Tracking Multiple Objects in Non-Stationary Video

Hoang Nguyen and Bir Bhanu
Center for Research in Intelligent Systems
University of Riverside, California

Presented at the RWA-1 Session at GECCO 2009
July 10, 2009
Object Tracking

- **Object tracking** is a staple computer vision problem of significant importance where the objective is to continuously locate the position of an object in sequential frames of a video.

Example: Head tracking
Problem

• Given a live video stream, **how to do real-time object tracking when there are multiple objects and the camera is moving?**

• Example:
  Tracking people in a crowd using a camera that is on an automated tour of a room.
Challenges

• **Real-time** means minimum of 30 frames/s

• **Multiple objects** means,
  – Occlusion can occur anywhere (objects can seemingly disappear and reappear arbitrarily)

• **Non-stationary video** means,
  – Even stationary objects will appear to move
  – No fixed point of reference to judge movement
<table>
<thead>
<tr>
<th><strong>Approach</strong></th>
<th><strong>Principle</strong></th>
<th><strong>Limitations</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Image stabilization</td>
<td>Try to stabilize camera movement across frames</td>
<td>Process intensive; susceptible to fast movement</td>
</tr>
<tr>
<td>Optical flow</td>
<td>Track large number of points to infer 2D vector field which models an object's movement</td>
<td>Process intensive, requires high frame rate video or slow/smooth-moving objects</td>
</tr>
<tr>
<td>Template/shape-based</td>
<td>Try to learn visual appearance of object</td>
<td>Requires setup; appearances may be highly application-dependent</td>
</tr>
<tr>
<td>Multi-camera</td>
<td>Use multiple cameras to overcome occlusion</td>
<td>Greatly increases amount of data to process</td>
</tr>
<tr>
<td>Kernel-based</td>
<td>Use a fitness function and some search strategy to estimate object location</td>
<td>Requires large number of particles to overcome fast-moving objects</td>
</tr>
<tr>
<td>Swarm intelligence</td>
<td>Evolutionary algorithms whose search strategy is modeled after the collective behavior of biological swarms</td>
<td>Same as kernel-based; tracking performance dependent on particular swarm’s behavior</td>
</tr>
</tbody>
</table>
Bacterial Foraging Optimization (BFO)
[Passino, 2002]

• Swarm intelligence algorithm which mimics behavior of E.coli bacteria

• Populates a search space with agents searching for areas of higher and higher fitness
Pros and Cons

• Bacterial Foraging Optimization
  – Pros:
    • robust to partial and full occlusion
    • robust to non-stationary video
  – Cons:
    • not suited for fast-moving objects
    • lots of parameters to configure (over 7)
    • potentially high computational cost
Bacteria Swarm’s Behavior

Example: Searching for a red object
The Classic Algorithm

- Randomly initialize \( n \) agents on the image
  - For each frame, do \( k \) reproduction steps:
    - Do \( j \) chemotactic steps:
      - For each agent \( i \), do this:
        - Evaluate fitness function at current location
        - Choose a random direction
        - For up to \( N_s \) times for this agent:
          » Swim forward in a step of size \( C \) pixels
          » Evaluate new fitness
          » If new fitness is worse than old fitness, stop swimming
      - Sort agents by current fitness
      - Relocate \( S_r \) worst agents to position of \( S_r \) top agents
  - Dispersal: randomly relocate agents with a \( p_{ed} \% \) probability to a new random position in the image
Observation I

• Agents move 1 step forward and then evaluate, continuing if fitness stays constant or gets better, or stopping once it gets worse.

• Why wait until you have already moved to a location of worse fitness before you stop?
Observation II

• In the same frame, all agents move at every reproduction step, including the top agents of the previous iteration

• What happens if we let some of them stay put?
Observation III

• Even if an object stops moving or does not move very far across frames, we continue to do a full search on every frame

• Can we take advantage of the fact that objects may stop moving, at least between two frames?
Improvements for Tracking

• Lookahead
  – Instead of moving and then evaluating, look ahead and evaluate fitness of potential location, then decide if you want to move there or not

• Elitism
  – Grant the $S_r$ agents with the best fitness after each reproduction exception from further movements for this frame, as well as immunity from dispersal

• Early termination
  – For each frame, check if the previous location is “good enough” before starting a full search of this image (fitness exceeds some dynamic threshold $T$)
Proposed Algorithm

