

A prototype computer-vision system for tracking bats

Stephen C. Crampton, Margrit Betke, Thomas H. Kunz, and Jason Horn

E-mails: stevec@cs.bu.edu, betke@cs.bu.edu, kunz@bu.edu, and jhorn@bu.edu

Boston University, 111 Cummington Street, Boston, MA 02215

We have developed a prototype computer-vision system to analyze infrared images of bats leaving their cave at night. The goals are to census the bat population and to analyze bats' flying patterns. The challenges in tracking are threefold: (1) the large number of bats in each frame, (2) the small apparent size of each bat, and (3) the speed and quick changes in velocity and direction that bats make.

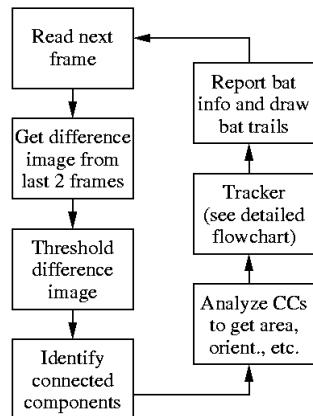


Figure 1 System overview

The prototype system is a tracker: it identifies and analyzes bat-like objects at each frame and then establishes correspondences between such objects in successive frames. From this the system counts and computes the motion of bats as captured by the imaging device. Currently, analysis is limited to 2D motion parallel to the image plane.

Figure 1 shows the system overview. After reading frame $I_t(i, j)$, the system computes the difference image $D_t(i, j) = I_t(i, j) - I_{t-1}(i, j)$ from the previous frame $I_{t-1}(i, j)$. The difference image effectively segments the bats from the background scene, because virtually the only things moving in the scene are bats. The system next converts the difference image into a binary image, using threshold $t = 1$ to remove any imaging noise:

$$B(i, j) = \begin{cases} 1 & \text{if } T(i, j) > t \\ 0 & \text{otherwise} \end{cases}$$

According to the manufacturer, the Merlin® Mid infrared camera has corrected uniformity of $< 0.1\%$.

Once foreground pixels are segmented, we need to group pixels into distinct objects. To do this, we use a sequential algorithm similar to the one described in [Jain, Kasturi, and Schunck, Machine Vision, p. 47 (1995)]. The algorithm labels each object in the binary image according to 8-connectedness. (A 1 pixel in a binary image is 8-connected with any other 1 pixel in its immediate neighborhood.)

Once distinct objects are identified, the system computes the area $A = \sum_{(i,j)} B(i, j)$ and centroid $(\bar{x}, \bar{y}) = (\frac{1}{A} \sum_{(i,j)} jB(i, j), \frac{1}{A} \sum_{(i,j)} iB(i, j))$ of each object. The system also computes each object's

orientation \mathbf{q} with respect to the x-axis as follows:

$$\mathbf{q} = \frac{1}{2} \arctan \left(\frac{s}{r - t} \right), \text{ where } r = \sum_{(i,j)} (i - \bar{x})^2 B(i, j),$$

$$s = 2 \sum_{(i,j)} (i - \bar{x})(j - \bar{y}) B(i, j), \text{ and } t = \sum_{(i,j)} (j - \bar{y})^2 B(i, j)$$

[Jain et al., p. 34]. The orientation \mathbf{q} corresponds to the slope of the line with the least-squared fit with respect to the pixels composing the object (see Figure 2). The line segment labeled \mathbf{a} in Figure 2 is the portion of that line that intersects with the object itself. If the object were an ellipse, \mathbf{a} would be the major axis. The line segment orthogonal to \mathbf{a} , corresponding to the minor axis of an ellipse, is labeled \mathbf{b} in

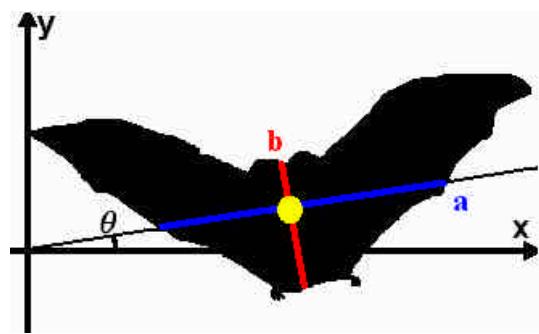


Figure 2 Centroid and orientation of a bat-like object

Figure 2. The geometrical information is collected into a feature vector $\mathbf{f} = (A, \bar{x}, \bar{y}, \mathbf{q}, |\mathbf{a}|, |\mathbf{b}|)$, which becomes the input to the tracking procedure.

