

Demonstration using *neclab*: The Network Embedded Control Lab

Nicholas Kottenstette, Panos J. Antsaklis and Simon Han
{nkottens, antsaklis.1}@nd.edu, simonhan@ee.ucla.edu

April 1, 2006

Abstract — Designing and implementing a wireless networked embedded control system (*wnecs*) is a challenging task. *neclab*, the network embedded control lab [1], is a software environment designed to assist the engineer with simulation, deployment, and monitoring of networked embedded control systems, in particular wireless networked embedded control systems (*wnecs*). A *wnecs* is a collection of interconnected sensors, digital controllers, and actuators connected to control plants which communicate with each other over wireless-channels via a radio interface. In particular, our demonstration will focus on *wnecs* which use the MICA2 Motes developed by Crossbow technology. An integral component of *neclab* is the *SOS* operating system [2] which is embedded on the MICA2 Motes. The demonstration will consist of implementing the simple wireless ball and beam control application described in [1].

In demonstrating the wireless ball and beam control application, we will discuss various technical design issues and how we addressed them. One challenging design was to provide an effective programming bridge between lower level embedded software design required for the MICA2 Motes and higher level scripting languages such as python. Using SWIG and building off the powerful *modules* concept introduced by *SOS*, it is now possible for us to design control programs running in python on a host computer. These control programs can directly use the *SOS* message passing interface in order to communicate with the MICA2 motes distributed in a wireless network. Using python we also utilize powerful scientific libraries such as NumPy and SciPy[3]. We will also demonstrate the ability to visualize data with the plotting tools provided by matplotlib [4]. These tools and others can be successfully integrated in to *neclab*, so that we can quickly start

developing *wnecs*.

SOS has matured and brought some powerful programming features which *neclab* is utilizing. *SOS* now allows memory to be allocated which is not limited to fixed block length segments. The multi-hop module loader now allows modules to be loaded to the internal flash of the MICA2 motes in a dormant state. This allows us to create a network of motes each with a common file system, similar to a network of computers. Using concepts such as the *networking message pipe* [1], the *configuring* module in *neclab* allows us to configure arbitrary networking routes. These routes allow messages to be exchanged to different modules in a manner which parallels passing information between processes on a PC. These processes; however, are running on individual motes, and our PC while communicating through a distributed wireless network. They can do this and still balance a ball on a beam.

References

- [1] N. Kottenstette and P. J. Antsaklis, “neclab: The network embedded control lab,” in *Workshop on Networked Embedded Sensing and Control*, Lecture Notes in Control and Information Sciences, Springer, October 2005. <http://www.nd.edu/~pantsakl/349-NESC05.pdf>.
- [2] C.-C. Han, R. Kumar, R. Shea, E. Kohler, and M. Srivastava, “A dynamic operating system for sensor nodes,” in *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, (New York, NY, USA), pp. 163–176, ACM Press, 2005.
- [3] E. Jones, T. Oliphant, P. Peterson, *et al.*, “SciPy: Open source scientific tools for Python,” 2001–. <http://www.scipy.org/>.
- [4] J. Hunter, “matplotlib: Matplotlib / pylab – matlab style python plotting,” 2003–. <http://matplotlib.sourceforge.net/>.