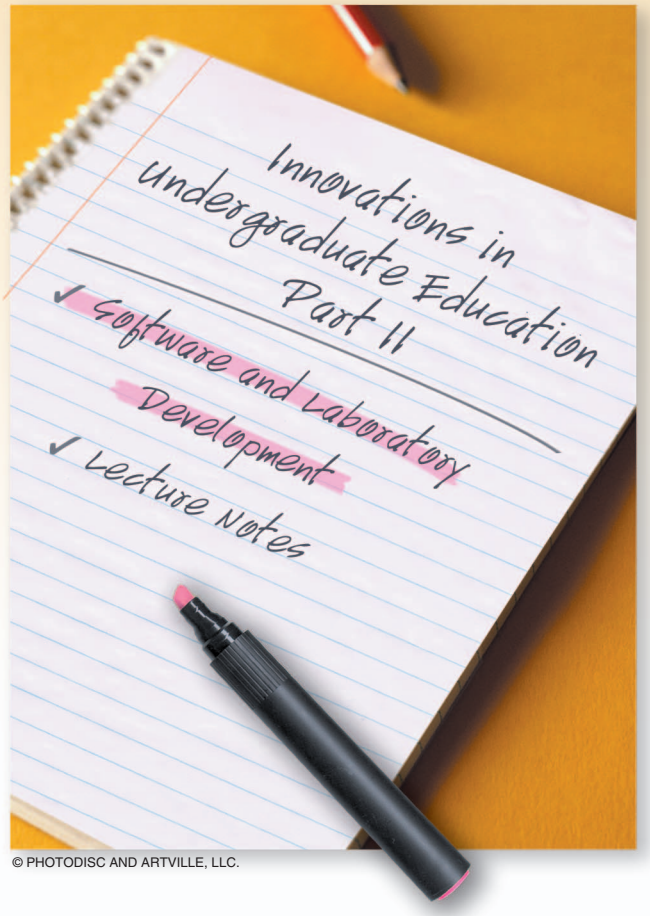
FEATURE

Experiments for Dynamic Resource Allocation, Scheduling, and Control

New challenges from information technology-enabled feedback control



By Nicanor Quijano, Alvaro E. Gil, and Kevin M. Passino

Advances in information technology such as object-oriented programming, real-time operating systems, and reliable computer communication networks enable the deployment of effective control and automation systems [1]. Industry exploits information technology in distributed control system (DCS) products, whose components include proportional-integral-derivative (PID) and programmable logic controllers. At the same time, DCSs can use distributed networked decision-making systems for factory-wide control solutions. There is an ongoing technological revolution in the application of networked embedded computing to control. Advancements in this area are driven by low-cost and high-performance microcontrollers that have onboard data acquisition as well as the controller area network (CAN) or wireless Ethernet technologies that interconnect these microcontrollers. There are software tools for developing and implementing such embedded distributed networked control systems. The codesign of software and feedback controls for both DCS and networked embedded controls is a topic of current investigation for control and software engineers [1]. For additional information on DCS products and networked embedded controls see "Additional Resources."

While industry continues to exploit information technology for distributed feedback control, universities face the challenge of revising their curricula to suit the educational needs of future engineers in this area. We can identify two research thrusts that support relevant educational experiences. First, cooperative robotics involves groups of ground, underwater, flying, or space robots that perform tasks such as cooperative search, cooperative pushing of an object, or coordinated group

0272-1708/05/$20.00©2005IEEE

Additional Resources

Distributed Control System (DCS) Products

There are a wide range of companies that sell DCS products. Major vendors include: Honeywell, Yokogawa, Emerson, Invensys Foxboro, Rockwell Automation, and ABB. See the Web sites of these companies or any issue of the Control Magazine: http://www.controlmagazine.com/.

Networked Embedded Systems

There are many companies that sell products for embedded networked control systems. See, for instance, the Web sites of Texas Instruments, Motorola, or Microchip Technology. For a product directed at industrial solutions, see the Honeywell "Smart Distributed Systems" product at: http://content.honeywell.com/sensing/prodinfo/sds/.

OSU Distributed Dynamical Systems Laboratory

For more information on the experiments described in this article, including videos of their operation, see http://www.ece.osu.edu/ ~passino/distdynamicsyslab.html.

motion. In these applications, there is typically a microcontroller on each vehicle supported by a wireless ethernet for vehicle-to-vehicle communication. Courses that provide educational experiences in this area are described in [2] and [3]. A second thrust is the research program on wireless sensor networks based on the University of California at Berkeley motes, which are electronic modules that have processor, sensor, and communication capabilities [4]. Networks of motes can be used for distributed networked control system problems, such as the tracking of an evader moving through a field of sensors.

This article introduces a third approach to distributed networked feedback control based on an educational laboratory. This laboratory emphasizes low-cost experiments that retain key elements of realism, problems that are different from those studied in the cooperative robotics and sensor network areas, and new challenges from information technology-enabled feedback control [1], [5], such as dynamic resource allocation, feedback scheduling of tasks, decision making over networks, and nontraditional control objectives. We have developed the following experiments to study these challenges.

- *Balls-in-tubes experiment:* Four balls are levitated to a common maximal height by dynamically allocating air flow to four tubes that hold the balls. In one scenario, only one air pulse can be applied at a time, and the control system juggles the balls. The juggling is challenging, owing to significant ball-height-sensor noise, air turbulence, ball-to-ball coupling through a common air source, actuator bandwidth limits, and the need for distributed decision making over a network.
- *Electromechanical arcade:* Two agents must cooperatively fire laser guns at photocell targets to maximize the team's point gain within a finite time period. Feedback scheduling is needed due to the unpredictable appearance and disappearance of targets. Distributed scheduling over a network makes coordination challenging.
- *Planar temperature grid:* This experiment has a planar grid of 16 zones, each of which has a temperature sensor, heater, and controller interface. The goal is to allocate a limited amount of electrical current to the heaters to maximally raise the temperatures of all of the zones to a common value. The dynamic allocation is complicated by interzone coupling, ambient temperature influences, and wind currents. Moreover, the presence of a communication network between zones, with possible topological constraints on information flow, necessitates the use of distributed decision making.
- *Building temperature control:* This experiment consists of a scale-model two-floor building with rooms and halls. There are heaters and sensors in each room, adjustable doors and windows, and electrically controllable wind flows. The building has many of the same characteristics and challenges as the planar temperature grid problem, but different interzone and ambient temperature effects.

In this article, we identify the role of these experiments in a university curriculum. The choice of laboratory software and hardware is discussed, followed by an introduction to the experiments. For each experiment, we describe the apparatus, the challenges that must be confronted to meet performance objectives, sample feedback control solutions, and alternative control methods that might be useful. In the concluding remarks, we explain how the modular nature of each apparatus provides opportunities to expand the experiments to confront additional challenges.

Laboratory Role in a University Curriculum

The Department of Electrical and Computer Engineering at Ohio State University (OSU) has a modern undergraduate control laboratory that uses dSPACE hardware and software [6] and Quanser experiments [7]. Within the OSU quarter system, we have two ten-week, dual-level graduate/undergraduate laboratories designated EE 757 and EE 758. Neither EE 757 nor EE 758 is a prerequisite for the other. The majority of students taking EE 757 and EE 758 are graduate students who have had at least one year of graduate course work in the control area. Students come primarily from the electrical, mechanical, industrial, and aerospace engineering disciplines.

EE 757 focuses on cooperative robotics problems such as platooning and cooperative search [2]. Small ground-based robots, connected by means of a wireless ethernet, are used as the experimental testbed. EE 758 is a service course that provides students with laboratory experiences in introductory linear, nonlinear, and robust control, along with more specialized topics [8]. The first half of EE 758 is dedicated to learning dSPACE, data acquisition, modeling and system identification, PID control with antiwindup schemes, the linear-quadratic regulator with an observer, and nonlinear control using feedback linearization. The experiments used in the first half of the course are Quanser products [7], [8].

The second half of EE 758 is dedicated to in-depth study on a topic of interest to the student. A faculty member can develop suitable experimental challenges for students who focus on system identification, sliding mode control, adaptive control, intelligent control, game-theoretic approaches, advanced robust control, or advanced nonlinear control. Additional Quanser experiments are available for the second half of EE 758, including tanks, a two-degree-of-freedom helicopter, and an inverted cube [7]. Moreover, we can provide students with projects that use the National Institute of Standards and Technology (NIST) Real-Time Control Systems (RCS) software library [9] or National Instruments hardware and LabVIEW software [10].

The default project sequence in the second half of EE 758 is distributed networked feedback control. We typically give the students one of the experiments described in this article and require them to design, implement, and test resource allocators, schedulers, and controllers. We often require the students to design strategies that can overcome instructor-induced disturbances, such as a puff of air on the multizone temperature control experiment. While such disturbances are difficult to quantify and repeat, they provide clear challenges and tangible phenomena for evaluating the student's understanding of distributed networked feedback control. Students are expected to provide a demonstration and written report for all experiments.

Laboratory Software and Hardware

Successful development of an educational laboratory for distributed networked feedback control depends on the choice of software, processor, and data acquisition hardware. One approach is to purchase DCS software and hardware products that are used in industry, which has the advantage of familiarizing students with state-of-the-art tools. Unfortunately, many industrial DCS products are too expensive for universities to purchase and maintain. Moreover, some DCSs lack flexibility and have a market niche that can lead to an over-specialization of student skills, resulting in a distraction from foundational principles.

An alternative is to use a networked embedded control approach, with multiple microcontrollers interconnected by means of a CAN bus. For this approach, competing software products must be evaluated. The networked embedded control approach has the advantage of being relatively easy and inexpensive to expand to a large number of decision-making nodes in some applications [11]. However, for some experiments, this approach can have the disadvantage of not facilitating comparisons to centralized approaches, which is often useful in research and education. Similar comments apply to the motes in [4].

Another alternative is to use software and hardware solutions that were developed for implementing a single controller but can support distributed control. For example, dSPACE [6] provides controller boards and software based on Mathworks' MATLAB and Simulink [12]. In the dSPACE products, there are a number of ways to communicate between controllers hosted by different computers, including a serial link or the Internet. The dSPACE approach has the advantage of building on students' MATLAB skills gained from courses on signal processing and control design. Moreover, each laboratory station at OSU has a computer with resident dSPACE hardware and software. When laboratory courses are not being offered, we can use the hardware and software for experimental research on distributed networked feedback control. An alternative to dSPACE is National Instrument's hardware and LabVIEW software [10].

