A Novel Trilateral Filter based Adaptive Support Weight Method for Stereo Matching

Dongming CHEN, Mohsen ARDABILIAN, Liming CHEN

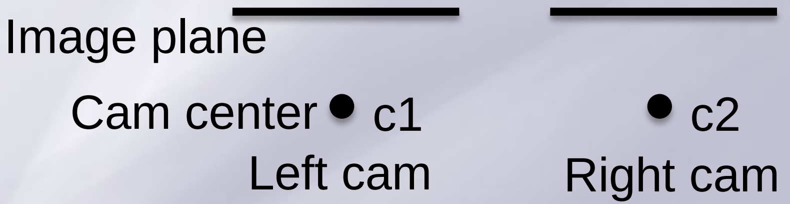
Laboratoire d'**I**nfo**R**matique en **I**mage et **S**ystèmes d'information

LIRIS UMR 5205 CNRS, Ecole Centrale de Lyon

http://liris.cnrs.fr

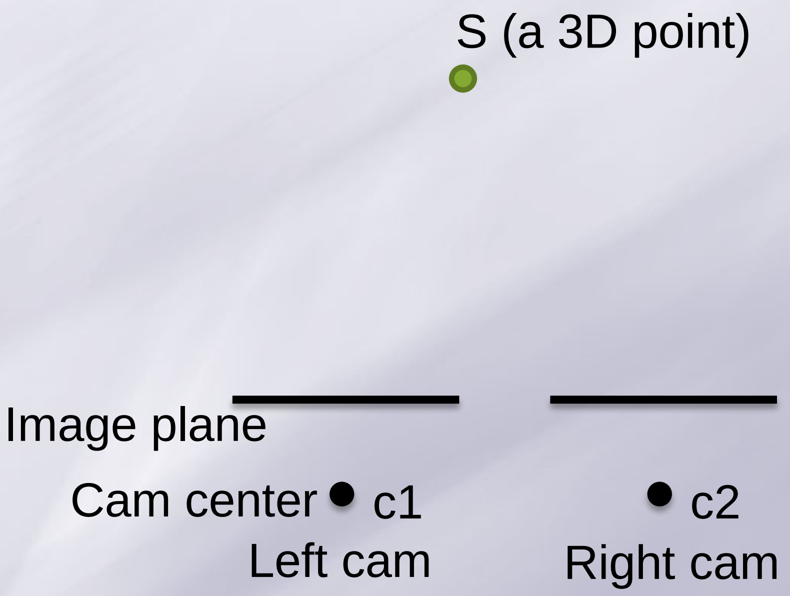
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



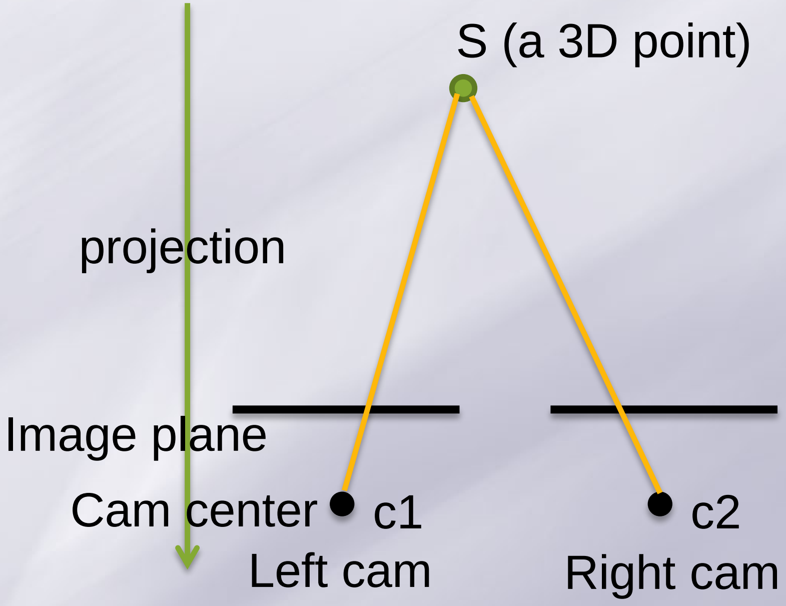
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



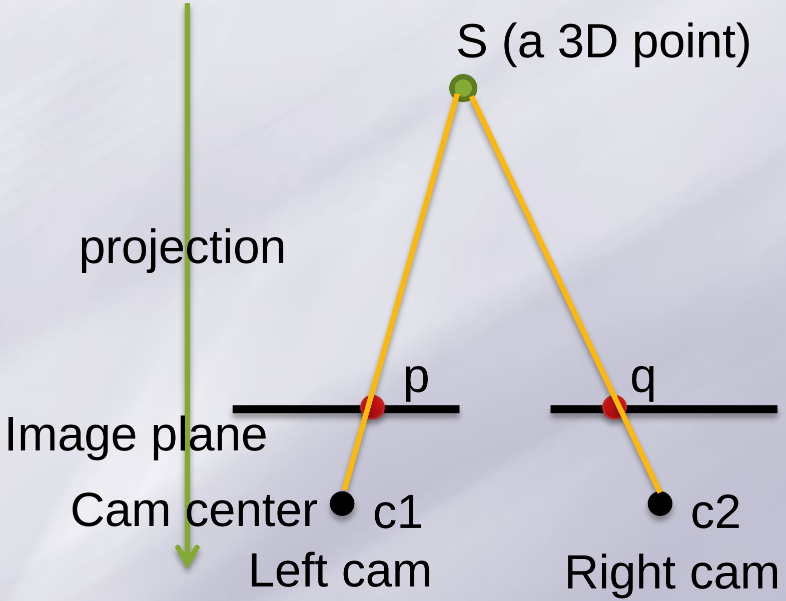
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



