SPM-BP: Sped-up PatchMatch Belief Propagation for Continuous MRFs

Yu Li, Dongbo Min, Michael S. Brown, Minh N. Do, Jiangbo Lu

Discrete Pixel-Labeling Optimization on MRF

- Many computer vision tasks can be formulated as a pixel-labeling problem on Markov Random Field (MRF)

Segmentation
$l=\{B,G\}$

Denoising
l = *intensity*

Stereo
$l = d$

Optical flow
$l = (u,v)$

$$E = \sum_p E_p(l_p; W) + \sum_p \sum_{q \in \mathcal{N}_p} E_{pq}(l_p, l_q)$$

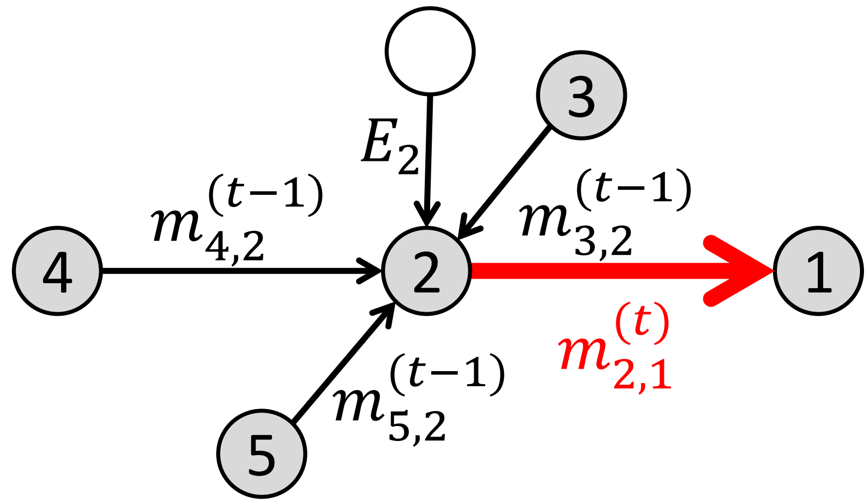
p: pixel, N_p: 4 neighbors

- Simple: data term + smoothness term
- Effective: labeling coherence, discontinuity handling
- Optimization: Graph Cut, $\boxed{\text{Belief Propagation}}$, etc

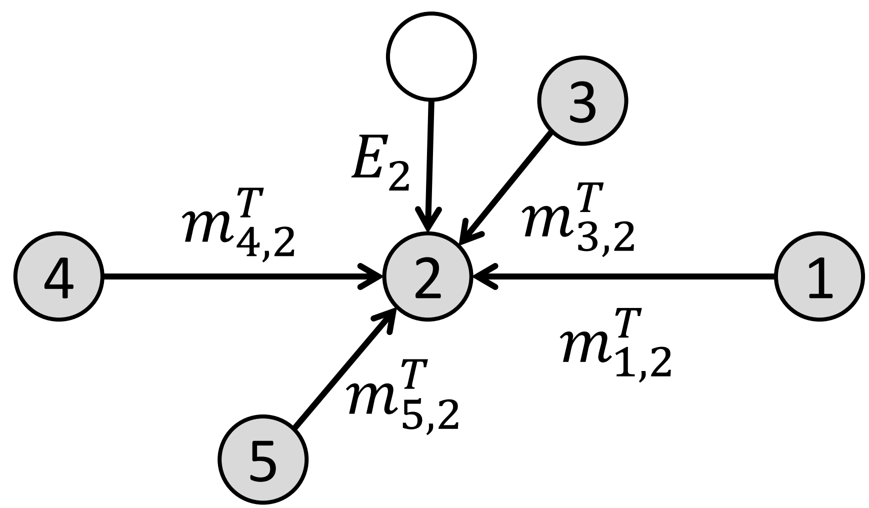
Belief Propagation (BP)

Iterative process in which neighbouring nodes "talk" to each other:

- Update message between neighboring pixels



- Stop after T iterations, decide the final label by picking the smallest dis-belief



- **Challenge**:

When the label set L is huge or densely sampled, BP faces prohibitively high computational challenges.

Particle Belief Propagation (PBP)

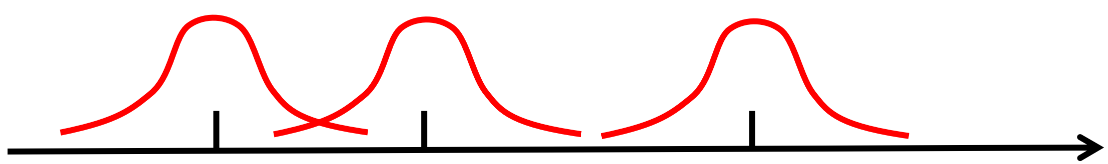
[Ihler and McAllester, "Particle Belief Propagation," *AISTATS*'09]

- **Solution**:

  (1) only store messages for *K* labels (particles)

l (discrete label)

  (2) generate new label particles with the MCMC sampling using a Gaussian proposal distribution



l

▪**Challenge**:
MCMC sampling is still inefficient and slow for continuous label spaces (e.g. stereo with slanted surfaces).

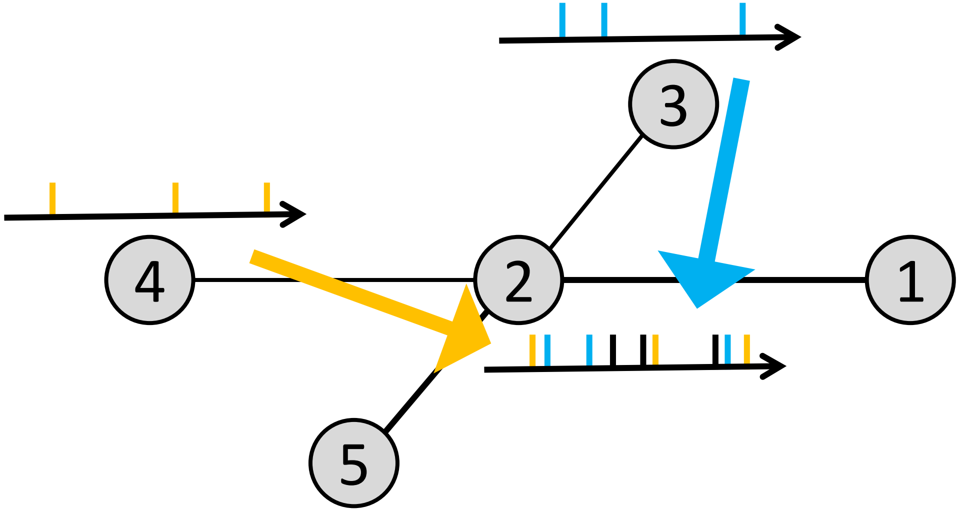
Patch Match Belief Propagation (PMBP)

[Besse et al, “PMBP: PatchMatch Belief Propagation for Correspondence Field Estimation,” *IJCV* 2014]

- **Solution**:

Use Patch Match[Barnes et al. Siggraph’09]’s sampling algorithm – augment PBP with label samples from the neighbours as proposals

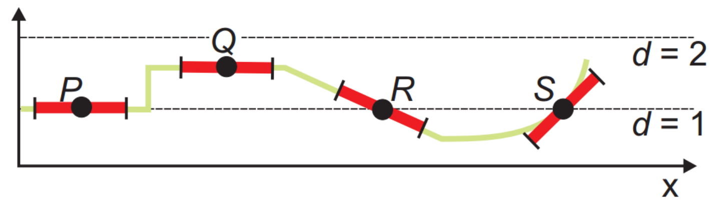
- Orders of magnitude faster than PBP



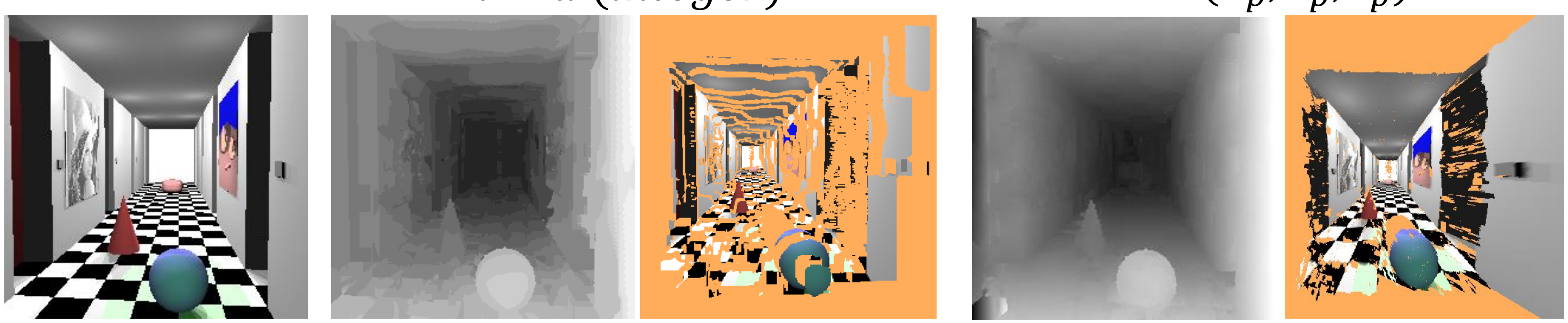
Patch Match Belief Propagation (PMBP)

- Effectively handles large label spaces in message passing
- Successfully applied to stereo with slanted surface modeling [Bleyer et al., BMVC'11]

