

# Tracking Pitches for Broadcast Television

*A cable network's desire to enhance its baseball telecasts resulted in K Zone, a computerized video tracking system that may have broader applications.*



André  
Guézic  
Triangle Software

In baseball, a pitcher's fame and fortune depend on his mastery of the strike zone. Pitches that pass outside the strike zone count as balls and can be safely ignored. Those that pass through it untouched, however, count as strikes, three of which will retire the batter to his team's dugout. Players, fans, and sports journalists thus have an intense interest in compiling statistics about these pitches—as do the umpires who determine each pitch's status when it crosses the plate.

During the 2001 major league baseball season, the strike zone received special attention when officials decided to enforce the game's original strike zone definition, placing the zone's upper limit between the batter's shoulders and belt. In the past, major-league umpires had rarely called a strike above the belt. League officials and journalists thought that the effect of enforcing the original definition would be so significant it might change the hierarchy of hitters and pitchers. Further, 2001 turned out to be a particularly exciting year for baseball as Barry Bonds pursued—and ultimately surpassed—Mark McGuire's single-season home run record set in 1998.

These developments made tracking pitches accurately more important than ever. Tracking the flight of a pitch during a live broadcast presents two major challenges, however: speed and image-processing reliability. Speed is an issue because ensuring rapid calculation of the trajectory practically requires real-time processing of the 60-fields-per-second video.

Ensuring image-processing reliability, on the other hand, requires overcoming several obstacles.

During a baseball game, dramatic changes in lighting conditions and the movement of objects and players can result in a shifting pattern of light and color that makes it especially difficult to track a pitched ball. Further, several ballparks have a net in place behind home plate, which contributes further to the visual clutter that the image-processing system must filter out when tracking the baseball.

Meeting these challenges required developing a complex system that fuses high-end computer graphics with a sophisticated algorithm for calculating flight trajectories. The ESPN K Zone system uses computer-generated graphics to create a shaded, translucent box that outlines the strike zone boundaries for viewers. Behind the flashy graphics, K Zone—named after a synonym for the strike zone—is a sophisticated computing system that monitors each pitch's trajectory.<sup>1</sup>

## K ZONE TAKES SHAPE

In February 2001, ESPN contracted with Sportvision to build a system for analyzing baseball pitches during its Major League Baseball broadcasts. ESPN wanted a system that would determine electronically, within one to two centimeters, whether each pitch qualified as a strike or a ball. The system would then draw a representation of the strike zone on the TV screen, superimposed over the replayed broadcast video, to clearly show the pitch's status. ESPN chose Sportvision for the project because of the company's track record in graphically enhancing sports broadcasts.

## System overview

Figure 1 shows K Zone in action during a television broadcast. ESPN insisted that the effect appear on the program video as an integral part of the scene, not as a separate graphical animation. To fulfill this requirement, the developers minimized the graphics so that they would not obscure any part of the game.

The overall broadcast enhancement system uses three subsystems to produce the final televised graphics:

- The *camera pan-tilt-zoom encoding subsystem* calibrates the broadcast cameras in real time.
- The *measurement subsystem* detects the baseball's trajectory, measures the batter's stance, and determines if the pitch is a strike or a ball.
- The *graphic overlay subsystem* uses these measurements to produce the televised graphics. To draw them in the proper position, this subsystem needs the real-time calibration data that the camera subsystem provides.

The trajectory component, which consists of three PCs connected to three video cameras, tracks a pitched baseball's flight toward the strike zone. Two cameras observe the baseball, while the third observes the batter to provide proper sizing for the strike zone.

