MEMOCODE 2013
Hardware/Software Co-design

Fast and Adaptive BP-based Multi-core Implementation for Stereo Matching

The all-around winner

Armin Ahmadzadeh, Hatef Madani, Kianoush Jafari,
Farzad Salimi-jazi, Shervin Daneshpajouh, Saeid Gorgin

Present By :

Farzad Salimi-Jazi

Outline

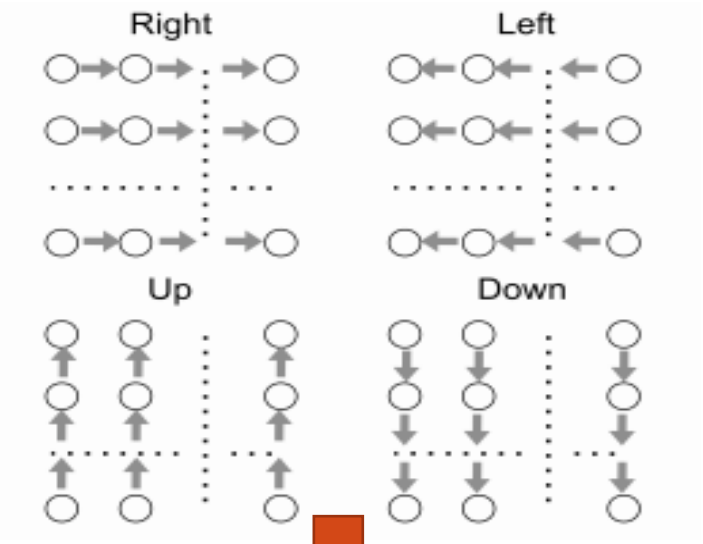
- Introduction
- Optimization techniques
- Implementation method
- Experimental results
- Conclusion

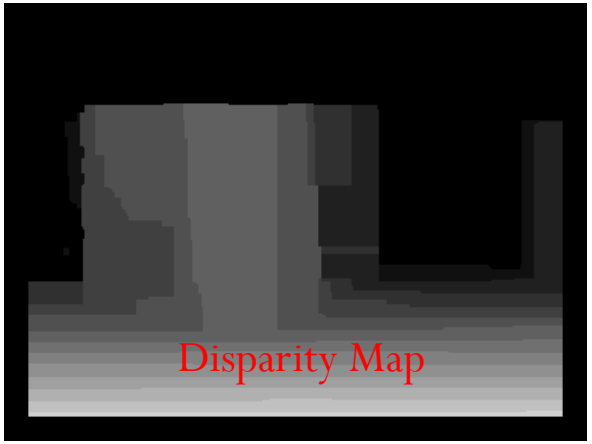
Introduction

LEFT

RIGHT

Loopy Belief Propagation Message passing [1]