An attractive but less commercially popular alternative is to use the NIST RCS software library [9], [13], [14]. The NIST RCS software was developed to provide the following features: platform-independent software for use on different computers with different operating systems; software support for a range of communication technologies and message formats; design support for hierarchical and distributed control modules; and a graphical user interface for online monitoring and debugging. The NIST RCS software is under continual development and is used in a range of applications [9], [13]. The primary disadvantage of the NIST RCS software is the lack of an interface to MATLAB and, in some cases, the need to develop an interface to data acquisition hardware. A significant advantage is that a mature design methodology for distributed networked feedback control is available [13].

OSU has been funded for a number of years by NIST to transfer RCS software and methodology to a university educational setting [13], [14]. Recently, NIST funded an OSU evaluation on the potential of transferring the RCS design methodology to dSPACE and National Instruments products. Based on these studies, our solution to the software/hardware choice problem is to use dSPACE products coupled with the NIST RCS design methodology. This approach allows exploitation of MATLAB in the lab, supports pedagogy for design methodology, allows us to integrate the hardware/software resources from each lab station, and provides support for comparisons between

distributed and centralized feedback control solutions. Since the full details of RCS are covered elsewhere [9], [13], [14], we only briefly describe its adaptation for use in our laboratory with dSPACE software. Adaptation of RCS methodology for LabVIEW is conceptually similar.

RCS software has design and diagnostic tools, which can be used to develop distributed control algorithms and monitor and change variables in real time. The functionality of these tools can be implemented in dSPACE products through Simulink and the dSPACE graphical user interface. To develop the control system block diagrams in Simulink for the RCS design methodology, we use preprocess, decision process, and postprocess submodules in a control module. Then, several control modules can be connected in a hierarchical and distributed fashion. In the preprocess submodule, we acquire data, perform conversions, and store information in global variables for use in the decision process submodule. Next, we develop several subsystems that are responsible for making control decisions or other tasks such as updating variables used in the postprocess submodule, which transmits data to other computers and updates the digital outputs.

The Balls-in-Tubes Experiment

The balls-in-tubes experiment is designed to be an inexpensive testbed for dynamic resource allocation strategies that exploit information from the plant in feedback.

Experimental Apparatus and Challenges

Figure 1 shows the balls-in-tubes experiment. There are four modules, each of which has a tube with a ball inside, a fan at the bottom to lift the ball, and a sensor at the top to sense the ball's height. For each tube, there is a box that the fan pressurizes. The box can be viewed as a stiff balloon that is blown up by the fan and whose outlet hole blows air into the tube. The tubes are connected at the fan inlets by means of an input manifold that has an inlet at the bottom, as indicated in Figure 1. Each tube has an output manifold at the top with an outlet as shown.

The manifolds are a key component of the experiment, as they necessitate the sharing of air at the input or restrict air flow at the output; both actions cause significant coupling among the four tubes. Characteristics of the coupling can be adjusted by designing different opening sizes for the inlet and outlet. Alternatively, solid objects can be placed inside the input or output manifolds to obstruct airflow. The air flow characteristics are complicated due to turbulence in the manifolds, pressurized box, and tubes. For a range of input manifold inlet sizes, a fan succeeds at lifting the ball only at the expense of other balls dropping. This feature leads to the need for resource allocation, where the resource is the air that elevates the balls. The input manifold effectively implements a multivariable saturation constraint. Under some conditions, a fixed amount of air coming from the input manifold can be allocated. When significant amounts of air are allocated to some tubes, the fans for the other tubes become limited in what they can allocate.

To measure the ball heights, we use Devantech SRF04 ultrasonic sensors. The raw data obtained from the sensors are quite noisy, with

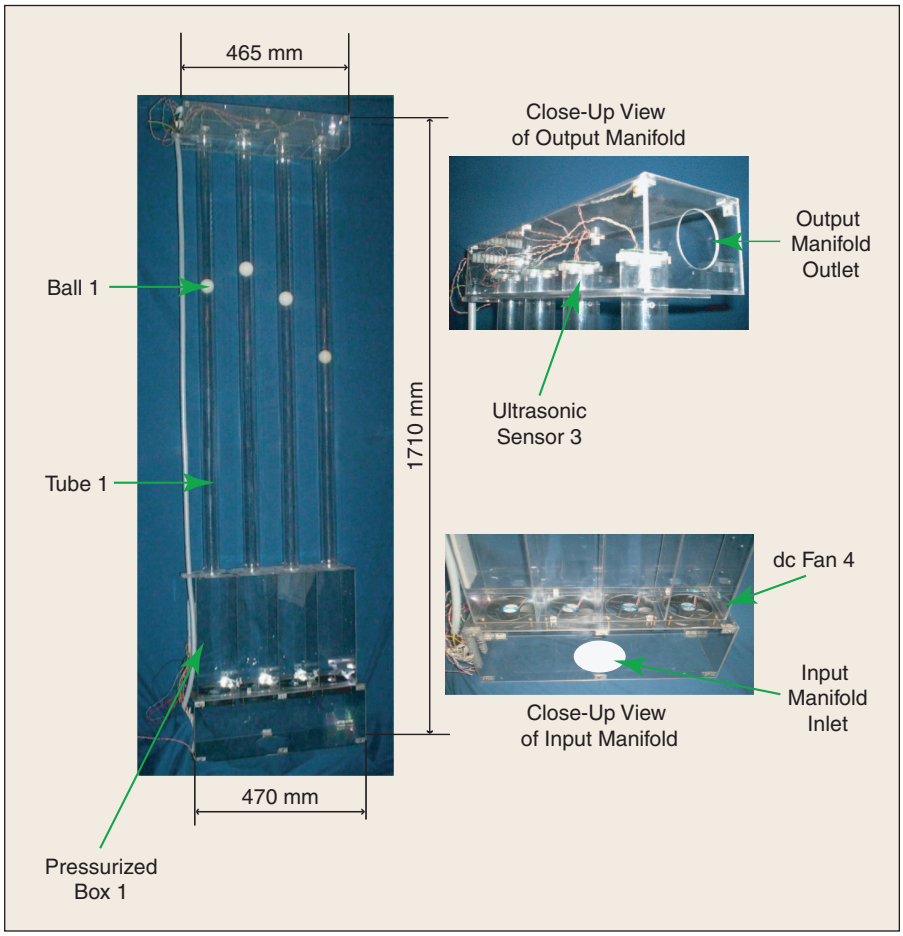


Figure 1. *Balls-in-tubes experiment. The tubes are numbered from left to right. Each tube has a dc fan, a ball, and an ultrasonic sensor. There is an input manifold inlet at the bottom and an output manifold outlet at the top. These manifolds necessitate the sharing of air at the input, or restrict air flow at the output.*

typical measurement spikes corresponding to errors greater than 10 cm. Using digital filters to smooth the sensor outputs, we achieved a resolution of ± 1 cm. The actuators are Dynatron DF1209BB dc fans, commonly found inside personal computers. We use a pulse-width-modulation (PWM) signal as an input to the fan. The sampling period is 100 μs, and the period for the PWM is 0.01 s. There is one digital input and one digital output for each sensor and one digital output for the PWM input to each fan, for a total of 12 digital input-output lines that connect to a DS1104 dSPACE card.

The total cost of the plant is less than US$200 for four tubes, including the sensors and actuators. The supporting electronics for the actuator and sensor interfaces are relatively inexpensive. Tube modules designed to be easily separated can be used at a separate laboratory station to study the control of ball height in a single tube. A project on ball height control in a single tube might be a useful challenge, perhaps even in an undergraduate laboratory.

In addition to balancing a single ball in a tube, there are a number of control objectives and challenges that can be studied for this experiment:

- balancing the balls inside the tubes, trying to allocate air flow to keep all the balls at fixed positions or maximally elevated at a uniform height
- balancing and reallocation in the presence of plant changes due to manifold inlet size changes or flow obstructions in a manifold
- effects of distributed networked decision making in the presence of an imperfect communication network.

Resource Allocation Strategies and Results

We describe two resource allocation strategies and analyze their performance in the presence of an inlet size change or an obstruction placed inside the input manifold.

Resource Allocation Strategy: Juggler

The goal of the juggler resource allocation strategy is to maintain the balls in the vicinity of a specified common height. Although the peak-to-peak oscillation of each ball height can be relatively large, we require the time averages of the heights of each of the four balls to be the same. We complicate this goal by allowing only one fan at a time to have a sufficient portion of the PWM duty cycle to raise the corresponding ball. This constraint leads to a juggling resource allocation strategy that is designed to minimize the differences between the average ball heights despite air turbulence, intertube coupling through the manifolds, fan bandwidth constraints, and significant sensor noise. The dynamics of allocation rely critically on feedback information on ball heights and the distributed decision-making that implements the allocation strategy.

The following characteristics of the experiment are relevant to the design of the juggler strategy.

- The time it takes for a ball to rise a fixed distance in a tube is generally greater than the time it takes the ball to drop the same distance.
- Since the inlet to the input manifold is closest to the fan inlets to tubes 2 and 3, air flow for these two tubes is less restricted than for tubes 1 and 4. Hence, for tubes 2 and 3, the amount of time it takes to raise the balls or have them drop is less than the corresponding times for tubes 1 and 4.
- Apparently due to module construction differences, the time it takes to lift the ball in tube 4 is generally greater than the corresponding time for tube 1.
- The minimal percentage of the duty cycle of the PWM signals that can lift all of the balls simultaneously is about 60%. This input is just strong enough to pressurize the boxes so that the air pressure can lift the balls.

For fan i at time t, let $D_u^i(t)$ and $D_d^i(t)$ be the duty cycles that cause the ith ball to go up or down, respectively. We chose these values using the previous insights and some trial-and-error tuning. The juggler strategy consists of the following steps:

1) *Find a ball to focus on:* Let i^* be the lowest ball at time t and $h_{i^*}(t)$ be its height.
2) *Push the low ball up:* For all $t' \in [t, t+T]$, for some fixed time period T, assign $D_u^{i^*}(t')$ to fan i^* and, for $j \neq i^*$, assign $D_d^j(t')$ but modify the input based on the following rules
   - *Truncation rule 1:* If $h_{i^*}(t') > 0.6$ m at time $t' \in [t, t+T]$, then the algorithm aborts the current allocation and returns to step 1, even if the time duration T has not elapsed. This rule avoids pushing ball i^* too high.
   - *Truncation rule 2:* If $h_i(t') > 0.6$ m for all i at time $t' \in [t, t+T]$, then the algorithm aborts the current allocation, assigns $D_d^i(t)$ to each fan for 0.6 s, and then goes to step 1. This rule helps avoid having all of the balls get stuck at the top of the tubes, which can happen in some cases if the inlet to the input manifold is not restricted in size.

In summary, the juggling strategy involves dynamically applying different size and duration air pulses to the tubes corresponding to the minimum ball height. We tuned the value of the time duration T until we found that $T = 3$ s was acceptable.