Label: 3D plane normal $l = (a_p, b_p, c_p)$



$l = d \ (integer)$ $\qquad$ $l = (a_p, b_p, c_p)$



Left image | Disparity map | 3D reconstruction | Disparity map | 3D reconstruction

Image courtesy of [Bleyer et al., BMVC'11]

- Also successfully applied to optical flow [Hornáček et al., ECCV'14]

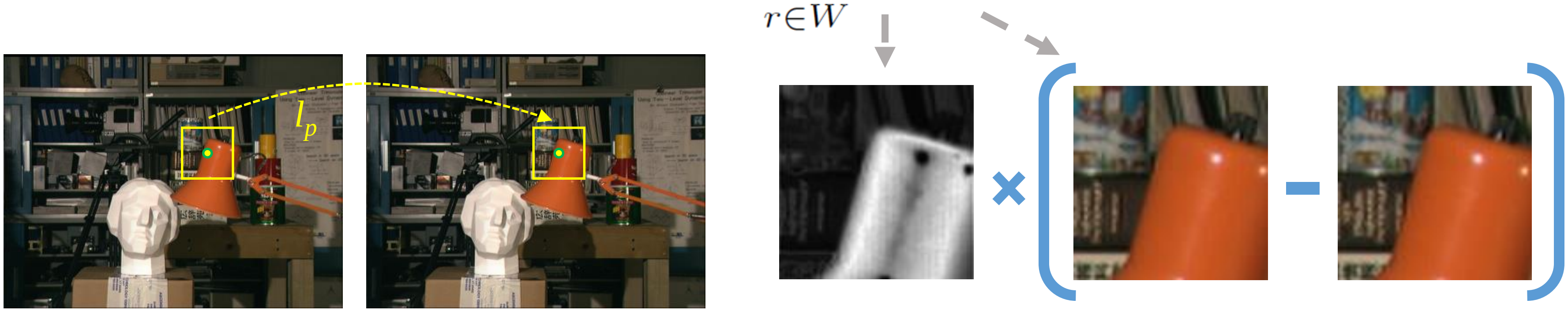
Problem of PMBP

- However, it suffers from a heavy computational load on the data cost computation

$$E = \boxed{\sum_{p} E_p(l_p; W)} + \sum_{p} \sum_{q \in \mathcal{N}_p} E_{pq}(l_p, l_q)$$

- Many works strongly suggest to gather stronger evidence from a local window for the data term

$$E_p(l_p; W) = \sum_{r \in W} \omega_{pr} C_r(l_p)$$

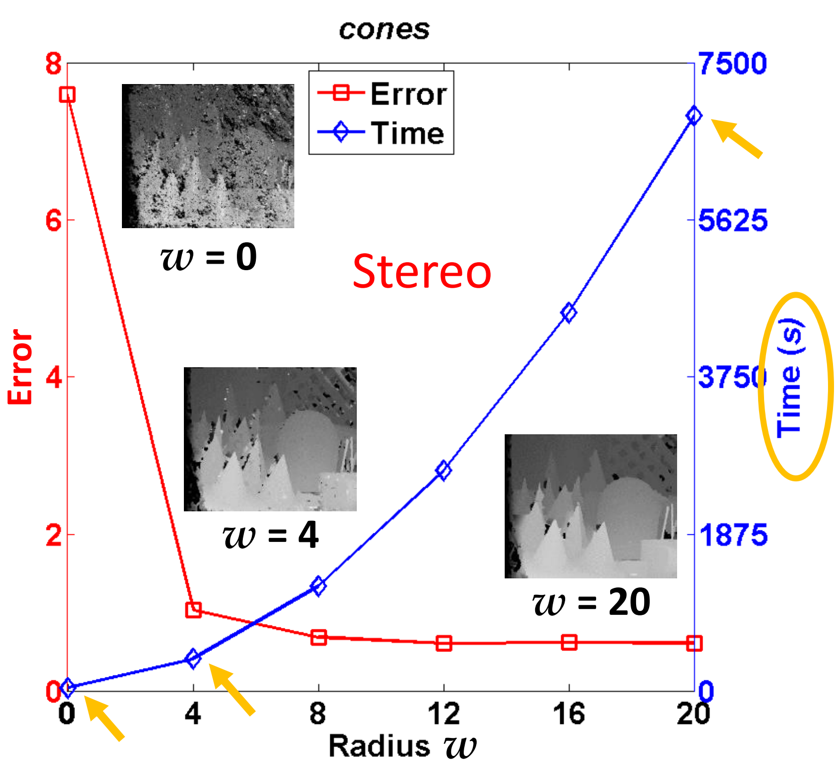


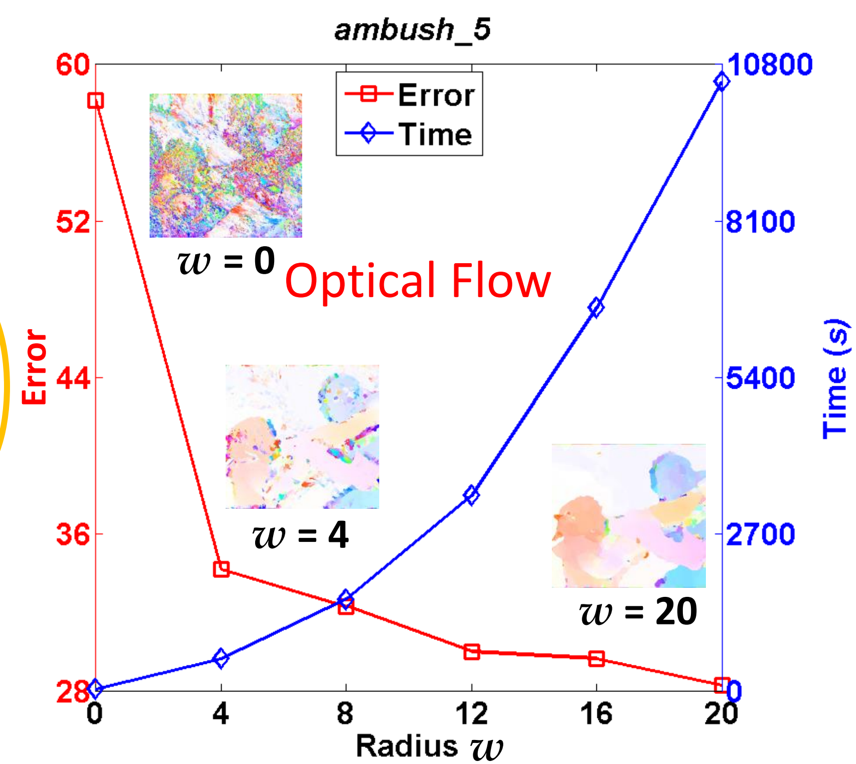
Left view Right view Weight Raw matching cost

Data term is important!

- *Better results with larger window sizes (2w+1)^2, but more computational cost!*

$$\boxed{E_p(l_p; W) = \sum_{r \in W} \omega_{pr} C_r(l_p)}$$





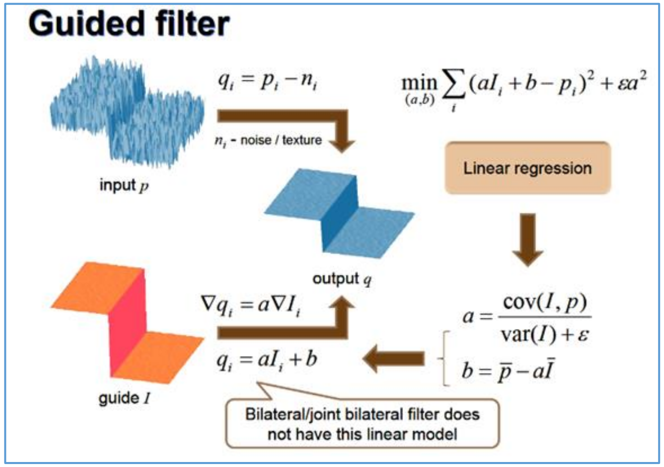
Aggregated data cost computation

- Cross/joint/bilateral filtering principles

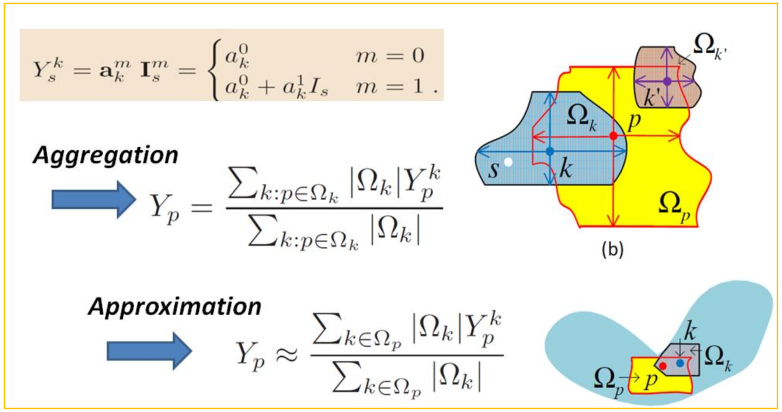
$$E_p(l_p; W) = \sum_{r \in W} \omega_{pr} C_r(l_p)$$

- **Local discrete labeling approaches** have often used efficient O(1)-time edge-aware filtering (EAF) methods [Rhemann et al., CVPR'11].
  - O(1)-time: No dependency on window size used in EAF

Guided Filter [He et al. *ECCV* 2010]



Cross-based Local Multipoint Filtering (CLMF) [Lu et al. *CVPR* 2012]



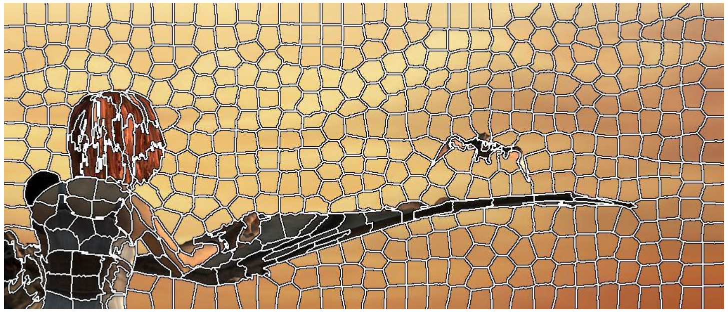
Why does PMBP NOT use O(1) time EAF?