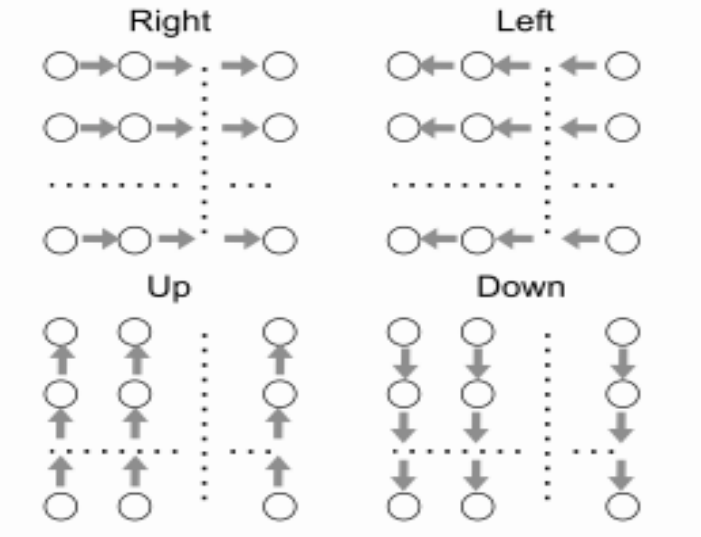




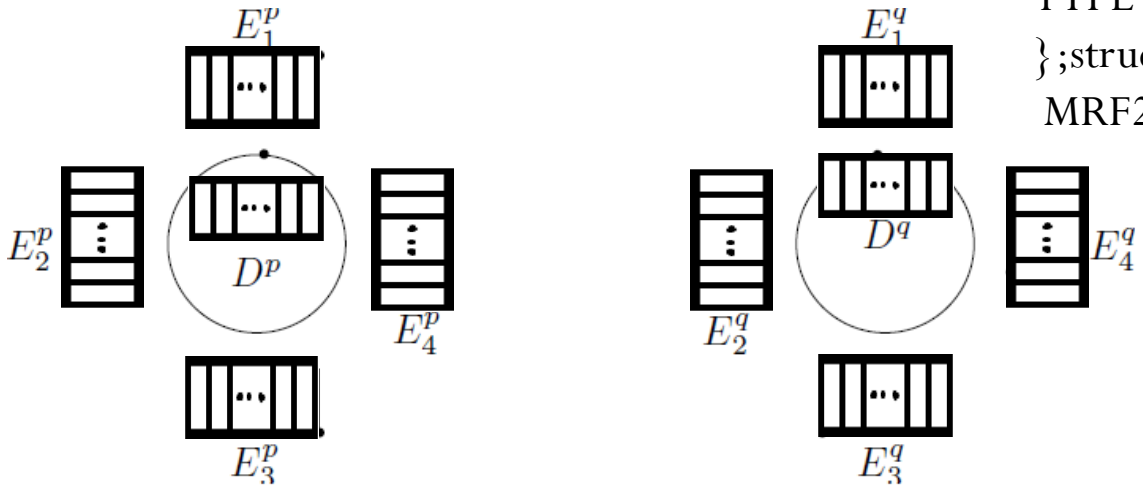
The closer The whiter

Belief Propagation Message passing

- Each vertex has 4 messages from its right/left/up/down edges and a data cost.
- Each message should update L labels (16)
- We do this in some iterations (t)
  - In each iteration we update all of vertexes (n)
    - In each message we update L labels using sender's values
    - Finding better probability in O(L2)




```
struct Vertex {
TYPE  msg[5][LABELS];
};struct
 MRF2D {std::vector <Vertex> grid;};
```



$$O\,(t.n.L^2)$$

Our Optimization techniques

- Min-convolution
- Hierarchy of vertices
- Bipartite method
- Bi-direct message passing

Min-convolution

- Using Min-convolution instead of nested loops to find better probabilities.

$$\textbf{for } f_q \textbf{ from } 1 \textbf{ to } k-1:$$
$$m(f_q) \leftarrow \min(m(f_q), m(f_q-1)+s).$$

$$\textbf{for } f_q \textbf{ from } k-2 \textbf{ to } 0:$$
$$m(f_q) \leftarrow \min(m(f_q), m(f_q+1)+s).$$

Linear algorithm for finding the min value for suitable label [2]

$$O(t.n.L^2) \longrightarrow O(t.n.L)$$

Hierarchy of vertices

- Original idea, several layers
  - like pyramid
- BP is used in coarse layers and data is passed to finer layers
- Data combination and migration between layers

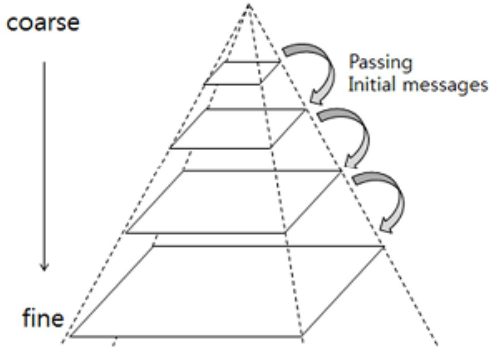


Illustration of fine to coarse levels [4]

