on large amounts of computing to achieve complex and robust behavior.

As a complementary approach, consider instead the control system for a fruit fly, depicted in the figure below. This system uses approximately 300,000 neurons with typical time constants in the range of 1 - 100 msec (10 - 1000 Hz) and is approximately the size of a sesame seed. Yet it is able to take off, land, avoid obstacles, find food and mate (among other things), often with performance that is beyond what we can do in engineered systems at this size scale. As just two specific instances, the control system of a fly is capable of executing saccades (rapid changes in direction) that occur at angular rates of up to 1800 deg/sec and it can fly in wind gusts that are up to twice its flight speed in air.



Contributor: Richard Murray, California Institute of Technology, USA

Control System for a Fruitfly

## Challenge

The goal of this challenge is to develop the fundamental insights and tools that will allow us to design control systems that can perform the tasks of modern high-speed control systems but using an architecture that is compatible with the speed of computation used by an insect. The development of such an architecture has the possibility of providing new ways of integrating control into systems where large amounts of fast computation are not easily available, because of limitations on power, physical size, or choice of computing substrate. It is likely that many of the tools and insights required to design such systems will prove to be central to the design of other classes of systems in which the effects of time delay, asynchronous execution of parallel computations, and highly complex interconnections play a defining role.

A candidate architecture for the implementation of such a system is shown below. The system consists of a set of agents that are interconnected through a network (the portion of the figure to the left of the network illustrates the architecture within an agent). Each individual agent has a highly structured inner loop, which interacts with a guarded command language (GCL) protocol engine that triggers rules to change the discrete state of the system. The "inner loop" control system makes use of an interconnection matrix  $L$ , a set of asynchronous delays (represented by the blocks labeled  $\tau$ ) and nonlinear elements  $N()$ . Internal feedback between the nonlinear block and the interconnection block allows a general set of dynamical systems to be formed from this simple structure. The protocol-based feedback system modifies the inner loop dynamics, but also controls communications between other agents, using a packet-based communications network. Multiple agents interact with each other across the communications network, using packet-based communication protocols. This network introduces another layer of variable time-delays and asynchronous behavior, enabling complex behaviors by the multi-agent control system.

The complexity of the architecture and of the resulting behavioral space will require the development of formal tools for specification, design and verification.

