S.B. Bowling, R.A. Ford, and F.W. Vote

Design of a Real-Time Imaging and Discrimination System

A novel architecture achieves an extraordinary operating speed by linking conventional computing devices — general-purpose computers, high-speed bulk storage, and array processors — in a parallel, distributed network. The network performs calculations at a rate of 450 million operations per second and transfers data at a rate of 160 MB/s, which satisfies the computation-intense requirements of automatic real-time target imaging, discrimination, and classification.

Computation-intense operations on large amounts of data can be performed by costly supercomputers — or by a newly developed network of general-purpose computing devices. By using this network, Lincoln Laboratory has cut the computer time required for analysis of radar data by many orders of magnitude.

For many years, the size, shape, and motion of targets have been determined by applying imaging techniques to radar data. These analyses were used for evaluation of target-identification procedures, establishment of an intelligence data base on foreign objects, and diagnosis of problems in the deployment of objects in space.

Until recently, computational limitations compelled us to analyze these data in a postmission, batch-processing mode. In 1985, however, Lincoln Laboratory initiated development of a parallel, distributed network of computing devices that could generate radar images and identify targets in real time. Although a supercomputer could achieve this vast performance improvement, our network is a much less expensive solution.

The architecture of the network exploits the parallel nature of the imaging and classification algorithms. These algorithms execute operations simultaneously on large amounts of data. Although commercial array-processing computers are designed to operate on this type of problem (see the box, "Array Processors Used with Radar Systems"), no single array processor can provide the operating speed required for our application. Therefore, we developed a parallel, distributed network that links conventional serial computers and array processors and that provides performance levels many times greater than that available from commercial networks.

Two nearly identical facilities have been built: the Lexington Discrimination System (LDS), located at Lincoln Laboratory in Lexington, Mass.; and the Kwajalein Discrimination System (KDS), located at the Kiernan Reentry Measurements Site (KREMS) in the Marshall Islands. KREMS is administered by the Laboratory under the direction of the U.S. Army at the Kwajalein Atoll. The LDS/KDS project is sponsored by the U.S. Army Strategic Defense Command.

The LDS is the primary research tool used for the development of the signal- and data-processing software that the Laboratory will eventually put into the field and support at the KDS. It enables Laboratory staff members to develop and test procedures and algorithms in a real-time environment. Moreover, the LDS can simulate a wide variety of sensors — at the same speed and in the same format as the actual sensors. This flexibility has made the LDS the focal point for discrimination research at the Laboratory.

The KDS, shown in the photograph in Fig. 1, is installed at the 35-GHz millimeter-wave (MMW) radar at KREMS, and is customized to that sensor. The KDS performs imaging and classification functions in live time — while an object is actually being tracked by the radar. Control of the radar is currently a manual process, but an

automatic mode is planned. That is, the KDS will be able to use intermediate results to change the radar's transmissions and thereby more efficiently to determine the size, shape, and speed of a target. Thus, the KDS will become part of a closed-loop system and will optimize use of the radar's resources.

LDS/KDS Operational Requirements

When the LDS/KDS program began in 1985, the Lincoln Laboratory team surveyed all radars of interest to the imaging and discrimination community. One of the basic requirements of the system was that it accommodate the real-time data-processing needs of the relevant radar systems. High-resolution range-Doppler imaging of rapidly spinning targets was the most computation-intense application.

The generation of target images required computing power that clearly exceeded the capability of super-minicomputers. The benchmark goal for our system was the production of 30 images per second in each of two orthogonal polarizations — a total of 60 frames per second. Each frame had 128 range gates, and a 128-point fast Fourier transform (FFT) had to be computed for each range gate. Thus the processing system had to calculate 7,680 ($2 \times 30 \times 128$) 128-point complex FFTs per second. This computation requirement translates into an operating speed of 150 million floating-point operations per second (Mflops), a memory-access speed that supports this operating speed, and an I/O subsystem capable of a sustained transfer rate of 20 MB/s (the data rate of the radar sensor).