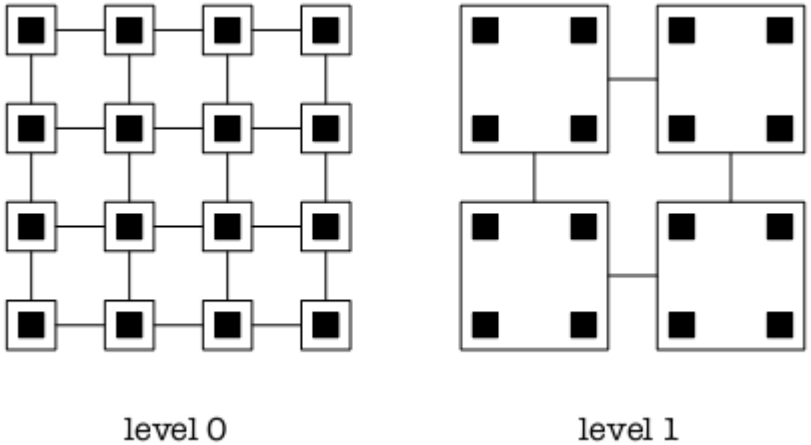
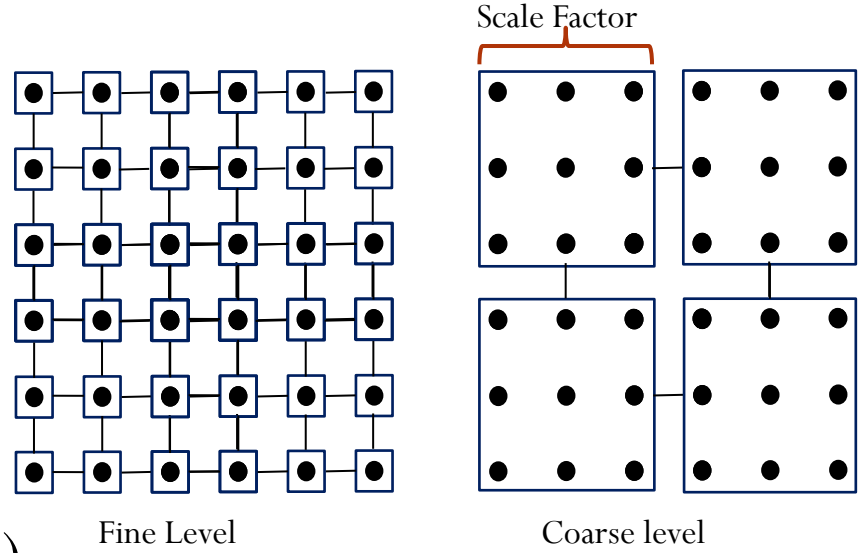


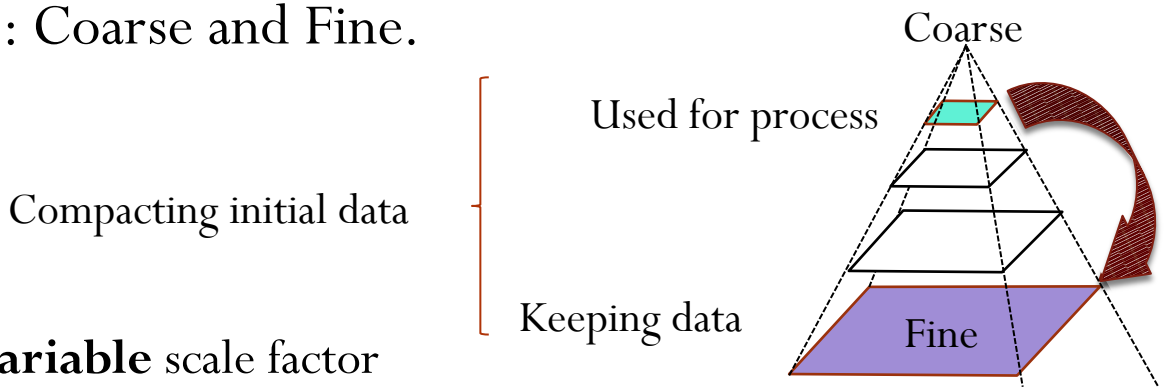
Illustration of two levels in our coarse-to-fine [3]

