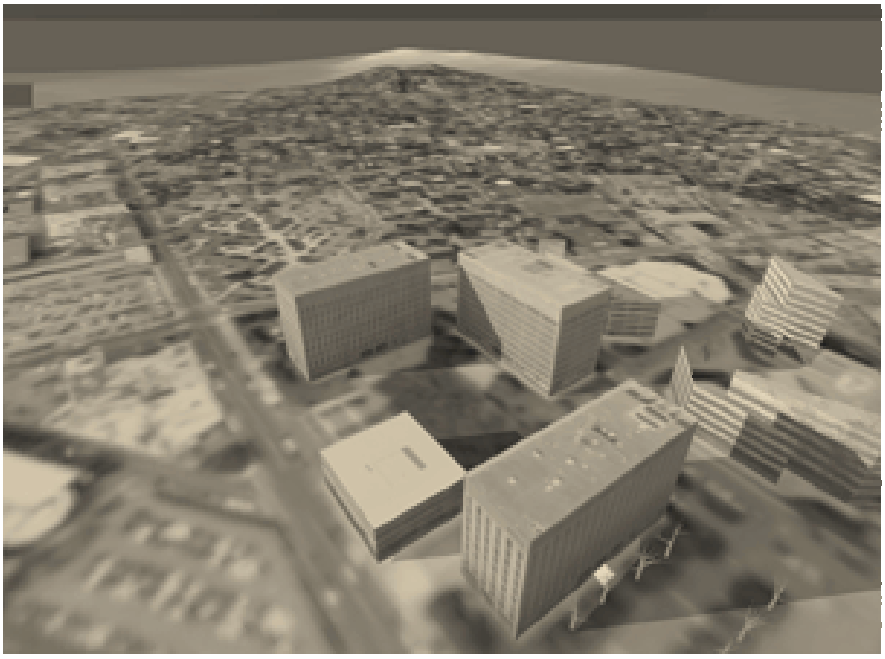
In GPS World, April 2000, pp. 20-30.

Eyes of Argus

Georeferenced Imagery in Urban Environments

Like its namesake, the 100-eyed creature of Greek mythology, Argus observes its surroundings from many angles. The device, developed by a team at the MIT Computer Graphics Group, combines navigation sensors, a high-resolution digital camera, and various algorithms to capture georeferenced imagery of urban environments.



A simple model of MIT's Technology Square, extracted from ground-level imagery. The model is overlaid onto georeferenced terrain and draped with a registered aerial image.

Michael Bosse
Douglas De Couto
Seth Teller

Computer Graphics Group,
Massachusetts Institute of Technology

All images courtesy of the authors

In cave drawings depicting the hunt, early cosmogonies in stone, and paper charts of mountain passes and maritime routes, people have long sought to capture useful representations of their environment in maps.

Whether constructed from memory or meticulously compiled from observation, maps enable myriad cultural functions, including delineating property boundaries, planning the movement of people and goods, taxation and representation, as well as zoning and developing communities.

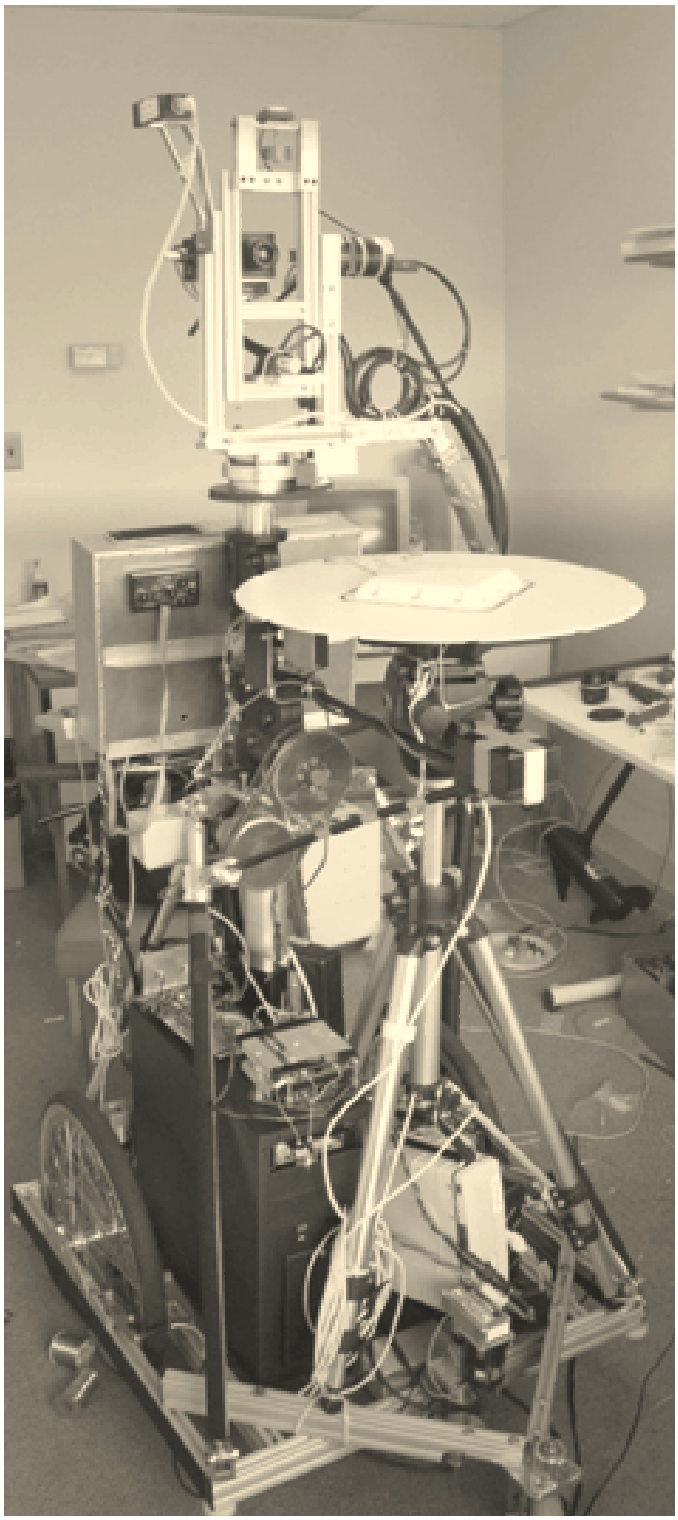
Modern maps, in electronic form and supplemented with a variety of associated data and applications programs, form the basis of geographic information systems (GIS). As the sheer magnitude of geospatial data increases, techniques for reducing the cost and increasing the speed and efficiency of map production have gained importance.