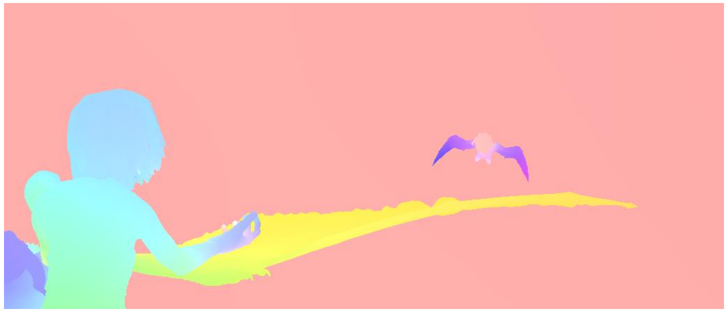
- Particle sampling and data cost computation are performed independently for each pixel

  ➔ Incompatible with EAF, essentially exploiting redundancy

- **Observation**

Labeling is often spatially smooth away from edges. This allows for shared label proposal and data cost computation for spatially neighboring pixels.





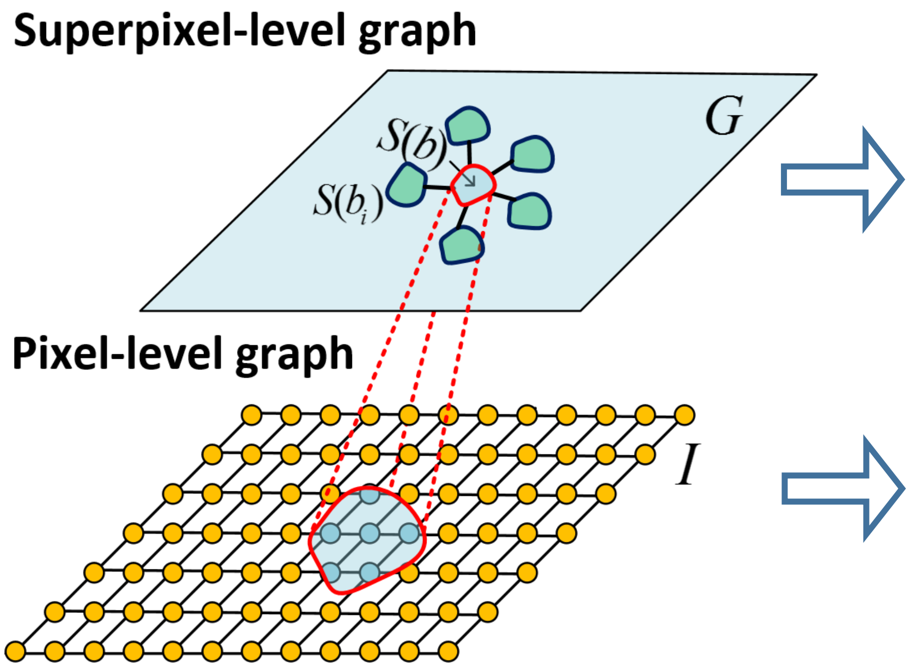
- **Our solution**

A superpixel based particle sampling belief propagation method, leveraging efficient filter-based cost aggregation

Sped-up Patch Match Belief Propagation (SPM-BP)

Sped-up Patch Match Belief Propagation

- **Two-Layer Graph Structures in SPM-BP**



1. Shared particle generation
2. Shared data cost computation

$$E = \boxed{\sum_p E_p(l_p; W)} + \sum_p \sum_{q \in \mathcal{N}_p} E_{pq}(l_p, l_q)$$

1. Message passing
2. Particle selection

$$E = \sum_p E_p(l_p; W) + \boxed{\sum_p \sum_{q \in \mathcal{N}_p} E_{pq}(l_p, l_q)}$$

- **Scan Superpixels and Perform :**
  - *Neighbourhood Propagation*
  - *Random Search*

Related works

Pixel based MRF

Local methods

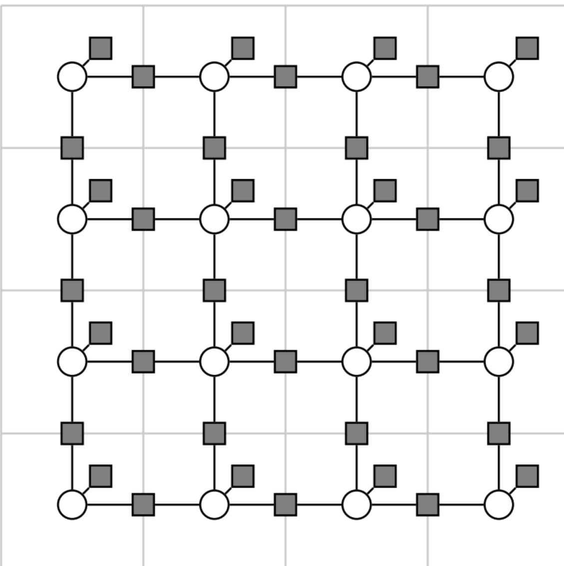
[Rhemann et al., CVPR’11]
[Lu et al., CVPR’13]

Only rely on data term

Superpixel based MRF

[Kappes et al., IJCV’15]
[Güney & Geiger, CVPR’15]

Superpixels as graph nodes



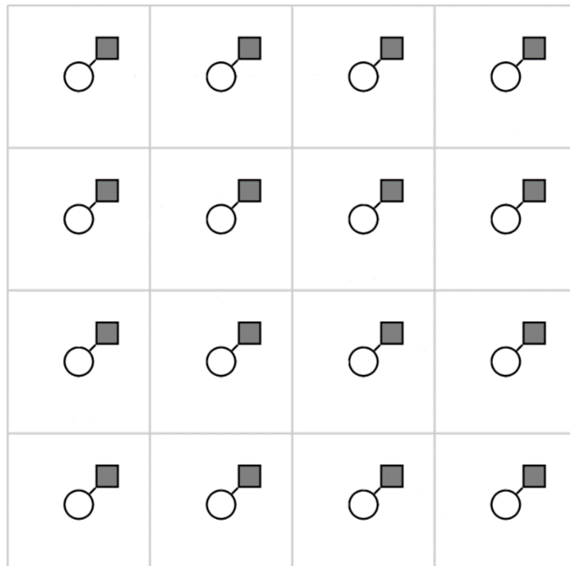
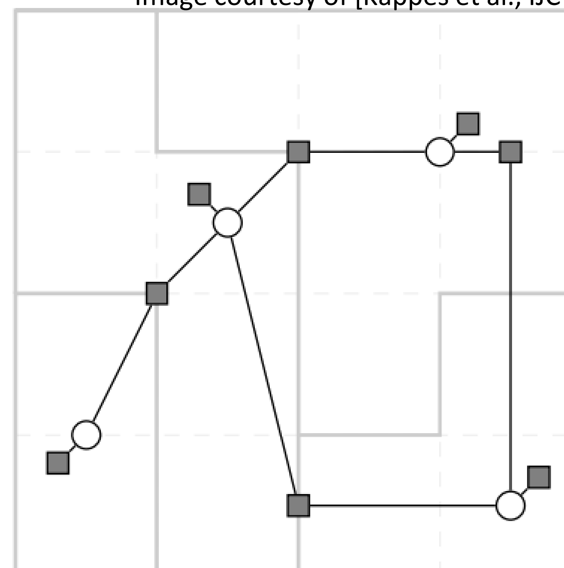


Image courtesy of [Kappes et al., IJCV’15]



Superpixel-based MRF: each superpixel is a node in the graph and **all pixels of the superpixel are constrained to have the same label**.

Our two-layer graph: superpixel are employed only for particle generation and data cost computation, the **labeling is performed for each pixel independently**.