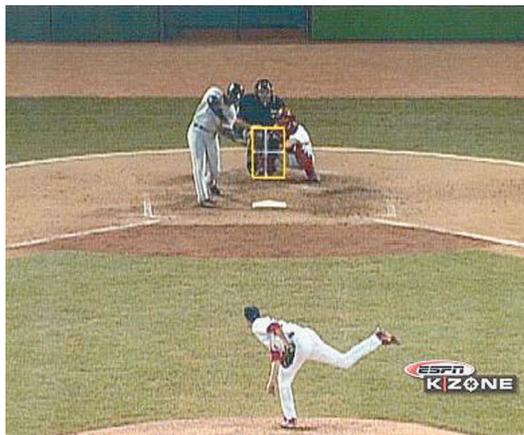
For calibrating the broadcast cameras, technicians install an encoder on each camera that measures the pan and tilt angles, zoom voltage, and zoom extender positions. The encoders collect these measurements 30 times per second and transmit them to the graphic overlay subsystem.

The graphic overlay subsystem renders a graphic and superimposes it on the broadcast video. This graphic consists of two video streams, the *fill*, which contains the actual graphic, and the *key*, which contains the transparency map that indicates the video pixels the graphic affects. These two streams are input to the linear keyer, a piece of video equipment that overlays the graphic on the broadcast video. The graphic overlay subsystem uses an SGI O2 computer to draw a three-dimensional representation of the strike zone in the position that the broadcast camera's pan, tilt, and zoom parameters specify.

Although Sportvision had used the camera and graphic-overlay systems in their broadcasts for several years, using them with K Zone required modifications. The measurement subsystem had to be built from scratch.

## Measurement subsystem

K Zone's measurement system uses two Pentium 4 PCs running Windows 2000, linked to two video



**Figure 1. K Zone during a televised game. The pitch-tracking effect is an integral and unobtrusive part of the telecast.**

cameras that observe each pitch. An operator uses a third camera and PC to locate the strike zone's top and bottom boundaries. All these components fit conveniently in one short equipment rack.

During operation, each PC processes the video for one camera in real time. The processing uses a four-way multithreaded software architecture. One thread reads the video frame into memory, a second displays the video, a third handles the image processing, and the fourth writes the video to disk.

We tested the pitch-tracking system extensively before using it in its broadcasts. Technicians checked various camera locations for tracking the baseball, selected views that permitted the most reliable detection, and refined the tracking algorithm. These tests took place over several weeks early in the season at baseball games played in Oakland, Minneapolis, and New York.

## TRACKING CONSTRAINTS

To accomplish pitch tracking, the developers needed to deal with four primary constraints.

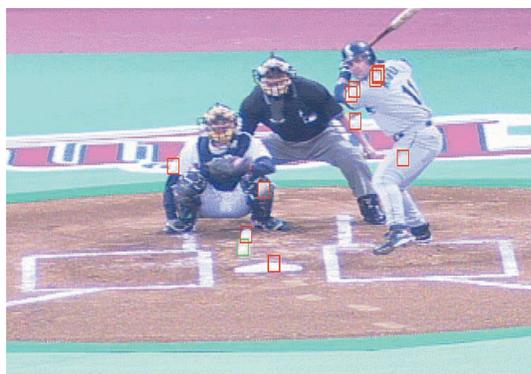
### Performance

Full-resolution digital NTSC (National Television System Committee) or PAL (phase alternating line) video requires 270 Mbits per second of bandwidth. Importing, displaying, and exporting this data in real time takes several passes through the personal computer's PCI bus, stretching it nearly to capacity. Doing all this data transmission in real time requires carefully optimized software engineering. At the very least, multithreading is essential to keep the CPU working on the video-processing pipeline while waiting for the next video field or frame to arrive. The system then decomposes the video-processing pipeline into tasks executed independently in a thread-safe manner.

### Real-time operation

Although ESPN planned to use the system for replays, we designed the image-processing pipeline to work in real time to keep pace with the video frame rate, with a delay of two seconds. Such a design

**Figure 2. Broken trajectory mapping. This ultimately unsuccessful algorithm seeks the baseball pattern in the color image, resulting in the scattered ball positions that the red squares denote. The algorithm became ineffective when lighting conditions, the field, and team colors combined to make parts of the uniforms and background look more like a baseball than the baseball itself.**



allows using the effect live, provided the program receives a two-second or longer delay when broadcast. Sports broadcasts commonly apply such delays.