No single commercial array processor was equal to the task. Had we combined five or more array processors, we might have been able to obtain sufficient required processing power, but orchestrating the operation of that many processors would be overwhelmingly complex. However, we found that a single Star Technologies ST-100 array processor was capable of generating 50 frames per second, so two of them, each dedicated to images of a specified polarization, could provide the required 60 frames per second with some processing power to spare. Yet even with the extra processing power provided

Array Processors Used with Radar Systems

Lincoln Laboratory pioneered the use of array-processing computers in large ground-based radars in 1973 when it installed one of the first commercially available digital signal processors, the SPS-81, in the Long Range Imaging Radar (LRIR). The LRIR, located in Westford, Mass., is a high-power wideband radar used to interrogate deep-space objects.

The Millstone radar, also located in Westford and used to maintain a catalog of deep-space satellites, was the second Lincoln Laboratory radar that used an array processor (installed in 1978-79): a Floating Point Systems AP-120B. The precision tracking performed by the Millstone radar requires substantial coherent integration. Furthermore, the experimental nature of the radar's applications demands a processing system that can accommodate a variety of algorithms and waveforms. Unlike most conventional radar signal processors, the Millstone computer does not use fast Fourier transforms. Instead, the array processor solves problems by the use of unconstrained optimization — a unique real-time application for an array processor.

The TRADEX and MMW radars use AP-120B processors, and the ALTAIR radar uses a Westinghouse PSP machine. Because new waveforms are regularly installed and tested on these radars, all located at KREMS, their processing systems must be flexible and programmable. The price for flexibility is usually a reduction in operating speed, but improvements in semiconductor technology have allowed us to maintain performance levels while increasing the flexibility of each radar system.

Fig. 1 — The Kwajalein Discrimination System (KDS) computer room at the 35-GHz millimeter-wave (MMW) radar, Kiernan Reentry Measurements Site (KREMS), Kwajalein Atoll, Marshall Islands.

by two ST-100s, the system architecture had to minimize the load on the data channels and take advantage of specialized parallel-processing hardware.

System Architecture

The LDS and the KDS are managed by a control computer and organized around a high-speed bulk-storage unit. Each provides the parallel operation and the relatively low demand on communication channels required for radar data analysis. To complete the system design quickly and to minimize the cost, we used commercially available hardware and software whenever possible. After an extended period of competitive procurement, we selected the equipment listed in Table 1. We also designed and built several high-speed interfaces for the various, and not fully compatible, processing elements.

We chose a configuration that is simple in concept: a set of independent distributed-processing elements linked to a large, high-speed, bulk-storage device (Fig. 2). This architecture provides the parallelism necessary for our computational goals, and yet its simplicity lets us coherently integrate processing elements

Table 1. Real-Time Discrimination Test-Bed Processors

Processor	*Function*
MMW Radar Interface	Source of all data for the Kwajalein Discrimination System. The processor includes a programmable signal processor for pulse compression, data windowing, coherent pulse integration, and delivery of radar signature pulses into bulk storage. Operating speed for four channels of information is 20 MB/s. The embedded FFT processor operates at 450 Mflops.
Recording System	A programmable processor subsystem that can store or retrieve four channels of radar data at a maximum rate of 20 MB/s, for a duration of 140/s. KDS and LDS each contain 2.8 GB of online IBIS disk storage. The system uses a Motorola 68020 microprocessor as its CPU. The recording system plays back data, and thus simulates real-time operation, for the Lexington Discrimination System.
Central Memory	Multiported Dataram 2000 memory processor supporting as many as 14 individual I/O ports; internal bandwidth is 160 MB/s.
Control Computer	Gould 9780 dual processor, rated at 3 MWhetstones/s (double-precision mode). It contains 8 MB of main storage; backplane bandwidth is 26 MB/s.
Display Computer	Gould 6780 dual processor, rated at 1 MWhetstone/s (double-precision mode). It includes three Raster Technology graphics processors that display image frames at a rate of 60 Hz, non-interlaced, with a resolution of 1,280 × 1,024 pixels. The main memory includes 8 MB of storage; the backplane bandwidth is 26 MB/s.
Front-End Computer	IBM PC/AT computer with outboard serial I/O processors and an integrated touch panel.
Machine Intelligence Computer	Symbolics 3670 Lisp-based processor, connected via Ethernet to the control and display computers.
Array Processor	KDS has three and LDS has two Star 100 array processors. Each processor operates at a speed of 100 Mflops; the I/O bandwidth of the DMA channel is 50 MB/s.

with diverse computational capacities. Distributed processors based on this architecture may have incompatible hardware and software, yet can still share a common data base.