Comparison of existing labeling optimizers

Local labeling approaches		Data cost computation	
		w/o EAF: $O(\lvert W \rvert)$	w/ EAF: $O(1)$
Label space handling	w/o PatchMatch: $O(\lvert L \rvert)$	Adaptive Weighting [PAMI'06]	Cost Filtering [CVPR'11]
	w/ PatchMatch: $O(\log \lvert L \rvert)$	PM Stereo [BMVC'11]	**PMF** [CVPR'13]

Global labeling approaches		Data cost computation	
		w/o EAF: $O(\lvert W \rvert)$	w/ EAF: $O(1)$
Label space handling	w/o PatchMatch: $O(\lvert L \rvert)$	BP [PAMI'06]	Fully-connected CRFs [NIPS'11]
	w/ PatchMatch: $O(\log \lvert L \rvert)$	**PMBP** [IJCV'14]	**?**

Comparison of existing labeling optimizers

Local labeling approaches		Data cost computation	
		w/o EAF: $O(\|W\|)$	w/ EAF: $O(1)$
Label space handling	w/o PatchMatch: $O(\|L\|)$	Adaptive Weighting [PAMI'06]	Cost Filtering [CVPR'11]
	w/ PatchMatch: $O(\log\|L\|)$	PM Stereo [BMVC'11]	**PMF** [CVPR'13]

Global labeling approaches		Data cost computation	
		w/o EAF: $O(\|W\|)$	w/ EAF: $O(1)$
Label space handling	w/o PatchMatch: $O(\|L\|)$	BP [PAMI'06]	Fully-connected CRFs [NIPS'11]
	w/ PatchMatch: $O(\log\|L\|)$	**PMBP** [IJCV'14]	**SPM-BP [This paper]**

SPM-BP: Neighbourhood Propagation

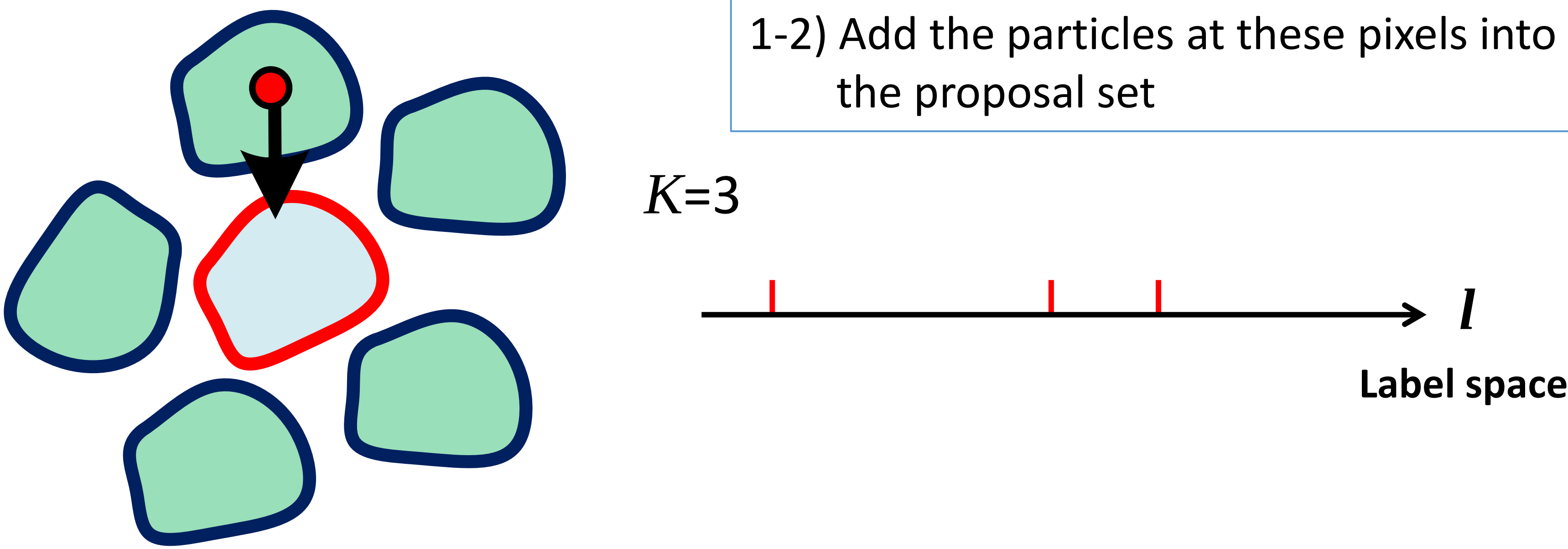
✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

1-1) Randomly select one pixel from each neighbouring superpixel

1-2) Add the particles at these pixels into the proposal set



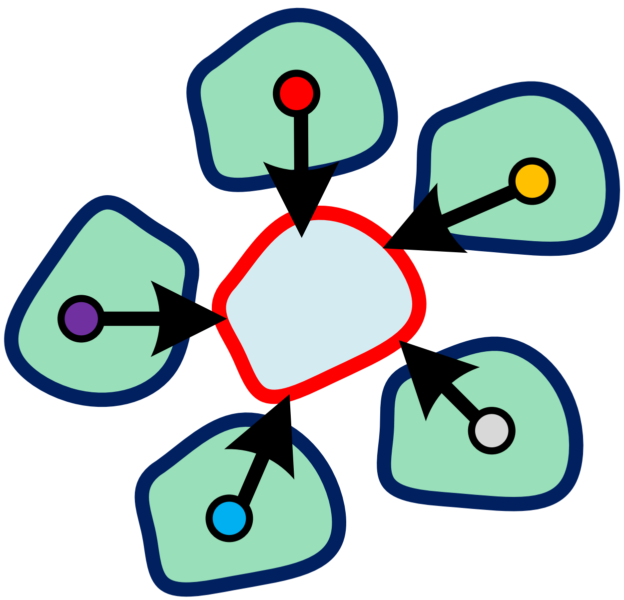
SPM-BP: Neighbourhood Propagation

✓Step 1. Particle propagation

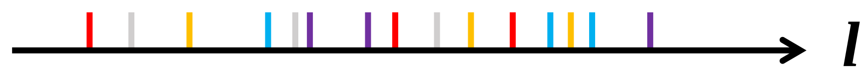
✓Step 2. Data cost computation

✓Step 3. Message update

1-1) Randomly select one pixel from each neighbouring superpixel
1-2) Add the particles at these pixels into the proposal set



K=3



SPM-BP: Neighbourhood Propagation

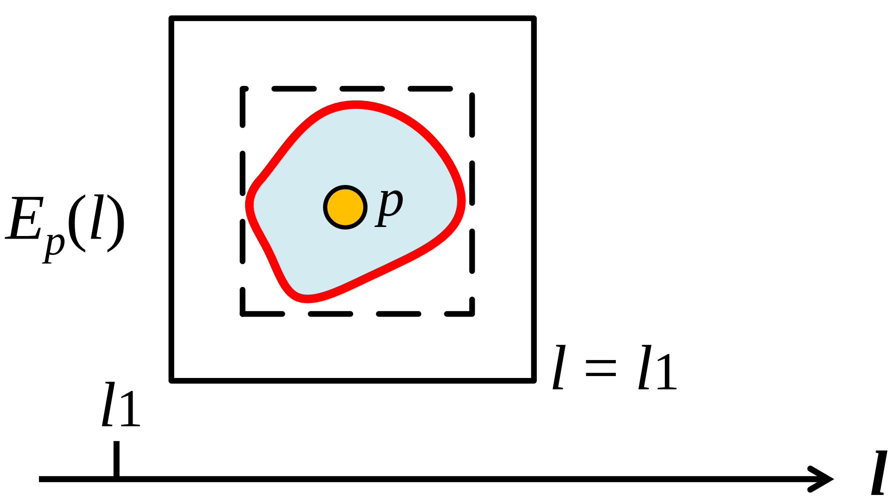
✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

2-1) Compute the raw matching data cost of these labels in a slightly enlarged region

2-2) Compute the aggregated data cost for each label by performing EAF on the raw matching cost

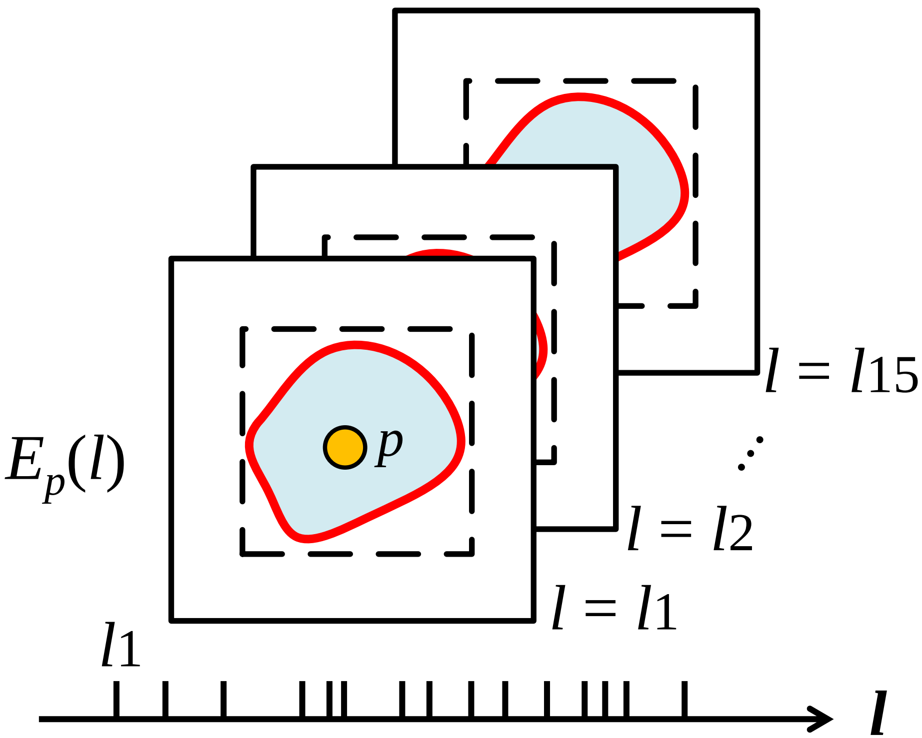


SPM-BP: Neighbourhood Propagation

✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update



2-1) Compute the raw matching data cost of these labels in a slightly enlarged region