Even when used for replays alone, the pitch-tracking system still needed to process video quickly. Creating a successful replay required executing several steps in rapid succession. In addition to processing the video, using the system required coordinating with ESPN's television operators to cue the appropriate footage. So pressing were these constraints that the show's director would periodically cancel replays for lack of time.

### Reliability

Image processing and computer vision have been well-established academic fields since the 1970s. Although many academics have published applications in this field, few have been highly reliable and successful in practice. Academic emphasis tends to be directed more toward pure innovation and mathematical elegance—demonstrated with a few carefully chosen test cases—than toward reliability. Partially because taking measurements only seconds before airing them on television can be extremely stressful, engineers who build applications for commercial broadcasting tend to emphasize reliability and repeatability.

Developing the image-processing system required meeting the particularly significant challenge of designing an algorithm that would work in all possible lighting conditions for an event staged outdoors. The image-processing system needed to function in sunny or overcast lighting, during the day and at night, on both well-lit and shadowed subjects, on scenes composed of different viewing angles or involving markedly different backgrounds, and on images filtered through a foreground net.

### Efficient detection

Developing a successful tracking algorithm provided the key to making K Zone work. To track pitches effectively, the system needed to isolate the ball and follow it throughout its trajectory. We began with color cameras working in the standard and less costly interlaced mode at 30 Hz, which

meant that the video would contain 60 fields—or half frames—per second. The interlacing, which is the television standard, displays two fields in alternating lines on a frame and thus represents two different moments in time.

Each camera has a field of view that covers about half the baseball's flight. As a consequence of the relatively wide field of view, the baseball's image consists of only a few pixels. Depending on the view, background, foreground, and other factors, the baseball can appear as no more than two pixels after detection. In one view, the ball passes near—and often over—the white foul line, creating a white-on-white image.

Several moving objects and shadows could be mistaken for the ball as well. The home plate umpire, catcher, and batter typically stay immobile, then move swiftly and precisely when the ball is pitched, while the computers are busy detecting it. Baseball uniforms typically have white or gray patches, such as a white handkerchief hanging from the umpire's pocket. Helmets exhibit specular highlights that can be mistaken for the ball in some views.

Figure 2 shows how a later-abandoned image-processing algorithm that uses multiresolution pattern matching created a misleading image in the video sequence taken from the centerfield position. Compare this image with Figure 3, generated by an algorithm that I developed. The system draws the successive detected positions of the baseball, whether successfully tracked or not, superimposing colored squares on a corresponding video sequence's images. In Figures 3b and 3c, the still frame represents two fields and shows two successive baseball positions, one for each field. The baseball locations in the images appear as small green blobs of just a few pixels, with the interlacing causing the color to skip every other line.

### TRACKING ALGORITHM

Instead of using pattern matching, the K Zone tracking algorithm exploits the kinematic properties of the baseball's flight. The first step, however, is to qualify a potential baseball position in the image.

A potential ball position corresponds to a number of adjacent pixels, or blobs, that satisfy simple criteria in terms of size, shape, brightness, and color. Specifically, the algorithm eliminates anything that is too colorful, looking instead for a grayish shape with perhaps a little red dirt or green from the grass on it. More importantly, these adjacent pixels must be significantly different from what occupied that section of the image in previous fields.