Inherent in the design were the problems of processor synchronization and interprocessor communication. We solved both problems by employing a message-passing scheme in concert with semaphore synchronization [1]. For example, display tasks on the display computers, feature extraction on the array processors, and discrimination on the mainframes can all run simultaneously. Communication among these processes is accomplished by passing messages [2]. The fixed-length messages contain FROM and TO addresses and an indication of the desired action (e.g., send display data, status and control information, and error-detection

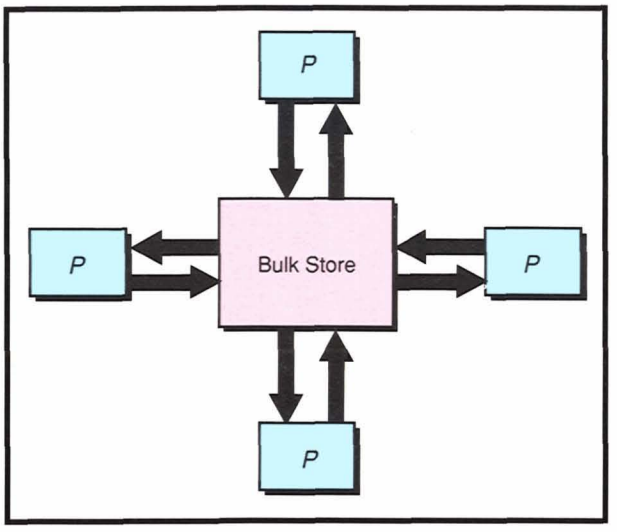


Fig. 2 — The architecture of LDS/KDS comprises a network of distributed-processing elements connected to a large, high-bandwidth, multiported bulk-storage device.

and correction information). Since each process is self-contained and independent, it will run until it is complete or until it requires information from another process. When a process requires information [3], it registers a request and halts its operations until the required information is returned. When a process lodges a request, it increments a counter, or "semaphore," that keeps track of how many requests are made, their order, and the process that made the request.

In this way, the various processors and processes are synchronized [4]. Synchronization ensures, for example, that the display process does not display an image that combines partial information from two frames. Instead, the display process must wait for information from the feature-extraction process, and the feature-extraction process will not return information until its operation is complete.

The network is, in effect, a data-flow machine. The system's operation is synchronized, or "clocked," by the availability of data. Data are transferred by a hardwired interrupt (a signal generated by the hardware that cannot be programmed) or via a complete message that can contain several bytes of information (Fig. 3). An interrupt generally indicates a change of status, such as data requested or data available. Interrupts are also used in conjunction with the more extensive messages.

Suppose, for example, that a display processor requests display information from the control computer. The mainframe will send an interrupt to the display computer when the data become available. The mainframe will also send a formatted message that tells the display computer the location of the data in the bulk-storage device. The display computer will then retrieve the information from the bulk-storage device (where it was placed by the radar interface).

Independent paths to the bulk storage provide a great deal of parallelism. Simply by providing another interface to the bulk storage, additional processors can be added to the existing system. However, growth is limited by the bandwidth (160 MB/s) and by the number of access ports to the memory. In the current configuration, the bulk-storage device has five ports, but could support as many as 14.

LDS/KDS Hardware

A block diagram of LDS/KDS is shown in Fig. 4. The Dataram is the repository for all raw and processed information. For the LDS, data from one or more sensors are stored on the parallel-transfer IBIS disks for high-speed playback into the Dataram bulk memory. In the KDS live mode, a 20-MB/s data stream flows directly from the MMW radar into the Dataram and is recorded simultaneously on the IBIS disks. Aside from this exception, the facilities are operationally identical.

When raw data enter the bulk memory, the control computer commands transfers of information between the memory and the peripheral units attached to the memory's ports. For example, suppose that the control computer sends a message to the array processors that radar pulses are available in the Dataram. The array processors will then read the data from the Dataram and create images of the target. The array processors scan the images for scattering features, convert the images into an array of pixels, and transfer the image back to the bulk memory. Then, by a similar process, the images will make their way to the display computer and