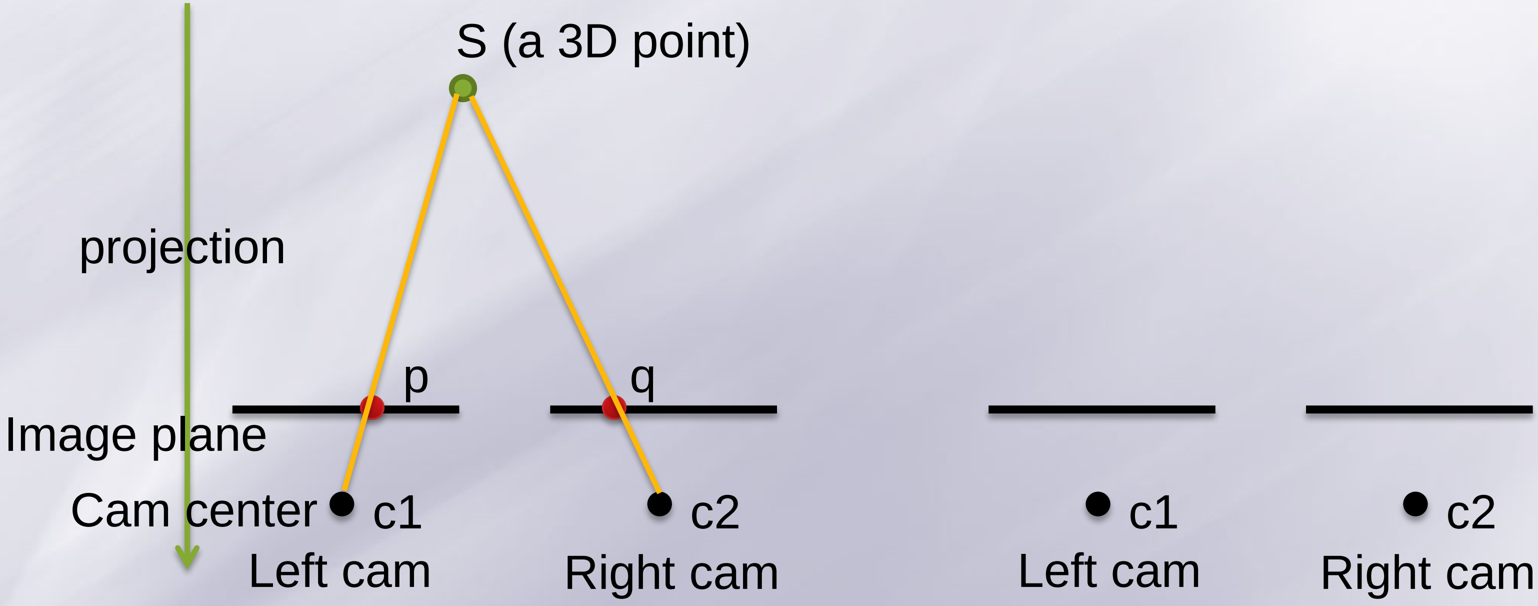
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



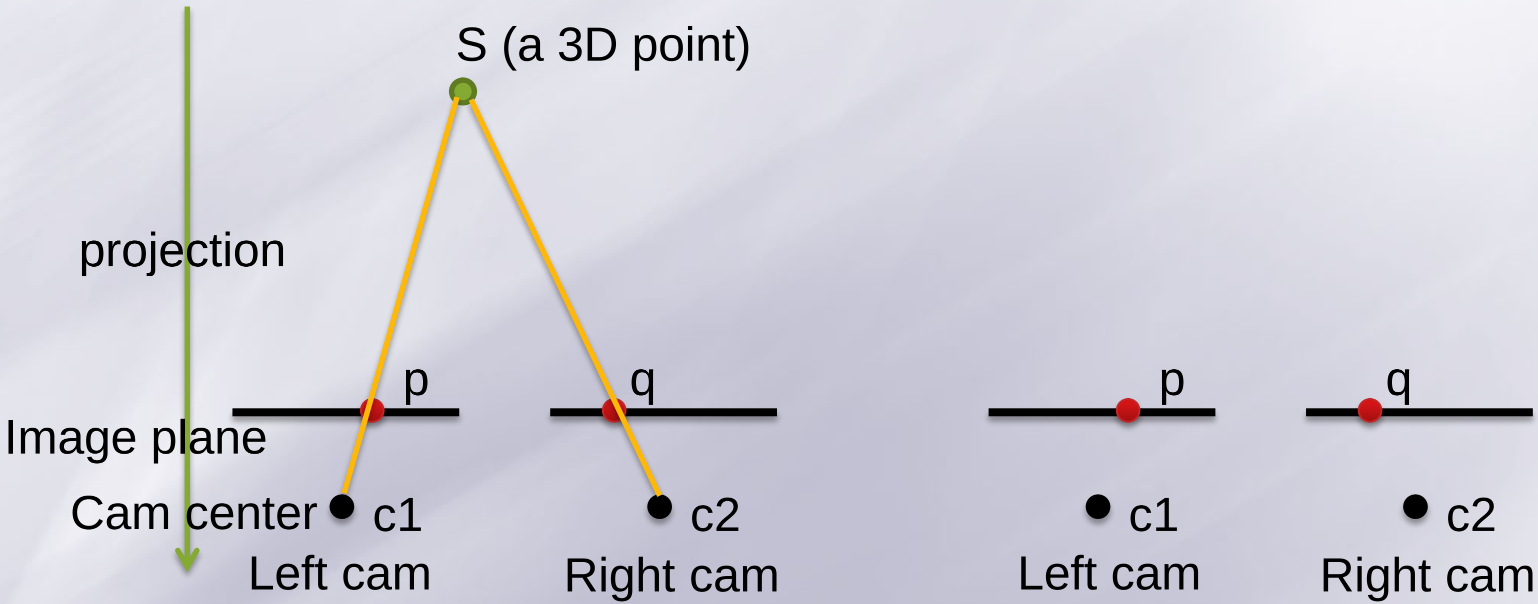
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



