High Throughput Hardware Architecture for Accurate Semi-Global Matching

Yan Li, Chen Yang, Wei Zhong, Zhiwei Li, Song Chen*
School of Information Science and Technology
University of Science and Technology of China (USTC)
January 19, 2017
Outline

• Introduction
• SGM Algorithm for Hardware Implementation
• Proposed Hardware Architecture
• Experimental Results
• Conclusion
Stereo Vision Systems

• Binocular vs. Monocular
  • Depth information by Triangulation
  • Vision distances

• Applications
  • Automobile, Robots, aerospace, etc.

Calibration (offline)  Rectification  Stereo Matching  Triangulation  Image Recognition and Analysis
Stereo Matching - basics

- Two cameras: Left and Right
- Optical centers: \(O_L\) and \(O_R\)
- **Virtual image plane**: projection of actual image plane through optical center
- **Baseline, B**: the separation between the optical centers
- Scene Point: \(P\)
- **Disparity, \(d\)** = \(p_L - p_R\) = 6

How to find the pixel \(p_L\) and its corresponding pixel \(p_R\) accurately and fast?

\[ \text{Depth, } z = \frac{Bf}{pd} \]

\(p = \text{pixel width}\)
Stereo Matching – Challenges & Algorithms

- **Challenges**: Accuracy, Real time, and Low power

- Global matching algorithms:
  - **High accuracy + Low throughput**
  - DP, BP, Graph cuts

- Local matching algorithms:
  - **Lower accuracy + High throughput**
  - SAD: foreground fattening
  - ADSW[Yoon, IEEE TPAMI 2006]: exponential operation, hardware unfriendly
  - GIF[Ttofis, DATE 2014]: poor disparity accuracies in the low-texture and occluded regions
Stereo Matching - Solution

- Semi-global Matching [Hirschmuller, IEEE TPAMI 2008]
  - High Accuracy:
    - Aggregate matching costs along paths (8/16)
    - Make trade-off between accuracy and hardware resources

- Customized Hardware Design
  - Real time processing & Low power consumption
  - Parallel processing
  - Full Pipeline
  - Hardware accelerators (FPGA, ASIC)
## Hardware SGM – Previous works

<table>
<thead>
<tr>
<th>Work</th>
<th>Path num.</th>
<th>Row parallelization</th>
<th>Weight path cost</th>
<th>External memory</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gehrig, IEEE ICVS 2009</td>
<td>8</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>ZSAD</td>
</tr>
<tr>
<td>Banz, IEEE SAMOS 2010</td>
<td>4</td>
<td>Yes</td>
<td>No</td>
<td>-</td>
<td>Rank Transform</td>
</tr>
<tr>
<td>Roszkowski, IEEE DDECS 2014</td>
<td>4</td>
<td>No</td>
<td>Yes</td>
<td>-</td>
<td>Census</td>
</tr>
<tr>
<td>Wang, IEEE TCSVT 2015</td>
<td>4</td>
<td>Yes</td>
<td>No</td>
<td>-</td>
<td>AD-census+ Cost Aggregation</td>
</tr>
<tr>
<td>Proposed</td>
<td>5</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>TAD</td>
</tr>
</tbody>
</table>
Contributions

- **Hardware Architecture**
  - Full pipeline
  - Two-row, path and disparity parallelization
  - High throughput: 197fps, 1280x960, 64 disparity levels

- **Cost aggregation**
  - Five aggregation paths
  - Adaptive weighted path cost aggregation
  - Laplace filter is used to enhance the edge of image

- **Disparity refinement**
  - Spike removal (SPF)
Outline

• Introduction

• SGM Algorithm for Hardware Implementation
  • Pre-processing
  • Initial matching cost volume construction
  • Semi-global cost aggregation
  • Disparity computation
  • Post-processing

• Proposed Hardware Architecture

• Experimental Results

• Conclusion
Stereo Matching

Stereo image pairs

Reference pixel

Stereo correspondence

Cost initialization

Cost aggregation

Disparity computation

Disparity refinement
Pre-processing & Initial matching cost volume construction

- Pre-processing
  - Gaussian filter removes the noise
  - Laplace filter enhances the edge

- Initial matching cost computation
  - The truncated absolute difference of intensity and Sobel gradient are used to compute the pixel-wise cost
    \[ Cost_{initial}(x, y, d) \]
  - The cost volume is constructed as follows:
    \[ M(x, y, d) = |I_l(x, y) - I_r(x, y - d)|, \]
    \[ G(x, y, d) = (|\nabla_x I_l(x, y) - \nabla_x I_r(x, y - d)| \]
    \[ +|\nabla_y I_l(x, y) - \nabla_y I_r(x, y - d)|)/2 \]
    \[ Cost_{initial}(x, y, d) = \min(M(x, y, d), T_{gray}) \]
    \[ +\min(G(x, y, d), T_{grad}) \]
Semi-global cost aggregation