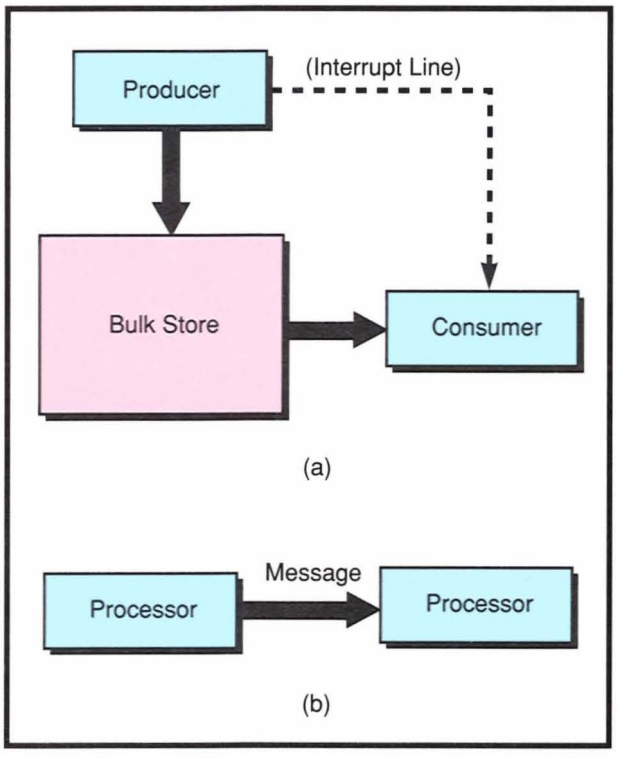


Fig. 3 — Two types of interprocessor communication: (a) hardwired interrupt lines, (b) direct-communication channels between processors.

a color monitor. Extracted feature information simultaneously becomes available to all CPUs. At many times during this operation, discrimination algorithms will interrogate the data streams and report results to the control and machine-intelligence computers.

Although the system's component processors were standard commercial products, no interfaces were available that would allow the processors to work together. Therefore, following our specifications, Star Technologies built an interface between the high-speed direct memory access (DMA) port on the ST-100 and the Dataram memory. In this interface, all devices connected to the bulk memory emulate the Star DMA port, and thus implement the Dataram interface scheme. Also under contract, Star Technologies designed and built the interface between the Dataram memory and the control and display computers. The link between the Star array processor and the Dataram operates at a sustained transfer rate in excess of 30 MB/s (160 MB/s for bursts of 32 kB).

The Dataram's 160-MB/s data-transfer rate is based on the combination of an interleaved memory access, a 16-byte–wide data path, and a high-speed 32-kB buffer located on each interface card. Data are transferred 16 bytes at a time, with a cycle time of 100 ns. On the side of the interface opposite the Dataram, a high-speed 32-kB buffer allows an attached processor — control computer, array processor, or display computer — to load as much as 32 kB at its own, lower data-transfer rate. The buffer is then emptied into the Dataram at 160 MB/s, a far faster data-transfer rate than the attached processors could provide.

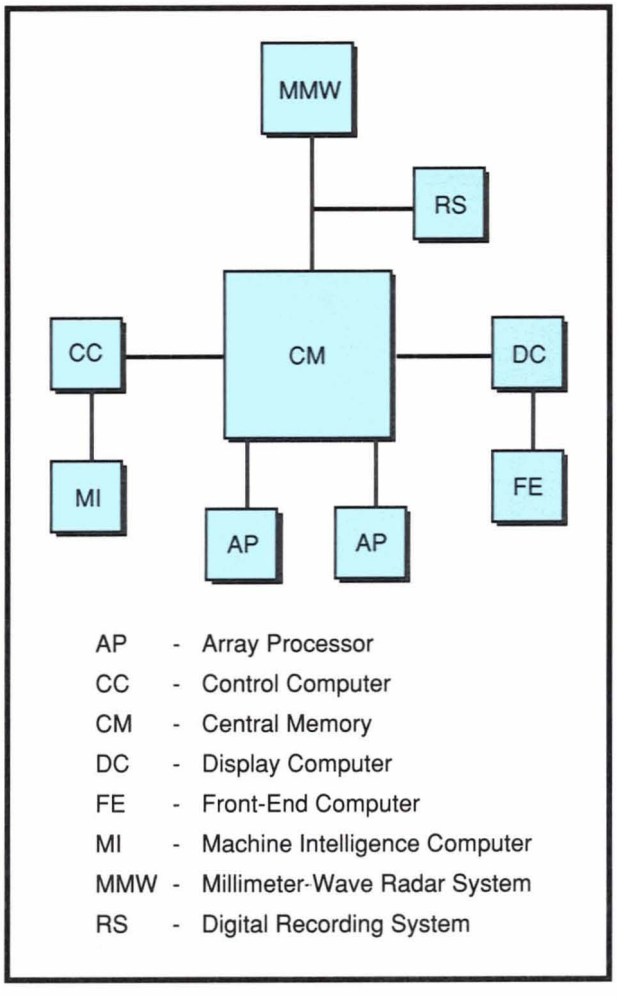


Fig. 4 — Test-bed distributed processors connected to the central-memory unit.