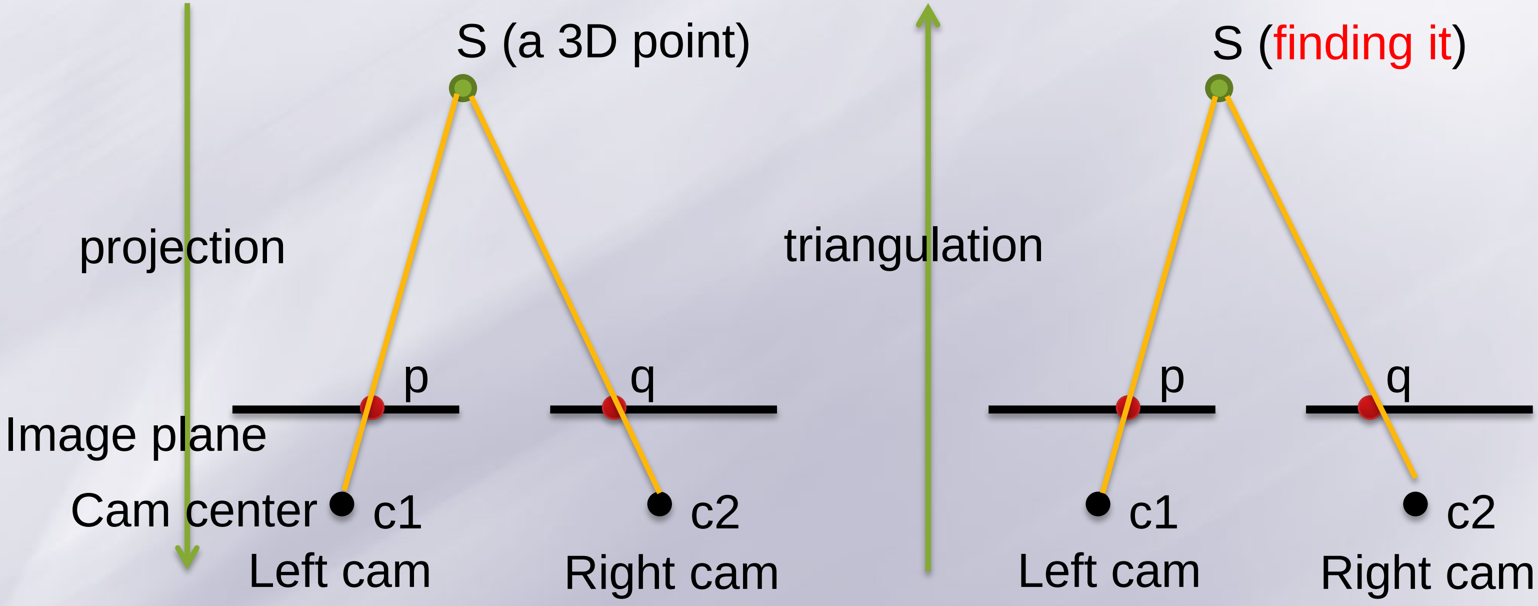
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



