Segmenting Video Into Classes of Algorithm-Suitability

Oisin Mac Aodha (UCL)
Gabriel Brostow (UCL)
Marc Pollefeys (ETH)
Which algorithm should I (use / download / implement) to track things in this video?
The Optical Flow Problem

- #2 all-time Computer Vision problem (disputable)
- “Where did each pixel go?”
Optical Flow Solutions

- Compared against each other on the “blind” Middlebury test set

<table>
<thead>
<tr>
<th>Optical flow evaluation results</th>
<th>Statistics: Average</th>
<th>SD</th>
<th>R0.5</th>
<th>R1.0</th>
<th>R2.0</th>
<th>A50</th>
<th>A75</th>
<th>A95</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error type:</td>
<td>angle interpolation</td>
<td>normalized interpolation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adapative [23]</td>
<td>avg. rank</td>
<td>Army (Hidden texture)</td>
<td>Mequon (Hidden texture)</td>
<td>Schefflera (Hidden texture)</td>
<td>Wooden (Hidden texture)</td>
<td>Grove (Synthetic)</td>
<td>Urban (Synthetic)</td>
<td>Yosemite (Synthetic)</td>
</tr>
<tr>
<td>avg.</td>
<td>5.2</td>
<td>0.09</td>
<td>0.26</td>
<td>0.06</td>
<td>0.23</td>
<td>0.78</td>
<td>0.18</td>
<td>0.54</td>
</tr>
<tr>
<td>Articulated OF [24]</td>
<td>6.6</td>
<td>0.11</td>
<td>0.28</td>
<td>0.10</td>
<td>0.18</td>
<td>0.73</td>
<td>0.12</td>
<td>0.31</td>
</tr>
<tr>
<td>Huber-L1 [25]</td>
<td>7.0</td>
<td>0.10</td>
<td>0.28</td>
<td>0.08</td>
<td>0.31</td>
<td>0.88</td>
<td>0.15</td>
<td>0.56</td>
</tr>
</tbody>
</table>
1st Best Algorithm (7th overall as of 17-12-2009)

(3rd overall as of 17-12-2009) 2nd Best Algorithm

Classic+Area [31]: Anonymous. Secrets of optical flow estimation and their principles. CVPR 2010 submission 477
Should use algorithm A

Should use algorithm B
(Artistic version; object-boundaries don’t interest us)
Hypothesis:

- that the most suitable algorithm can be chosen for each video automatically, through supervised training of a classifier
Hypothesis:

- that the most suitable algorithm can be chosen for each video automatically, through supervised training of a classifier.

- that one can predict the space-time segments of the video that are best-served by each available algorithm.
  - (Can always come back to choose a per-frame or per-video algorithm.)
Experimental Framework

Image data → Feature extraction → Learning algorithm → Trained classifier → Estimated class labels

- Groundtruth labels
Experimental Framework

- Image data
- Feature extraction
- Learning algorithm
- Groundtruth labels
- Trained classifier
- Estimated class labels
- New test data
Experimental Framework

Image data → Feature extraction → Learning algorithm

Learning algorithm → Trained classifier

Estimated class labels

Random Forests

Breiman, 2001
Experimental Framework
“Making” more data
Formulation

\[ \mathcal{D} = \left\{ (x_i, c_i) \middle| x_i \in \mathbb{R}^d, c_i \in \mathbb{Z}^k \right\}_{i=1}^{n} \]

- Training data \( \mathcal{D} \) consists of feature vectors \( x \) and class labels \( c \) (i.e. best-algorithm per pixel)

- Feature vector \( x \) is multi-scale, and includes:
  - Spatial Gradient
  - Distance Transform
  - Temporal Gradient
  - Residual Error (after bicubic reconstruction)
Training data $\mathcal{D}$ consists of feature vectors $\mathbf{x}_i$ and class labels $c_i$ (i.e., best algorithm per pixel).

Feature vector $\mathbf{x}$ is multi-scale, and includes:
- Spatial Gradient
- Distance Transform
- Temporal Gradient
- Residual Error (after bicubic reconstruction)

The formulation is:

$$\mathcal{D} = \left\{ \left( \mathbf{x}_i, c_i \right) \mid \mathbf{x}_i \in \mathbb{R}^d, c_i \in \mathbb{Z}^k \right\}_{i=1}^n$$

$$\mathbf{x}_i = \{g(x, y, [1, z]), d(x, y, [1, z]), t_x(x, y, [1, z]), t_y(x, y, [1, z]), r(x, y, [1, k])\}$$

$$g(x, y, z) = ||\nabla I_1||$$
Training data \( \mathcal{D} \) consists of feature vectors \( x \) and class labels \( c \) (i.e. best algorithm per pixel).