The Dataram's interleaved memory reduces the memory-access time by pipelining memory accesses; while one memory segment's contents are being fetched, the address and enable signals for the next memory segment are being output by the controller. Latches between the memory and the data path prevent contention for the bus. This technique significantly reduces the cycle time for the memory. The access time is reduced to nearly that of the high-speed latch, not to the access time of the comparatively slow memory.

Additional speed is obtained from the interface by operating it in a DMA mode. In this approach, lists that contain starting addresses, number of bytes to be transferred, and status and control information are used to transfer data between the buffer and the Dataram. The advantage of DMA is that the interface controller can start a counter that sequentially accesses the memory locations being addressed without any decoding of instructions or making of decisions. The difference in transfer methods can be described in the following way. Without the use of DMA, the transfer process is

Fetch instruction
Decode instruction
Fetch data
Transfer data
Fetch instruction
Decode instruction
Fetch data
Transfer data.

By using DMA, the transfer process becomes

Fetch instruction
Decode instruction
Transfer data
Transfer data
Transfer data
Transfer data
Transfer data
Transfer data.

The savings in machine cycles are evident and result in a much faster data-transfer rate.

Real-Time Program

Extensive software development was needed for the control and display subsystems. The control and display computers (Gould 9780 and 6780, respectively) each contain two CPUs; several real-time tasks execute simultaneously and in parallel on the four CPUs. Basic process changes are triggered by data-available interrupts, message-arrival notices [5], and external events.

The Gould computers use the MPX-32 real-time operating system. This operating system uses a software priority structure to provide selective process control, which is necessary to provide real-time operation. MPX-32 is a commercially available real-time operating system; however, we added device drivers (software that provides an interface to a specific piece of equipment), interrupt handlers, and a data-flow manager that maintains real-time system operation.

The I/O drivers were integrated into a scheme that uses software interrupts and message passing. The drivers use messages to deliver data packets to tasks that summon data. A message can be sent to a task on any processor without using dedicated I/O links. For example, a message can start on the front-end processor (IBM PC/AT), pass through the display computer, and be delivered to a task in the control computer. The I/O driver for the front-end computer is used to obtain user requests, but no I/O driver is required for communication between the front-end and the display computer, or between the display computer and the control computer.

The MMW radar's 20-MB/s data-transfer rate overruns standard I/O interfaces. In LDS/KDS we use dedicated programmable controllers that accept I/O commands with embedded interrupt-enable flags. The flags tell all processors attached to the bulk memory that data have been deposited in the memory.

Operator Interface

Four major processes are ongoing during real-time operations: data acquisition and recording,

signal processing in the array processors, execution of discrimination algorithms, and display of results on the console screens (see Fig. 5). A touch screen on the console provides operator control.

A touch on the screen generates an interrupt and a message to the desired device. During execution of a mission, for example, the operator can touch the screen and view the execution of an algorithm. The touch screen can also be used to override a decision made by the system. We chose a touch screen for the real-time operator interface because keyboard entry is too slow and error-prone for reliable operation at KREMS, where missions typically last 30 *seconds*.

Discrimination Algorithms

The ultimate goal of the LDS and the KDS is the support and execution of algorithms that interpret the data streams from both active and passive sensors — as the data are obtained, not in a postmission batch mode.

During a mission, radars measure a target's position and scattering properties by the use of a variety of interrogating waveforms. Optical sensors measure the target's radiance in selected passbands in order to estimate its thermal properties. All this information must be processed in real time. The discrimination tasks must not lag behind the incoming data stream. The algorithms must be causal, which seems obvious; but the requirement for causality contrasts strongly with the usual methods of postmission data analysis, which can access the entire data set on an object without regard to temporal order.