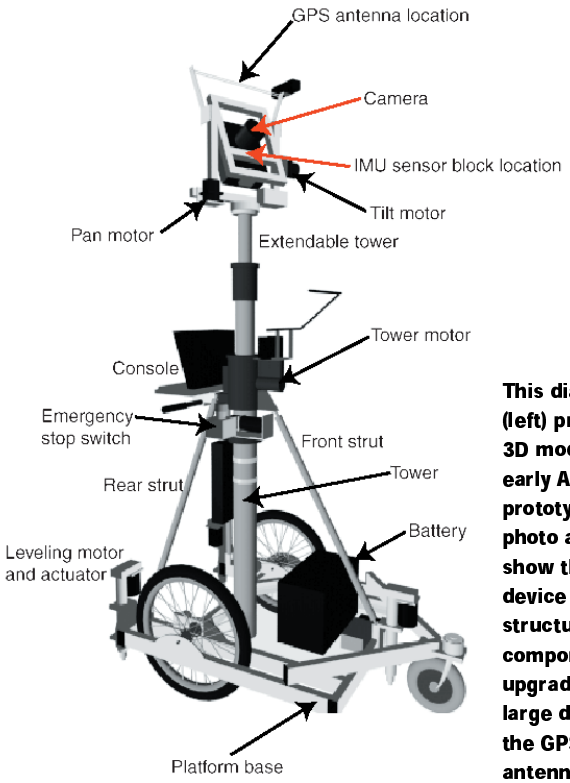
Modern navigation techniques, including GPS data, are critical in this regard: GPS provides straightforward tagging of observations, for example photographs or environmental features, with location information that places each observation in context on the Earth's surface.

FROM MAPS TO MODELS

Traditional paper maps include information about road networks, land usage, and points of interest, often draped over a topographical surface representing height through elevation markers or contour lines. Increasingly, such maps can be viewed online, enabling the user to change scale or center of interest dynamically. When the semantic content of the map is available for further processing — for example by programs that suggest travel routes, estimate fuel usage, or generate synthetic vistas from points on or above the map using computer graphics techniques — the map data are enhanced in a fundamental sense, essentially becoming a model of the environment.

Today, modern maps mimic reality ever more closely, enabling a host of new applications. Using a process known as





This diagram (left) presents a 3D model of an early Argus prototype. The photo at far left show the Argus device after structural and component upgrades. The large disk is the GPS antenna.

geometric modeling —acquiring a representation of an object in a form useful for computer simulation —we can create three-dimensional (3D) models of the urban environment for use in visualization, simulation, and computer-aided design (CAD).

Indeed, 3D CAD models are the starting point for many realistic simulations. Architects and urban planners, for instance, use 3D models to visualize the relationships between existing and planned buildings. Emergency and military personnel use 3D models for training and rehearsal in urban theaters. These models even form part of popular culture in advertising, movies, and electronic games.

Capturing Reality. How are 3D models captured in electronic form? Much as word processing software enables typists to transcribe paper documents into electronic files, geometric modeling software enables skilled human operators to input 3D model data.

This method of model entry is manually intensive, usually accounting for a significant fraction of the time and expense involved in developing a given visualization or simulation capability. Modeling an extended urban area poses particular challenges of scale, demanding the capture and faithful representation not only of coarse structural information, but also of fine-scale geometric details, such as windows and variations in appearance related to underlying surface color and material properties.

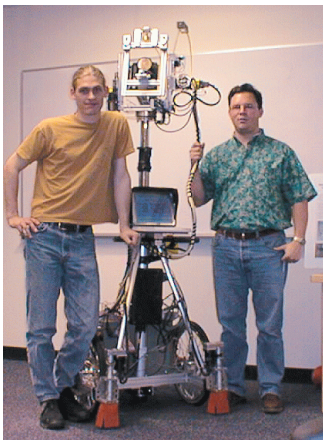
THE HUMAN ELEMENT

To exploit the rich information inherent in terrestrial (near-ground) photographs, researchers in photogrammetry and computer vision have developed many techniques for extracting geometric and appearance information directly from images.

These efforts typically rely on a "human in the loop" —manual intervention by a skilled operator —for one or more tasks, including establishing a working coordinate system, initializing or constraining estimates of camera placement, indicating the structures to be modeled, or distinguishing buildings from their attendant clutter. Because a human operator is indispensable to such systems, and the underlying computer performance can inexorably increase, the human eventually becomes the fundamental factor limiting system performance.

Moving Automation. To overcome this limitation —and make system performance solely a function of the underlying sensor, storage, and processing technology —we at the Massachusetts Institute of Technology (MIT) Computer Graphics Group have pursued the City Project, aiming to develop efficient techniques for fully automated capture of high-fidelity, textured geometric models of urban environments. Streamlining the model production process should have considerable potential payoff for a number of applications based on 3D CAD models.