Reduce Layers in 2 layers

- Motivation:
  - Using modified hierarchical BP
  - Reduce number of iterations



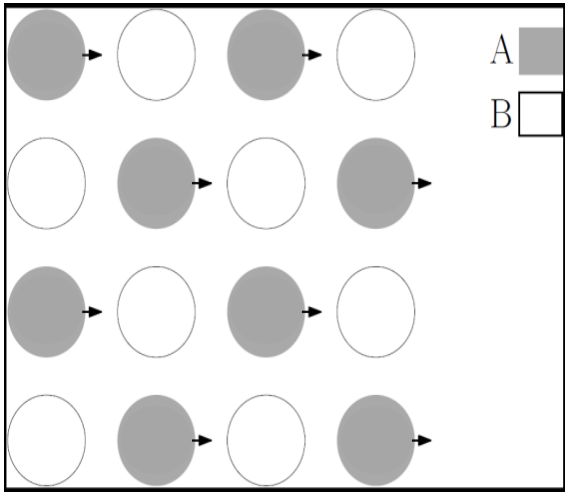
- Idea:
  - Concentrate coarser Level (Layer 1)
  - Reduce computation time rather than finer levels
  - Create two layers : Coarse and Fine.



Also, we have a **variable** scale factor

Bipartite method

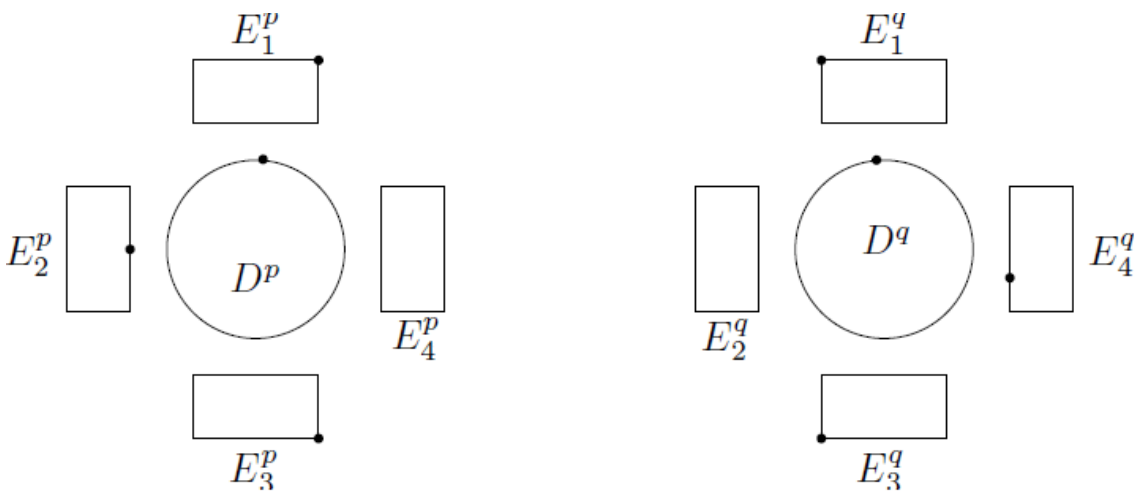
- Bipartite method's two subsets
- Performance improvement
- Reducing memory requirement by half



MRF graph split into two subsets. Subsets are specified through different colors. In each iteration one of the subsets pass message.

