A Robust Digital Watermark Using Feature Points and Image Normalization

By

Jon Weinheimer
Watermark Goals

- Robust
  - Watermark withstands a variety of common signal processing attacks.

- Invisible
  - Watermark is imperceptible to the human visual system.

- Blind
  - Detection of watermark without access to the original image.
Inspiration

- Patrick Bas, *et al.* “Geometrically Invariant Watermarking Using Feature Points.”
  - **Pro:** Feature point detector analysis.
  - **Con:** Computationally intensive.

- Chih-Wei Tang, *et al.* “A Feature-Based Robust Digital Image Watermarking Scheme.”
  - **Pro:** Localized use of image normalization.
  - **Con:** Not robust towards large rotations.
Method Components

- Watermark Embedding
- Watermark Detecting

Method Sub-Components

- Feature Point Extraction
- Image Normalization
- Fourier Domain Embedding
Feature Point Extraction

Purpose – Provide stable reference points for both embedding and detecting.

Detectors
- Mexican Hat Wavelet
- Achard-Rouquet Detector
- SUSAN Detector
- Harris Corner Detector
Detector Analysis

Results of Patrick Bas, *et al.*
- 5x5 averaging pre-filter aides robustness.
- Harris detectors scores highest for a variety of images.
Harris Corner Detector

Usual Response Function:

\[ R = \text{Det}(M) - k \ast \text{Trace}^2(M) \]

where

\[ M = \begin{pmatrix}
\frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\
\frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2}
\end{pmatrix} \]

and \( k \) is an arbitrary constant.
Harris Detector (cont.)

Chosen Response Function:

\[ R = \left( \frac{\partial^2 I}{\partial x^2} \right) \left( \frac{\partial^2 I}{\partial y^2} \right) + \left( \frac{\partial^2 I}{\partial x^2} \right) - \left( \frac{\partial^2 I}{\partial x \partial y} \right)^2 \]
Feature Point Extraction

- Selection Criterion
  - Response function threshold.
  - Local maximum.
Image Normalization

Purpose – Transform an image into its normal form to compensate for rotation, skewing, and scaling attacks.

Process – Rotation and normalization transformation matrices are computed using the central moments of the image.

\[
\begin{pmatrix}
    x' \\
    y'
\end{pmatrix} = \begin{pmatrix}
    \cos \alpha & \sin \alpha \\
    -\sin \alpha & \cos \alpha
\end{pmatrix} \begin{pmatrix}
    \frac{c}{\lambda_1} & 0 \\
    0 & \frac{c}{\lambda_2}
\end{pmatrix} \begin{pmatrix}
    e_{1x} & e_{1y} \\
    -e_{1y} & e_{1x}
\end{pmatrix} \begin{pmatrix}
    x - C_x \\
    y - C_y
\end{pmatrix}
\]

Image Normalization (cont.)

Normalization

Normalization
Image Normalization (cont.)

Inverse Normalization

Inverse Normalization
Fourier Domain Embedding

Purpose – Fourier domain embedding uses middle frequencies to resist common signal processing attacks like JPEG compression.

Process – The magnitudes of pairs of frequencies are altered so that their difference represents a single bit of the watermark.
Fourier Domain Embedding (cont.)

A middle frequency $F_A$ is chosen and is paired with frequency $F_B$ 90° apart on the upper half of the DFT plane. Their magnitudes are altered in the following way according to the watermark strength $\alpha$:

For a watermark bit of 1:

$$F_A' = F_A + \frac{1}{2} \left[ \alpha - (F_A - F_B) \right]$$

$$F_B' = F_B - \frac{1}{2} \left[ \alpha - (F_B - F_A) \right]$$

$$F_A' - F_B' = \alpha$$

For a watermark bit of 0:

$$F_A' = F_A - \frac{1}{2} \left[ \alpha - (F_A - F_B) \right]$$

$$F_B' = F_B + \frac{1}{2} \left[ \alpha - (F_B - F_A) \right]$$

$$F_A' - F_B' = -\alpha$$
Watermark Embedding Process

Original Image

Extract Feature Points

Compute Normalization of Disks

Mean and Angle for Each Disk

Transform Selected Point back to Original Image

Select 32x32 Blocks in Each Disk

2-D FFT of Blocks

Watermarked Image

Insert the Watermarked Blocks into Watermarked Image

2-D IFFT of Blocks

Watermark Embedding

Secret Key K
Watermark Embedding Process

1. Extract Feature Points
2. Compute Normalization of Disks
3. Transform Selected Point back to Original Image
4. Select 32x32 Blocks in Each Disk
5. 2-D FFT of Blocks
6. Watermark Embedding
7. Secret Key K
8. 2-D IFFT of Blocks
9. Insert the Watermarked Blocks into Watermarked Image
10. Mean and Angle for Each Disk
11. Original Image
12. Watermarked Image

Inverse Normalization
Watermark Embedding Process

1. Original Image
2. Extract Feature Points
3. Compute Normalization of Disks
4. Mean and Angle for Each Disk
5. Transform Selected Point back to Original Image
6. Select 32x32 Blocks in Each Disk
7. 2-D FFT of Blocks
8. Insert the Watermarked Blocks into Watermarked Image
9. 2-D IFFT of Blocks
10. Watermark Embedding
11. Secret Key K
Watermark Detecting Process

1. Attacked Image
   - Extract Feature Points
2. Compute Normalization of Disks
3. Transform Selected Point back to Original Image
4. Correlate Disks and Compute Restoration Angle
5. Rotate Image Based on Restoration Angle
6. Mean and Angle for Each Disk
7. Mean and Angle for Original Disks
8. 2-D FFT of Blocks
9. Select 32x32 Blocks in Each Disk
10. Extract New Feature Points
11. Detect Watermark
12. Watermark Detection
13. Secret Key K
Watermark Detection Process (cont.)

Inverse Normalization

Inverse Normalization
Watermark Detection Process (cont.)

- Restoring angle for each disk is calculated:
  \[ \beta = A - A' \]
- For \( n \) correlated disks, the angles are scored:
  \[
  \text{Score}(\beta_i) = \sum_{j=1}^{n} f(\beta_i, \beta_j)
  \]
  where
  \[
  f(\beta_i, \beta_j) = \begin{cases} 
  1 & \text{if } |\beta_i - \beta_j| < .01 \\
  0 & \text{if } |\beta_i - \beta_j| \geq .01
  \end{cases}
  \]
Watermark Detecting Process