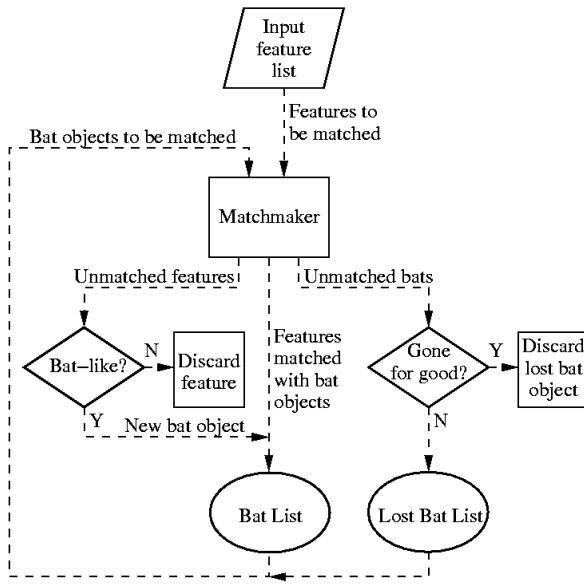


Figure 3 Data flow through the tracker

predicted position: $r = \frac{1}{2} \left[\frac{1}{B} \max(B - \Delta b, 0) + \frac{1}{D} \max(D - \|\mathbf{p} - (\bar{x}, \bar{y})\|, 0) \right]$ B is the maximum allowable size difference. Δb is the difference between $|\mathbf{b}|$ for the feature and $|\mathbf{b}|$ for the bat object. The predicted position $\mathbf{p} = (x + \frac{dx}{dt}, y + \frac{dy}{dt})$ is the bat object's last known position plus its predicted velocity. (We discuss below why we use $|\mathbf{b}|$ as a measure of a bat object's size and how we compute a bat object's predicted velocity.)

The matchmaker pairs the feature vector and bat object with the highest rating, updates the bat object, and removes both from consideration. The process repeats until all feasible matches are made. At that point, unmatched feature vectors are examined to determine whether they are sufficiently bat like, based upon their geometrical characteristics. If so, a new bat object is created from the feature vector and added to the bat list. Unmatched bat objects are examined to determine whether they are likely to reappear: If a bat-like object's trajectory would take it outside the field of view or if the same bat object has been missing for more than a few frames, the object is considered gone for good and removed from the lost-bat list.

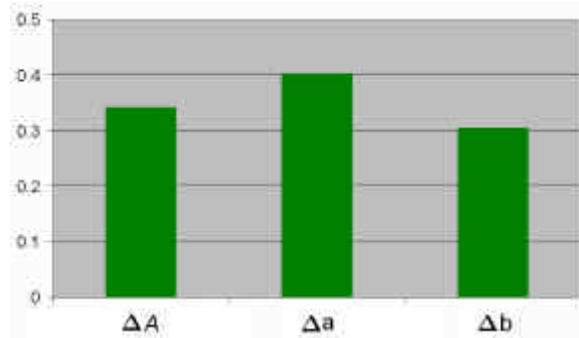


Figure 4 Mean rate of change for different measures of size

$N \times M$ image and the length of the object along \mathbf{b} : $\left(\frac{dx}{dt}, \frac{dy}{dt} \right) \propto (|\mathbf{b}|(\frac{N}{2} - \bar{x}), |\mathbf{b}|(\frac{M}{2} - \bar{y}))$. We chose this methodology, because, in video sequences we observed that a bat newly discovered near the edge of the field of view is most likely to be flying toward the center of the image. Also, a bat's velocity across the

Figure 3 shows the data flow through the tracker. At each frame, the tracker inputs a list of feature vectors for each object and updates two data structures: a “bat list” and “lost-bat list.” The bat list contains geometrical information and predicted velocities for bat-like objects identified in earlier frames. The lost-bat list contains objects from the bat list that have gone missing for one or more frames. From our observations, the main reason a bat becomes “lost” is because another bat flies in front of it, occluding it.

The heart of the tracker is the “matchmaker” procedure, which couples feature vectors with bat objects. For each bat object, the matchmaker examines all feature vectors whose centroid is within a threshold distance (D) of the bat object’s predicted position (\mathbf{p}). Each feature vector receives a similarity rating between 0.0 and 1.0 (1.0 indicates a perfect match). The similarity rating (r) depends upon the difference in size between the feature and bat object and the distance between the feature’s centroid and the bat object’s

predicted position: $r = \frac{1}{2} \left[\frac{1}{B} \max(B - \Delta b, 0) + \frac{1}{D} \max(D - \|\mathbf{p} - (\bar{x}, \bar{y})\|, 0) \right]$ B is the maximum allowable size difference. Δb is the difference between $|\mathbf{b}|$ for the feature and $|\mathbf{b}|$ for the bat object. The predicted position $\mathbf{p} = (x + \frac{dx}{dt}, y + \frac{dy}{dt})$ is the bat object’s last known position plus its predicted velocity. (We discuss below why we use $|\mathbf{b}|$ as a measure of a bat object’s size and how we compute a bat object’s predicted velocity.)