## Background subtraction

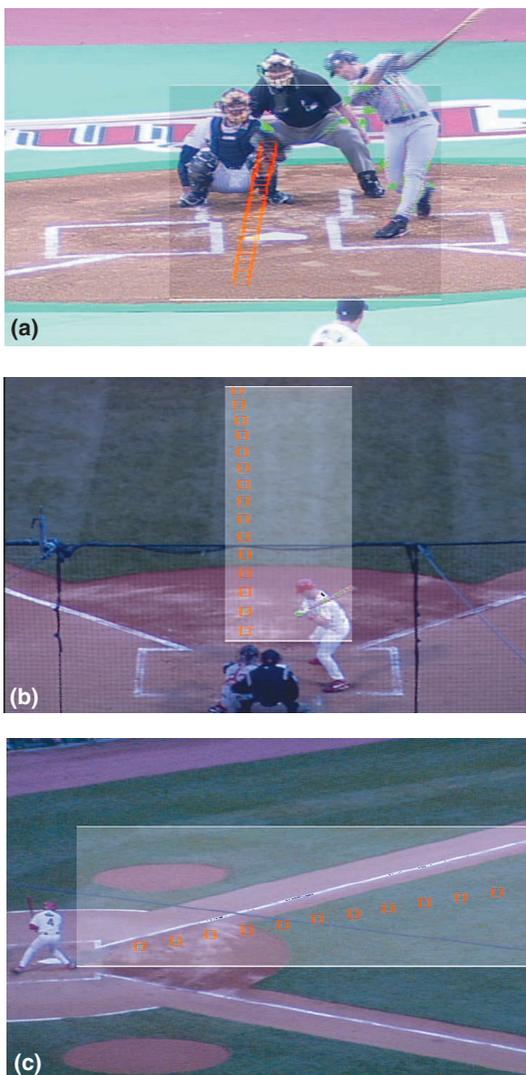
To estimate the probability of obtaining a given pixel value, the tracking system draws samples of intensity values from previous frames or fields.<sup>2</sup> A Gaussian distribution works well for a background pixel that does not change over time. To handle variations over time, the system uses a mixture of Gaussian distributions centered about recently observed pixel values for a given location. Pitch tracking updates these models continuously to monitor changes in the background caused by moving shadows and lighting variations such as switching from natural to artificial light. If the current pixel deviates significantly from a predicted value, the system registers the change as motion. The various pixel locations a ball occupies when it passes through the camera's field of view generally satisfy this criterion.

Size, shape, color, and background differencing criteria—although significantly constraining—cannot by themselves provide enough data to distinguish a pitched ball from various other moving elements in a video sequence. Specifically, depending on the view, perhaps 10 to 15 blobs of pixels satisfy these criteria at a given time, with at most one corresponding to the ball's correct location. Figure 3, for example, shows all ball candidates as green blobs of pixels. Narrowing the cameras' field of view could resolve this problem at least partially. But doing so would result in more cameras showing less of the trajectory so that they could track larger and thus less ambiguous pitched balls.<sup>3</sup> Even if there had been time to implement this upgrade, it would have increased the system's cost and complexity significantly.

## Finite state machine

The system does not necessarily select the correct blob instantly from among all candidates. Rather, the selection depends on the past and future blob sets, and the consistency the system can find among them. The selection algorithm uses a finite state machine with two states. In the first state, the algorithm looks for physically plausible trajectories. It checks the angle, the velocity in pixels per field, and the trajectory's deviation with respect to a locally linear fit of the samples. If the tracking system can match enough candidates in successive fields, they serve as seeds for starting a trajectory in the image's two-dimensional plane.

To acquire a plausible trajectory, the system delays the image processing by a small number of fields so that the algorithm can look both ahead and back a few fields—corresponding to a time



**Figure 3. Successful trajectory mapping.** The final algorithm chooses the correct ball positions, shown as red squares, from among a variety of candidates, shown as green patches, whose shape, color, and brightness patterns resemble those of the baseball. K Zone provided three views of each pitch: (a) batter close-up (for sizing the strike zone only), (b) high home, and (c) high first base. In these stills, used for real-life detection, the shutter speed necessary to keep the ball's image from blurring causes some of the shots to be dark.

lapse of perhaps one tenth of a second—for potential ball positions that form a consistent trajectory.