Eliminating the human in the loop, though, is a significant challenge from both an engineering and research standpoint. Although a number of automated algorithms have been proposed for various subtasks, their operating assumptions and regimes of effectiveness vary so widely



MIT Computer Graphics Lab researchers Michael Bosse (left) and Douglas De Couto with Argus.

that no straightforward composition of existing techniques yields a functioning automated system for model recovery.

For example, existing algorithms often assume only a small number of images to be processed, and that most pairs of images are related, thus expending computational resources in proportion to the square of the number of input images, or worse. This is clearly an untenable strategy for datasets consisting of thousands or millions of input images, many of which observe little or nothing in common.

We have therefore pursued an alternative approach with two major components: a sensor that observes the urban environment to produce georeferenced imagery, and a suite of algorithms that refine and process the sensor observations to extract a model of the observed structures, appropriately situated in a global coordinate system.

In developing the sensor, our challenge lay in capturing images georeferenced with sufficient accuracy to enable fully automatic image registration and subsequent extraction not only of coarse structural elements but also of fine detail.

NECESSARY TRADEOFFS

Our goal is to capture high-resolution digital representations of built structures over areas of square kilometers, to a feature size of a few centimeters. Presently, automated computer vision techniques for completely general scenes are widely believed to be decades away from realization. To achieve full automation in the near term, we made four important tradeoffs.

Urban Focus. First, we focused our efforts on recovering models of urban scenes. On one hand, this is a significant restriction; it means we cannot handle marinas, deserts, or jungles. On the other hand, urban scenes are quite common, forming the surroundings of much of the world's population. Our strategy is to get an end-to-end reconstruction system working for simple urban scenes, then gradually tackle increasingly complex environments as our sensor and related algorithms mature.

Novel Sensor. Second, we combined a digital camera with GPS and other navigation instrumentation to produce a new kind of sensor we call a *pose camera*, which stamps each acquired image with the approximate location, orientation, and time of its acquisition. This georeferencing information enables us to insert imagery into a spatial index as it is acquired. Thus, we can determine which observations are likely to be related without performing expensive pairwise image analysis.

Myriad Images. Our third tradeoff was to acquire many thousands of urban images, rather than just a few dozen as with earlier, manually operated modeling systems. Although buildings are typically static, we observe them under changing lighting and environmental conditions, and among a clutter of people, vehicles, and foliage. Only by correlating many observations can we overcome these variations to reliably and automatically register the image observations and recover models of building structure and appearance.

Computational Command. Finally, whereas existing manual modeling systems typically require modest, intermittent computational resources, we purposefully crafted our techniques to take full advantage of the significant storage and computational resources embodied by modern computer systems. Our system currently expends hours of sustained processing to recover even simple environments, but we anticipate that as computers become faster and more capacious, our techniques will eventually outperform even skilled human operators.

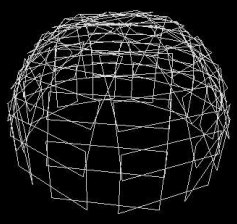
ARGUS IN THE CITY

A major component of the City Project is the development of our pose camera, a sensor with a unique physical design. We call the sensor "Argus," after the 100-eyed creature of Greek myth, for its ability to observe the environment from a multitude of positions and directions.

Specifically tailored for use in crowded urban environments where mobility is a concern, Argus is a heavy-duty, three-wheeled cart propelled by an operator on foot. It is equipped with a high-resolution, computer-controlled digital camera mounted on an electrically actuated pan/tilt head that rotates about the camera's optical center.

CREATING A 3D WORLD

Reconstructing 3D models of complex urban environments begins with collecting high-resolution, georeferenced images and culminates in a 3D CAD model of the environment, including buildings, structures, and texture.



Acquiring Pose Imagery: the tiling pattern Argus uses to create each hemispherical pose image of its surroundings. Overlapping each image with its neighbors enables a correlation-based alignment process to register the images to subpixel accuracy.

A detailed view of the tiling pattern shows one set of images as the camera rotates about a fixed optical center.

The raw images comprise one complete node. The above node, made up of 47 images at 1,000 × 1,500 × 24 bits each, was acquired near ground level on the MIT campus.

The entire assembly is placed atop a telescoping stalk that raises to a height of 3 meters, enabling Argus to see over most people, shrubbery, and vehicles —each of which are considered obstacles in our application.

Argus acquires high-resolution, georeferenced color images called pose images, which serve as input to our CAD model reconstruction system. *Pose* refers to a rigid object's position and attitude, that is, the six degrees of freedom that describe the object's location and orientation in a 3D coordinate system. A pose image is simply the combination of an image and the camera's pose when the image was taken, along with a date and (GPS-based) timestamp.

Argus uses a variety of navigation sensors to estimate camera pose, including a differential GPS receiver (the crucial link to a global coordinate system), an inertial measurement unit (IMU), and wheel encoders that report motion with respect to the ground (see Figure 1). An onboard processor records image and navigation data, controls most aspects of the cart's operation, and provides visual feedback to the operator.

When Argus returns to the laboratory, we reconnect it to our network, uploading and registering the collected pose-imagery and extracting model geometry on our lab's computation cluster. Our ultimate goal is an online system, in which Argus moves around an area while onboard processing performs image registration and geometry extraction, generating 3D CAD models of the captured area on the fly.

A NEW SET OF EYES

Traditional cameras supply either a wide field of view (such as with a fisheye lens), or high effective resolution (with a telephoto lens), but not both. Argus uses a pan/tilt head to achieve both high resolution and a large field of view (FOV) simultaneously. This eliminates the classical ambiguity between sensing translational and rotational motion inherent in limited-FOV imagery, enabling some of our computer vision algorithms to succeed where their counterparts on standard imagery would fail.

These advantages come at the cost of increased mechanical complexity and increased acquisition time as the camera "tiles" the sphere with many distinct images. However, our pan/tilt mount is electrically actuated, allowing the camera to be moved through each tiling much faster than would be possible with a human operator.