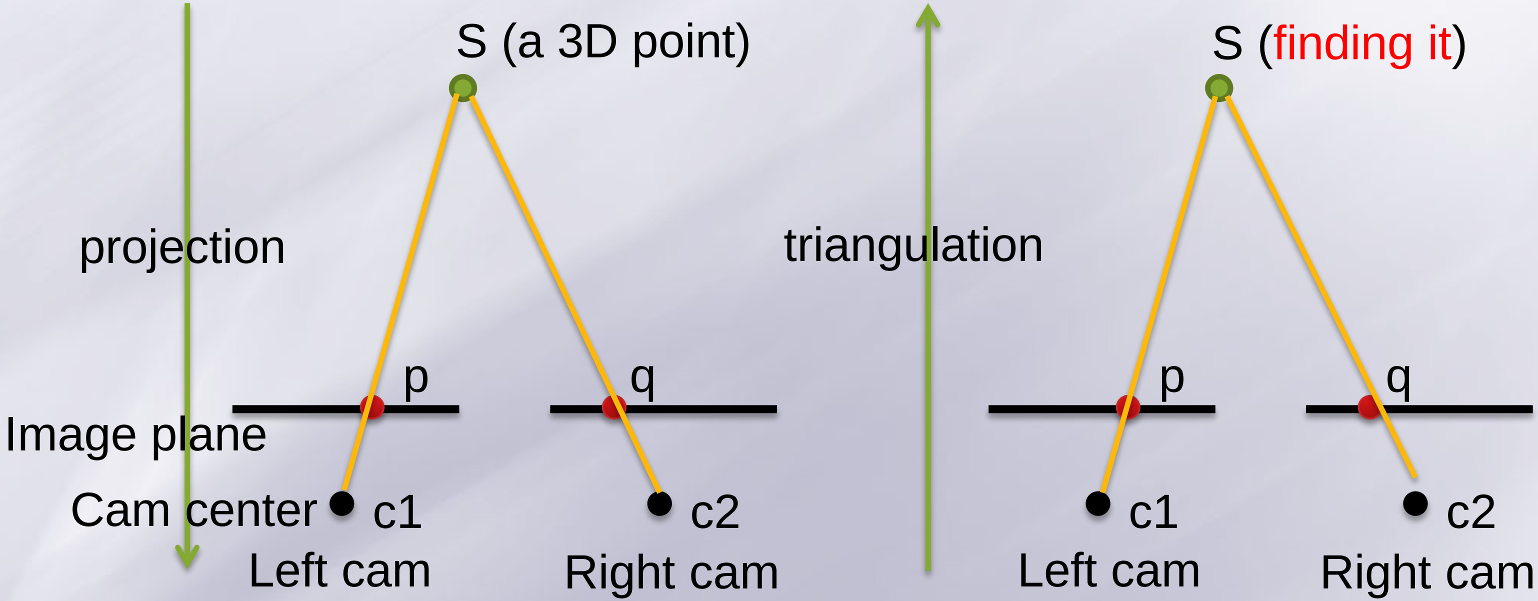
Objective

- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



Objective

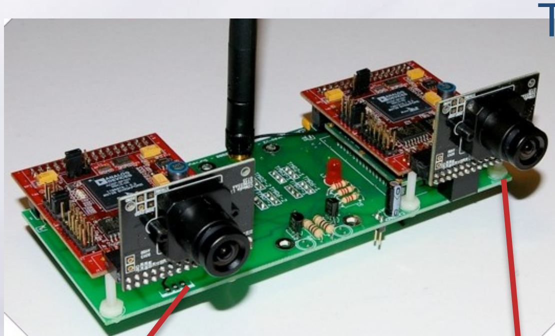
- Endowing the computer with human like depth skill
  - Human: two eyes + a brain = depth perception
  - Stereo vision system: two cameras + PC/FPGA = depth perception



How to find the corresponding points？

Objective

- How to find the corresponding points in a pair of stereo images?
  - Stereo matching algorithms



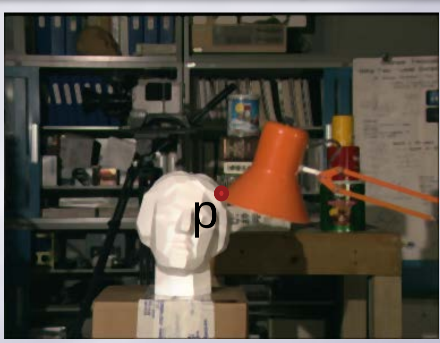
Two cameras are calibrated

After rectification: the corresponding points are in the same height (py = qy)

image rectification

disparity means px - qx

x



p

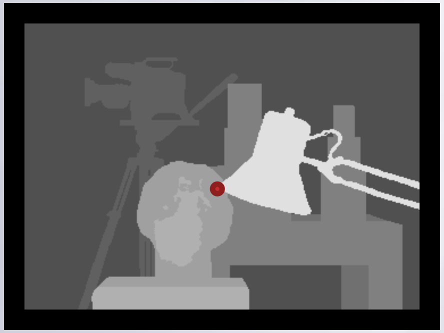
y

left image

x

y

right image



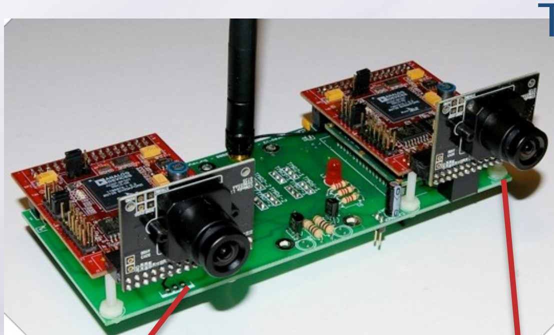
disparity map

inversely proportional to depth map

stereo matching

Objective

- How to find the corresponding points in a pair of stereo images?
  - Stereo matching algorithms

