Multiple target detection in video using quadratic multi-frame correlation filtering

Ryan Kerekes
Oak Ridge National Laboratory

B. V. K. Vijaya Kumar
Carnegie Mellon University

March 17, 2008
Outline

PART I
• Correlation filtering overview
• Database

PART II
• Multi-frame correlation theory
• Multi-frame simulation results
Why use correlation filters?

- **Shift invariance**: useful when we don’t know:
  - *Target locations* in the scene
  - *Number of targets* in the scene

- **Distortion tolerance**: CF’s can be trained to tolerate some distortion and reject false targets

- **Graceful degradation**: noise, occlusion
Correlation peak metric

- Often we will measure peak sharpness
  - Peak-to-sidelobe ratio (PSR)
Correlation peak metric

- Often we will measure peak sharpness
  - Peak-to-sidelobe ratio (PSR) metric
  - Invariance to overall image brightness

\[
PSR \triangleq \frac{\text{peak} - \mu}{\sigma}
\]

\[PSR = 7.46\]
Correlation filter usage

- Typically computed in the **frequency domain**
  - 2 FFTs
  - complexity reduction $O\left(n^4\right) \rightarrow O\left(n^2 \log n\right)$
- PSR computed at every point in frequency domain
  - 4 more FFTs
Linear filter vs. quadratic filter

- Consider **thresholding** the output
  - results in a **discriminant** at each shift

![Diagram showing linear filter vs. quadratic filter](image)

- Example: elliptical boundary

- **Linear filter**
- **Quadratic filter**
Turntable database

- Captured rotational imagery of three targets
- Depression angles used: 17°, 19°, 21°, 23°
- Green background used to aid in segmentation
Filter training

- Each filter trained on \textit{scaled} and \textit{out-of-plane-rotated} images of true- and false-class targets
Filter comparisons

- Showed that **RQQCF filters** outperform others
  - each filter bank trained to divide up 360 azimuth range
  - different number of filters/bank, equal computation

**LEGEND**

<table>
<thead>
<tr>
<th>Filter Design</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EOTSDF</td>
<td>Eigen-filter</td>
</tr>
<tr>
<td>UOTSDF</td>
<td>Unconstrained OTSDF</td>
</tr>
<tr>
<td>EMACH</td>
<td>Extended MACH</td>
</tr>
<tr>
<td>CPCF/UPCF</td>
<td>Polynomial CFs</td>
</tr>
<tr>
<td>RQQCF/SSQSDF</td>
<td>Quadratic filters</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1×MINACE</th>
<th>12×EOTSDF</th>
<th>12×MACH</th>
<th>10×RPCF</th>
<th>1×RQQCF (6)</th>
<th>2×RQQCF (3)</th>
<th>3×RQQCF (1)</th>
<th>4×RQQCF (1)</th>
<th>8×RQQCF (7)</th>
<th>8×RQQCF (4)</th>
</tr>
</thead>
</table>
PART II

Multi-frame correlation filtering
CF in video sequences

- Three types of approaches to the problem

**TYPE 1: SINGLE-FRAME**

Test images → CF → Correlation planes → THRESHOLD → Detections
CF in video sequences

- Three types of approaches to the problem

**TYPE 2: DETECTION-FIRST (CLASSICAL TRACKING)**
CF in video sequences

- Three types of approaches to the problem

**TYPE 2: DETECTION-FIRST (CLASSICAL TRACKING)**

- Information loss due to thresholding step
- Data association problem due to false alarms

...
CF’s in video sequences

- Three types of approaches to the problem

**TYPE 3: RELATION-THEN-DETECTION (our work)**
Efficient output combination

• We want to combine outputs while preserving information
  – Avoid preliminary thresholding step
  – Avoid assumptions on number of targets
  – Avoid a large computational increase

• We have developed a probabilistic approach
  – We can derive defensible theory
  – We can confidently utilize all available information

\[
\Pr(\text{target} \mid \text{frame1, frame2,} \ldots)
\]
Typical detection schematic

Input video frames → Correlation filter(s) → Output arrays → MFCF algorithm → THRESHOLD → tracking → recognition

point detections
Probability mapping

- We map each correlation value to **probability of target presence using all available information**

PSR array

- high likelihood of target (PSR ≈ 10)
- reasonable likelihood of target (PSR ≈ 4)

probability array
PSR score distributions

- Must assume distributions for **target scores** and **non-target scores** for each filter
  - (we used Gaussians)

**Example from a FLIR sequence**

![Histograms showing target and non-target (clutter) distributions](chart.png)
Target motion model

- Must assume **transition probability** function between adjacent frames
  - can take velocity into account
- Function of **displacement** (position-independent)
  - (we used centered, rotationally symmetric functions)
Posterior probability array

- We can derive a *pixel-wise* mapping function for the posterior probability array.
Mapping function

- Various properties of the mapping function

\[ \pi^i_{j}(x) = \frac{\alpha^i_{j}(x) \prod_{k=1}^{N} p^k_{j}(s_k)}{\sum_{k=0}^{N} \alpha^i_{k}(x) \prod_{l=1}^{N} p^l_{k}(s_l)} \]

<table>
<thead>
<tr>
<th></th>
<th>i-th frame posterior probability array</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pi^i_{j}(x) )</td>
<td></td>
</tr>
<tr>
<td>( \alpha^i_{j}(x) )</td>
<td>i-th frame prior probability array</td>
</tr>
<tr>
<td>( p^l_{k}(s_l) )</td>
<td>score pdf of filter ( l ) on target ( k )</td>
</tr>
</tbody>
</table>

- Uses PSR scores from all filters

- Already normalized

- Frame index

- Sub-class (filter) index

- Uses prior arrays from all filters

- Various properties of the mapping function
Mapping function

- Example of a mapping in a two-class problem
  PSR value $\rightarrow$ **target probability**
Prior probability array

- We can derive the prior probability array for the next frame: based on convolution

- Motion model

- Frame 3 posterior probs.