Our approach differs from methods based on aerial and satellite photography, which benefit from improved GPS visibility but suffer in other ways. These techniques provide monocular views, often of each area only once or a few times, and typically from a great distance and at high obliquity (if viewed from above) or with significant occlusion (if viewed from the side).

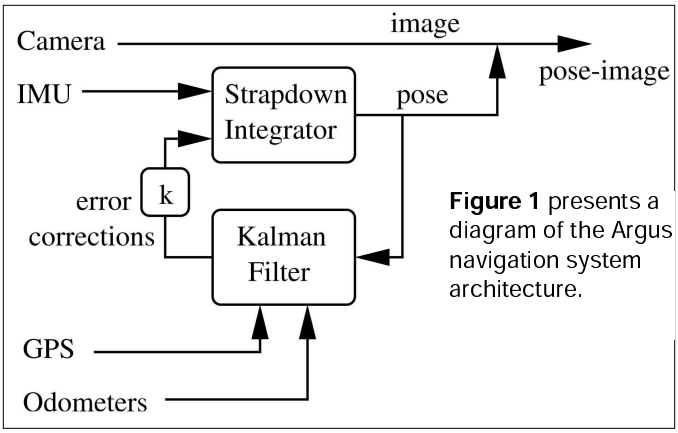


Figure 1 presents a diagram of the Argus navigation system architecture.

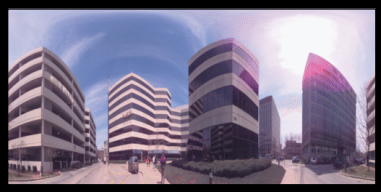
In contrast, Argus observes the world at high resolution from ground level, in and among the scene's urban canyons. Most building surfaces are likely to appear in several images, from several vantage points. When Argus observes a building facade from many views, our algorithms can overcome obstructions from people, trees, and other objects, essentially "erasing" the blocking elements.

As far as we know, Argus and the City Project are unique. Other projects acquire a combination of images and position data, but none acquire the same quantity of data, nor do others acquire images or pose data at a granularity comparable to ours.

For example,vehicle-mounted systems have been developed to record time-tagged video from cameras fixed to the vehicle along with continuous GPS data. The video frames are later correlated to the GPS solution, and the video stream can be roughly spatially indexed for inspection.

Argus is different; it images the acquisition area from many discrete viewpoints. And the dynamics of Argus, which is deployed at a walking pace, but can change direction quite quickly, present a different challenge to the integrated navigation system than would a vehicle.

The next step produces a seamless, nearly spherical node by aligning each image. This is performed by associating a quaternion with each image and then performing numerical optimization to best correlate overlapping regions of neighboring images.



To ease visualization, a spherical coordinate representation of these mosaics is produced, by coalescing each one into a rectangular pixel array whose horizontal coordinate corresponds to phi (azimuth, from 0 to 360 degrees), and whose vertical coordinate corresponds to theta (altitude, from -90 to 90 degrees).

Pose images are acquired so as to observe every building from multiple viewpoints. Above, 81 node positions are shown, along with the extracted 3D model and a map of a part of the MIT campus.

THE CITY'S CHALLENGES

To determine design targets for Argus, we considered our accuracy and operational requirements. Our goal is to resolve and localize centimeter-scale building features from a standoff of about 10 meters. Optics with angular resolution of roughly 1 milliradian per pixel (such as a 1-radian FOV projected onto a 1,000-pixel raster) supply the necessary resolution.

Localizing an observed feature to 1 centimeter requires that the camera position —latitude, longitude, and altitude —be estimated to a centimeter, and the camera attitude —heading, pitch, and roll —be estimated to 1 milliradian, or about a twentieth of a degree. (1 radian, about 57.3 degrees, is the angle swept by a circular arc whose length is equal to the radius of the circle.)

Filtering Out the Buzz. In a rural, open field, our GPS receiver reliably estimates its position to a fraction of a meter. However, Argus operates in a busy urban environment, a challenging venue for GPS receivers. The MIT campus in Cambridge, and nearby Boston, are electronically noisy environments due to high-power radio and television transmissions emanating from Boston's Prudential building, as well as the area's high volume of cellular calls.

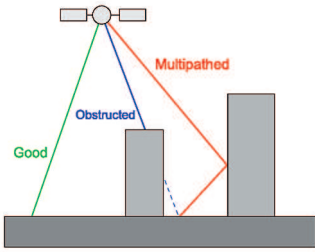


Figure 2 shows GPS dropout and multipathing caused by urban canyons.

Argus, using GPS at walking speed in this noisy environment, is also subject to dropout (caused by satellite obscuration) and multipath (caused by signals arriving along different paths from the same satellite, see Figure 2), producing position estimates good only to a few tens of meters.

In spite of these obstacles, the operating modes of Argus also yield some advantages. Because Argus rolls continuously along the ground, the wheel encoders function as odometers and sense when it is stopped. When Argus is moving, its navigation system combines the data stream from the wheel encoders and IMU for dead reckoning over short timescales. By fusing complementary GPS, IMU, and odometry data through Kalman filtering, Argus produces position and heading estimates good to about 1 meter and 1 degree, respectively. Finally, offline image-based registration techniques refine the pose estimates to within a fraction of a meter and a fraction of a degree.

ARGUS TAKES SHAPE

Because our aim was developing new end-to-end computer vision techniques, rather than developing new support technologies, we used commercial-off-the-shelf (COTS) parts wherever possible. This kept our costs and development time relatively low.

We also designed Argus to be propelled by a person, not a vehicle, so its physical configuration was constrained by typical urban (and suburban) pedestrian spaces: sidewalks, ramps, gates, paths, doors, and other routes that people take. And of course, it was necessary that the sensor fit into an elevator and be maneuverable in and out of our lab.