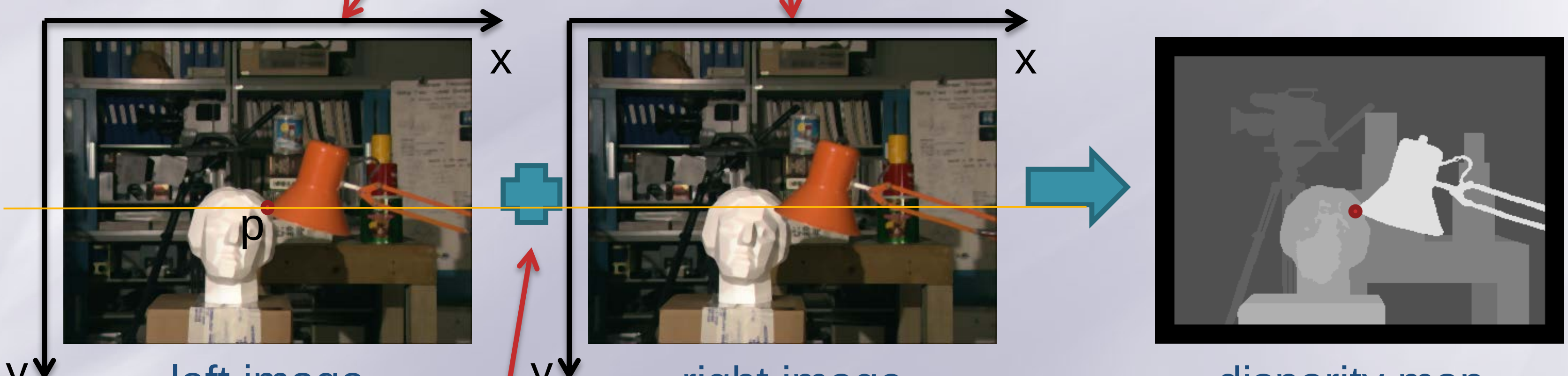


Two cameras are calibrated

After rectification: the corresponding points are in the same height (py = qy)

image rectification

disparity means px - qx



left image

right image

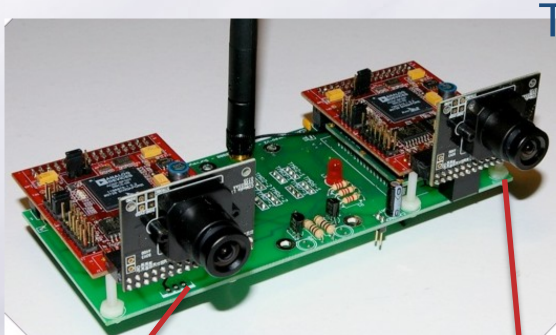
disparity map

inversely proportional to depth map

stereo matching

Objective

- How to find the corresponding points in a pair of stereo images?
  - Stereo matching algorithms

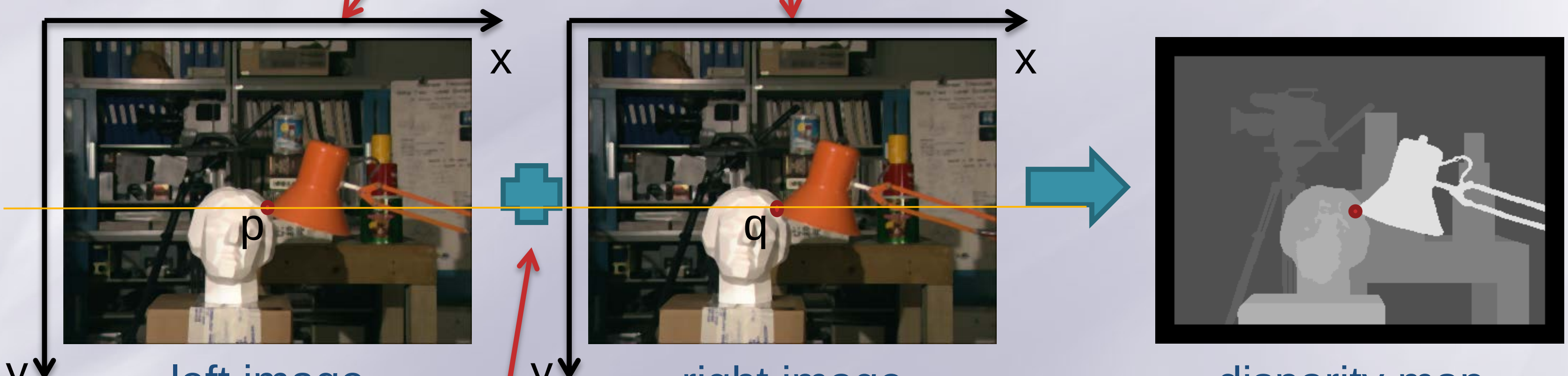


Two cameras are calibrated

After rectification: the corresponding points are in the same height (py = qy)