Feature vector \( x \) is multi-scale, and includes:
- Spatial Gradient
- Distance Transform
- Temporal Gradient
- Residual Error (after bicubic reconstruction)

Formulation:

\[
\mathcal{D} = \{(x_i, c_i) \mid x_i \in \mathbb{R}^d, c_i \in \mathbb{Z}^k\}_{i=1}^n
\]

\[
x_i = \{g(x, y, [1, z]), d(x, y, [1, z]), t_x(x, y, [1, z]), t_y(x, y, [1, z]), r(x, y, [1, k])\}
\]

\[
d(x, y, z) = \text{disTrans}(\|\nabla I_1\| > \tau)
\]
Formulation Details

- Temporal Gradient

\[ t_x = \| \nabla (x + \bar{u}) \| \]
\[ t_y = \| \nabla (y + \bar{v}) \| \]

- Residual Error

\[ r_i(x, y, k) = I_1(x, y) - bicubic(I_2(x + u_i(k), y + v_i(k))) \]
Application I: Optical Flow
<table>
<thead>
<tr>
<th>Image Sequence</th>
<th>BA</th>
<th>TV</th>
<th>HS</th>
<th>FL</th>
<th>TrivComb</th>
<th>Ours</th>
<th>OptCombo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.344</td>
<td>0.271</td>
</tr>
<tr>
<td>FL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.527</td>
<td>0.398</td>
</tr>
<tr>
<td>HS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.435</td>
<td>0.256</td>
</tr>
<tr>
<td>TV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.096</td>
<td>0.074</td>
</tr>
<tr>
<td>08</td>
<td>0.549</td>
<td></td>
<td>0.350</td>
<td></td>
<td>0.387</td>
<td>0.603</td>
<td>0.398</td>
</tr>
<tr>
<td>32</td>
<td>1.329</td>
<td></td>
<td>0.527</td>
<td></td>
<td>0.589</td>
<td>0.603</td>
<td>0.398</td>
</tr>
<tr>
<td>06</td>
<td>0.772</td>
<td></td>
<td>0.435</td>
<td></td>
<td>0.428</td>
<td>0.436</td>
<td>0.256</td>
</tr>
<tr>
<td>35</td>
<td>0.182</td>
<td></td>
<td>0.096</td>
<td></td>
<td>0.133</td>
<td>0.097</td>
<td>0.074</td>
</tr>
<tr>
<td>96</td>
<td>0.276</td>
<td></td>
<td>0.164</td>
<td></td>
<td>0.191</td>
<td>0.165</td>
<td>0.123</td>
</tr>
<tr>
<td>45</td>
<td>0.873</td>
<td></td>
<td>0.622</td>
<td></td>
<td>0.670</td>
<td>0.628</td>
<td>0.466</td>
</tr>
<tr>
<td>20</td>
<td>0.285</td>
<td></td>
<td>0.170</td>
<td></td>
<td>0.189</td>
<td>0.171</td>
<td>0.111</td>
</tr>
<tr>
<td>11</td>
<td>0.171</td>
<td></td>
<td>0.144</td>
<td></td>
<td>0.147</td>
<td>0.147</td>
<td>0.115</td>
</tr>
<tr>
<td>64</td>
<td>5.195</td>
<td></td>
<td>3.724</td>
<td></td>
<td>4.582</td>
<td>3.748</td>
<td>2.448</td>
</tr>
<tr>
<td>15</td>
<td>17.891</td>
<td></td>
<td>12.634</td>
<td></td>
<td>17.634</td>
<td>9.607</td>
<td>3.373</td>
</tr>
<tr>
<td>31</td>
<td>2.702</td>
<td></td>
<td>0.709</td>
<td></td>
<td>2.177</td>
<td>1.833</td>
<td>0.443</td>
</tr>
<tr>
<td>42</td>
<td>0.876</td>
<td></td>
<td>0.344</td>
<td></td>
<td>0.389</td>
<td>0.342</td>
<td>0.212</td>
</tr>
<tr>
<td>10</td>
<td>0.267</td>
<td></td>
<td>0.250</td>
<td></td>
<td>0.221</td>
<td>0.252</td>
<td>0.186</td>
</tr>
<tr>
<td>76</td>
<td>0.683</td>
<td></td>
<td>0.403</td>
<td></td>
<td>0.474</td>
<td>0.437</td>
<td>0.368</td>
</tr>
<tr>
<td>59</td>
<td>8.443</td>
<td></td>
<td>8.607</td>
<td></td>
<td>8.52</td>
<td>8.612</td>
<td>8.11</td>
</tr>
<tr>
<td>06</td>
<td>1.281</td>
<td></td>
<td>1.021</td>
<td></td>
<td>1.057</td>
<td>0.988</td>
<td>0.762</td>
</tr>
<tr>
<td>96</td>
<td>0.464</td>
<td></td>
<td>0.473</td>
<td></td>
<td>0.456</td>
<td>0.467</td>
<td>0.296</td>
</tr>
</tbody>
</table>
FlowLib Decision Confidence
Application II: Feature Matching
Comparing 2 Descriptions

- What is a match? Details are important...
  - Nearest neighbor (see also FLANN)
  - Distance Ratio
  - PCA
- Evaluation: density, # correct matches, tolerance

“192 correct matches (yellow) and 208 false matches (blue)”
SIFT Decision Confidence

ROC Curve Scene 17

![ROC Curve Image]

Number of Correct Matches

![Number of Matches Image]
Hindsight / Future Work

Current results don’t quite live up to the theory:

- Flaws of best-algorithm are the upper bound (ok)
- Training data does not fit in memory (fixable)
- “Winning” the race is more than rank (problem!)
Summary

- Overall, predictions are correlated with the best algorithm for each segment (expressed as Pr!)

- Training data where one class dominates is dangerous – needs improvement

- Other features could help make better predictions
  - Results don’t yet do the idea justice

- One size does NOT fit all
  - At least in terms of algorithm suitability
  - Could use “bad” algorithms!
FlowLib Based on Prediction

White = 30 pixel end point error