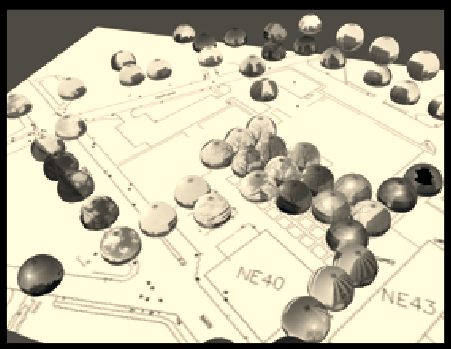
In its current form, the Argus system comprises four main parts: navigation, imaging, mechanical, and the control software that links it all together. We began acquiring components in 1996, with the navigation system consisting of the GPS receiver, IMU, and wheel encoders, along with software to combine their data streams using a Kalman filter. Each of the navigation sensors is powerful but exhibits significant individual limitations.

Integrating GPS. Our first GPS receiver was a small, credit card-sized, user-grade unit that fit inside our control computer. This receiver had a small antenna that could be mounted rigidly with the camera and IMU.

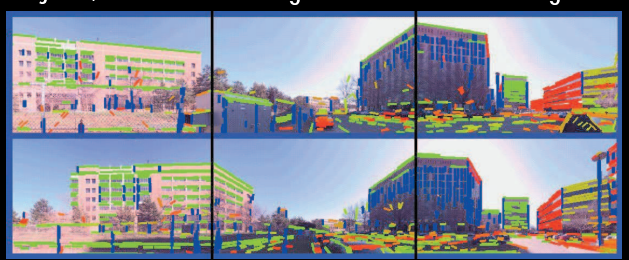
As we tested our initial configuration, we found that urban canyons degraded GPS accuracy significantly. This prompted our decision to incorporate a survey-grade, L1/L2 full cycle carrier, L1, C/A-code, nine-channel GPS receiver, which had a larger antenna and a 19-inch ground plane. The new receiver had to be mounted on the base of the Argus platform, and the new antenna in front of the camera tower; thus, the antenna no longer moved with the camera.

Another GPS receiver was deployed as a base station atop our office building, connected to our lab's network. A computer on the roof receives and stores raw GPS data for later download and differential GPS solutions. The increase in signal and satellite tracking quality was well worth the added complexity of compensating for the changed position of the camera and GPS receiver on Argus, and handling an additional receiver on the roof.

Below is a similar view of the growing node network, situated on the MIT campus. Each hemisphere in this dataset is a 50-megapixel mosaic, georeferenced to a global coordinate system.



After spherical mosaicing and absolute pose determination are complete, the dataset of pose imagery is ready for further processing. Our model extraction algorithms treat pose images as first-class data objects, much as earlier algorithms treated raw images. Our overall goal is the reconstruction of 3D models of urban scenes directly from large numbers of pose images. We have developed a variety of algorithms for model and texture extraction, each with different properties. Some are sparse, or feature-based; they infer structure from repeated observation of points or lines in the world. Other algorithms are dense, or texture-based; they infer structure such as building facades by identifying pixels that appear to lie on the same surface. The above image is from the work of Satyan Coorg, who developed a robust technique for identifying vertical facades. His algorithm determines facade orientations by histogramming edges under the assumption that the edges arise from vertical surfaces.



MEASURING ON THE MOVE

GPS suffers dropouts in urban environments because of satellite signal blockages and multipath solution degradation. Moreover, GPS is a point position sensor and gives no direct information about the direction in which the camera is pointing.

IMU. To overcome these limitations, we incorporated an inertial measurement unit into Argus. The IMU provides linear acceleration and angular velocity measurements for each of three mutually orthogonal axes. The IMU's output can be integrated over short timescales to provide camera position, attitude, and velocity information. However, the IMU cannot be integrated over long timescales while retaining accuracy.

Most gyrocompasses and inertial navigation systems are large, bulky, power-hungry, and expensive, and thus would be impractical for our project. They are designed for aviation use where high linear velocities allow the gyro outputs to be calibrated, or for automobile applications, where the gyro's orientation is partially constrained (for example, always parallel to the ground).

We eventually located a dual-gyro, military-grade IMU that was small and light enough to attach rigidly to the camera and worked well at walking speed. We worked with the IMU's supplier to develop an ISA (Industry Standard Architecture) expansion board that allowed us to interface with the IMU. This board also monitors the GPS receiver's one-pulse-per-second output, which is used to synchronize all Argus data streams and to timestamp all acquired pose-imagery.

Rolling Sensors. Argus's two wheel encoders provide another source of translational data. They are connected to the cart's two rear wheels, measuring movement parallel to the local ground plane and total distance traveled from a given starting point. Using the differences of measurements over small time periods, we can treat the wheel encoders as a speedometer.

In combination, the IMU and odometry sensors provide a dead-reckoning system that can overcome GPS data gaps when not enough satellites are in view. In addition, the Kalman filter automatically accounts for slowly increasing position errors, recovering properly when the satellites later become visible.

Ultimately, integrating the GPS, IMU, and wheel encoder data with a Kalman filter allowed us to meet our system requirements. By fusing information from all of the sensors, we were able to overcome the inadequacies of each individual component and achieve a whole greater than the sum of the parts (see Figure 3).

COMPUTING CONTROL

The final control component of Argus is the software system, running on a 450-MHz Pentium II dual processor under Windows NT. Most of the software is written in C++, with some interface components in Visual Basic and some visualization components in OpenGL, a cross-platform standard for 3D graphics.