Figure 2 illustrates the closed-loop performance of the juggler strategy. The oscillations in ball heights are a direct consequence of the constraint that only one fan at a time can be given an input that can lift the corresponding ball. The average heights of the balls are reasonably close. We could tune the strategy to obtain a closer correspondence between the averages of the ball heights, however, such tuning can be tedious and futile for the following reasons. First, the module differences that arise from sensors, actuators,

air leaks, and physical construction are dynamically coupled through turbulent air flows. Tuning the juggler strategy affects this unpredictable coupling, and the small differences in average ball heights emerge. Second, we have seen temperature changes in the room affect the performance.

A typical feedback control goal, such as zero steady state tracking errors between the ball heights and a commanded height, is simply not achievable for the given constraints. Moreover, the achievable common average ball height is not known a priori; hence, the problem is not one of trying to regulate the average ball heights to a commanded value. The closed-loop performance objective for the experiment is nontraditional but nonetheless quite natural, as most people have seen a human juggler perform and understand that juggling is a feedback control problem.

Next, we conducted an experiment using the juggler strategy, but with an unmodeled plant change. To change the plant, we placed an object inside the input manifold to partially obstruct the air flow. The obstruction especially affected tube 4 since the object was placed beneath its fan. The plant change necessitated reallocation of the air flow relative to the nominal case discussed previously. To illustrate the required reallocation, we ran the experiment with and without the obstruction and computed the average ball heights. We also computed the percentage of the 200 s experiment run time that each ball was pushed up. In other words, we computed the fraction of time that $D_u^{i^*}(t)$ was applied to each ball; see Table 1 for the results. Due to the characteristics of the experiment, the allocator attends to tube 4 the most. The average values for the ball heights are comparable to the ones for the nominal case but generally lower due to the flow obstruction. The key feature is how the amount of time allocated to tube 4 increases to compensate for the air flow obstruction. Basically, the strategy takes some of the air that was allocated to tubes 2 and 3 in the nominal case and reallocates it to maintain the relatively equal average heights. This phenomenon is the essence of dynamic feedback resource allocation.

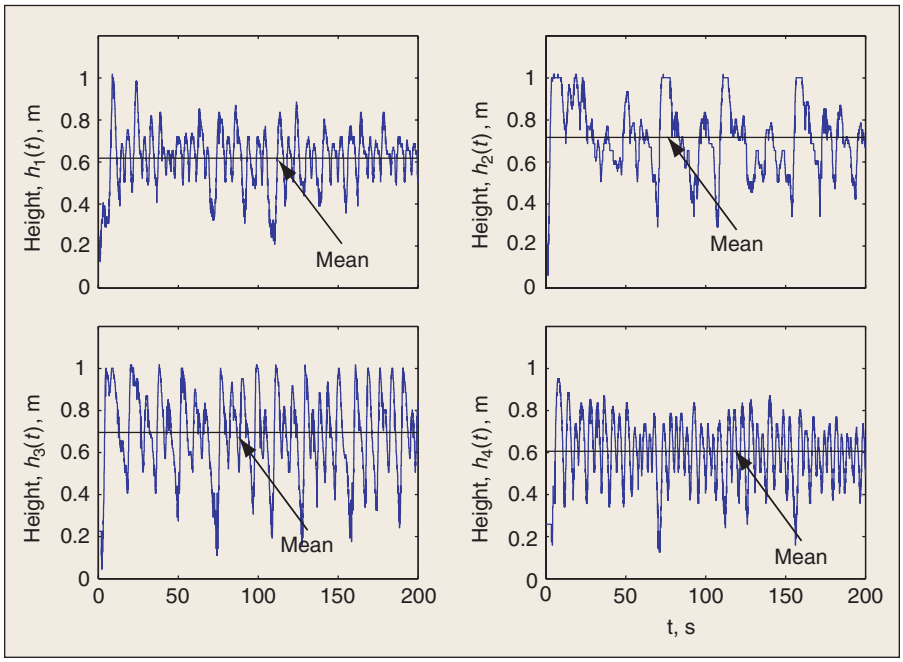


Figure 2. *Heights of the balls in each tube for the juggler strategy. The oscillations in the ball heights are expected since only one fan at a time is given an input sufficient to raise the corresponding ball. The indicated average heights are reasonably close. The standard deviations of the heights are 0.14, 0.17, 0.20, and 0.15 m for tubes 1 through 4, respectively.*

Table 1. Allocation results for the nominal and modified plants. Note that the obstruction to flow for tube 4 demands a reallocation that results in more attention given to tube 4 to render the values of the average ball heights close to each other. A side effect is that the average ball heights are generally lower than in the nominal case.

	Nominal Plant		Plant with Obstruction	
	Percent of time each tube is attended	Height avg. (m)	Percent of time each tube is attended	Height avg. (m)
Tube 1	23.22	0.62	24.41	0.59
Tube 2	10.95	0.72	6.98	0.64
Tube 3	22.69	0.70	13.61	0.71
Tube 4	43.13	0.61	54.99	0.51

Resource Allocation Strategy: Dynamic Proportioning

For the dynamic proportioning resource allocation strategy, we first design inner-loop ball height control systems for each tube. We use a simple proportional controller that has a static nonlinear gain on the error $e_i(t) = h_d(t) - h_i(t)$, where $h_i(t)$ is the ball height for the ith tube and $h_d(t)$ is the desired height.

Although a PID controller might perform better, such design improvements for the inner-loop ball height control system are not our focus. Our proportional controller changes the duty cycle of the PWM used to drive the fans, and its gain depends on the sign of the error. The controller will have one slope if the error $e_i(t)$ is positive, and another slope if the error is negative. The two slope values are chosen so that the balls do not drop so fast that they are too difficult to lift at a later time due to inertia.

For the outer-loop resource allocation strategy, we change $h_d(t)$ every 10 s by letting

$$h_d(t) = \max\{h_i(t) : i = 1, 2, 3, 4\}.$$

This strategy directs the fan corresponding to the lowest ball to provide sufficient air to lift the ball to the same height as the highest ball. A time of 10 s is sufficient to achieve the modified ordering. The dynamic iterative application of this simple allocation rule results in elevating the balls to a maximum uniform height. It is essentially impossible for the balls to be at exactly the same height in implementation, owing to noise, air turbulence, and ball overshoot. One ball will be higher than the others at virtually all times. The outer-loop resource allocation strategy commands the lower balls to achieve the position of the highest ball. Over time, the strategy persistently drives the balls to successively higher uniform positions. What emerges is a proportioning of air flow with relative amounts of air allocated to overcoming flow differences between the tubes, as well as disturbances and plant changes. Since the resource allocation strategy proportionally allocates the PWM input, this strategy is different from the juggler strategy. The juggler allocates air pulses over time, whereas the dynamic proportioning strategy continuously allocates air both spatially across the tubes and temporally due to the changes in the commanded value of $h_d(t)$.

To test this allocation strategy, we limit the amount of air entering at the inlet of the input manifold; otherwise, all of the balls will hit the top. To restrict flow at the inlet, a piece of paper with a hole cut in the middle is used to cover the air inlet. In the following, we describe how holes of different sizes in the paper affect dynamic reallocation.

As shown in Figure 3, all the balls reach the desired height $h_d(t)$, except the first one. The balls converge to a common height of around 0.7 m and after two minutes, their heights are within approximately 0.2 m of each other. However, even though the ball in tube 1 is in the 0.2 m range, ball 1 does not reach the maximum desired height $h_d(t)$. This shortcoming is due to several factors. First, air enters in the middle of the input manifold, which is close to tubes 2 and 3. The inner-loop ball height controller for the height of ball 1 continually tries to obtain more air but is at a disadvantage due to the flow restriction. Although we attempted to overcome the differences between the tubes by tuning the inner-loop proportional controllers, we did not dedicate much time to tuning since the inner-loop controls were not our main focus. Adding an integral term to the inner-loop controllers might have reduced the error seen in Figure 3. However, since we did not try a PI controller and since there is strong coupling between the inner- and outer-loop controls, we cannot be certain that such improvements could be obtained. One dominant factor in this coupling is air flow. We cannot measure the patterns of air flow inside the manifold, and the flow patterns make the dynamics inside the manifold change unpredictable due to turbulence. Some effects of air turbulence are easy to see while the experiment is in operation. For instance, even with constant PWM signals to the fans, the balls bounce and spin during operation. We find that the air turbulence creates significant coupling between the inner- and outer-loop controls and hence limits the

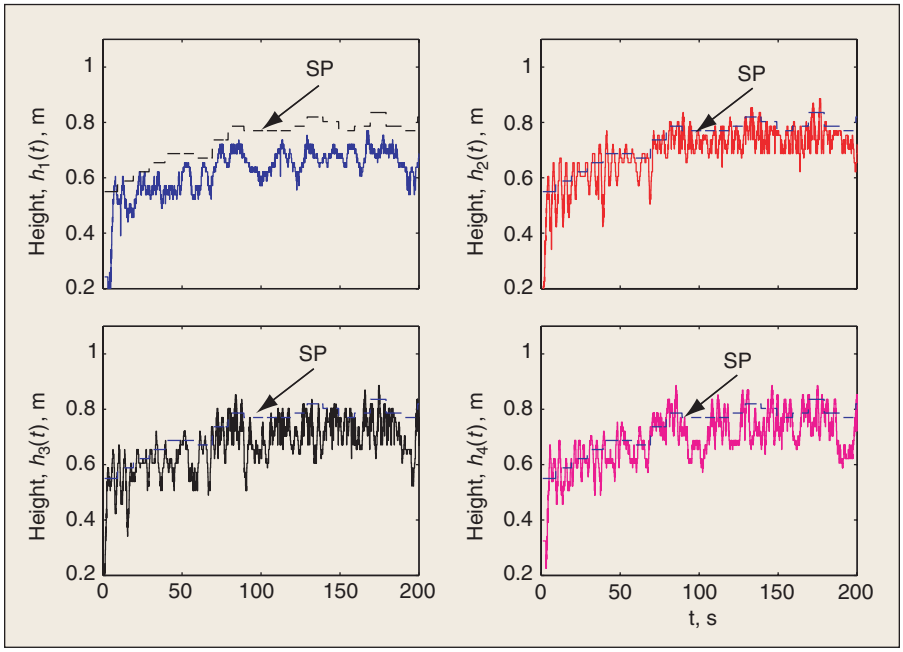


Figure 3. *Dynamic proportioning strategy results. The figure shows the ball height in each tube and the desired height $h_d(t)$, represented by SP. The desired height changes every 10 s to the height of the highest ball.*

achievable performance. Such features have been studied in a variety of applications and are a key challenge in designing distributed and hierarchical controllers for large-scale systems [13], [15], [16].

Experiment modularity facilitates the construction of new configurations for additional research and educational objectives.