image rectification

disparity means px - qx



left image

right image

disparity map

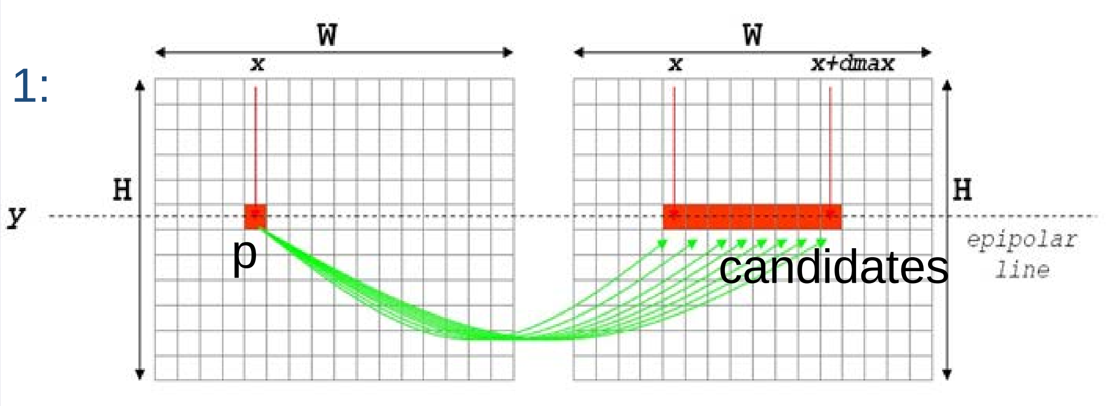
inversely proportional to depth map

stereo matching

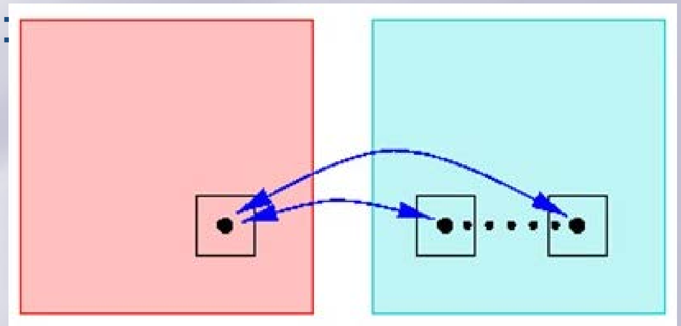
Pipeline

- Generally, local stereo matching algorithms follow four steps:
  - Cost computation: calculate the matching cost between p and its candidate points
  - Cost aggregation: aggregate all costs within a support window → compare windows
  - Disparity optimization: winner take all strategy, local optimal among candidate windows
  - Refinement: post-processing step to remove mismatches

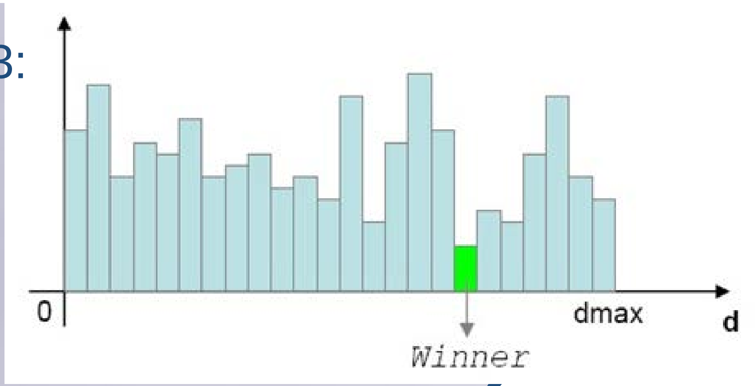
Step 1:



Step 2:



Step 3:

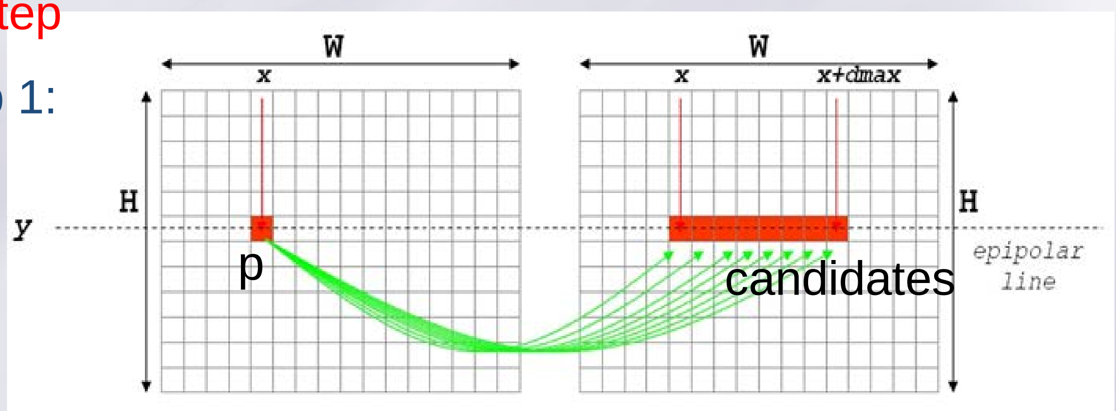


Pipeline

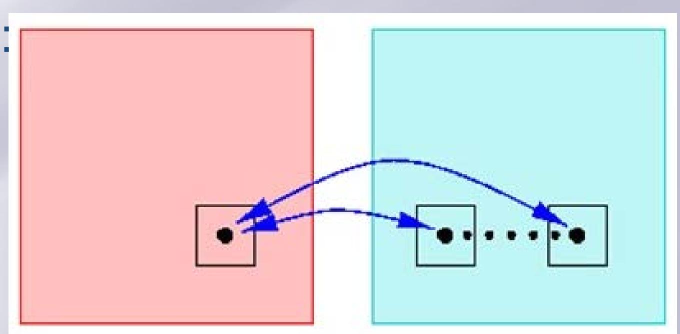
- Generally, local stereo matching algorithms follow four steps:
  - Cost computation: calculate the matching cost between p and its candidate points
  - Cost aggregation: aggregate all costs within a support window → compare windows
  - Disparity optimization: winner take all strategy, local optimal among candidate windows
  - Refinement: post-processing step to remove mismatches