The navigation software was originally written to run in HyperKernel, a real-time extension and programming interface for Windows NT. HyperKernel provides a simple model for device input/output (I/O) under Windows NT, which we used for communicating with the IMU's ISA interface board. However, after encountering difficulties with programs running in HyperKernel and because of its need for special setup within Windows NT, we gradually moved the navigation calculations out of HyperKernel, leaving in it only the bare minimum of IMU I/O code.

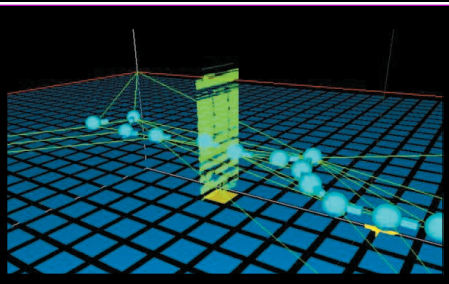
The navigation software considers each GPS satellite's messages individually. This allows the software to calculate fine-grained error measurements for each satellite, an improvement over relatively simple indications of error such as horizontal dilution of precision (HDOP). The IMU control module transmits data by way of shared memory to the rest of the navigation software, running as a normal Windows NT program. Eventually we wrote our own Windows NT device driver for the IMU, allowing us to run the navigation system on both processors.

TACKLING OBSTACLES

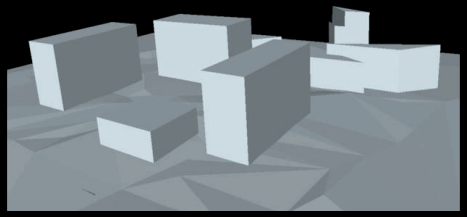
We encountered several other challenges in developing Argus.For example, " smart motors"drove the cart's mechanical systems for leveling and camera pointing. These small stepper motors contain integrated microprocessors and provide their own control language, a version of BASIC.

We used the control language to avoid handling many of the details of motor control in the Argus software. However, the motor control language had its own set of idiosyncrasies, and significant effort was required to achieve reliable motor control.

GPS Bugs. One of our toughest challenges was obtaining good GPS data. We originally had severe multipath problems with our user-grade GPS receiver. However, even after switching to



The histogramming technique yields the orientations of dominant facades in the scene. Finding their positions requires sweeping a virtual plane through the relevant spatial region of the dataset, as shown here.



The histogramming technique yields facade geometries, but not texture, which is also known as reflectance or albedo.

To create details closer to real urban environments, a diffuse texture is estimated for each facade using weighted median statistics. The algorithm combines multiple, rectified observations of each facade, removing most clutter such as trees, signs, and cars. Below is a synthetically generated image of the resulting 3D model, overlaid on a georeferenced aerial image.

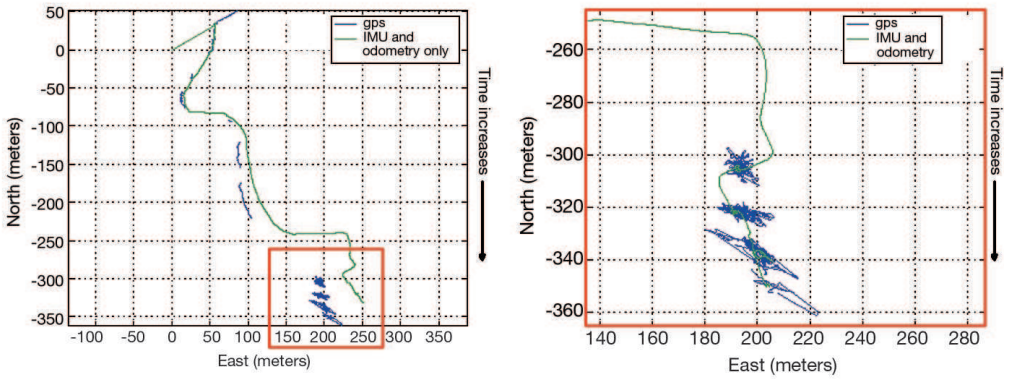


Figure 3 shows, in the graph at left, the raw GPS position estimate (intermittent) overlaid with the navigation solution using only IMU and odometry data (smooth). Time proceeds roughly downward along the path. The GPS solution provides accurate position but is intermittent; the IMU/encoder solution is continuous but drifts increasingly with time. The graph at right shows a detailed view of the full navigation solution (smooth), again overlaid on the GPS solution (intermittent). When the IMU/odometry and GPS data streams are combined, the resulting navigation solution is both continuous and drift-free.

the high-end GPS receiver, we still observed kinematic errors on the order of tens of meters. Upgrading the receiver firmware eliminated a bug, making our solution much more accurate.

Other parts of the GPS data had to be handled carefully to avoid introducing errors. For example, GPS clock bias discontinuities introduce errors that are hard to diagnose, as they occur intermittently. On days with good reception and coverage, the clock bias term exhibits few discontinuities, producing good position solutions even when the bias term is mishandled.

Antenna Arrangement. Placing the GPS antenna also tested our ingenuity. There is an inherent conflict between the camera's need for an unobstructed hemispherical field of view around Argus, and the GPS antenna's need for an unobstructed view of the satellites.

To simplify the navigation filter, we wanted the GPS antenna in a fixed position on Argus. We couldn't place the antenna near the bottom of Argus, where the metal superstructure, motors, and electronic equipment would block or interfere with the GPS signal. But we could not fix the antenna on the top of Argus, as it would be in the way of the camera.

Using the small GPS antenna, we settled on a solution in which the antenna was mounted with a gimbal on top of the camera bracket. The antenna swivels and moves with the camera, while the gimbal keeps it level and out of the way of the camera. Once we switched to using the second receiver, we mounted its large antenna and ground plane on the front of Argus. Although GPS signals from satellites behind Argus are corrupted by the platform's frame, the navigation system is not too badly affected; it tracks each GPS satellite, discarding signals that arrive from a region directly behind Argus.