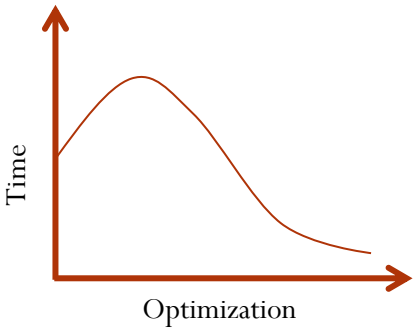
Bidirect message passing

- Bi-direct message passing scheme
- Messages can pass in parallel
- Cache hit improvement

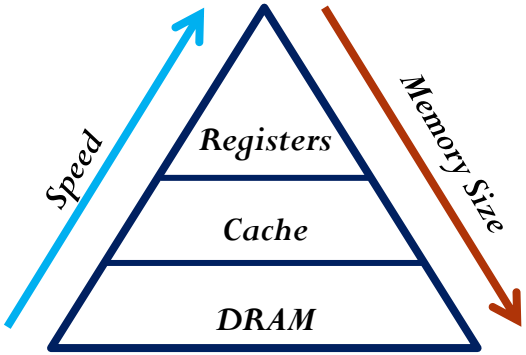


Implementation Method

- Computational Optimization



- Memory Optimization



Computational Optimization

- Initialize Optimization
  - Initiate data in two layers (Coarse and Fine)
  - Compact initial data with Scale Factor
  - Parallel initiate data

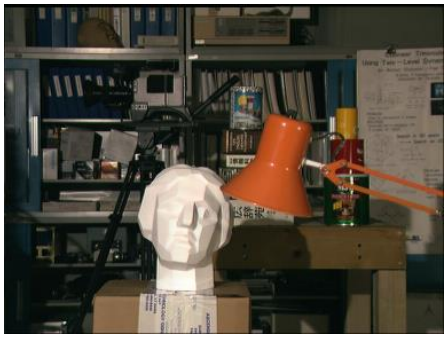
- Message Passing Parallelizing
  - Parallel message passing in two direction (row and column)
  - Utilizing CPU cores (Bidirect message passing)
    - Compute each thread one row (Left & right) or column (up & down)

Memory Optimization

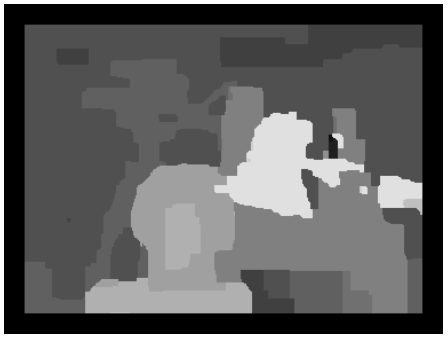
- System Level Optimization
  - Linearizing initial data at memory
  - Vector processing
- Bidirect Memory Optimization
  - Improve cache hit rate (data already exist in catch)
  - Message passing across a direction prefetches data for the other

Experimental Results

Experimental Results
Accuracy



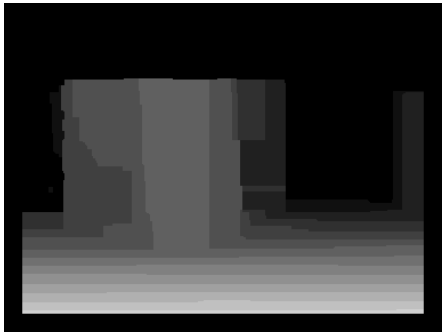
Contest reference image



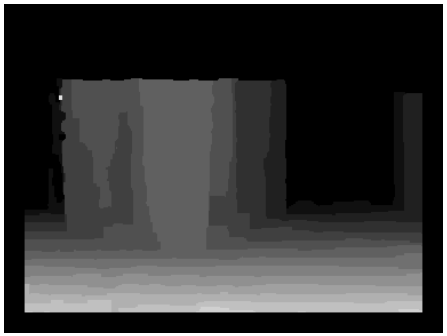
The reference code's output
(17743 mismatches)

Our method's output
(11367 mismatches)