The most computationally intensive task currently performed by the LDS and the KDS is the generation of a range-Doppler image sequence of a target. Range-Doppler imaging uses lines detected in the frequency spectra, calculated for the range cells that span a target along the radar's line of sight, to reveal the physical and dynamic characteristics of the target. The procedure is similar to synthetic aperture radar (SAR), except that the target itself provides the motion relative to the radar. Range-Doppler imaging is therefore akin to inverse SAR.

Summary

The LDS and KDS are fully operational and provide the means to test system and discrimi-

Fig. 5 — The KDS control console includes three high-resolution color screens for real-time information display, a touch screen for operator control, and devices that move on-screen cursors.

nation concepts within the constraints of the real-time environment. The flexible architecture employed by these systems provides a variety of operational modes and growth paths. This variety supports practical investigations into methods of high-speed control, signal processing, and target classification. Whether we are playing back data at LDS, or taking live missions in the field at KDS, the experience gained on the test bed encompasses many of the computational problems that will be encountered in the design and operation of a strategic defense system.

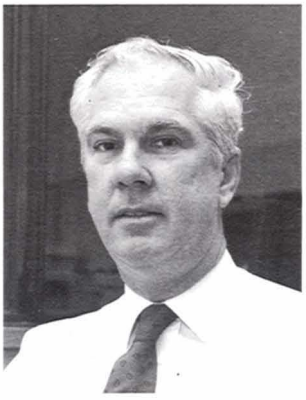
Acknowledgments

The original design team for the LDS and KDS included the authors and three other Lincoln Laboratory staff members: S.C. Crocker, T.L. Sangiolo, and G.W. Titi. Their contributions have endured and ensured the successful operation of the test bed. Software engineers J.F. Baldassini and T.C. Bressoud later joined the team and implemented much of the code used in the real-time program and the data-base manager. Throughout the course of the project, from equipment layout to hardware installation, R.R. Roberge has managed and maintained the computer facility. We also acknowledge the guidance and direction of George Drake of the U.S. Army Strategic Defense Command.

References

1. E.W. Dijkstra, "The Structure of 'THE'-Multiprogramming System," *Commun. ACM* **11,** 341 (1968).
2. R. Atkinson and C. Hewitt, "Synchronization in Actor Systems," *Fourth ACM Symp. on Principles of Programming Languages, Los Angeles, 17–19 Jan. 1977,* p. 267.
3. F. Baskett, J.H. Howard, and J.T. Mantague, "Task Communication in DEMOS," *Operating Systems Rev.* **11,** 23 (1977).
4. P. Brinch Hansen, "The Nucleus of a Multiprogramming System," *Commun. ACM* **13,** 238 (1970).
5. R.B. Kieburtz, "A Hierarchical Multicomputer for Problem-Solving by Decomposition," *Proc. of the First Int. Conf. on Distributed Computing Systems, Huntsville, AL, 1–5 Oct. 1979,* p. 63.

STEPHEN B. BOWLING is Leader of the Field Systems Group. His work at Lincoln Laboratory has concentrated on radar systems and discrimination research. In 1984, he and three other Lincoln Laboratory co-authors received the Best Paper Award from the *Journal of Defense Research* for their work on radar-imaging techniques. Steve received a B.S. degree in physics from the University of North Carolina at Chapel Hill in 1970 and M.S. and Ph.D. degrees in space physics and astronomy from Rice University in 1973 and 1975, respectively. While in school, Steve was elected to Phi Beta Kappa and was chosen to be a National Merit Scholar and a Rice Endowed Scholar. He is currently a member of the solar-magnetospheric section of the American Geophysical Union.



FREDERICK W. VOTE is a staff member in the Field Systems Group. He joined Lincoln Laboratory after he received a B.A. in mathematics and history from Boston College in 1971. He participated in the original design effort for CP67, a virtual memory operating system for IBM 360 systems. He had a brief sojourn at Los Alamos National Laboratory, where he helped to develop a Cray operating system and a control system for a proton storage ring. He then returned to Lincoln Laboratory, where he has worked on the design of a system of multiple parallel distributed processors.

ROBERT A. FORD is a staff member in the Field Systems Group. His current work focuses on real-time signal processing for wideband radar systems. Bob received B.S.E.E. and M.S.E.E. degrees from the Columbia University School of Engineering. Before joining Lincoln Laboratory, he worked for Ford Aerospace in Newport Beach, Calif. Bob is a member of Eta Kappa Nu. He enjoys sailing and running 10-k road races.