- Prior formula

- Frame 4 prior probs.

“predictor” for Frame 4

“knowledge” from Frames 1, 2, and 3
Prior array formula

- Various properties of the prior array formula

\[
\alpha_{j}^{i+1}(x) \approx \exp \left\{ - \sum_{k=1}^{N} \left[ \tau_{k} \times (1 - \delta) \pi_{k}^{i}(x) \right] \right\} \\
\times \left\{ 1 - \varepsilon(x) \right\} \\
+ \varepsilon(x) \cup_{N+1,j}
\]

- Probability of target emergence
- Non-zero probability region
- Zero probability region

| \( \alpha_{j}^{i}(x) \) | \( i \)-th frame posterior array |
| \( \alpha_{j}^{i}(x) \) | \( i \)-th frame prior array |
| \( \tau_{k}(x) \) | Motion model function |
| \( \varepsilon(x) \), \( \delta(x) \) | Emergence/disappearance probability functions |
| \( \nu_{k,j} \) | Subclass transition probs. |
Prior array formula

• Various properties of the prior array formula

\[ \alpha_{j}^{i+1}(x) \approx \exp \left\{ - \sum_{k=1}^{N} [\tau_k \ast (1 - \delta) \pi_k^i](x) \right\} \]

| \( \pi_{k}^{i}(x) \) | \( i \)-th frame posterior array |
| \( \alpha_{j}^{i}(x) \) | \( i \)-th frame prior array |
| \( \tau_{k}(x) \) | Motion model function |
| \( \epsilon(x), \delta(x) \) | Emergence/disappearance probability functions |
| \( \nu_{k,j} \) | Subclass transition probs. |
Prior array formula

- Various properties of the prior array formula

\[ \alpha_{i+1}^j (x) \approx \exp \left\{ - \sum_{k=1}^{N} \left[ \tau_k \ast (1 - \delta) \pi_k^i \right] (x) \right\} \]

only **one convolution** per filter

\[
\begin{align*}
\tau_k & \quad (1 - \delta) \pi_k^i \\
\alpha_i^j (x) & \quad i\text{-th frame posterior array} \\
\alpha_0^j (x) & \quad i\text{-th frame prior array} \\
\tau_k (x) & \quad \text{Motion model function} \\
\epsilon(x), \delta(x) & \quad \text{Emergence/disappearance probability functions} \\
\nu_{k,j} & \quad \text{Subclass transition probs.}
\end{align*}
\]
Prior array formula

The prior array formula uses posterior arrays from all "filter communication" subclass transition probability

\[
Pr\left(Class_j \rightarrow Class_k\right) = \sum_{i=1}^{N} \left[ \tau_k \ast (1 - \delta) \pi^i_k \right] (x) \\
\times \left[ 1 - \epsilon(x) \right] \sum_{k=1}^{N} \nu_{k,j} \left[ \tau_k \ast (1 - \delta) \pi^i_k \right] (x) \\
+ \epsilon(x) \nu_{N+1,j}
\]

Table:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\pi^i_j(x)</td>
<td>i-th frame posterior array</td>
</tr>
<tr>
<td>\alpha^i_j(x)</td>
<td>i-th frame prior array</td>
</tr>
<tr>
<td>\tau_k(x)</td>
<td>Motion model function</td>
</tr>
<tr>
<td>\epsilon(x), \delta(x)</td>
<td>Emergence/disappearance probability functions</td>
</tr>
<tr>
<td>\nu_{k,j}</td>
<td>Subclass transition probs.</td>
</tr>
</tbody>
</table>
Multi-frame schematic

Frame $i$ priors → map → Frame $i$ → Motion model → Frame $i+1$ priors → prior formula → Frame $i$ posterior probabilities → THRESHOLD → detections

(to next iteration)
Multi-frame schematic

- Frame $i$ priors
- Map
- Frame $i+1$ priors
- Motion model
- Big formula
- Table lookup
- 2 FFTs
- Frame $i$ posterior probabilities
- Threshold
- Detections

(to next iteration)
Synthetic sequence demo

![Sequence Image](image1.png)

- **Sequence**
- **Original probability array**
- **Prior probability array**
- **Posterior probability array**

- **PSR values**
- \[ \log_{10} \left( \frac{p}{1-p} \right) \]

31
Synthetic sequence demo

- MFCF can handle **multiple targets**
  - Example: sequence with 3 true-class targets, 3 false-class
Synthetic sequence demo

- False alarms reduced in target and non-target areas
  @ 90% detection rate
FLIR sequence results

- Two FLIR sequences tested
- Two types of noise each
- MFCF offers greater performance improvement in white noise than in compression noise

Sequences L2117

Detection Rate vs. Avg. #FA/frame

- AWGN, SNR=20dB
- H.264 compression noise (65:1)
Noise vs. clutter in MFCF

- MFCF favors filters with **better clutter rejection**
  - Example: sequence *Synthetic3*, noisy (SNR = 20dB)
  - Varied number of eigenvalues ($N_e$) retained in filter design
Summary of main findings

*MFCF algorithm:*

- MFCF is best-suited for handling temporally uncorrelated noise
- **Clutter rejection** should be handled by the filter(s)
- **Filter communication** can help reject false targets
- MFCF degrades gracefully under parameter changes
Future work

- Multi-view correlation filtering (multiple cameras)