2-2) Compute the aggregated data cost for each label by performing EAF on the raw matching cost

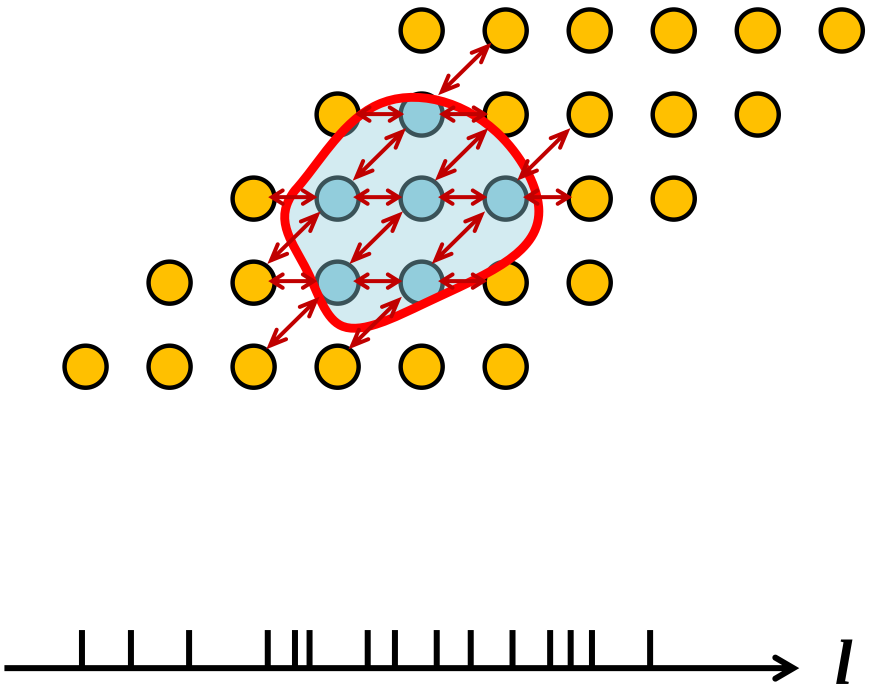
SPM-BP: Neighbourhood Propagation

✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

3-1) Perform message passing for pixels within the superpixel.



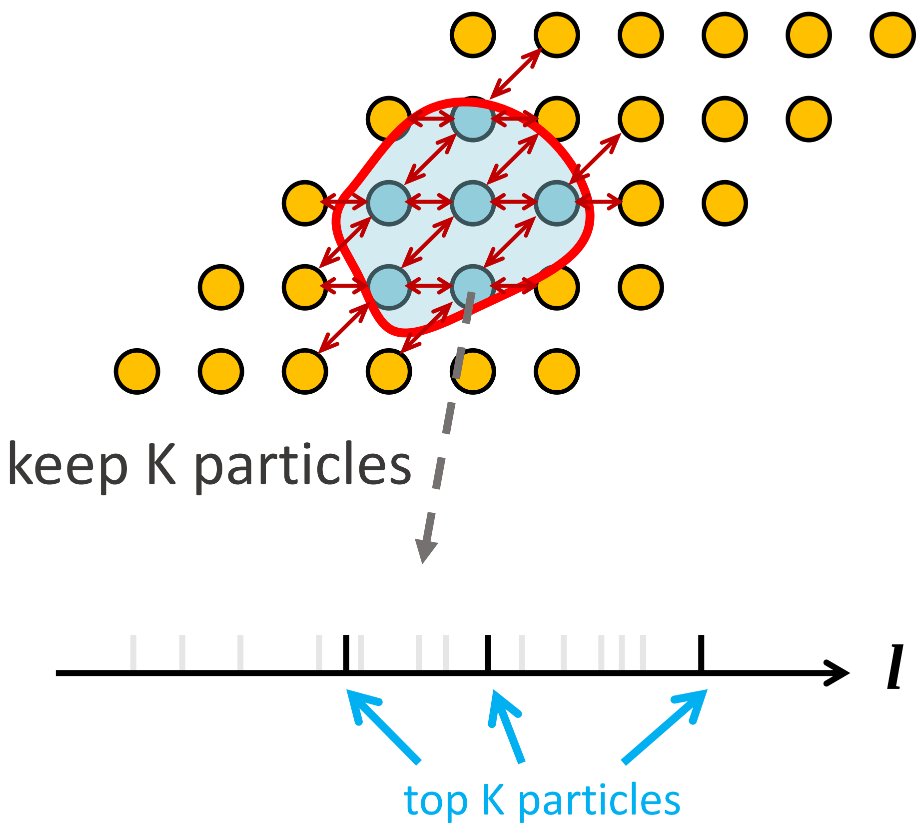
SPM-BP: Neighbourhood Propagation

✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

3-1) Perform message passing for pixels within the superpixel.
3-2) Keep K particles with the smallest disbeliefs at each pixel.



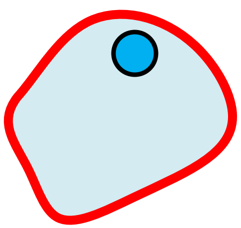
SPM-BP: Random Search

✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

1-1) Randomly select one pixel in the visiting superpixel



SPM-BP: Random Search

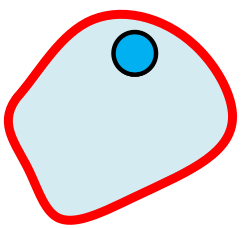
✓Step 1. Particle propagation

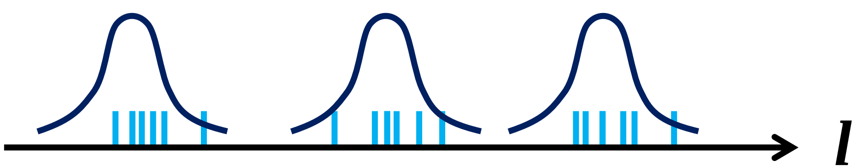
✓Step 2. Data cost computation

✓Step 3. Message update

1-1) Randomly select one pixel in the visiting superpixel

1-2) Generate new proposals around the sampled particles





SPM-BP: Random Search

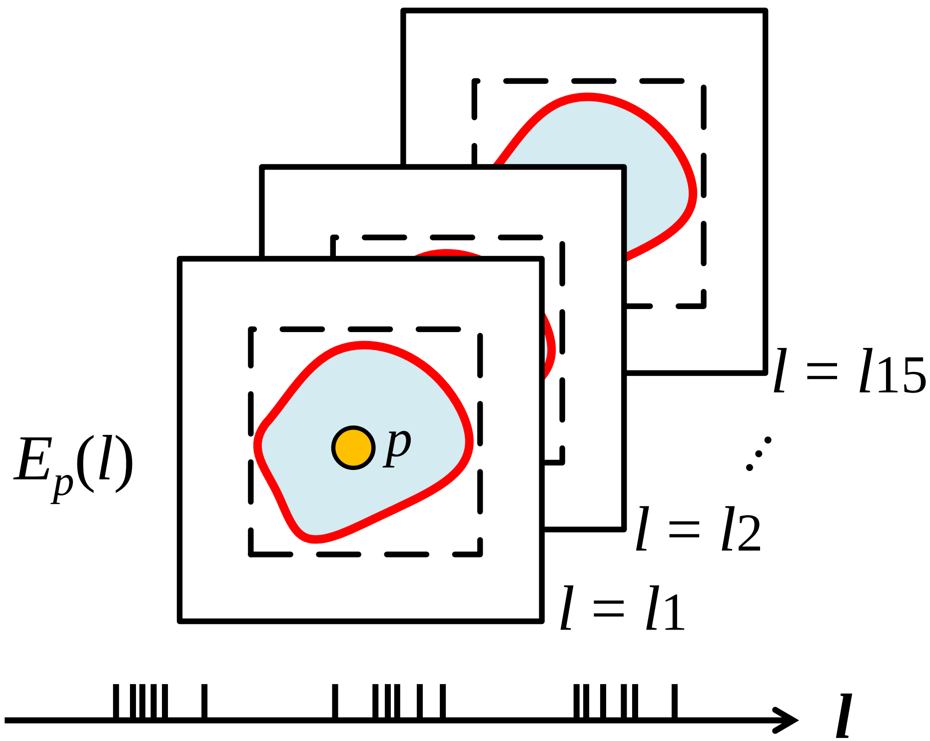
✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

2-1) Compute the raw matching data cost of these labels in a slightly enlarged region