Finding the Source. Subtle aspects of the system, such as handling the bias term, must be implemented and verified carefully. It is tempting to attribute poor navigation solutions to multipath. However, especially in the face of harsh environments, it is necessary to systematically test and verify each component of the system, from data acquisition through navigation software.

Poor GPS data can result from a wide variety of errors, or even a combination of error sources. In our case, we faced a firmware bug, clock bias discontinuities, poor satellite geometries, multipath interference, and interference from the platform itself. As we gained experience, we learned to tease apart the error sources. By systematically addressing each in isolation, we eliminated the errors intrinsic to the platform, eventually obtaining a useful GPS solution.

Sizing Up the Software. Our final challenge was developing the software to monitor and control the array of devices on Argus: the GPS, IMU, wheel encoders, motors, camera, display, and storage subsystem. The software system also includes an extended Kalman filter, which calculates an online navigation solution and provides a variety of statistics about the solution's quality.

The most visible software component is the Argus display, which presents the state of all subsystems to the operator. For example, there is a graphical display of the pan-tilt head motor activity and the image acquisition and processing subsystem, which shows each pose image as it is acquired. This display allows the operator to verify that Argus is successfully acquiring pictures. Alternatively, it may reveal that the camera requires adjustment.

There is also a 3D display of the navigation solution, along with the locations of the GPS satellites in the sky and their corresponding individual signal qualities. A poor signal may prompt a pause, a turn, or a move to another location. The control software also avoids aiming the camera directly at the sun, which would saturate the camera's charge-coupled device (CCD) and corrupt subsequent image processing.

ARGUS ON THE STREETS

A typical outing with Argus is relatively straightforward. In the lab, the student operator boots the control computer and powers up the various subsystems. She runs through a pre-outing checklist, verifying that the batteries are charged, that the mechanical subsystem is properly aligned, and that the navigation sensors are operational.

The student then configures the control software, with a glance at the system logs on the computer screen. Finally, she ensures that the GPS base station is activated and logging data. The operator then wheels Argus down the hall, through a door into the second-floor lobby, and into the elevator for the trip to the first floor. She then takes Argus outside using the wheelchair ramp. Often an assistant, carrying a notebook and other gear (such as a cell phone for communicating with the lab), accompanies the operator, because maneuvering Argus can take two hands.

Collecting Pictures. Once outside, the operator starts the navigation system, spinning up the IMU. From a starting location, ideally one with good GPS coverage, she then walks a small circular path, during which Argus initializes its position and heading estimates and other model parameters.

Finally, about 10 minutes after power-up, the operator starts acquiring pose images. In a process of " punctuated acquisition,"she wheels Argus to a sequence of locations. At each location, she releases the deadman brake to prevent rolling and initiates an image acquisition sequence by pressing a single

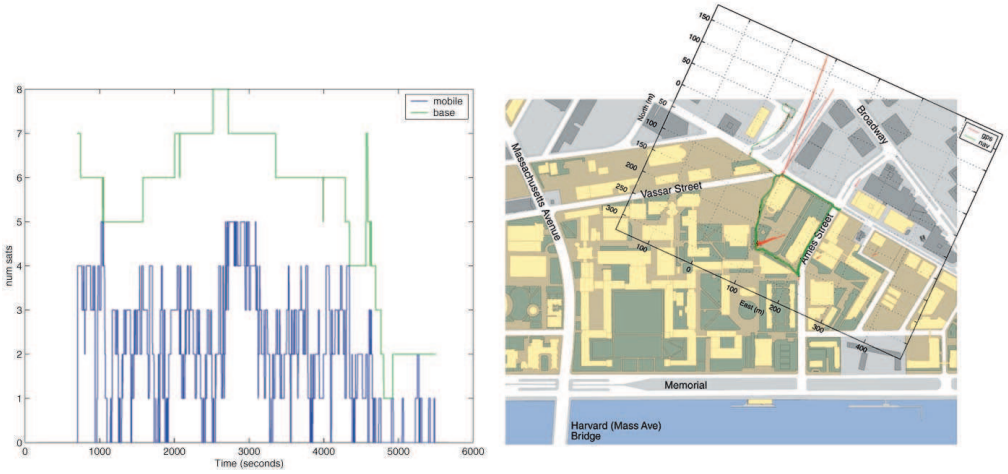


Figure 4 shows, in the graph at left, the time history of GPS visibility, with the mobile receiver of Argus seeing an average of only two GPS satellites at one time. The diagram at right illustrates the trajectories of Argus, shown in context with the local environment. Note the long spikes in trajectory, an effect of multipath in the urban canyon.

key or clicking a mouse.

Argus starts each acquisition sequence by extending three pins to the ground, stabilizing and leveling itself. The motorized pan/tilt head then sweeps through a predefined set of camera orientations tiling a roughly hemispherical field of view.

At each orientation, Argus acquires several high-resolution digital images at different exposures. Each image is merged with a pose estimate provided by the navigation subsystem, and written to onboard storage as a pose-image. Upon completing the sequence, Argus retracts its leveling pins. The operator then moves Argus to the next location.

Argus occupies each location for about five minutes, depending on the number of pose-images acquired (typically 40–80). During acquisition, the operator and her assistant fend off curious passers-by while monitoring the subsystem status displays.

When the operator finishes collecting pose imagery, she wheels Argus back to the lab and plugs it into network and power outlets (its battery is typically nearly depleted upon return). She then transfers the image and navigation data from Argus and the GPS base station onto our file server. The data then flow into a cluster of work stations running our pose refinement and model reconstruction algorithms.

TESTING ARGUS

At present, we are assessing the performance of Argus through a procedure to evaluate and validate the accuracy and precision of our navigation solution.