Figure 4 illustrates the results of dynamic reallocation when the inlet area is reduced at $t = 20$ s by suddenly covering the inlet with a piece of paper with a small hole cut in it. Since there is not as much air to proportion after the plant change, the balls fall. However, the balls fall somewhat uniformly owing to the dynamic reallocation. For this case, we had to include a constraint that indicates that when all of the balls are below 0.2 m, the desired height is set at a fixed value of 0.4 m. This rule is used in some runs of the experiment to keep the balls from getting stuck at the bottom.

The necessity for such heuristic rules highlights the need to understand the theoretical underpinnings of the dynamic resource allocation strategy. We need to model the plant and analyze the choice of $h_d(t)$ and the effects of dynamics and noise on ultimate achievable height and height uniformity. Resource allocation was originally conceived as a static optimization problem [17]. Extending resource allocation concepts to feedback control systems requires innovations in resource allocation theory to cope with dynamics and disturbances.

Electromechanical Arcade

The electromechanical arcade experiment is designed to be an inexpensive testbed for networked scheduling strategies where information must be shared to achieve success.

Experimental Apparatus and Challenges

This experiment is composed of two main components: guns mounted on the shafts of motors and targets (Figure 5). Each of the two guns has a laser and a photodetector. To fire a gun, the gun's laser is activated by a computer. There are eight targets, which are black boxes arranged in an arc. Each target has a single photodetector and two lasers mounted on top, with a laser pointed at each gun. We represent the appearance and disappearance of targets by whether the target's two lasers are both on or both off. When a target appears, we consider it to have "popped up." A gun can sense the target's appearance by detecting its laser, but only if the gun is pointed directly at that target by its motor. A gun cannot detect the appearance of more than one target at a time. If a gun fires at a target that is currently popped up and the gun's laser successfully triggers the photodetector on the target, then the gun receives a point for hitting the target. However, the targets also disappear and if the target is fired upon at that time, the gun receives no points. The appearance frequencies of the eight targets are independent of each other. The target lasers are driven by simple timing circuits that can be adjusted by hand to provide different frequencies. We use Radio Shack #277-1101 lasers and Radio Shack #276-1657 photodetectors for both the targets and guns.

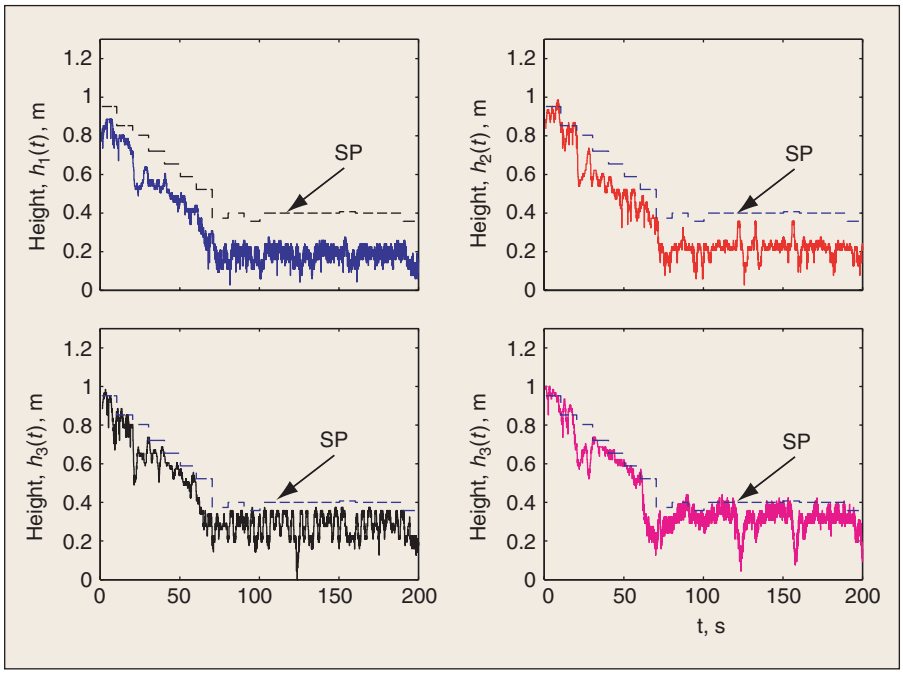


Figure 4. *Dynamic reallocation results. The figure shows the ball height in each tube and the desired height $h_d(t)$, represented by SP. The desired height changes every 10 s to the height of the highest ball. Here, the inlet area is reduced at $t = 20$ s to demonstrate dynamic reallocation.*

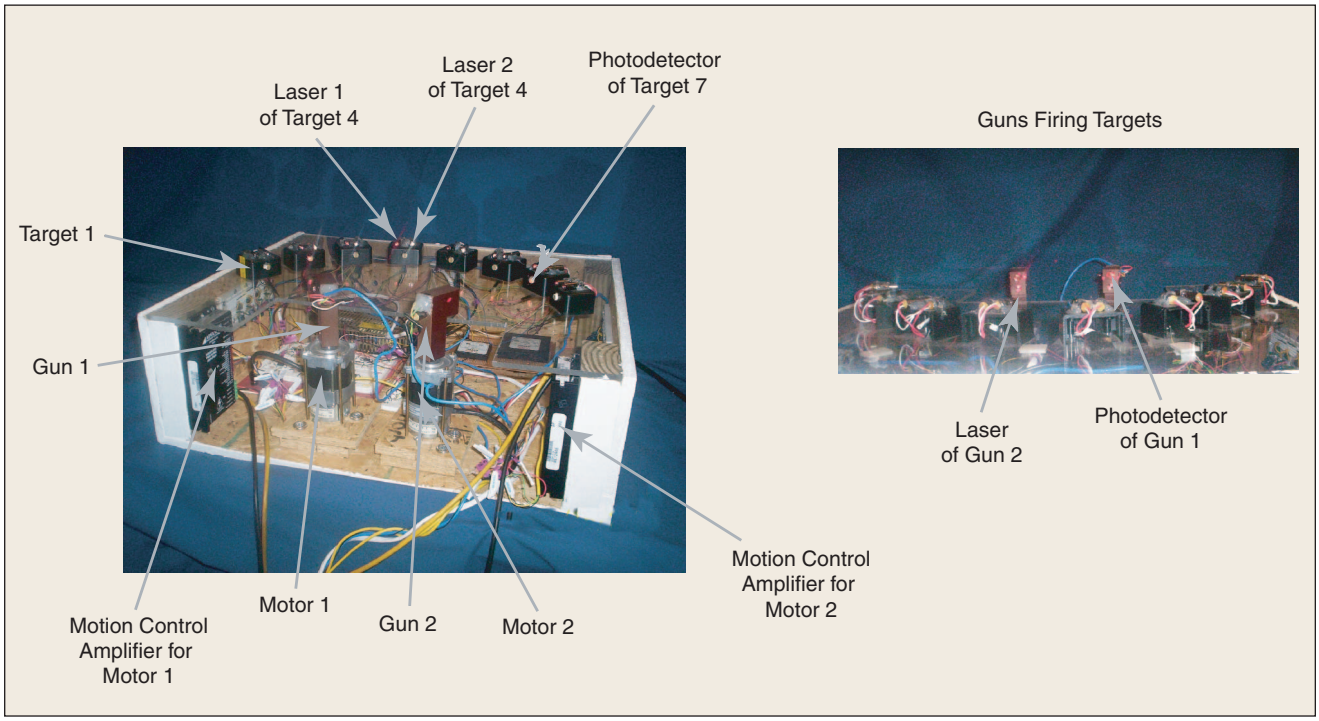


Figure 5. *Electromechanical arcade experiment. The experiment has two guns and eight targets. The guns detect the targets by sensing lasers mounted on the targets. Each gun receives points by firing its laser at targets when the targets appear. Each gun is mounted on a motor that is under computer control so that the gun can be pointed at different targets.*

The two guns are mounted on the shafts of Shinano Kenshi #LA052-040E4N02 motors driven by Advanced Motion Control #BE12A6 amplifiers. Each motor has a quadrature encoder to measure the angular position of the motor. We limit the angular velocity of each motor shaft to make the problem more challenging and to make it possible to view the firing of guns at targets. We use a PID controller to point each gun to the target that needs to be shot, and these PID control system loops are tuned so their performance does not affect our scheduling strategies. The data acquisition system requirements are ten digital inputs to the computer from the target and gun photodetectors, two digital outputs to fire the gun's lasers, two motor shaft encoder interfaces, and two analog outputs for the PID controller inputs to the motor amplifiers. We use the DS1104 dSPACE card.

The cost of the experiment is relatively low, with the greatest expenses occurring from the motors and amplifiers. However, the specifications for the motors are not overly demanding since high torques are not needed. Relatively low shaft rotation speeds are desirable to allow viewing of the arcade. Required gun pointing accuracy is driven by the laser's light dispersal pattern and the size of the photodetector. As long as the targets are kept close to the guns, highly accurate pointing is not needed. In fact, the target appearance timing circuits can be eliminated if additional digital outputs are available as they are on the DS1104. Computer control of the targets provides a flexible way to command the pattern of appearances.

We assume that the guns have no information about the rates of appearance of the targets, although strategies could be invented for estimating appearance sequences. The guns know the positions of all of the targets, and the guns can communicate their decisions to process or pursue targets. The challenges for this experiment are as follows:

- *to schedule a sequence of firings in real time to maximize the number of points received by the team of two guns.* Since target detection and shooting requires movement of the guns, a good schedule typically minimizes the motion of the guns and, at the same time, maximizes point gain. Feedback is required to overcome uncertainty about when targets appear and to develop target appearance time estimation strategies.
- *to cooperatively schedule the shooting of the guns in the presence of imperfect communication between the two guns.* While the communication network can be the Internet, and a computer can be dedicated to each gun, networked schedulers can also be simulated within one computer. Communication imperfections such as random but bounded delays, bandwidth constraints, or message misordering can be considered.

Finally, this distributed feedback scheduling problem can be thought of as a resource allocation problem, analogous to the one faced by the juggler. However, the dynam-

ics, uncertainty sources, and performance objectives are quite different.

Scheduling Strategies and Results

The arcade experiment involves an uncertain environment due to unpredictable target appearance times. Imperfect communication makes it difficult for the two guns to coordinate their actions. Because of uncertainty, it is not generally possible to accurately predict far into the future, and hence optimization approaches are not effective for developing long sequences of target firings either offline or online. Hence, we develop a different approach here.