Once the algorithm has seeded a trajectory, it works in the second state, in which it tests all the potential blobs for a fit with the existing trajectory. The actual mathematics of the trajectory use a Kalman filter, as the “Using Kalman Filtering to Track Trajectories” sidebar describes.

Since the process uses regular interlaced NTSC video, a still frame contains two successive positions of the baseball, shown as green blobs in the figures. Background and foreground variations can complicate trajectory acquisition. Figure 3b, for example, shows how the foregrounds with net or no net, and the backgrounds with grass or dirt, vary dramatically within the same trajectory.

## DETERMINING THREE-DIMENSIONAL TRAJECTORIES

Our two pitch-tracking computers work as a team to compute each final trajectory. One observes the view from high home, the other from high first base, exchanging the two-dimensional positions

## Using Kalman Filtering to Track Trajectories

Invented in 1960 by Rudolph E. Kalman, Kalman filtering<sup>1</sup> is a commonly used technique for removing measurement errors and estimating a system's variables. A vector represents each system parameter and measurement. Linear equations must describe the system and measurement evolutions over time. The Kalman filter provides optimal estimates of the system parameters, such as position and velocity, given measurements and knowledge of a system's behavior.

In general, the Kalman filter assumes that the following two relations can describe a system:

$$\mathbf{x}_k = \mathbf{A}_k \mathbf{x}_{k-1} + \mathbf{w}_k \quad (1)$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \quad (2)$$

$\mathbf{x}_k$  is the state vector, such as a position and velocity, perhaps an acceleration or other parameters, while  $\mathbf{z}_k$  is the measurement, such as a position.  $\mathbf{x}_0$ ,  $\mathbf{w}_k$ , and  $\mathbf{v}_k$  are mutually uncorrelated vectors, and  $\mathbf{w}_k$  (process noise, or process evolution) and  $\mathbf{v}_k$  (measurement noise) are white noise sequences. The first equation determines the evolution of the state over time, and the second relates measurement and state.

Once this is established, a recursive algorithm estimates  $\mathbf{x}_k$  optimally.<sup>2</sup> In this case, "optimally" relates to minimizing the mean squared error in Equation 2 over all the measurements.

The Kalman filter algorithm uses the following notations:  $:=$  designates the assignment operator,  $\mathbf{P}$  is the covariance matrix associated with the state,  $\mathbf{Q}$  is the process noise covariance matrix,  $\mathbf{R}$  is the measurement noise covariance matrix, and  $\mathbf{K}$  is the *Kalman gain* matrix. The algorithm consists of two main steps, as follows.

The *time update* or prediction step:

$$\mathbf{x} := \mathbf{A} \mathbf{x}; \quad (3)$$

$$\mathbf{P} := \mathbf{A} \mathbf{P} \mathbf{A}^T + \mathbf{Q}; \quad (4)$$

and the *measurement update* step, which adapts the state to the new measurement value:

$$\mathbf{K}_1 := \mathbf{H} \mathbf{P} \mathbf{H}^T + \mathbf{R}; \quad (5)$$

$$\mathbf{K} := \mathbf{P} \mathbf{H}^T \mathbf{K}_1^{-1}; \quad (6)$$

$$\mathbf{x} := \mathbf{x} + \mathbf{K} (\mathbf{z} - \mathbf{H} \mathbf{x}); \quad (7)$$

$$\mathbf{P} := (\mathbf{I} - \mathbf{K} \mathbf{H}) \mathbf{P}; \quad (8)$$

After a time update, the algorithm can reject a measurement if the predicted

value in Equation 3 is too different from the measurement  $\mathbf{z}$ , given the current uncertainty on the prediction:  $\mathbf{H} \mathbf{P} \mathbf{H}^T + \mathbf{R}$ . Also, if the algorithm can choose between several potential measurements to feed the filter, with only one being the correct measurement, the correct choice will probably be close to the predicted value within the uncertainty region.