Procedure. We designed a course of roughly 20 well-separated waypoints around the MIT campus, each visible to at least two other waypoints.We hired a surveying firm to determine independently the true position of each waypoint, and thus the heading vector between any two intervisible points.

Our experimental procedure involves traversing the course many times throughout the day,

allowing for variations in the effective GPS constellation. At each waypoint, Argus is precisely positioned over a survey mark on the ground, and its camera is centered (with the help of automatic image feedback) on a small target positioned above another survey mark.

We can then compare the variance and drift of the computed navigation solutions with the true locations of the waypoints. The estimated variances reported by the Kalman filter can also be checked against the true variances for accuracy. Thus our waypoint procedure yields a rigorous characterization of the overall accuracy and precision of the Argus sensor.

On the Fly

This application . . .

discovered

- to create 3D CAD models of urban environments involves acquiring multitudes of high-resolution images aligned into spherical mosaics
- eliminating the "human-in-the-loop" from that process requires a novel sensor to acquire the georeferenced imagery
- the limitations of such a system's individual sensors can be overcome by combining multiple components with a Kalman filter and targeted processing algorithms
- despite severe multipathing and other GPS challenges that typically produced position accuracies of 20–30 meters, the final navigation solution proved to be about 1 meter in position and 1 degree in heading, which met the project's accuracy and operational requirements, with room for growth as technology advances

Results. We have repeatedly traversed a subset of the validation waypoints with Argus. Using the procedure outlined above, we measured how much the reported pose varied across different visits to each waypoint.

Interim tests in our urban campus indicate an overall precision of about 1 meter in position and about 1 degree (20 milliradians) in heading, both to a one-sigma error bound. We are now working to establish comparable results for the absolute accuracy of the sensor.

The quality of the navigation solution depends on the dead-reckoning drift rates of the IMU and wheel encoders. These drift rates are consistent from day to day. The navigation solution also depends on the quality of GPS data, as it is the only source of absolute position information. The quality and frequency of the GPS updates varied significantly across different times and locations.

Typically, eight GPS satellites orbit above the horizon. In our dataset, however, only two satellites were visible on average, with a maximum of five visible at any time.

The Kalman filter requires only two visible satellites to make use of GPS data. Even then, it could perform GPS updates less than 60 percent of the time. A full GPS solution, requiring four or more satellites, was possible only 18 percent of the time.

Argus tracked, on average, only two of the eight satellites above the horizon (see Figure 4). The GPS data exhibited severe multipathing in our urban environment, resulting in a raw GPS accuracy of only 20–30 meters. Despite these limitations, the final navigation solution reduced this error by more than an order of magnitude and produced heading information to about a degree of precision.

Next Steps. In the next few months, we plan to complete rigorous evaluation and validation of the navigation system. Future plans for improving the platform include incorporating onboard, vision-based updates directly into the navigation systems.

During the spring and summer, we will collect and process several thousand pose images covering most of the MIT campus, or about 1 square kilometer. We expect a total input datasize to approach several terabytes (trillion bytes) of image data, and the extracted model to include several hundred structures, each with detailed geometry and texture information.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of the National Science Foundation (through Career Award IRI-9501937), the Defense Advanced Research Projects Agency (under contract DACA76-97-K-0002), the Office of Naval Research (under MURI award SA 1524-2582386), the MIT Lincoln Laboratories (under grant ACC-233), and Intel Corporation. For more information about the City Project and citations for all work mentioned in this article, see <http://city.lcs.mit.edu/city>. ■

MANUFACTURERS

The developers of the Argus employed a wide variety of equipment. For GPS, they used a **Motorola** (Northbrook, Illinois) Oncore receiver and a **Trimble** (Sunnyvale, California) 7400MSi system. The IMU was a Longbow model from **GEC-Marconi** (Warrendale, Pennsylvania), developed in cooperation with **Korbin Systems** (Andover, Massachusetts). The wheel encoders are from **Encoder Technology International** (Fallbrook, California). Both a DCS-420 digital camera from **Kodak** (Rochester, New York), housed in a **Nikon** N-90 camera body, and a **Wintriss** (San Diego, California) OPSIS 1300 ASC color camera with digital I/O board were used. **Animatics** (Santa Clara, California) supplied the smart motors. **Peace River Studios** (Cambridge, Massachusetts) worked with the Argus team to design and build the pan/tilt head and physical cart structure. The HyperKernel software is by **Nematron** (Ann Arbor, Michigan). **Cullinan Engineering** (Boston, Massachusetts) provided surveying services, and GeoTrans coordinate transformation software from the **Army Corps of Engineers** was used.

Seth Teller *obtained a B.A. in Physics from Wesleyan University, and a Ph.D. in Computer Science from the University of California at Berkeley, focusing on accelerated rendering of complex architectural environments. After postdoctoral research at the Computer Science Institute of the Hebrew University in Jerusalem and Princeton University's Computer Science Department, he joined MIT's Electrical Engineering and Computer Sciences Department, and MIT's Lab for Computer Science (LCS), founding the Computer Graphics Group there. He pursues research in computer graphics, computer vision, and computational geometry, with the common theme of acquiring, representing, manipulating, and interacting with complex geometric datasets.*

Michael Bosse *is a Ph.D. candidate in the Computer Graphics Group at LCS, where he has been developing the navigation system for Argus. He earned his B.S. and M.S. in 1997 from Boston University, where he built a vision-aided navigation system for an autonomous helicopter. His interests include computer vision, robotics, and navigation.*

Douglas S.J. De Couto *is a Ph.D. candidate in the Parallel and Distributed Operating Systems group at LCS. He received his B.S. and M.Eng. in Computer Science in 1998 from MIT, where he worked on Argus as a member of the Computer Graphics Group at LCS. His current work is on large mobile computer networks that take advantage of GPS (and other) location information to route data. His interests are mainly in computer systems.*