The targets are numbered, the set of targets is denoted by $P = \{1, 2, \ldots ., 8\}$, and the set of guns is denoted by $Q = \{1, 2\}$. Let p_i, $i \in P$, denote the priority of processing target i, where processing means moving to and shooting at the target when it appears. The priority $p_i > 0$ is proportional to the importance of a target. Let $T_i(t) = p_i t_i$, where $i \in P$ and $t \geq 0$, denote what we call the "prioritized time" elapsed since target i was last shot, where t_i denotes the amount of time elapsed since target i was last shot. Suppose that, initially, $T_i(0) = 0$ for all $i \in P$ so that we act as though we had simultaneously shot all of the targets, which is clearly physically impossible. However, this initialization is useful since the scheduling strategies make decisions about which target to process based on the sizes of the prioritized times $T_i(t)$ for all $i \in P$.

If no gun is used, then, for all $i \in P$, $T_i(t) \to \infty$ as $t \to \infty$, since no targets are processed. The goal of a scheduling strategy is to avoid $T_i(t) \to \infty$ for all $i \in P$. Moreover, a successful strategy tries to keep the values of $T_i(t)$ as small as possible to reflect that the guns have recently shot each target.

There are three types of delays in the experiment that have a significant impact on the choice of scheduling strategy. First, there is the delay between the appearances of each target. Second, there is travel time to move the gun to point at a target. Third, there is a random but bounded delay incurred for each communication between the guns. We implement an asynchronous scheduler over a simulated communication network to pick $i_j^*(t)$, the target that gun j is commanded to process at time t. To define the scheduler, let $U(t) \subset P$ be the set of unattended targets and let $U_j^a(t) = \{i_j^*(t)\} \cup U(t)$ be the set of targets that gun j can consider after processing (shooting at) target $i_j^*(t)$. We pass the set $U(t)$ around the network, let each gun choose a target from $U_j^a(t)$, and include in $U(t)$ the target that gun j just finished processing. A distributed mutual exclusion algorithm is used, and a communication delay is incurred each time $U(t)$ is passed. While a gun is waiting to receive $U(t)$, the gun can keep shooting at $i_j^*(t)$ but receives only one point per appearance of the target. Hence, we consider $i_j^*(t)$ to be attended to until gun j makes the decision and switches to processing a different target.

Distributed Scheduling Strategy: Focus on the Target Ignored the Longest

Next, we introduce a scheduling strategy from [18] and [19] that is an extension of a strategy defined in [20] and for which stability properties have been investigated in [19] and [21]. First, let D_j denote the decision time for gun j. At each decision time, gun j chooses a target to process. Only one gun can choose a target to process at each time due to the mutual exclusion algorithm. Suppose that $D_j = 0$ for all $j \in Q$ so that at $t = 0$ one gun chooses which target to process.

One candidate scheduling strategy chooses to process a target if the target is ignored longer than the average of all the lengths of time that the targets were ignored. Under such a strategy, gun j chooses to process target $i_j^*(D_j)$ at time D_j if

$$T_{i_j^*}(D_j) \geq \frac{1}{N - M + 1} \sum_{i_j \in U_j^a(D_j)} T_{i_j}(D_j).$$

Here, $N = 8$ targets and $M = 2$ guns. Gun j makes no other decision until it has finished shooting target i_j^* and receives $U(t)$. A special case of this process is a strategy that picks $i_j^*(D_j)$ such that $T_{i_j^*}(D_j)$ is bigger than every other $T_{i_j}(D_j)$, i.e., a process-the-target-ignored-the-longest strategy. If there is more than one maximizer for any gun at any time, then the strategy simply chooses one of these targets at random.

Figure 6 shows the results for the process-the-target-ignored-the-longest strategy. In the following, we use a fixed communication delay of 10 s to facilitate comparison to the approach. Figure 6 shows the values of $T_i(t)$ for all $i \in P$ as well as the targets selected by the two guns during the time the experiment is running. First, consider the initial choices made by the guns. Before the experiment starts, both guns are pointing at target 1. Once the experiment starts, gun 1 chooses a target to process, while gun 2 must wait 10 s for the arrival of the set $U(t)$ sent by gun 1. During this 10 s interval, gun 2 detects target 1, fires on it and receives a point, and keeps processing target 1. Therefore, $T_1 = 0$ in this interval.

Next, consider the pattern of the trajectories in Figure 6. First, consider the effect of the priority values p_i on the variable T_i. We assigned to the targets the priorities $p_1 = 0.9$, $p_2 = 0.85$, $p_3 = 0.8$, $p_4 = 0.75$, $p_5 = 0.4$, $p_6 = 0.4$, $p_7 = 0.5$, and $p_8 = 0.2$. These values were chosen by observing the illumination frequencies of each target and assigning higher priorities to the targets that appear more frequently. With these priorities, the guns return more frequently to targets that appear more frequently. Second, consider the effects of the communication delay on the performance of the strategy. For instance, notice how the values of T_i contained in the set $U(t)$ are ignored during the communication delays. Third, notice how the guns

allocate their processing capabilities to shoot and gain more points for as many targets as possible in a cooperative manner. Cooperative behavior can be seen by the sequence of choices over the time range in the latter half of the experiment in Figure 6.

Distributed Scheduling Strategy: Focus on Closest Highest Priority Target

We now introduce a strategy that schedules the next target to avoid ignoring high priority targets for too long and, at the same time, minimize travel time to process targets. We view the electromechanical arcade experiment as analogous to a type of cooperative scheduling for autonomous military vehicles firing on targets. From this perspective, it is important to pay attention to minimizing the travel time between targets. This consideration motivated us to define the strategy [21] that the gun processes target $i_j^*(D_j)$ at time D_j if

$$T_{i_j^*}(D_j) - \delta_{ji_j^*}(D_j) \geq \frac{1}{N-M+1} \sum_{i_j \in U_j^a(D_j)} \left[T_{i_j}(D_j) - \delta_{ji_j}(D_j) \right]. \tag{1}$$

Here, $N = 8$, $M = 2$, and δ_{ji_j} is the amount of time required for the gun to move from target j to target i_j. Choice of the maximum value on the left hand side of (1) results in a process-the-closest-highest-priority-target policy. Ties are broken with an arbitrary choice. The left side of (1) can be viewed as a cost function, and the scheduler selects a target in order to minimize the time the targets are ignored and the travel time. A bound on the ultimate longest time that any gun ignores target i is given in [21].

Figure 7 shows the results for the process-the-closest-highest-priority-target strategy. As in the previous case, the communication delay is fixed and equal to 10 s, and the priority values are the same. The number of points in this case is greater than or equal to the number of points obtained in Figure 6. Figure 7 shows that the value of the peak for each T_i is less than the peaks seen in Figure 6. By attempting to minimize travel time, we improve our score. Data in Figure 7, specifically in the time range $t \in [0, 25]$, show that the closest targets are chosen by the guns more frequently than those for the corresponding time range in Figure 6.

We quantify the characteristics highlighted previously by using performance metrics. There are many ways to measure performance of the scheduling strategies. Here, at each step k, we compute the average $(1/N)\sum_{i=1}^{N} T_i(k)$ of the time elapsed since any target has been shot. First, we compute the time average of this quantity, which is the time average of the average values. In addition, we compute the maximum average value achieved over the entire run of 100 s. At each time step k, we compute the maximum time $\max_i\{T_i(k)\}$ that any target has been ignored. Second, we compute the time average of this quantity, which is the time average of the maximum values. Next, we compute the maximum of the maximum values achieved over the entire run. Finally, we compute the number of points obtained by the two guns during the experiment, which is the total number of targets shot.

Table 2 shows the results for the two scheduling strategies. The process-the-closest-highest-priority-targets

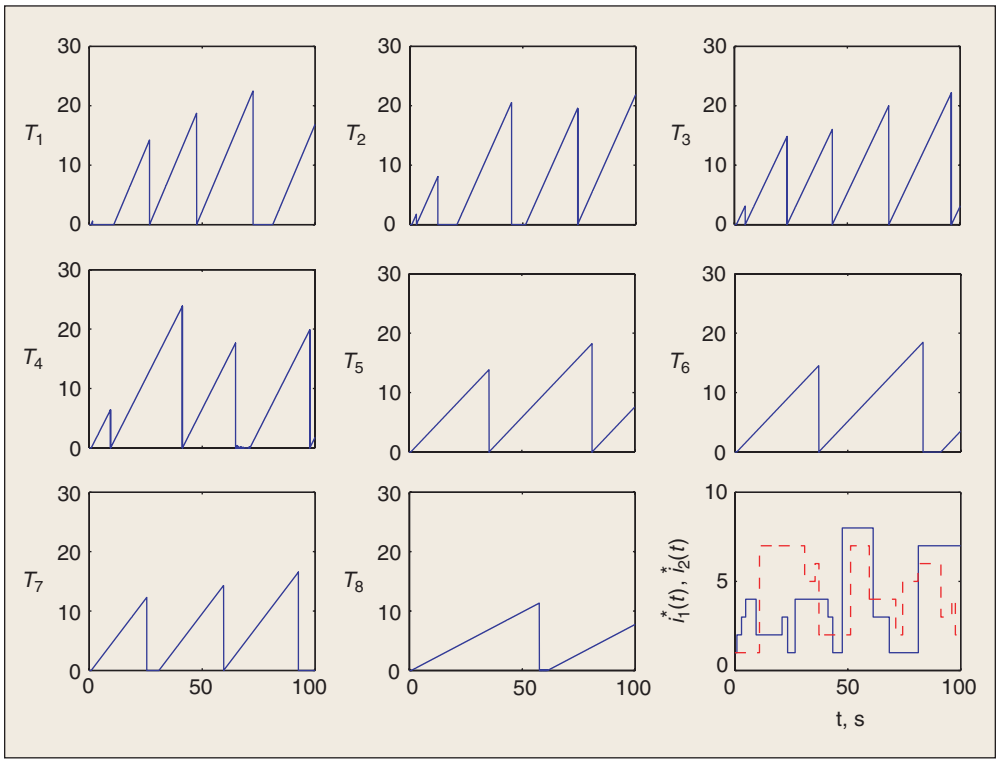


Figure 6. *Performance of the process-the-target-ignored-the-longest strategy. At time $t = 0$, both guns are pointing to target 1. There is a fixed communication delay of 10 s. The values of T_i plotted versus t show the prioritized time at which target $i \in \{1, \ldots, 8\}$ was last shot. In the plot on the lower right-hand corner, the solid line corresponds to $i_1^*(t)$, and the dashed line corresponds to $i_2^*(t)$.*