important step

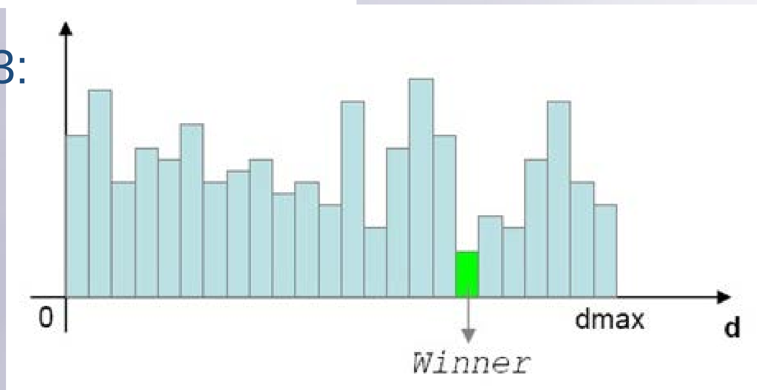
Step 1:



Step 2:



Step 3:



Related work

- How to choose a window?
  - Important assumption: all pixels within the support window should be at the same depth with the center pixel

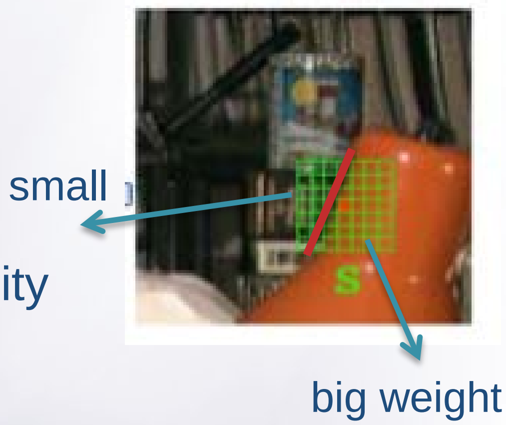
Occlusion at depth discontinuity
（background & foreground）

S. Mattoccia's slides

  - Approaches:
    - Multiple window approach: change the window shape
    - Variable window approach: change the window size & shape
    - Shiftable window approach: change the window center offset
    - Adaptive support weight (ASW) approach – best local method

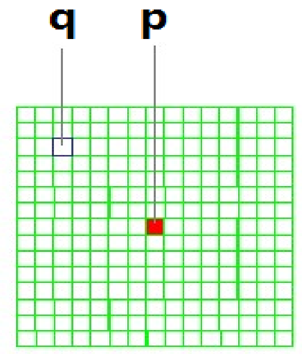
Related work

- Adaptive support weight (ASW) methods
  - Idea:
    - assign a weight for each pixel of the support window
    - aggregate the weighted costs
  - Weight ~ Likelihood of two pixels at the same disparity
    - more likely in the same disparity → big weight
    - less likely in the same disparity → small weight
  - Assign weight = change window size, shape, center offset



- How to calculate the weight for each pixel?
  - Bilateral filter weight function *[Yoon PAMI’06]* (BF)
  - Segmented bilateral filter weight function *[Tombari PSIVT’07]* (BFSeg)
  - Geodesic weight function *[Hosni ICIP’09]* (GEO)
  - Guided filter weight function *[Rhemann CVPR’11]* (GF)
  - ……
- Which function is the most accurate?
  - Hosni *[Hosni CVIU’13]* evaluated various weight functions on a large datasets, suggesting that both BF and GF are the most accurate functions

Bilateral filter weight function



- Two rules (for pixel p and q):
  - Spatially close → more likely in the same disparity
  - Similar in color → more likely in the same disparity

- Bilateral filter weight function *[Yoon PAMI'06]* :
  - Spatial proximity term: Euclidean distance between coordinates (x, y)