2-2) Compute the aggregated data cost for each label by performing EAF on the raw matching cost



$$E_p(l_p; W) = \sum_{r \in W} \omega_{pr} C_r(l_p)$$

SPM-BP: Random Search

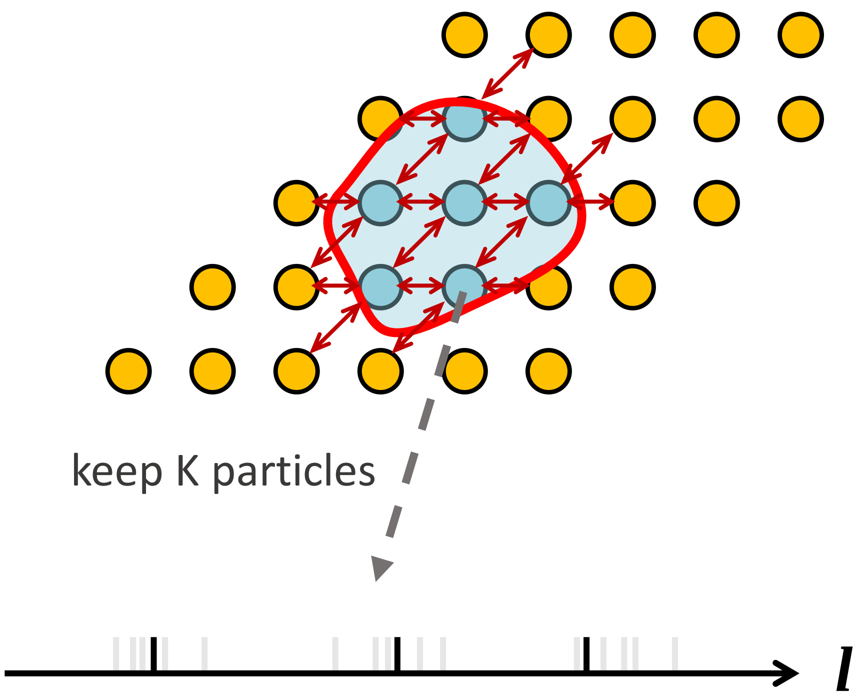
✓Step 1. Particle propagation

✓Step 2. Data cost computation

✓Step 3. Message update

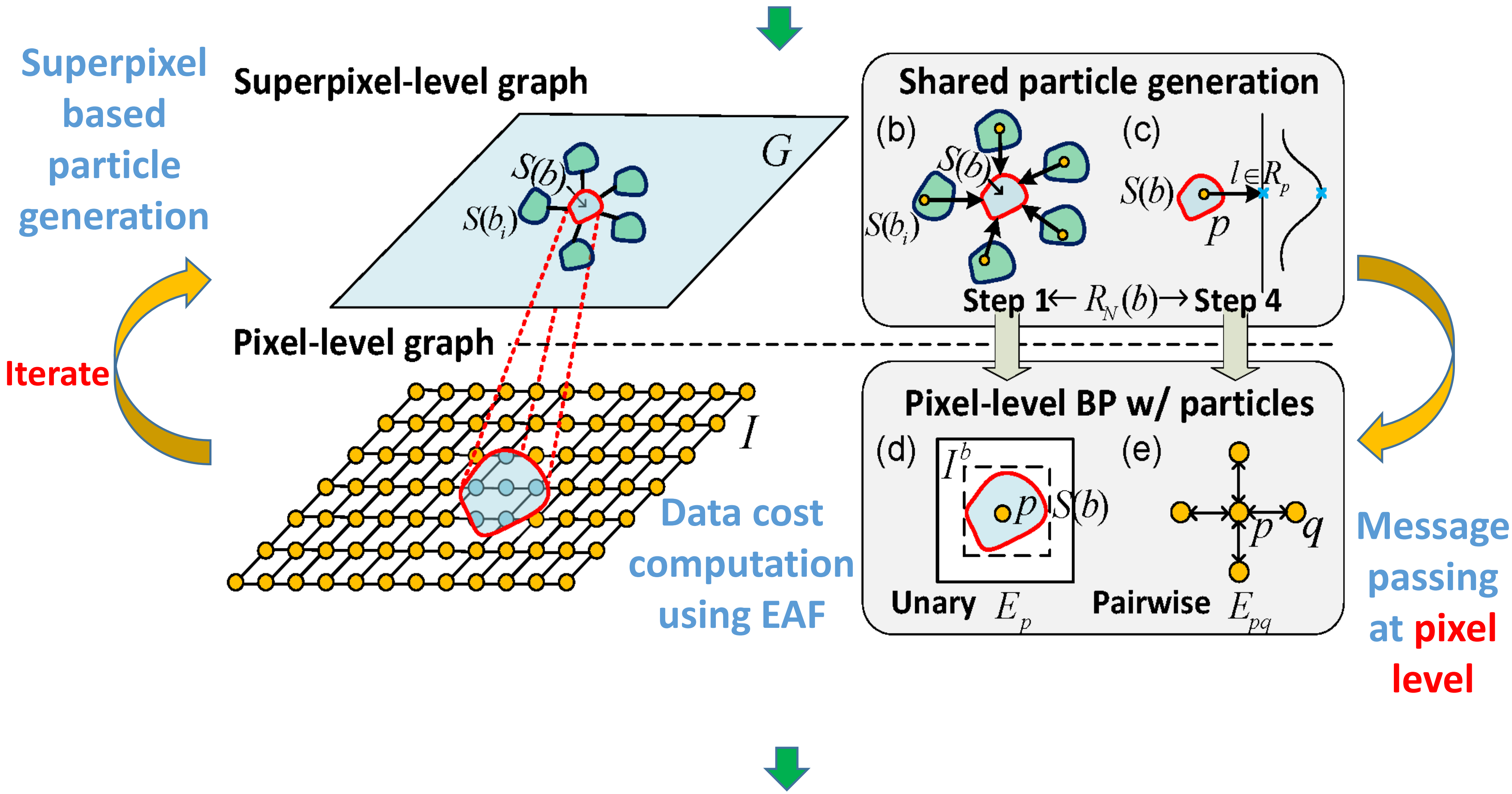
3-1) Perform message passing for pixels within the superpixel.

3-2) Keep K particles with the smallest disbeliefs at each pixel.



SPM-BP: Recap

Random Initialization



Final labels

Complexity Comparison

	PMF* [32]	PMBP [8]	**SPM-BP**
Data Cost	$O(N \log L)$	$O(\lvert W \rvert K N \log L)$	$O(K N \log L)$
Message Passing	-	$O(K^2 N \log L)$	$O(K^2 N \log L)$

$|W|$ – local window size (e.g. 31x31 for stereo)

K – number of particles used (small constant)

N – number of pixels

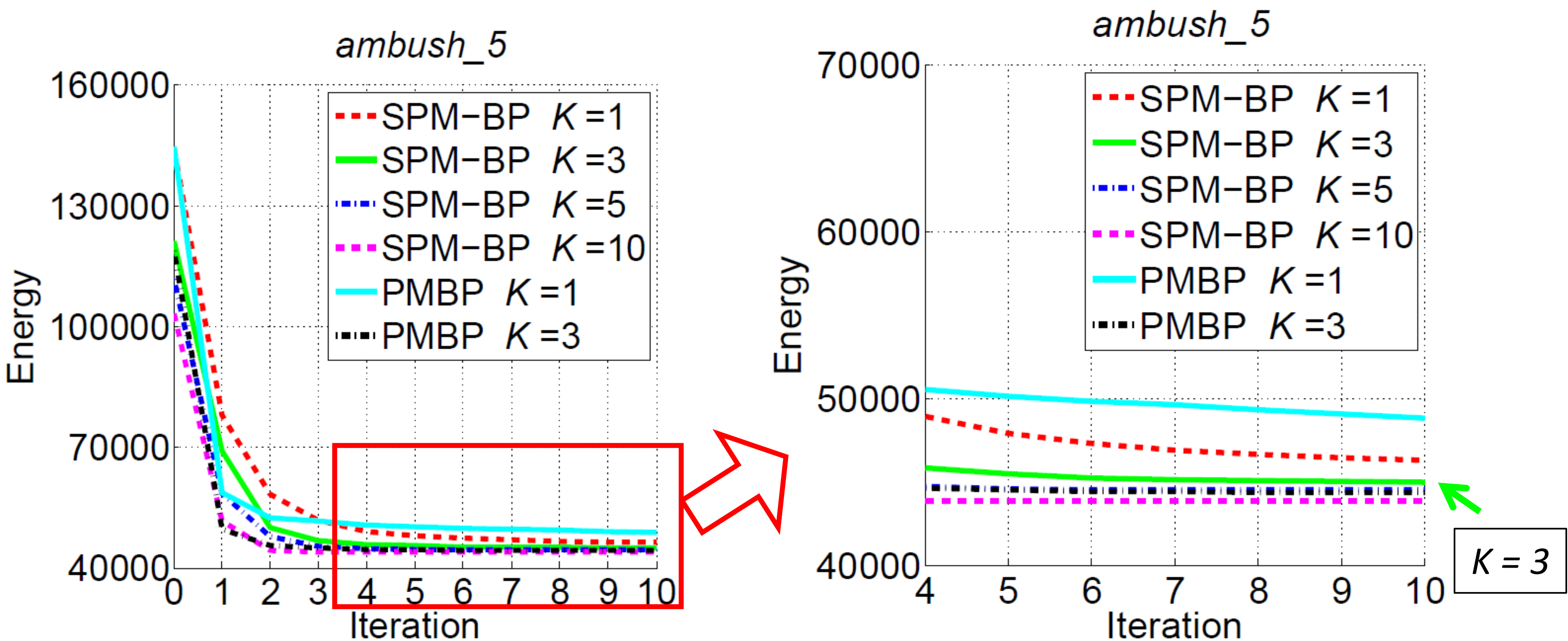
L – label space size (e.g. over 10 million for flow)

*PMF stores only one best particle (K = 1) per pixel node, thus requiring more iterations than the other two methods.