Contest Barrels image

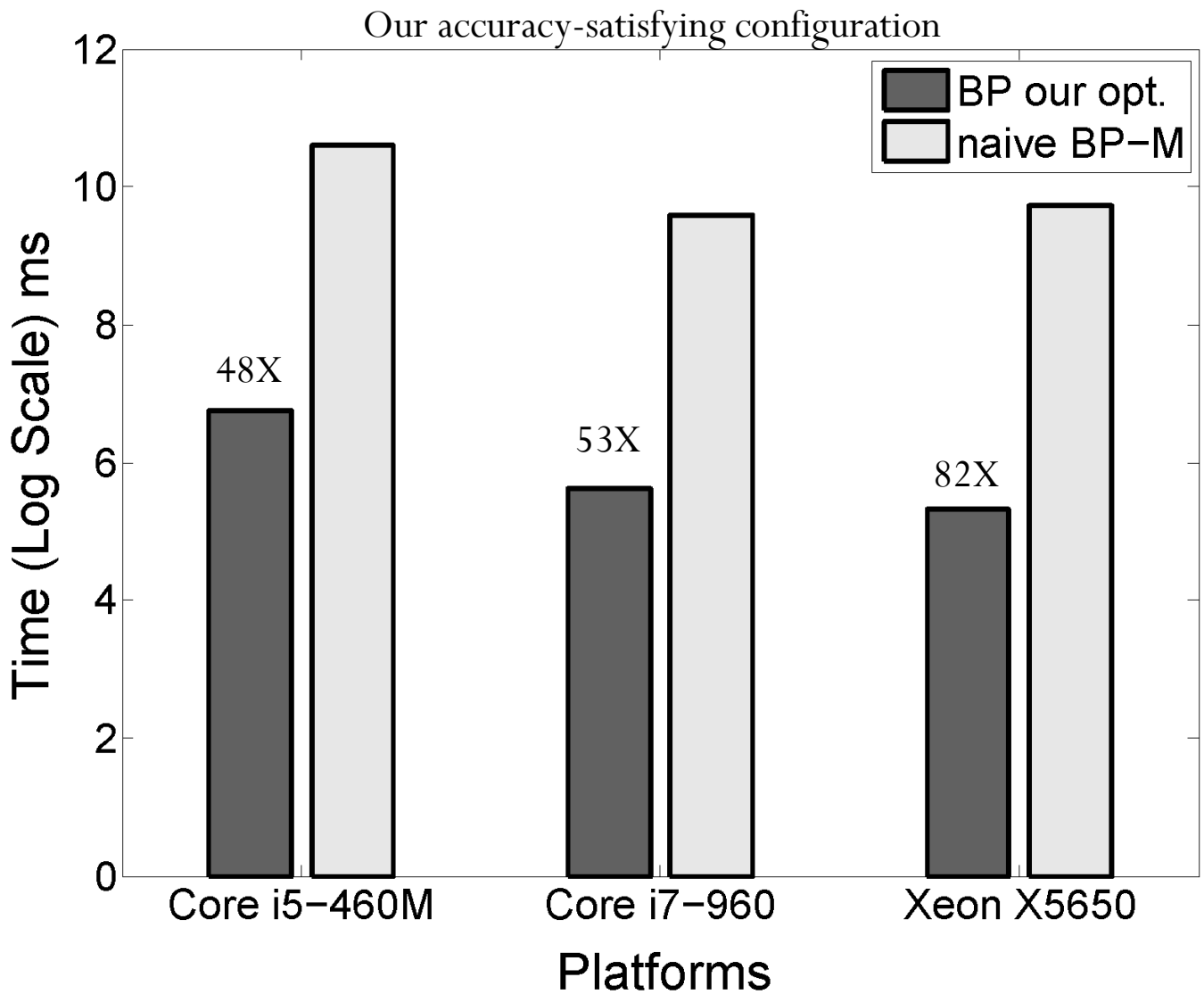


The reference code's output
(11870 mismatches)



Our method's output
(11833 mismatches)

Experimental Results
Time Results



Experimental Results
Time Results I : Xeon X5650

Our implementation results in detail:

Intel(R) Xeon(R) CPU X5650 @ 2.67GHz [5]						
Implementation Methods	Iter.	Scale Factor	tsukuba Image		Barrels Image	
			Mismatch	Time Req.	Mismatch	Time Req.
Reference Code	***40***	***-***	***17743***	***16.7 s***	***11870***	***16.7 s***
Our – Config. 4-8	***4***	***8***	***13695***	***0.6 ms***	***17084***	***0.6 ms***
Our – Config. 4-4	4	4	15047	1.1 ms	15348	1.1 ms
Our – Config. 40-4	40	4	11367	6.3 ms	15328	6.3 ms
Our – Config. 40-1	***40***	***1***	***12727***	***204 ms***	***11833***	***204 ms***
Our – Config. 14-1	14	1	17487	73 ms	13318	73 ms
Our – Config. 20-1	20	1	16333	103 ms	12570	103 ms
Our – Config. 30-1	30	1	14073	155 ms	12048	155 ms

The effect of itter and scale factor are obvious

For generating appropriate accuracy and runtime

Experimental Results
Time Results II : Core i7

Our implementation results in detail:

Intel(R) Xeon(R) CPU Core i7 960 @ 2.6 GHz [5]						
Implementation Methods	Iter.	Scale Factor	tsukuba Image		Barrels Image	
			Mismatch	Time Req.	Mismatch	Time Req.
Reference Code	***40***	***-***	***17743***	***14.7 s***	***11870***	***14.7 s***
Our – Config. 4-8	***4***	***8***	***13695***	***0.8 ms***	***17084***	***0.8 ms***
Our – Config. 4-4	4	4	15047	1.5 ms	15348	1.5 ms
Our – Config. 40-4	40	4	11367	8.1 ms	15328	8.1 ms
Our – Config. 40-1	***40***	***1***	***12727***	***277 ms***	***11833***	***277 ms***
Our – Config. 14-1	14	1	17487	98 ms	13318	98 ms
Our – Config. 20-1	20	1	16333	139 ms	12570	139 ms
Our – Config. 30-1	30	1	14073	206 ms	12048	206 ms

Experimental Results
Time Results III : Core i5

Our implementation results in detail:

Intel(R) Xeon(R) CPU Core i5 @ 2.57GHz [5]						
Implementation Methods	Iter.	Scale Factor	tsukuba Image		Barrels Image	
			Mismatch	Time Req.	Mismatch	Time Req.
Reference Code	***40***	***-***	***17743***	***40.5 s***	***11870***	***40.5 s***
Our – Config. 4-8	***4***	***8***	***13695***	***2.9 ms***	***17084***	***2.9 ms***
Our – Config. 4-4	4	4	15047	5.9 ms	15348	5.9 ms
Our – Config. 40-4	40	4	11367	40 ms	15328	40 ms
Our – Config. 40-1	***40***	***1***	***12727***	***850 ms***	***11833***	***850 ms***
Our – Config. 14-1	14	1	17487	300 ms	13318	300 ms
Our – Config. 20-1	20	1	16333	426 ms	12570	426 ms
Our – Config. 30-1	30	1	14073	635.3 ms	12048	635.3 ms

Acknowledgments

- Prof. Hamid Sarbazi Azad

- Ahmad Lashgar
- Mohsen Mahmoudi-Aznaveh
- Alireza Majidi
- Soroush Bateni
- Kamyar Mirzazad
- Mehrshad Vosoughi

http://cs.ipm.ac.ir

Any Question

References

[1] MEMOCODE 2013 Hardware/Software Co-design Contest: Stereo Matching can be find: http : //memocode.irisa.fr/2013/.

[2] P. F. Felzenszwalb and D. P. Huttenlocher Efficient Belief Propagation for Early Vision, International Journal of Computer Vision 70(1): 41-54(2006).

[3] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, C. Rother, A Comparative Study of Energy Minimization Methods For Markov Random Field with Smoothness-Based Priors, IEEE Transaction on pattern analysis and machine intelligence, 30(6): 1068-1080 (2008).

[4] C. Liang, C. Cheng, Y. Lai, L. Chen, H. Chen, Hardware-Efficient Belief Propagation, IEEE Transaction On Circuits And Systems For Video Technology, 21(5): 525-537 (2011).

[5] Product Information available at http://ark.intel.com.