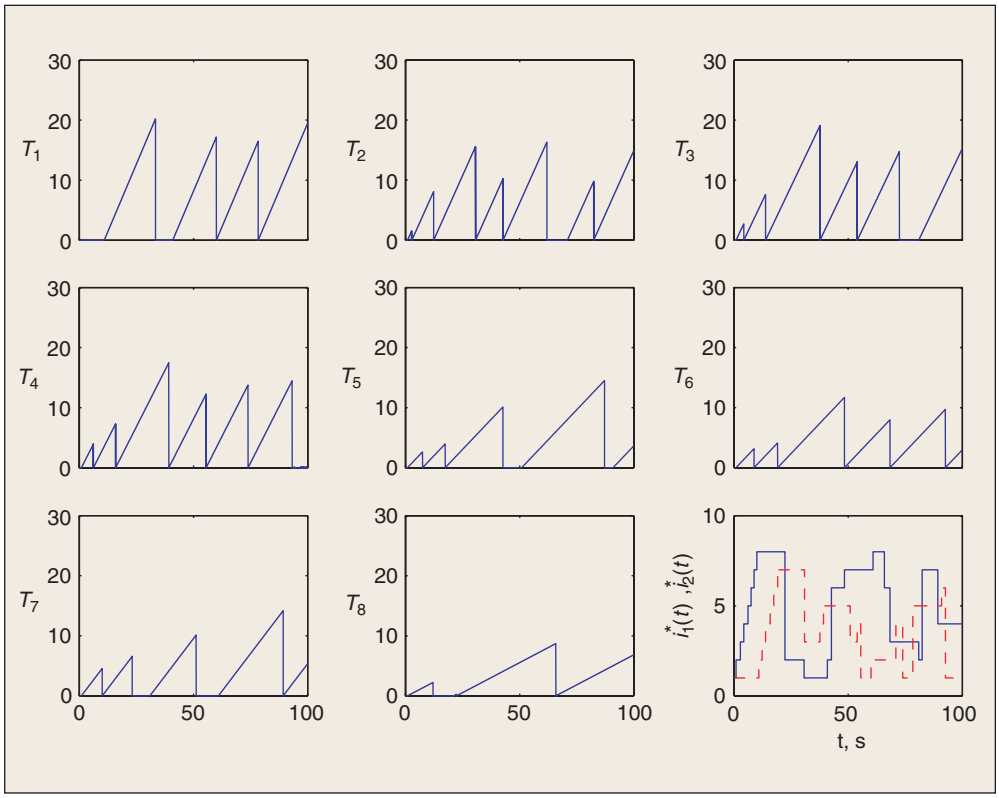


Figure 7. *Performance of the process-the-closest-highest-priority-targets strategy. At $t = 0$, both guns point to target 1. There is a fixed communication delay of 10 s. The values of T_i plotted versus t show the prioritized time at which target $i \in \{1, \ldots, 8\}$ was last shot. In the plot in the lower right-hand corner, the solid line corresponds to $i_1^*(t)$, and the dashed line corresponds to $i_2^*(t)$. Performance of this strategy is superior to the process-the-targets-ignored-the-longest strategy since inter-target travel times are used by the scheduler.*

strategy performs better for all performance measures, including the total number of points obtained. Seeking to minimize travel time allows significant point improvement. The strategy does not, however, partition the targets so that each gun is always responsible for a few adjacent targets. What emerges in the bottom right plot in Figure 7 is a type of cooperation that balances the objectives quantified in (1) for the process-the-closest-highest-priority-targets strategy.

Table 2. Performance measures for the arcade. The process-the-closest-highest-priority-targets strategy is superior for each performance measure due to its persistence in seeking to fire on nearby targets.

	Ignored-the-Longest	Closest-Highest-Priority
Average of average	7.28	5.22
Maximum of average	11.15	8.69
Average of maximum	14.46	11.56
Max of max	23.87	20.21
Number of points	25	35

A number of other methods can be tested on the electromechanical arcade. In addition to extensions of the strategies in [20], there are manufacturing system scheduling methods, both deterministic and stochastic, that might be useful for this experiment [22], [23]. Moreover, it would be interesting to include manual control and challenge two teenagers to beat the above strategies for this game.

Multizone Temperature Control

Temperature control experiments are perhaps the least expensive experiments that possess interesting distributed networked control problems. Here, we briefly describe two multizone temperature control experiments, namely, a planar grid of sensors and heaters, and a building temperature control problem. These experiments include traditional multivariable tracking objectives, as well as objectives found in dynamic resource allocation problems.

Experimental Apparatus and Challenges

Our first experiment has 16 individually controlled zones arranged in a planar grid as shown in Figure 8. Each zone has a light for a heater and an analog temperature sensor. Ambient temperature variations and wind currents have a significant impact. Since some of the heaters are close to other zones' sensors, there is significant inter-zone temperature coupling. Moreover, experiments show that air currents can be generated by heater-induced spatial temperature gradients.

There are several practical challenges in implementing this experiment. First, since each DS1104 dSPACE card supports only eight analog inputs, we need two computers to acquire the data from the sensors. Sensed information and actuation signals are transmitted

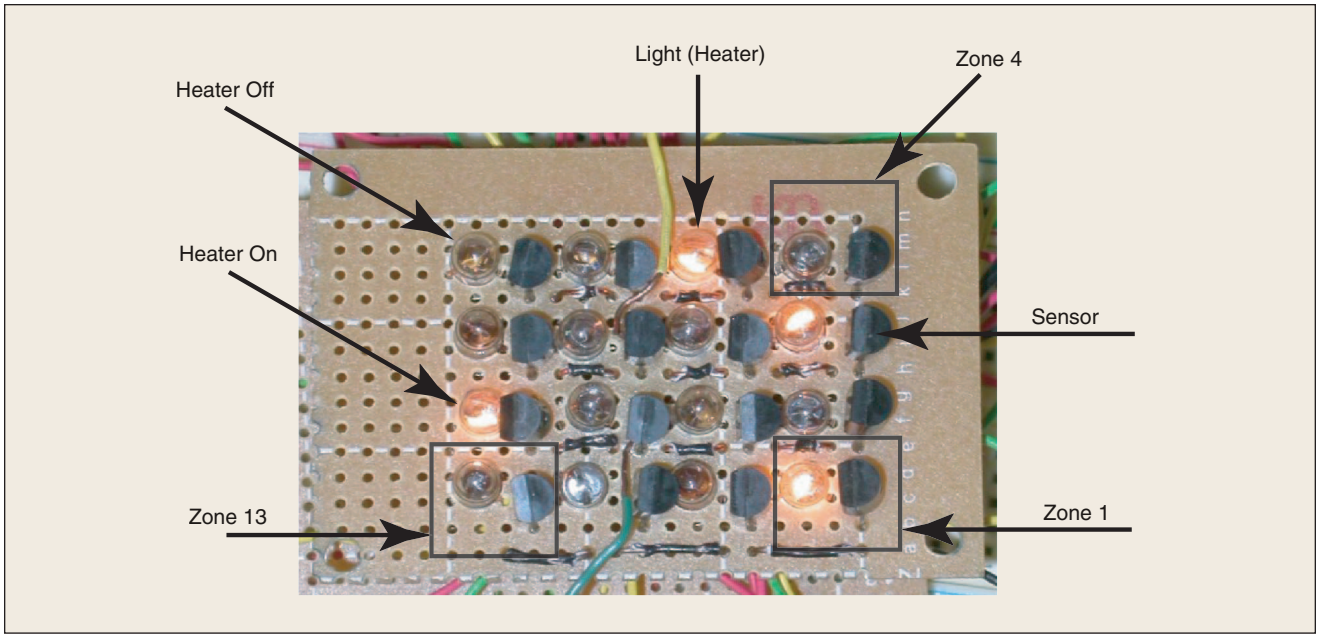


Figure 8. *Planar temperature grid control experiment. The experiment has 16 individually controlled zones, each with a light for a heater and an analog temperature sensor.*

from one computer to the other by means of an RS-232 link. The master computer, which acquires data from the slave, is in charge of the control strategies. Second, there is a significant calibration problem. We chose 16 LM35CAZ sensors from National Semiconductor, which have essentially identical characteristics. However, soldering the sensors to the board might change the sensor characteristics in an unknown way. Although sensor calibration after construction of the experiment is possible, it is tedious, time consuming, and error prone. We used a Fluke Meter 179 temperature probe to provide the ambient temperature at the start of the experiment since it provides a measure for comparing the achieved temperatures in two successive runs of the experiment.

The cost of the sensors and actuators is less than $2 per zone. Supporting electronics includes inexpensive drivers for the lamps. A single-zone version of the experiment provides students with experience using the dSPACE system. An eight-zone version can be used in conjunction with a single dSPACE DS1104 board, and a communication network can be simulated in that case. Students are able to construct the experiment themselves due to the simplicity and low cost.

The main challenges of this experiment are as follows:

- temperature regulation uniformly across the grid but with a fixed maximally elevated value; alternatively, we can seek a fixed or dynamic two-dimensional pattern of temperature values
- temperature tracking in which one set of zones tracks the average temperature in another zone, or a reference temperature value
- distributed control with different controllers for each zone, and a communication network with random but bounded delays for neighbor-zone sensed information, and delayed control inputs.

Distributed Control Strategies and Results

In the following, we show results for centralized and distributed resource allocation strategies. It is useful, however, to highlight several other control strategies that can be studied for this experiment. For instance, we have implemented a temperature tracking system with the 16 zones partitioned into four superzones. One goal is to force each superzone to track desired temperatures $T_1^{ds}, \ldots, T_4^{ds}$, that are set by the user. Another challenge is the requirement that superzones 3 and 4 track the current average temperatures of superzones 2 and 1, respectively, when the user sets the desired temperatures for superzones 2 and 1.

Suppose that the control objective is to reach a maximally elevated common temperature for all of the zones, but given the constraint that only one light can be on at a time. For this problem, we first consider a centralized resource allocation strategy. Let the temperatures in the ith zone be denoted by T_i and the ith heater input by u_i. At each time instant our strategy applies a 100 ms duration pulse to the zone that has the lowest temperature. This strategy allocates a resource defined by the time that the light is on or, equivalently, the electrical current available for heating. In other words, at each time k and for each zone i,

$$u_i(k) = \begin{cases} 1, & \text{if } T_i(k) \le T_j(k) \quad \text{for all } j \in \{1, 2, \ldots, 16\}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Intuitively, this strategy seeks to iteratively increase the minimum temperature, thereby forcing temperature uniformity while maximizing all of the temperatures. Resource allocation emerges from the simple optimization strategy implemented by (2). Notice the conceptual similarity to the juggler strategy for the balls-in-tubes experiment.

For this strategy, the whole grid reaches a practically uniform final temperature value as shown in Figure 9. We start with a room temperature of 22.8 °C, and the average final value is about 24.1 °C. Some of the zones are above this value; the maximum temperature is 24.8 °C. However, considering the disturbances associated with the experiment, the performance is quite good relative to all other strategies we have studied for this experiment. Since the room temperature is variable, each time we run an experiment, the final average steady state value is different.