- The $L_r(p, d)$ along a path $r$ of the pixel $p$ at disparity $d$ is computed as follows [Hirschmuller, IEEE TPAMI 2008]:

$$L_r(p, d) = \text{Cost}_{\text{initial}}(p, d) + \min \left( \frac{L_r(p - r, d),}{L_r(p - r, d \pm 1) + P_1}, \min L_r(p - r, k) \right) - \min L_r(p - r, k)$$

$$P_2 = \frac{P_2'}{|I_{bp} - I_{bq}|}, \quad r = \text{ltb}(1,1), \text{tb}(1,0), \text{rtb}(1,-1), \text{lr}(0,1), \text{rl}(0,-1)$$
Semi-global cost aggregation & Disparity computation

- The adaptive weight of path costs: $6 \times 11$ windows
  - $W_r(p) = \sum_{i=0,1,2,\ldots,R} W_{ir}(p - i \cdot r), i = D_{\text{chebyshev}}(p, p - i \cdot r)$,
  - $W_{ir}(p - i \cdot r) = \begin{cases} (R + 1) - |i| & |I(p) - I(p - i \cdot r)| < Th \\ 0 & \text{else} \end{cases}$
  - $S(p, d) = \sum_r W_r(p) \cdot L_r(p, d)$.

- Disparity computation
  - The disparity of each pixel is computed by WTA approach.
  $$D_p = \text{argmin}_{0 \leq d \leq d_{\text{max}} - 1} S(p, d)$$
Post-processing

- Detect the mismatching and occluded pixels by L-R check.
  \[ D_p \begin{cases} D_l & |D_l - D_r| \leq 1 \\ 0 & \text{else} \end{cases} \]

- Remove spikes by SPF
  \[ W_p(x + i, y + j) = \begin{cases} 1 & \text{if } \frac{D_p(x,y) \times (256 - \text{spf}_r)}{256} \leq D_p(x + i, y + j) \leq \frac{D_p(x,y) \times (256 + \text{spf}_r)}{256} \\ 0 & \text{else} \end{cases} \]

- \[ N_p(x, y) = \sum_{-r \leq i \leq r, -r \leq j \leq r} W_p(x + i, y + j) \]

- \[ D_p(x, y) = \begin{cases} D_p(x,y) & \text{if } N_p(x,y) \geq Th \\ 0 & \text{else} \end{cases} \]

- Fill the disparity of invalid pixel
  \[
  \text{valid} \quad \text{min} \quad \text{valid}
  \]

\[\begin{array}{c}
\text{LR-check} \\
\text{SPF} \\
\end{array}\]
Outline

• Introduction

• SGM Algorithm for Hardware Implementation

• Proposed Hardware Architecture
  • Overview
  • Pre-processing module
  • Initial matching cost volume construction
  • Semi-global cost aggregation architecture
  • Post-processing

• Experimental Results

• Conclusion
Overview – Hardware Architecture

• Hardware Architecture

- Fully pipeline
- Disparity, path, and row parallelization
- Throughput: 960p/197fps
- $F_{max}$: 156MHz
- Disparity range: 64/16
- Flexible Resolution
Pre-processing module – Gaussian and Laplace filter

- Two line FIFOs capture pixels from four adjacent rows.
- CONV performs convolution of 3x3 windows consisting of a set of registers with the filter kernels.
Initial matching cost volume construction

- Disparity parallel: 64
- Scanline buffer: FIFO
- Window buffer: 3x3 register
- GCU: Sobel gradient computation
- ICCU: 64x2, initial matching cost computation
Overview - Semi-global cost aggregation architecture

- **Horizontal group**: lr and rl.
- **Vertical group**: ltb, tb and rtb. The path cost computation depends on the path costs in the upper row.

```plaintext
\[ \text{Lr}(p,d) \]
```

Initial cost_odd\((x)\)
\((d=0,1,\ldots,d_{\text{max}}-1)\)

Initial cost_even\((x+1)\)
\((d=0,1,\ldots,d_{\text{max}}-1)\)

Gray Lap_odd\((x)\)

Gray Lap_even\((x+1)\)

HDRIU

VBU

HDROU

Adaptive weighted path cost accumulation

Weight

\[ S_{\text{odd}} \]

\[ S_{\text{even}} \]
Semi-global cost aggregation architecture – Path cost computation

• PCCU(path cost computation)
  • Compare tree
  • Three-stage pipeline
  • fixed point divider (P2): four-stage pipeline

\[ L_r(p, d) = \text{Cost}_{\text{initial}}(p, d) + \min \left( \frac{L_r(p - r, d),}{L_r(p - r, d \pm 1) + P_1}, \frac{\min_k L_r(p - r, k) + P_2}{\text{min}_{k} L_r(p - r, k)} \right) \]

\[ P_2 = \frac{P_2'}{|I_{bp} - I_{bq}|} \]
Semi-global cost aggregation architecture – Horizontal path cost computation

