Real-Time Accurate Stereo Matching using Modified Two-Pass Aggregation and Winner-Take-All Guided Dynamic Programming

Xuefeng Chang
Beihang University
2011.5.15
Stereo Algorithms

- Global Methods
  - Minimize a certain energy function
  - Graph-cut, Belief propagation
  - High accuracy, Low speed

- Local Methods
  - Aggregate matching cost in a local support window
  - Winner-Take-All (WTA) Strategy
  - Fast but inaccurate
Locally Adaptive Support-Weight Approach

(Yoon and Kweon CVPR 2005)

- A fixed-size support window with per-pixel varying support weight.
- The support weight is computed based on color similarity and geometric distance to the center pixel of interest.

\[
C_d(p_x, p_y) = \sum_{c \in \{r, g, b\}} |I_c(p_x, p_y) - \tilde{I}_c(p_x + d, p_y)|
\]

\[
S(p, q) = \exp\left(-\left(\frac{\Delta C_{pq}}{r_c} + \frac{\Delta g_{pq}}{r_g}\right)\right)
\]

\[
W_d(p) = \frac{\sum_{p' \in \Omega_p, q' \in \Omega_q} S(p'^i, p)S(q'^i, q)c_d(p_x, p_y)}{\sum_{p' \in \Omega_p, q' \in \Omega_q} S(p'^i, p)c_d(p_x, p_y)}
\]

- Very good results (Avg. Error 6.67%).
- Computationally expensive, off-line.
Real-Time Stereo using Adaptive Weight

(Liang Wang 3dpvt2006)

- The per-pixel matching cost is only aggregated in a one dimensional vertical window.
- The Adaptive weight scheme is integrated into a DP framework.
- Achieves over 50 million disparity evaluations per second (MDE/s) when using the graphics hardware.
- The quality is not quite satisfactory (Avg. Error 9.82%).
Our approach

- **Framework**

  - Maching Cost Computation
  - Two-Pass Aggregation
  - Disparity Computation with WTA guided DP
Our Approach

- Weight computation by color similarity

\[ \Delta C_{pq} = \sqrt{\sum_{c \in \{r,g,b\}} (I_c(p_x, p_y) - I_c(q_x, q_y))^2} \]

\[ S(p, q) = \exp\left(-\frac{\Delta C_{pq}}{r_c}\right) \]

Only color similarity is used to calculate support weight.

Color Only

Color And Distance
Our Approach

- Two-Pass Aggregation

Simplifies the matching cost aggregation process by multi-use of computational results

\[ O(N^2) \rightarrow O(N) \]
Our Approach

- **Accuracy loss during two-pass aggregation**

\[
H^r(u, v, d) = \frac{\sum_{x=-r}^r w(u, v, u + x, v) C(u + x, v, d)}{\sum_{x=-r}^r w(u, v, u + x, v)}
\]

\[
V^r(u, v, d) = \frac{\sum_{y=-r}^r w(u, v, u, v + y) H^r(u, v + y, d)}{\sum_{x=-r}^r w(u, v, u, v + y)}
\]

\[
S'(c, p) = S(c', p) S(c, c')
\]

\[
S'(c, p) = \exp(-\frac{\Delta c'_{cc'} + \Delta c'_{pc}}{r_c})
\]

\[
S(c, p) = \exp(-\frac{\Delta c_{pc}}{r_c})
\]
Our Approach

- We can have a clear view of the difference between $S(c,p)$ and $S'(c,p)$ in Color Space

$$0 \leq \Delta c_{pc'} + \Delta c_{cc'} - \Delta c_{pc} \leq 2 \times \min(\Delta c_{pc'}, \Delta cc')$$

The larger $\Delta c_{pc'}$ and $\Delta cc'$, the larger the possible accuracy loss.
Our Approach

- **Credibility estimation mechanism**

Computes a credibility value for each pixel

\[
R(c, p) = T(\exp(-\frac{\Delta c_{pc}}{K}))T(\exp(-\frac{\Delta c_{cc}}{K}))
\]

\[
T(x) = \begin{cases} 
0, & x < T_{w1} \\
0.5, & x < T_{w2} \\
1, & \text{else}
\end{cases}
\]

Aggregates in a fixed support window\((35 \times 35)\)

Excludes points which may be unreliable from two-pass aggregation.
# Performance comparison of different two-pass aggregation approaches

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Error[%]</th>
<th>Tuskuba nonocc all disc</th>
<th>Venus nonocc all disc</th>
<th>Teddy nonocc all disc</th>
<th>Cones nonocc all disc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Credibility Estimation</td>
<td>8.52</td>
<td>3.10, 4.70, 13.3</td>
<td>1.89, 2.86, 11.0</td>
<td>7.00, 14.0, 16.9</td>
<td>4.22, 12.0, 11.2</td>
</tr>
<tr>
<td>Without Credibility Estimation</td>
<td>6.56</td>
<td>1.40, 3.07, 5.86</td>
<td>0.73, 1.74, 3.86</td>
<td>6.81, 14.0, 15.4</td>
<td>3.99, 11.8, 10.1</td>
</tr>
</tbody>
</table>
Our Approach