Typically,  $\mathbf{P}$  gradually decreases as the algorithm incorporates more measurements: Confidence in the state builds up. Equation 7 shows that if  $\mathbf{K}$  is large—which is the case if  $\mathbf{R}$  is small, meaning that there is little noise in the measurements—the new measurement  $\mathbf{z}$  is weighted heavily. Instead, if  $\mathbf{K}$  is small, the value the current state  $\mathbf{x}$  predicts has a higher weight. Thus  $\mathbf{K}$  works as a gain.

### References

1. G. Welch and G. Bishop, "An Introduction to the Kalman Filter," <http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html>.
2. L. Levy, "The Kalman Filter: Navigation's Integration Workhorse," [http://www.cs.unc.edu/~welch/media/pdf/Levy0997\\_kalman.pdf](http://www.cs.unc.edu/~welch/media/pdf/Levy0997_kalman.pdf).

the tracking algorithm computes in each view in real time over a local Ethernet connection. K Zone uses synchronized time-code generators to inscribe a field-accurate time code into each field of each camera view. The pitch-tracking computers use the time codes to tag each two-dimensional position unambiguously.

One of the two pitch-tracking computers intersects two lines of sight from each camera to combine two two-dimensional positions that correspond to the same time code into a three-dimensional position. More specifically, the computer calculates the point of closest approach between the two straight lines. Each two-dimensional position, or pixel, in a camera view can be associated with a straight line in three-space that essentially depicts the path of the photons that hit that pixel. Figure 4 illustrates the process of using intersecting line-of-sight pairs to locate a baseball's successive positions. Associating each pixel with a line of sight is called camera calibration<sup>4</sup> and presents several challenges in terms of accuracy, especially in a large environment such as a ballpark.

To determine the final trajectory, the system feeds successive three-dimensional positions into another Kalman filter.

The third computer and camera provide a close-up view of the batter. A TV camera operator uses a joystick to locate the strike zone's top and bottom on the live video. Since the rule book says a strike occurs when any part of the baseball enters any part of the strike zone, K Zone computes the intersection of the strike zone's volume—a pentagonal prism—with a cylinder that has the same radius as the baseball, centered on the computed trajectory. The tracking system reports the intersection as a strike and the absence of an intersection as a ball. In either case, the system reports the baseball trajectory's point of impact with the front plane of the strike zone and draws this intersection on the TV screen.

### USER AND VIEWER FEEDBACK

K Zone launched on 1 July 2001 during ESPN's *Sunday Night Baseball*, and the network used it to augment every edition of the program aired that summer. Estimates from ESPN's published ratings indicate that about 13 million viewers tuned into the show each week and watched the graphical representation of the strike zone as each pitch either hit the zone for a strike or missed it for a ball.

Even though replays presented only a small fraction of the 300 to 400 pitches thrown each game, the

system tracked every pitch with an extremely low failure rate. In this context, failure means that the system did not accurately detect the pitched baseball's trajectory. During the entire season, at worst the system missed only a handful of pitches per game, and frequently none at all. Significantly, operator error accounted for most of the few reported failures.

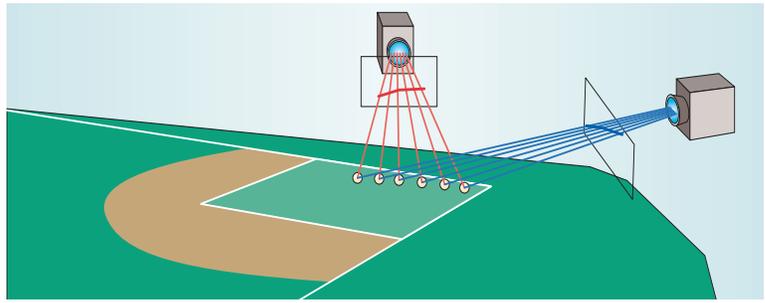
ESPN used the system intensively, showing 17 replays in K Zone's debut game, then 20 to 30 replays per game thereafter. Viewers and critics alike responded positively to K Zone's visual effects through messages posted to various Internet forums. Several critics went further, urging ESPN to use the system on controversial umpire calls. Although the network rarely used K Zone in this capacity early in the season, it did so occasionally later in the season.