Example Applications

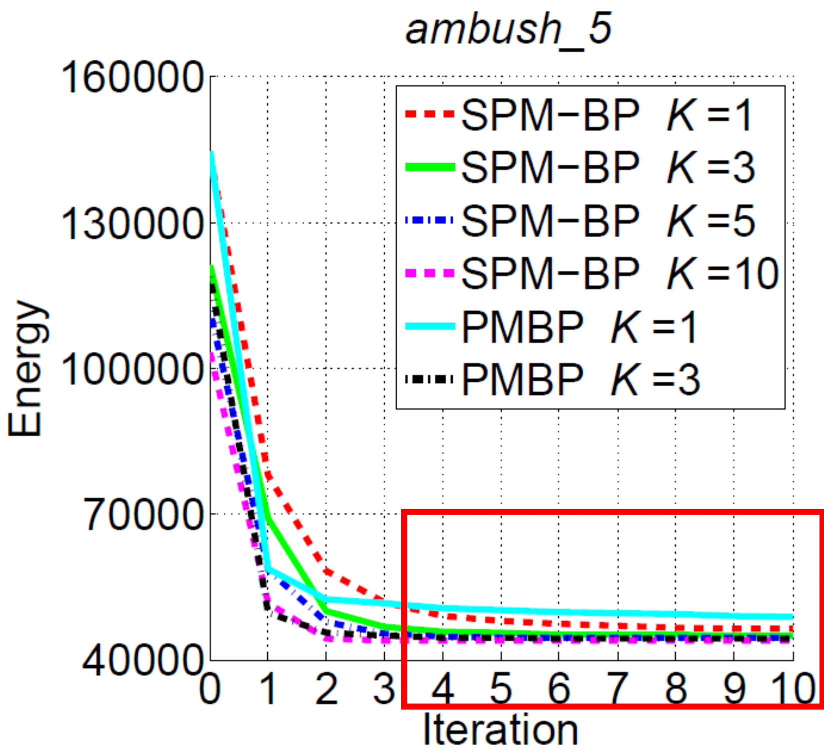
- **Stereo with slanted surface supports**
  - **label**: 3D plane normal $l_p = (a_p, b_p, c_p)$
  - **Matching features**: color + gradient
  - **Smoothness term**: deviation between two local planes
  - **Cross checking + post processing for occlusion**
- **Large-displacement optical flow**
  - **label**: 2D displacement vector $l_p = (u, v)$
  - **Matching features**: color + Census transform
  - **Smoothness term**: truncated L_2 distance
  - **Cross checking + post processing for occlusion**

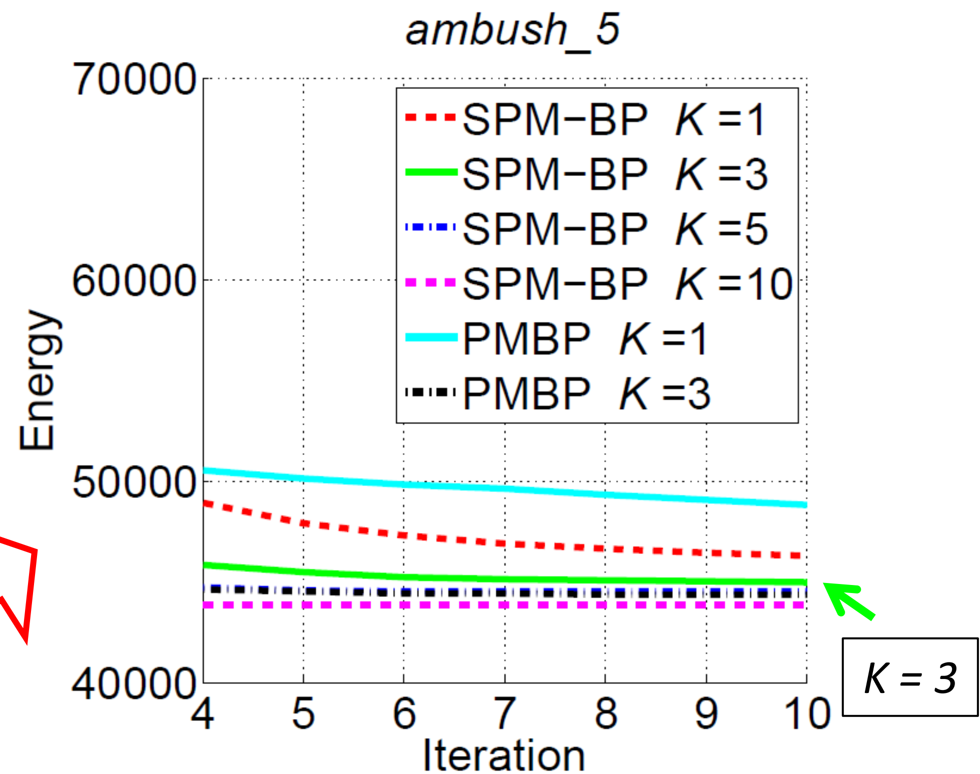
Convergence

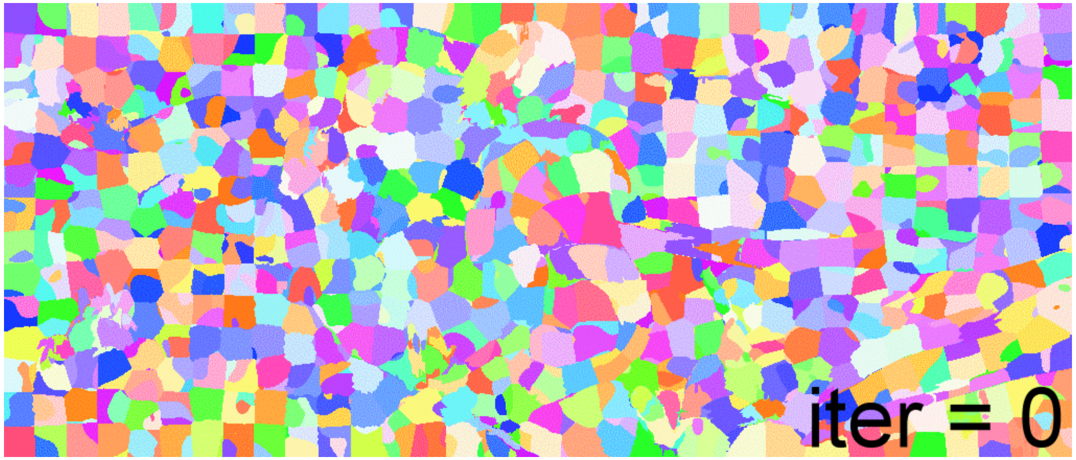


#iteration = 5, K = 3

Convergence

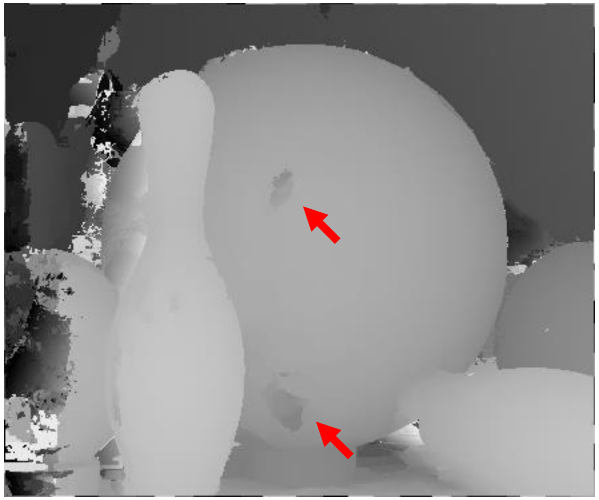




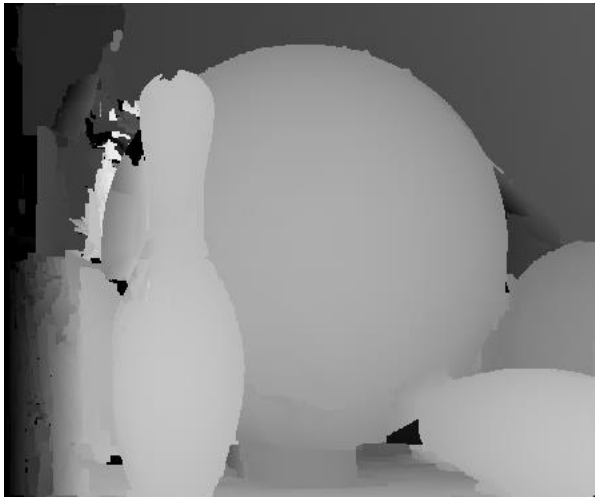


Stereo results

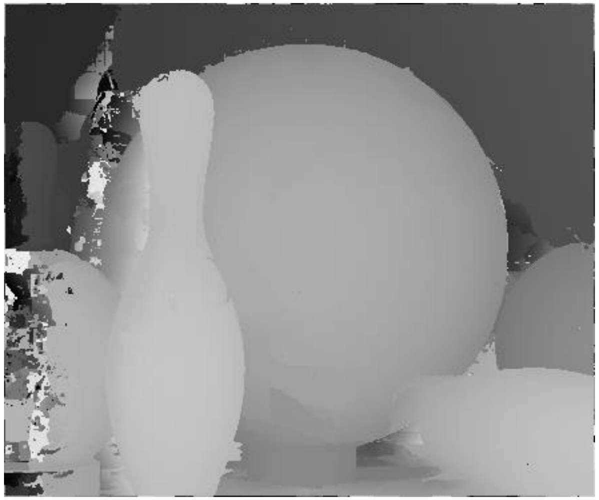
Stereo input



PMF
20 sec.