- Disparity computation with Scan-line DP

- Energy Function
  \[ E(d) = E_{\text{data}}(d) + E_{\text{smooth}}(d) \]

  \[ E_{\text{smooth}} = \sum_{x=1}^{\text{Width}} |d(x) - d(x-1)| \gamma \]

  \[ E_{\text{data}} = W_d(p) \]

- For \( p' = (p_x - 1, p_y) \), we only need to consider \( d(p)-1, d(p), d(p)+1 \) as disparity smoothness constrain.

  \[ F(p, d(p)) = W_{d(p)}(p) + \min_{d \in [d(p)-1, d(p), d(p)+1]} \{ F(p', d) + \gamma \cdot \text{abs}(d(p) - d) \} \]
Our Approach

- But disparities computed by this method change slowly at depth discontinue areas, and may blur the borders.
- So we adopt a winner-take-all guided dynamic programming.

\[ d_{x-1} = \arg \min_d C'(p', d) \]

\[ F(p, d(p)) = W_{d(p)}(p) + \min_{d \in \{d(p)-1, d(p), d(p)+1, d_{x-1}\}} \{ F(p', d) + \gamma \cdot \text{abs}(d(p) - d) \} \]

DP + WTA → DP+WTA
# Performance comparison of different disparity computation method

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Tuskuba nonocc all disc</th>
<th>Venus nonocc all disc</th>
<th>Teddy nonocc all disc</th>
<th>Cones nonocc all disc</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DP</strong></td>
<td>1.54, 3.30, 6.68</td>
<td>0.79, 1.95, 5.15</td>
<td>6.89, 14.2, 15.6</td>
<td>5.02, 13.1, 13.0</td>
</tr>
<tr>
<td><strong>WTA</strong></td>
<td>3.20, 5.21, 7.04</td>
<td>2.49, 3.93, 9.66</td>
<td>10.3, 18.0, 18.5</td>
<td>5.92, 15.4, 12.2</td>
</tr>
<tr>
<td><strong>DP+WTA</strong></td>
<td>1.40, 3.07, 5.86</td>
<td>0.73, 1.74, 3.86</td>
<td>6.81, 14.0, 15.4</td>
<td>3.99, 11.8, 10.1</td>
</tr>
</tbody>
</table>
Experiment

Resulting depth maps from Middlebury stereo data set

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Error[%, nonocc all disc]</th>
<th>Tuskuba</th>
<th>Venus</th>
<th>Teddy</th>
<th>Cones</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our approach</td>
<td>6.56, 1.40, 3.07, 5.86</td>
<td>0.73, 1.74, 3.86</td>
<td>6.81, 14.0, 15.4</td>
<td>3.99, 11.8, 10.1</td>
<td></td>
</tr>
<tr>
<td>Adaptive Weight</td>
<td>6.67, 1.38, 1.85, 6.90</td>
<td>0.71, 1.19, 6.13</td>
<td>7.88, 13.3, 18.6</td>
<td>3.97, 9.79, 8.26</td>
<td></td>
</tr>
<tr>
<td>RealTimeABW</td>
<td>7.90, 1.26, 1.67, 6.83</td>
<td>0.33, 0.65, 3.56</td>
<td>10.7, 18.3, 23.3</td>
<td>4.81, 12.6, 10.7</td>
<td></td>
</tr>
<tr>
<td>RealTimeBP</td>
<td>7.69, 1.49, 3.40, 7.87</td>
<td>0.77, 1.90, 9.00</td>
<td>7.78, 14.9, 17.3,</td>
<td>4.58, 12.4, 10.7</td>
<td></td>
</tr>
<tr>
<td>RTCensus</td>
<td>9.73, 5.08, 6.25, 19.2</td>
<td>1.58, 2.42, 14.2</td>
<td>7.96, 13.8, 20.3</td>
<td>4.10, 9.54, 12.2</td>
<td></td>
</tr>
<tr>
<td>Real-Time GPU</td>
<td>9.82, 2.05, 4.22, 10.6</td>
<td>1.92, 2.98, 20.3</td>
<td>7.23, 14.4, 17.6</td>
<td>6.41, 13.7, 16.5</td>
<td></td>
</tr>
</tbody>
</table>
Experiment

Experiment on dynamic scene

Speed: 20 fps on 320×240 video with GPU acceleration (disparity search range =24)

Depth maps generated by our proposed stereo algorithm.

Pixel depth is represented by pixel brightness, that is: the greater the pixel depth, the darker it will be.
Thank you!

Q&A