## BROADER APPLICATIONS

A potential future application for the technologies used in K Zone may lie in computerized medical-image analysis. The medical literature documents that radiologists sometimes make mistakes in reading x-rays and other imaging studies—as do experienced, well-trained, and dedicated umpires when watching a baseball flying straight toward them at 90 miles per hour.

Although much progress has been made lately in the automated processing and understanding of medical images, the process of examining a chest x-ray, even digitally, remains surprisingly similar to what it was several decades ago. Healthcare providers could apply some of the computer vision techniques used to detect a baseball to these analyses, either directly or in a closely related form. Image differencing, which relates to background differencing, is commonly used in digital subtraction angiography, for example, to show a patient's arterial network after injection of a contrast material. Likewise, technicians could compute an anatomical shape's motion and trajectory on image sequences to determine how a clinical condition evolves over time or in response to treatment.

Computer vision is coming of age, after decades of mostly academic pursuits. We finally have the computational power—at a reasonable price—and proper expertise to apply this technology to challenging technical problems. However, while solving some problems might seem easy intuitively, in reality, doing so may be tremendously difficult. In particular, the visual nature of these issues may give a false impression of simplicity: Humans are typically very skilled at pattern recognition, when



**Figure 4. Locating a baseball's successive positions. To locate a baseball's position in three dimensions, K Zone finds the point of closest approach between the two video cameras' lines of sight: high above home plate and high above first base.**

identifying faces, for instance, whereas computers obviously have none of these human perceptual skills built in. K Zone, built on a tight deadline that allowed only four months for its development, augmented existing technology to create a sophisticated and reliable system that has proven commercially viable and a valuable enhancement for sports fans. Further, ESPN is so pleased with the technology it is seeking an Emmy nomination for K Zone. This development model may well prove effective in future computer vision projects, regardless of the application domain. ■

## Acknowledgments

I thank Sportvision's Rick Cavallaro, Mike Cramer, Matt Lazar, Jim McGuffin, Alon Moses, and Marv White; J.R. Gloudemans; and Sportvision, ESPN, and Reality Check.

## References

1. A. Guézic, "Tracking a Baseball Pitch for Broadcast Television," [http://www.trianglesoftware.com/pitch\\_tracking.htm](http://www.trianglesoftware.com/pitch_tracking.htm).
2. A. Elgammal, D. Harwood, and L. Davis, "Non-parametric Model for Background Subtraction," <http://fizbin.eecs.lehigh.edu/FRAME/Elgammal/bgmodel.html>.
3. G. Pingali, Y. Jean, and A. Opalach, "Ball Tracking and Virtual Replays for Innovative Tennis Broadcasts," *Proc. 15th Int'l Conf. Pattern Recognition (ICPR)*, IEEE CS Press, Los Alamitos, Calif., 2000.
4. O. Faugeras, *Three-Dimensional Computer Vision*, MIT Press, Boston, 1993.

*André Guézic is the founder and CEO of Triangle Software (<http://www.trianglesoftware.com>). He is the main developer and designer of baseball tracking in K Zone. His interests include image processing and computer vision, notably applied to medical imaging, 3D shape modeling, and processing (particularly simplification), as well as modeling, simulation, and animation of road traffic. Guézic received a PhD in computer science from the University of Paris at Orsay, where he specialized in medical image analysis. He is a senior member of the IEEE and holds several US and international patents. Contact him at [andre@trianglesoftware.com](mailto:andre@trianglesoftware.com).*