We use $|\mathbf{b}|$ as a measure of a bat object’s size because we find it varies less from frame to frame than other geometrical measures. For example, Figure 4 shows average values of ΔA , Δa , and Δb for nine bat objects whose trajectories were verified by hand. In performing the analysis in Figure 4, we first normalized ΔA , Δa , and Δb , so that comparisons between them would be meaningful.

We predict a bat object’s velocity $(\frac{dx}{dt}, \frac{dy}{dt})$ differently, depending upon its tracking status: For a newly discovered bat object, the system predicts the velocity based upon the distance from the center of the

image plane is related to its distance (z) from the camera. Adult Brazilian free-tailed bats vary less than 10% in size. Treating the actual size as constant, a bat's distance from the camera is proportional to its apparent size: $z \propto |\mathbf{b}|$. The velocity of a bat flying parallel to the image plane is likewise proportional to the distance, $\mathbf{v} \propto z$, and therefore $\mathbf{v} \propto |\mathbf{b}|$. When recording images of the bats, we placed the camera near the cave entrance where the great majority of bats leaving or entering the cave fly across the field of view.

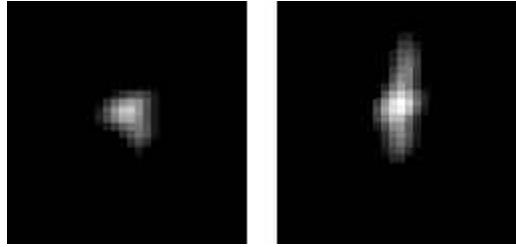


Figure 5 A single bat in two subsequent frames. For a bat that has been tracked in two consecutive frames, estimated velocity is computed as the distance traveled between the two frames: $(\frac{dx}{dt}, \frac{dy}{dt}) = (\bar{x}_t - \bar{x}_{t-1}, \bar{y}_t - \bar{y}_{t-1})$. For a bat that has been tracked for more than two consecutive frames, we compute estimated velocity as a weighted average of the most recent distance traveled and the previous estimated velocity: $\mathbf{v}_t = \mathbf{a}(\bar{x}_t - \bar{x}_{t-1}, \bar{y}_t - \bar{y}_{t-1}) + (1 - \mathbf{a})\mathbf{v}_{t-1}$. Finally, for a “lost” bat, we multiply its previous velocity by a “damping” factor (\mathbf{b}) to reflect uncertainty as to its trajectory: $\mathbf{v}_t = \mathbf{b}\mathbf{v}_{t-1}$ where $0 \leq \mathbf{b} < 1$.

In designing the tracker, we also considered the standard tracking method of template matching. Template matching, however, was confounded by the fact that a single bat's image can vary dramatically from frame to frame (see Figure 5).

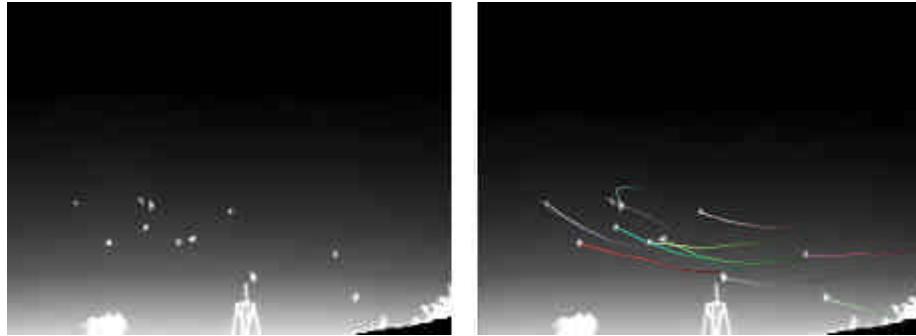


Figure 6 Infrared image (left) and the same image annotated to show bat trails (right).

annotated by the system. The trails show each bat object's last six positions, giving a visual representation of its trajectory. The trails also show how well the tracker is working. A user can check by hand at a reduced frame rate to determine whether the bats' trails properly depict their trajectories.



Figure 7 Tracking a multitude of bats (image cropped to show detail).

After the tracker finishes updating its data structures, the system uses the bat list and lost-bat list to report bat objects' geometrical information and velocity. The system also annotates each frame with colored bat “trails.” Figure 6 shows a frame from the infrared camera juxtaposed with the same frame as

Figure 7 is cropped and enlarged from the system's display of a frame containing hundreds of bats. The system appears to scale well to tracking numerous bat objects.

We propose further work on the prototype system. Initially we plan to evaluate and adjust the tracker's parameters and determine its error rate. To do this, we will compare hand-marked images with the tracker's annotations and analyze the position, trajectory, and geometrical characteristics of each bat as reported by the system. We also plan to refine our model of a bat object into a probabilistic model, incorporating random variables reflecting a bat's tendency to change velocity and direction. Further along these lines, we would like to incorporate motions of nearby bats into the model, given that bats appear to exhibit group behavior with respect to their flight patterns. Finally, we would like to extend the system to track bats in 3D. To accomplish this, we propose using two cameras in a stereo configuration.