Another strategy is a distributed version of the previous strategy. In particular, we assume that there is a local controller for each zone that can make decisions about whether to turn its heater on or off, but based only on the temperature in its own zone and possibly those in some adjacent zones. However, the local controllers can make decisions using only temperature data obtained from adjacent zones by way of a communication network that has random but bounded delays. To define the strategy, let $N(i)$ denote the set of neighbors of zone i, including zone i, from which the controller for zone i is given temperature data. For all i, let

$$T_i^{\min}(k) = \min \{T_j(k) \ : \ j \in N(i)\}$$

and

$$u_i(k) = \begin{cases} 1, & T_i(k) = T_i^{\min}(k), \\ 0, & \text{otherwise.} \end{cases}$$

Intuitively, this control law is a distributed version of the strategy in (2). More than one zone can have its heater on at a time, and hence the strategy can simultaneously increase multiple temperature values. For an ambient temperature of 21.8 °C, we obtain the results shown in Figure 10. After 200 s, the temperature in some of the

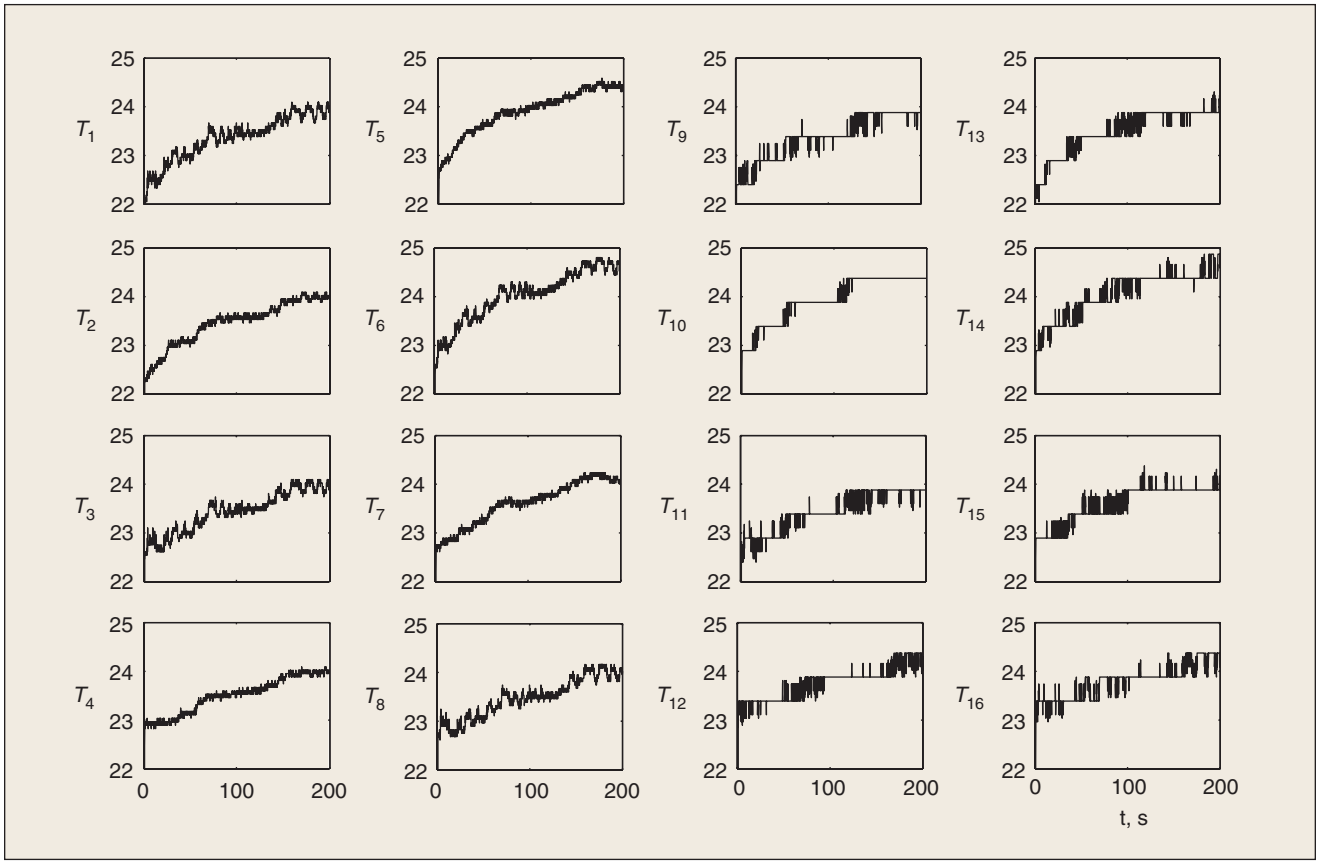


Figure 9. *Temperature values for a centralized resource allocation experiment. This experiment was performed with a 22.8 °C ambient temperature. After 200 s the average final value for the temperature is 24.1 °C.*

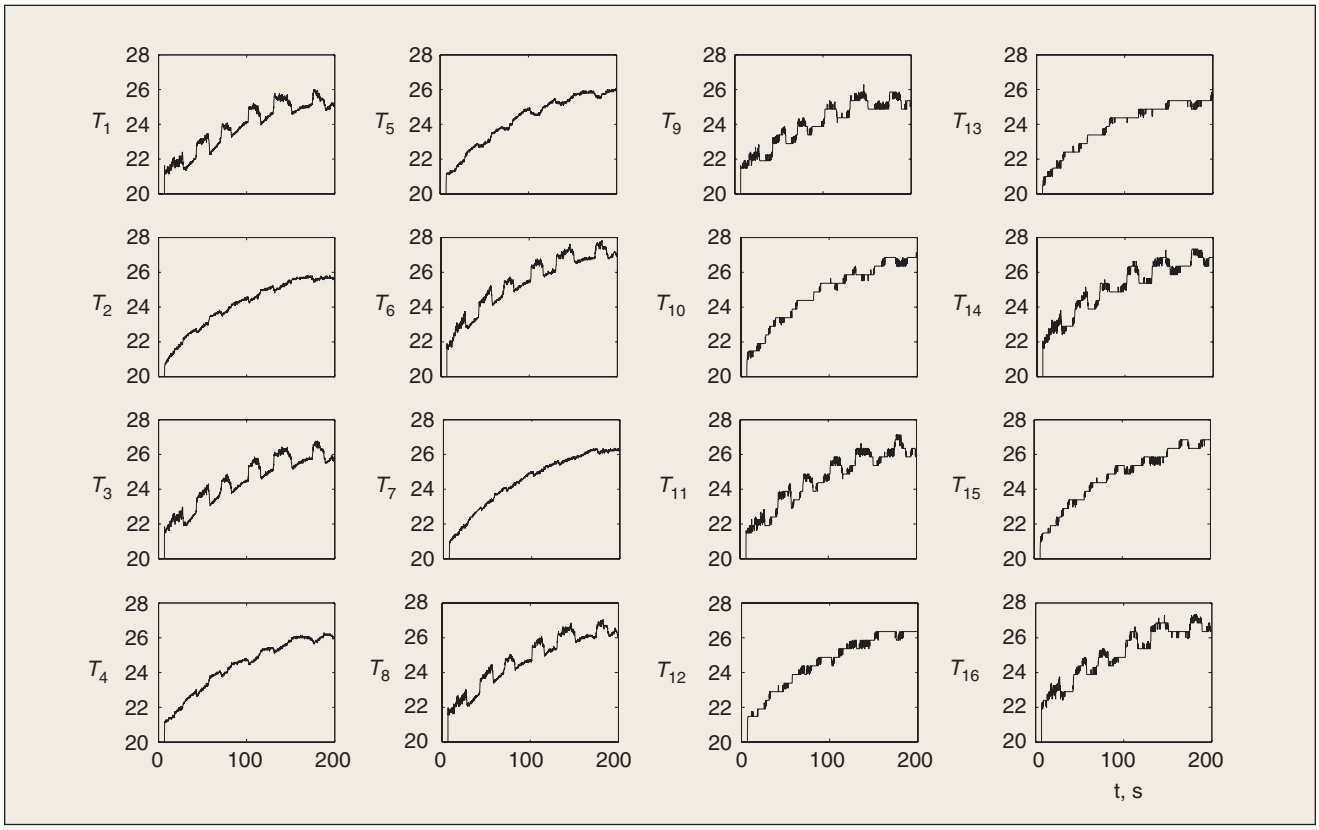


Figure 10. *Temperature values for a distributed resource allocation experiment. Using a sensing topology based on gathering sensed temperature data from neighboring zones only, and starting from 21.8 °C, the final temperatures are between 26 °C and 27 °C. Some of the zones on the edges cannot reach this value because of the large disturbance effects. The controller's attempts to overcome such disturbances amplify the grid's temperature differences by heating the middle zones while trying to raise the edge zones' temperatures.*

zones is between 26–27°C. There are other zones on the edges that cannot reach this value because of the large disturbances. On the other hand, there are several zones in the middle of the grid, such as T_6, that have relatively large temperature values. These large values result from the lamps on the edge zones being on practically all the time. Moreover, note that relative to the centralized resource allocation case in Figure 9, there are more oscillations in the temperature values. Note that different vertical scales are used in the two cases to facilitate viewing the data. The oscillations are due to the effects of the limited amount of information used by each local controller in the distributed controller.

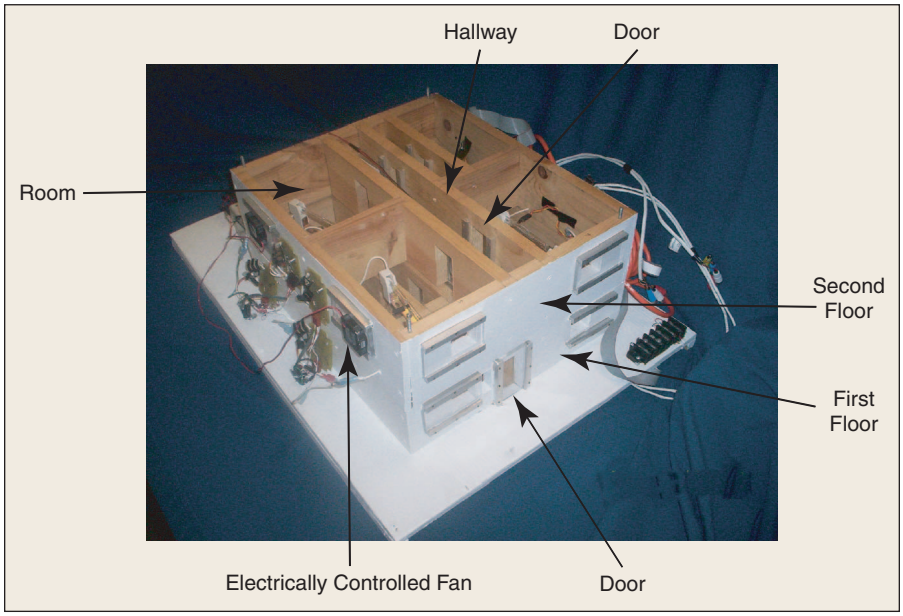


Figure 11. *Building temperature control experiment. This experiment is a two-floor model with four rooms and hallways on each floor. Each room has a temperature sensor and a heater. The model building is 39 cm wide, 46 cm deep, and 22 cm high.*

Building Temperature Control

We constructed the scale-model building in Figure 11 to study multizone temperature control. The building temperature control experiment uses a two-floor model with four rooms on each floor. Each of the rooms has a temperature sensor and a heater. The two floors are separable so that coupling between floors can be considered or eliminated. Each floor has windows and doors with adjustable openings, allowing the interaction pattern between the temperatures in the rooms to be adjusted. Moreover, there are several electronically controllable fans that can be inserted into windows or doors to simulate ambient wind effects and drafts. These fans speed up the temperature interaction effects. The cost of the experiment is dominated by the power supply.