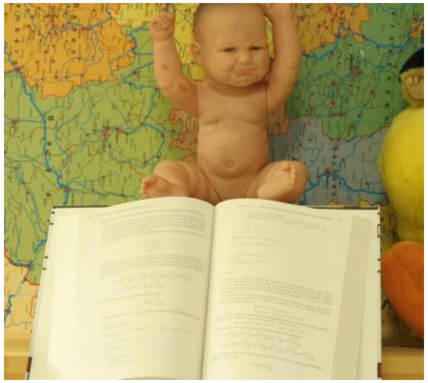
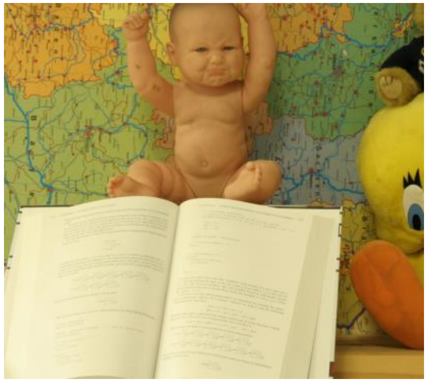
PMBP
3100 sec.



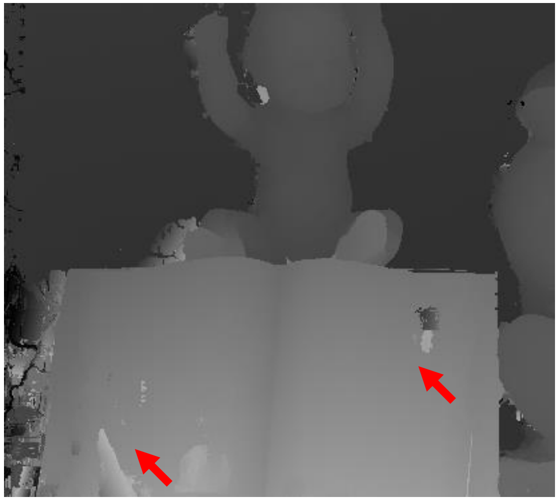
SPM-BP (ours)
30 sec.

Much faster than PMBP, and much better than PMF for textureless regions

Stereo results

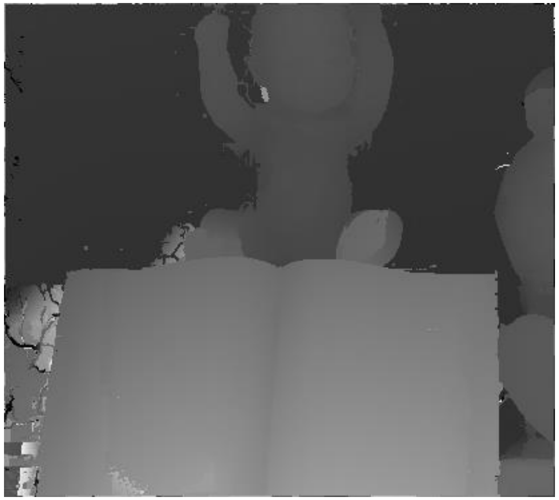



Stereo input



PMF
20 sec.

PMBP
3100 sec.



SPM-BP (ours)
30 sec.

Optical flow results

Optical flow input

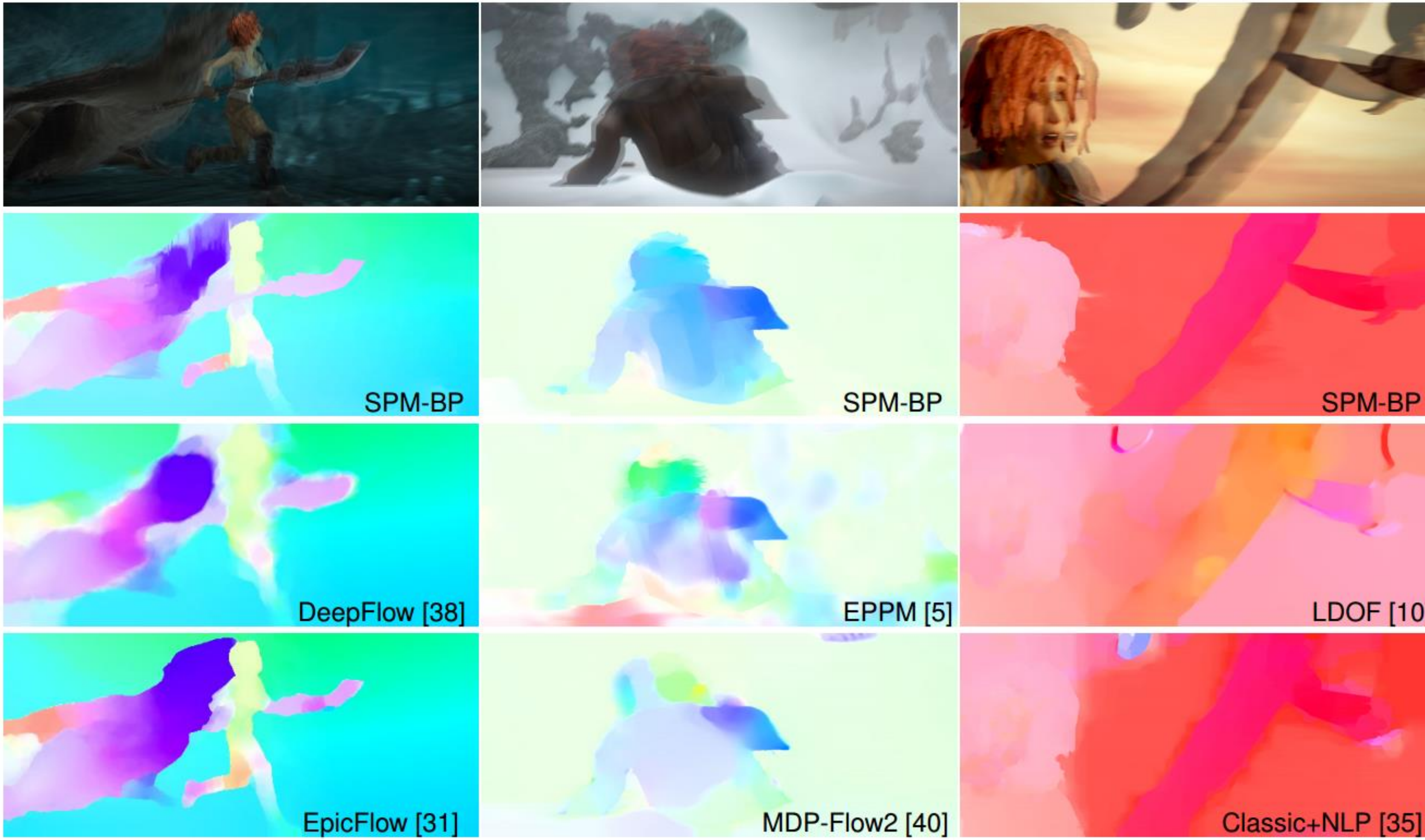
PMBP
2103 sec.

PMF
27 sec.

SPM-BP (ours)
42 sec.

Much faster than PMBP, and much better than PMF for textureless regions

Optical flow results



Performance Evaluation

Middlebury Stereo Performance (Tsukuba/Venus/Teddy/Cones)

Method	Avg. Rank	Avg. Error	Runtime(s)
PM-PM [39]	8.2	7.58	34 (GPU)
PM-Huber [17]	8.4	7.33	52 (GPU)
SPM-BP	12.1	7.71	30
PMF [24]	12.3	7.69	20
PMBP [7]	19.8	8.77	3100

Middlebury Stereo 2006 Performance

Dataset	PMF [25]	PMBP [7]	**SPM-BP**
Baby2	15.34	16.85	**12.82**
Books	**22.15**	27.57	22.52
Bowling2	15.95	15.20	**14.35**
Flowerpots	**24.59**	27.97	24.80
Lampshade1	25.02	30.22	**23.39**
Laundry	**26.77**	33.90	27.32
Moebius	21.47	25.09	**21.09**
Reindeer	**15.04**	21.57	16.02
Mean	20.79	24.79	**20.29**

Optical Flow Performance on MPI Sintel Benchmark
(captured on 16/04/2015)

Method	EPE all Clean	Rank	EPE all Final	Rank	Runtime (Sec)
EpicFlow [30]	4.115	1	6.285	1	17
PH-Flow [41]	4.388	2	7.423	8	800
SPM-BP	5.202	5	7.325	6	42
DeepFlow [36]	5.377	7	7.212	4	19
LocalLayering [33]	5.820	13	8.043	13	-
MDP-Flow2 [38]	5.837	14	8.445	21	754
EPPM [5]	6.494	18	8.377	20	0.95*
S2D-Matching [21]	6.510	19	7.872	10	2000
Classic+NLP [34]	6.731	21	8.291	19	688
Channel-Flow [32]	7.023	24	8.835	26	>10000
LDOF [10]	7.563	25	9.116	28	30

Remarks

- A simple formulation, without needing *complex* energy terms nor a separate *initialization*
- Achieved top-tier performance, even when compared to *task-specific* techniques
- Applied on the full pixel grid, avoiding *coarse-to-fine* steps

Conclusion

- SPM-BP is simple, effective and efficient
- Takes the best computational advantages of
  - **efficient edge-aware cost filtering**
  - and **superpixel-based particle-sampling for message passing**
- Offers itself as a general and efficient global optimizer for continuous MRFs
- Future work
  - *Robust* dense correspondences for cross-scene matching
  - Dealing with *high-order* terms in MRF

Code available online:
http://publish.illinois.edu/visual-modeling-and-analytics/efficient-inference-for-continuous-mrfs/