1. Attacked Image
   - Extract Feature Points

2. Compute Normalization of Disks
   - Transform Selected Point back to Original Image

3. Correlate Disks and Compute Restoration Angle
   - Rotate Image Based on Restoration Angle

4. Select 32x32 Blocks in Each Disk
   - Extract New Feature Points

5. 2-D FFT of Blocks

6. Watermark Detection
   - Detected Watermark

7. Secret Key K
Results

Watermarked Image

Difference Image
<table>
<thead>
<tr>
<th>Attack</th>
<th>Lena</th>
<th>Peppers</th>
<th>Plane</th>
<th>Baboon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watermarked Image</td>
<td>10/10</td>
<td>11/11</td>
<td>12/12</td>
<td>12/12</td>
</tr>
<tr>
<td>Rotate 1°</td>
<td>9/10</td>
<td>9/11</td>
<td>12/12</td>
<td>11/12</td>
</tr>
<tr>
<td>Rotate 2°</td>
<td>1/10</td>
<td>10/11</td>
<td>2/12</td>
<td>12/12</td>
</tr>
<tr>
<td>Rotate 4°</td>
<td>7/10</td>
<td>10/11</td>
<td>12/12</td>
<td>9/12</td>
</tr>
<tr>
<td>Rotate 5°</td>
<td>7/10</td>
<td>8/11</td>
<td>4/12</td>
<td>0/12</td>
</tr>
<tr>
<td>Rotate 10°</td>
<td>4/10</td>
<td>0/11</td>
<td>11/12</td>
<td>8/12</td>
</tr>
<tr>
<td>Rotate 15°</td>
<td>0/10</td>
<td>6/11</td>
<td>1/12</td>
<td>6/12</td>
</tr>
<tr>
<td>Rotate 20°</td>
<td>2/10</td>
<td>0/11</td>
<td>8/12</td>
<td>5/12</td>
</tr>
<tr>
<td>Rotate 30°</td>
<td>2/10</td>
<td>2/11</td>
<td>2/12</td>
<td>5/12</td>
</tr>
<tr>
<td>Rotate 60°</td>
<td>2/10</td>
<td>2/11</td>
<td>6/12</td>
<td>2/12</td>
</tr>
<tr>
<td>Rotate 90°</td>
<td>10/10</td>
<td>3/11</td>
<td>12/12</td>
<td>12/12</td>
</tr>
<tr>
<td>Median Filter 2x2</td>
<td>6/10</td>
<td>0/11</td>
<td>12/12</td>
<td>8/12</td>
</tr>
<tr>
<td>Median Filter 3x3</td>
<td>6/10</td>
<td>10/11</td>
<td>11/12</td>
<td>4/12</td>
</tr>
<tr>
<td>Sharpening 3x3</td>
<td>6/10</td>
<td>9/11</td>
<td>8/12</td>
<td>4/12</td>
</tr>
<tr>
<td>Gaussian Filter 3x3</td>
<td>8/10</td>
<td>8/11</td>
<td>5/12</td>
<td>7/12</td>
</tr>
<tr>
<td>JPEG 90</td>
<td>9/10</td>
<td>9/11</td>
<td>11/12</td>
<td>12/12</td>
</tr>
<tr>
<td>JPEG 80</td>
<td>2/10</td>
<td>2/11</td>
<td>6/12</td>
<td>7/12</td>
</tr>
<tr>
<td>Scaling .9</td>
<td>3/10</td>
<td>6/11</td>
<td>6/12</td>
<td>1/12</td>
</tr>
<tr>
<td>Cropping 5%</td>
<td>8/10</td>
<td>6/11</td>
<td>6/12</td>
<td>11/12</td>
</tr>
<tr>
<td>Rotate 1° + Crop 5%</td>
<td>2/10</td>
<td>4/11</td>
<td>5/12</td>
<td>2/12</td>
</tr>
<tr>
<td>Rotate 5° + Crop 5%</td>
<td>1/10</td>
<td>0/11</td>
<td>3/12</td>
<td>1/12</td>
</tr>
<tr>
<td>Rotate 30° + Crop 5%</td>
<td>3/10</td>
<td>1/11</td>
<td>1/12</td>
<td>2/12</td>
</tr>
<tr>
<td>Rotate 60° + Crop 5%</td>
<td>3/10</td>
<td>0/11</td>
<td>2/12</td>
<td>2/12</td>
</tr>
<tr>
<td>Rotate 90° + Crop 5%</td>
<td>8/10</td>
<td>0/11</td>
<td>5/12</td>
<td>8/12</td>
</tr>
<tr>
<td>Median 2x2+JPEG 90</td>
<td>3/10</td>
<td>0/11</td>
<td>8/12</td>
<td>7/12</td>
</tr>
<tr>
<td>Median 3x3+JPEG 90</td>
<td>4/10</td>
<td>5/11</td>
<td>8/12</td>
<td>3/12</td>
</tr>
<tr>
<td>Sharp 3x3 + JPEG 90</td>
<td>4/10</td>
<td>9/11</td>
<td>8/12</td>
<td>2/12</td>
</tr>
</tbody>
</table>
Conclusions

- Robust?
  - The proposed method is quite robust to a large variety of common attacks. However, there is always room for improvement.

- Invisible?
  - If an appropriate value for the watermark strength is applied, the image is imperceptible to the human visual system.

- Blind?
  - The method does not need the actual original image, but it does need to be provided with information about the original (i.e. the means and angles of the original disks).