The main challenge for this experiment is multivariable temperature tracking. We can also consider challenges analogous to the resource allocation objectives for the planar grid. For more information, see "Additional Resources."

Finally, we highlight alternative strategies that can be useful for the temperature control problems. First, distributed temperature control is used in the semiconductor processing area. For instance, achieving a uniform temperature on a plate is studied in [24]. Distributed control of wafer temperature for multizone rapid thermal processing systems is studied in [25]. In [26], the authors describe a multizone space heating system that maintains a desired temperature in different zones. Accurate multizone temperature control is of significant importance in personal computers, and several current strategies are described in [27]. Another approach for distributed control design with a tracking objective is described in [28], where the authors show how systems with a spatial interconnection topology can be controlled using conditions that can be expressed as linear matrix inequalities.

Concluding Remarks

We introduced several inexpensive experiments for studying the design and implementation of distributed and networked dynamic resource allocation, scheduling, and control strategies. In each case, we presented an overview of the experimental apparatus, the challenges involved, and, for three of the testbeds, showed experimental results for distributed decision-making strategies.

Due to their modular nature, the experiments are easily expandable. Adding more tubes and a reconfigurable input manifold can provide interesting new challenges for the balls-in-tubes experiment. With a grid of tubes, the experiment's operation will be more visually appealing since it will present a sheet of balls that can dynamically deform in three dimensions. An easily reconfigured input manifold can provide opportunities for studying the effect of plant configuration changes on allocation performance. The electromechanical arcade can be expanded in many ways. For example, it can be extended to two or three dimensions with two opposing teams. The planar grid for multizone temperature control was chosen to be of minimum size, but still interesting since it has edge effects and interior zones. It is of interest to make the grid larger or in a different shape, since different disturbance patterns will arise. Significant expansion of the experiments will increase the data acquisition requirements so that our software/hardware choices might need to be reevaluated. One potential solution is the emerging networked embedded systems approach, as described in "Additional Resources."

Overall, these ideas for experiment redesign highlight the existence of tradeoffs among experimental apparatus design, the choice of hardware and software, and research and educational objectives. It is hoped that this article will help educators design laboratories that support the study of information technology–enabled distributed feedback control.

Acknowledgments

This work was supported by the Intelligent Systems Division of the NIST. The inputs of James Albus, Fred Proctor, and William Shackleford at NIST are gratefully acknowledged, as are the helpful suggestions of the reviewers. A number of other people contributed to the construction of the experiments that we describe here, and we gratefully acknowledge their assistance. First, we would like to thank William Thalgott and Emmet Evans for building the balls-in-tubes experiment and helping with construction details for the electromechanical arcade and multizone temperature control experiments. The balls-in-tubes experiment and the arcade were constructed by groups of undergraduates in an EE 682P design project. The arcade was later improved by Kristina Guspan and Dawn M. Kin. The building temperature control experiment was initially constructed by a group of undergraduates in an EE 682P design project and significantly improved by Todd Broceus and Tyson Rathburn. Jorge Finke added the electrically controllable fans, interface cables, and implemented a controller in dSPACE.

References

[1] T. Samad and G. Balas, Eds., *Software-Enabled Control: Information Technology for Dynamical Systems*. Piscataway, NJ: IEEE Press, 2003.

[2] OSU Cooperative Control Laboratory [Online]. Available: http://www.ece.osu.edu/ ~umit/ee757.htm

[3] Carnegie Mellon Robotics Institute [Online]. Available: http://www.ri.cmu.edu/lab_ lists/index.html

[4] Crossbow Wireless Sensor Networks [Online]. Available: http://www.xbow.com/Products/Wireless_Sensor_Networks.htm

[5] R. M. Murray, "Future directions in control, dynamics and systems: Overview, grand challenges and new courses," *Eur. J. Contr.*, vol. 9, no. 2, pp. 144–158, 2003.

[6] dSPACE [Online]. Available: http://www.dspace.de/ww/en/pub/home.htm

[7] Quanser Consulting [Online]. Available: http://www.quanser.com

[8] OSU Control Systems Implementation Laboratory, *EE 758* [Online]. Available: http://www.ece.osu.edu/~passino/ee758.html

[9] NIST RCS [Online]. Available: http://www.ece.osu.edu/~passino/RCSweb/

[10] National Instruments and LabVIEW [Online]. Available: http//www.ni.com/labview/

[11] M.P.J. Fromherz and W.B. Jackson, "Force allocation in a large-scale distributed active surface," *IEEE Trans. Contr. Syst. Technol.*, vol. 11, no. 5, pp. 641–655, 2003.

[12] Mathworks [Online]. Available: http://www.mathworks.com/products/simulink/

[13] V. Gazi, M. Moore, K. Passino, W. Shackleford, F. Proctor, and J. Albus, *The RCS Handbook: Tools for Real Time Control Systems Software Development*. New York: Wiley, 2000.

[14] M. Moore, V. Gazi, K. Passino, W. Shackleford, and F. Proctor, "Design and implementation of complex control systems using the {NIST-RCS} software library," *IEEE Control Syst. Mag.*, vol. 19, no. 3, pp. 12–28, 1999.

[15] A.N. Michel and R.K. Miller, *Qualitative Analysis of Large Scale Dynamical Systems*. New York: Academic, 1977.

[16] D.D. Šiljak, *Large-Scale Dynamic Systems Stability and Structure*. New York: Elsevier, 1978.

[17] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches.* Cambridge, MA: MIT Press, 1988.

[18] K. Passino, *Biomimicry for Optimization, Control, and Automation*. London: Springer-Verlag, 2004.

[19] A.E. Gil and K.M. Passino, "Stability analysis of network-based cooperative resource allocation strategies," in *Proc. IEEE Conf. Decision Control*, 2003, pp. 1206–1211.

[20] J. Perkins and P. Kumar, "Stable, distributed, real-time scheduling of flexible manufacturing/assembly/disassembly systems," *IEEE Trans. Automat. Contr.*, vol. 34, no. 2, pp. 139–148, 1989.

[21] A.E. Gil, K.M. Passino, S. Ganapathy, and A.G. Sparks, "Cooperative scheduling of tasks for networked uninhabited autonomous vehicles," in *Proc. IEEE Conf. Decision Control*, 2003, pp. 522–527.

[22] M. Pinedo, *Scheduling: Theory, Algorithms, and Systems.* Englewood Cliffs, NJ: Prentice-Hall, 2002.

[23] S.B. Gershwin, *Manufacturing System Engineering*. Englewood Cliffs, NJ: Prentice-Hall, 1994.

[24] C.D. Schaper, K. El-Awady, and A. Tay, "Spatially-programmable temperature control and measurement for chemically amplified photoresist processing," in *SPIE Conf. Process, Equipment, Materials Control Integrated Circuit Manufacturing V*, 1999, vol. 3882, pp. 74–79.

[25] C.D. Schaper, T. Kailath, and Y.J. Lee, "Decentralized control of wafer temperature for multizone rapid thermal processing systems," *IEEE Trans. Semiconduct. Manufact.*, vol. 12, no. 2, pp. 193–199, 1999.

[26] M. Zaheer-uddin, R.V. Patel, and S.A.K. Al-Assadi, "Design of decentralized robust controllers for multizone space heating systems," *IEEE Trans. Contr. Syst. Technol.*, vol. 1, no. 4, pp. 246–261, 1993.

[27] P.E. Ross, "Beat the heat," *IEEE Spectr.*, vol. 41, no. 5, pp. 38–43, 2004.

[28] R. D'Andrea and G.E. Dullerud, "Distributed control design for spatially interconnected systems," *IEEE Trans. Automat. Contr.*, vol. 48, no. 9, pp. 1478–1495, 2003.

Nicanor Quijano received his B.S. degree in electronical engineering from Pontificia Universidad Javeriana (PUJ), Bogotá, Colombia, in 1999. In 2002, he received his M.S. degree in electrical engineering from the Ohio State University, where he is currently working toward the Ph.D. degree in electrical engineering. From 1999 to 2000, he was an instructor at PUJ. He has also worked for BPX Colombia, and Capitel, Telecom. His research interests include hierarchical and distributed methods, dynamic resource allocation, and evolutionary game theory.

Alvaro E. Gil received his B.S. and M.S. degrees in electrical engineering from Instituto Universitario Politécnico, Barquisimeto, Venezuela, in 1990 and 1998, respectively, and the Ph.D. degree in electrical engineering from the Ohio State University, Columbus, in 2003. He is currently a postdoctoral researcher at Ohio State University. From 1990 to 1999, he held engineering positions in the Automation Department at Petróleos de Venezuela in Maracaibo, Venezuela. From 1999 to 2002, he was supported by FUNDAYACUCHO and PDVSA, and from 2002 to 2003 he was a research associate at the Department of Electrical Engineering, the Ohio State University. His current research interests include stability analysis of networked cooperative resource allocation strategies.

Kevin M. Passino (passino.1@osu.edu) received his Ph.D. in electrical engineering from the University of Notre Dame in 1989. He is currently a professor of electrical and computer engineering at the Ohio State University (OSU) and director of the OSU Collaborative Center of Control Science. He has served the IEEE Control Systems Society in many roles: as the vice president of technical activities, an elected member of the Board of Governors, Program chair of the 2001 IEEE Conference on Decision and Control, and a Distinguished Lecturer. He is coeditor of *An Introduction to Intelligent and Autonomous Control* (1993), coauthor of *Fuzzy Control* (1998), coauthor of *Stability Analysis of Discrete Event Systems* (1998), coauthor of *The RCS Handbook: Tools for Real Time Control Systems Software Development* (2001), coauthor of *Stable Adaptive Control and Estimation for Nonlinear Systems: Neural and Fuzzy Approximator Techniques*, (2002), and author of *Biomimicry for Optimization, Control, and Automation* (2004). He can be contacted at the Department of Electrical and Computer Engineering, The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210 USA.