• If previous frame’s best agent ≥ threshold $T$, skip frame
• For each frame, do $k$ reproduction steps:
  – Do $j$ chemotactic steps:
    • For each agent $i$ which is not immune, do this:
      – Evaluate fitness function at current location
      – Choose a random direction
      – For up to $N_s$ times for this agent:
        » Evaluate new fitness $c$ pixels ahead
        » If new fitness is better, swim there, else stop swimming
      – Sort agents by current fitness
      – Relocate $S_r$ worst agents to position of $S_r$ top agents
    – Grant top $S_r$ agents of this reproduction with immunity
• Dispersal: randomly relocate agents which are not immune with a $p_{ed}$% probability to a new random position in the image
• Reset immunity of all agents
Parameters

• Classic BFO:
  – 50 agents/object, 10 reproductions/frame, 1 chemotactic step/reproduction, 1 relocation/reproduction, 1% dispersal probability, 2px step size

• Proposed BFO:
  – Same as in Classic BFO, except 50% dispersal probability, 10px step size

• Particle Swarm Optimization:
  – 2048 agents/object, 1% decay of previous best swarm score
Dataset

- Three videos, 30 sec. each x 30 Frames/s, 704×480
- **Video I**: 3 moving objects in stationary video
- **Video II**: 4 moving objects in stationary video
- **Video III**: 4 moving objects in a non-stationary video
Swarm’s Behavior in Fitness Space

- 50 agents
- 10px step size
- 704×480 image
- Fitness function = 9×9 color histogram

darker = lower fitness,  brighter = higher fitness
Impact of each Improvement on Tracking Accuracy (Video III)

- Classic BFO: 29.67%
- Lookahead only: 46.81%
- Elitism only: 73.81%
- Early Termination only: 76.86%
- with all 3: 96.06%
- 20.00% (higher is better)
Speed Impact of each Improvement (Video III)

<table>
<thead>
<tr>
<th>Improvement</th>
<th>Speed per Second (higher is better)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classic BFO</td>
<td>36.04</td>
</tr>
<tr>
<td>Lookahead only</td>
<td>48.48 (+34%)</td>
</tr>
<tr>
<td>Elitism only</td>
<td>39.88 (+10%)</td>
</tr>
<tr>
<td>Early Termination only</td>
<td>44.83 (+24%)</td>
</tr>
<tr>
<td>with all 3</td>
<td>57.49 (+59%)</td>
</tr>
</tbody>
</table>
Track Accuracy - Video I

Track Accuracy (higher is better)

- Exhaustive Search
- CamShift
- CamShift w/ full window
- PSO
- Classic BFO
- Proposed BFO

Accuracy:
- Exhaustive Search: 100.00%
- CamShift: 63.45%
- CamShift w/ full window: 48.11%
- PSO: 75.25%
- Classic BFO: 87.18%
- Proposed BFO: 93.33%
Track Accuracy - Video III

Exhaustive Search
CamShift
CamShift w/ full window
PSO
Classic BFO
Proposed BFO

Track Accuracy (higher is better)

100.00%
98.69%
96.66%
98.45%
86.10%
89.13%
90.11%
65.69%
63.01%
49.18%
38.88%
14.78%
2.99%
Speed Comparison (Frames/s)

Frames per Second (higher is better)

<table>
<thead>
<tr>
<th>Method</th>
<th>Video I</th>
<th>Video II</th>
<th>Video III</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Processing</td>
<td>81.51</td>
<td>84.48</td>
<td></td>
</tr>
<tr>
<td>Exhaustive Search</td>
<td>0.51</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>CamShift</td>
<td>62.25</td>
<td>62.30</td>
<td></td>
</tr>
<tr>
<td>CamShift w/ full window</td>
<td>56.8</td>
<td>57.14</td>
<td></td>
</tr>
<tr>
<td>PSO</td>
<td>47.42</td>
<td>39.68</td>
<td></td>
</tr>
<tr>
<td>Classic BFO</td>
<td>33.5</td>
<td>34.69</td>
<td></td>
</tr>
<tr>
<td>Proposed BFO</td>
<td>64.33</td>
<td>53.80</td>
<td></td>
</tr>
</tbody>
</table>
Conclusions:
Proposed BFO vs. Classic BFO

• Capable of 5 - 7% higher tracking accuracy over classic BFO

• Capable of 28 - 74% faster performance than classic BFO

• Changes are easy to implement
Conclusions:
Proposed BFO vs. Other Current Techniques

• Capable of 1-8% higher tracking accuracy over PSO using similar resources

• Successfully tracks through occlusion and non-stationary video (93 - 94% track accuracy versus 14 - 57% of CamShift)

• Linear scaling with multiple objects (initialize one swarm per object)
Future Research

• **Adaptive step sizes** (e.g., automatically adapt to fitness landscape and density of objects in video)

• **Human tracking in multiple cameras**