- Horizontal Architecture
  - ping-pong buffer
  - LIFIFO (dual-port RAM), LIFO
Semi-global cost aggregation architecture – Horizontal path cost computation

- Horizontal Architecture
  - Compute three pixels from three different rows in three clock cycles

![Diagram showing the horizontal architecture with PCCU and registers](image)
Post-processing: L-R check, SPF, Subpixel interpolation and Median filter

- Occlusion and mismatching detection: ①
- Spike removal: ②
- Invalid pixel filling: ③
Outline

• Introduction
• SGM Algorithm for Hardware Implementation
• Proposed Hardware Architecture
• Experimental Results
  • Quality of results
  • Hardware performance and consumption
• Conclusion
Experimental results – Quality

- Test bench: Middlebury
- Test images: Tsukuba, Venus, Teddy and Cones
- Average error rate: 6.03%
  - Lower accuracy in Venus test image with large plain area
Experimental results – Quality

- The influence of the path number and adaptive weighted path costs are given in the following table.
- Using one more path and adaptive weighted path cost aggregation improve the accuracy by 3.69%.

<table>
<thead>
<tr>
<th>Variants</th>
<th>Average error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>nonocc</td>
</tr>
<tr>
<td>5path+weight</td>
<td>2.58</td>
</tr>
<tr>
<td>5path</td>
<td>3.19</td>
</tr>
<tr>
<td>4path+weight</td>
<td>3.37</td>
</tr>
<tr>
<td>4path</td>
<td>4.38</td>
</tr>
</tbody>
</table>
Experimental results – Hardware performance and consumption

- **Quartus II, Altera Stratix V**

<table>
<thead>
<tr>
<th>Work</th>
<th>Tsukuba</th>
<th>Venus</th>
<th>Teddy</th>
<th>Cones</th>
<th>nonocc</th>
<th>all</th>
<th>disc</th>
<th>overall</th>
<th>Image size</th>
<th>D</th>
<th>Speed (fps²)</th>
<th>MDE/s (10⁶)</th>
<th>Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wang[11]</td>
<td>2.39</td>
<td>0.38</td>
<td>6.08</td>
<td>2.12</td>
<td>2.74</td>
<td>6</td>
<td>8.10</td>
<td>5.61</td>
<td>1600×1200</td>
<td>128</td>
<td>42.61</td>
<td>10472</td>
<td>FPGA</td>
</tr>
<tr>
<td>Proposed</td>
<td>2.79</td>
<td>0.68</td>
<td>4.18</td>
<td>2.67</td>
<td>2.58</td>
<td>6.19</td>
<td>9.32</td>
<td>6.03</td>
<td>1280×960</td>
<td>64</td>
<td>197</td>
<td>15492</td>
<td>FPGA</td>
</tr>
<tr>
<td>Ttofis[6]</td>
<td>4.04</td>
<td>1.55</td>
<td>7.52</td>
<td>2.77</td>
<td>3.97</td>
<td>6.80</td>
<td>9.30</td>
<td>6.69</td>
<td>1280×720</td>
<td>64</td>
<td>60</td>
<td>3538</td>
<td>FPGA</td>
</tr>
<tr>
<td>Hirschmuller[8]</td>
<td>3.26</td>
<td>1.00</td>
<td>6.02</td>
<td>3.06</td>
<td>3.34</td>
<td>6.87</td>
<td>12.3</td>
<td>7.50</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>CPU</td>
</tr>
<tr>
<td>Banz[9]</td>
<td>4.1</td>
<td>2.7</td>
<td>11.4</td>
<td>8.4</td>
<td>6.7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>640×480</td>
<td>128</td>
<td>103</td>
<td>4050</td>
<td>FPGA</td>
</tr>
<tr>
<td>Shan[14]</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>17.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1280×1024</td>
<td>256</td>
<td>46</td>
<td>15437</td>
<td>FPGA</td>
</tr>
</tbody>
</table>

¹D: Disparity ²fps: frame per second ³MDE/s: Million Disparity Estimation per second (M × N × D × fps)

- **Throughput:** 1280x960/197fps at 64 disparity levels, MDE=15492M, $f_{max} = 156 MHz$
- **Hardware resource utilization:** logic utilization 96,084/234,720 (41%), total registers 83,156/469,440 (18%), block memory bits 20,329,470/52,428,800 (39%)
- **Power:** 16.603W
Outline

• Introduction
• SGM Algorithm for Hardware Implementation
• Proposed Hardware Architecture
• Experimental Results
• Conclusion
Conclusion

- This work proposed a high-throughput hardware architecture for accurate semi-global matching using two-row parallelization and disparity parallelization.
- Five path costs are adaptively weighted to improve the disparity accuracy by 3.69%.
- The implementation on FPGA shows a throughput of 1280×960/197fps at 64 disparity levels in 156MHz.
- The results show high quality.
Thank you very much!
Any questions?