$$\Delta g_{pq} = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

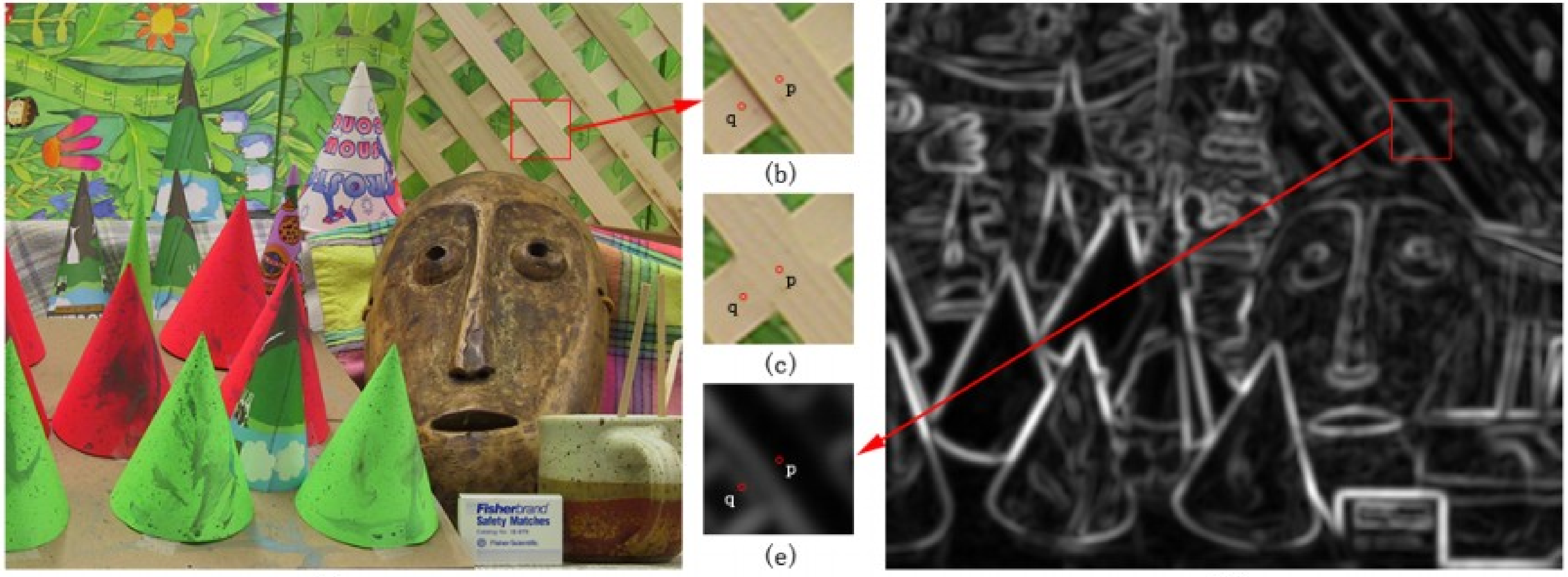
  - Color similarity term: Euclidean distance between the colors, CIELab color space

$$\Delta c_{pq} = \sqrt{\sum_{j \in (L,a,b)} (I_j(p) - I_j(q))^2}$$

  - → bf weight:

$$w_{bf}(p,q) = e^{-\frac{\Delta c_{pq}}{\gamma_c}} e^{-\frac{\Delta g_{pq}}{\gamma_g}}$$

Drawback of the bilateral filter weight



(a) (d)

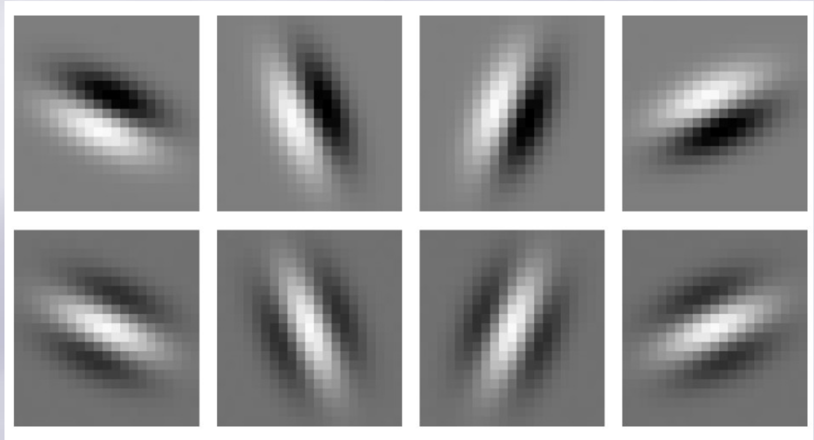
- From bilateral filter, W(p, q) in (b) equals to W(p, q) in (c)
- Actually, they should not equal, because in (c) these two pixels are in the same depth but in (b) they are not.
- Our method: use boundary cue in (d)
  - If there is a boundary between two pixels, their weight should be small
  - If not, their weight should be big

The proposed method

- Trilateral filter weight function
  - A new boundary strength term
  - The boundary strength at pixel p is defined as [Robbins IVC'97]:

$$E(p) = \sum_{\theta} \sqrt{(I(p) * F_{\theta,odd})^2 + (I(p) * F_{\theta,even})^2}$$

$F_{\theta,odd}$ and $F_{\theta,even}$ is a pair of quadrature filters in different orientation



Four odd filters

Four even filters

  - The boundary strength term is defined as:

$$e^{-\frac{\Delta E_{pq}}{\gamma_e}}$$

The proposed method

- If there is not a boundary between p and q, ΔE_{pq} is zero
- If there is a boundary, ΔE_{pq} is not zero, but proportional to the boundary strength between them.

Our trilateral filter weight function

- use the boundary strength term to modify the bilateral filter
- inspired by the cue combination strategy [Cour CVPR'05]

$$w_{tf}(p,q) = e^{-\frac{\Delta c_{pq}}{\gamma_c}} e^{-\frac{\Delta g_{pq}}{\gamma_g}} \left(e^{-\frac{\Delta E_{pq}}{\gamma_e}} + e^{-\frac{\Delta c_{pq}}{\gamma_c}} e^{-\frac{\Delta g_{pq}}{\gamma_g}} \right)$$

$$\rightarrow \quad w_{tf}(p,q) = \underbrace{e^{-\frac{\Delta c_{pq}}{\gamma_c}} e^{-\frac{\Delta g_{pq}}{\gamma_g}} e^{-\frac{\Delta E_{pq}}{\gamma_e}}}_{\text{Modification term}} + \underbrace{e^{-\frac{2\Delta c_{pq}}{\gamma_c}} e^{-\frac{2\Delta g_{pq}}{\gamma_g}}}_{\text{BF weight}}$$

Modification term

BF weight

Experimental results

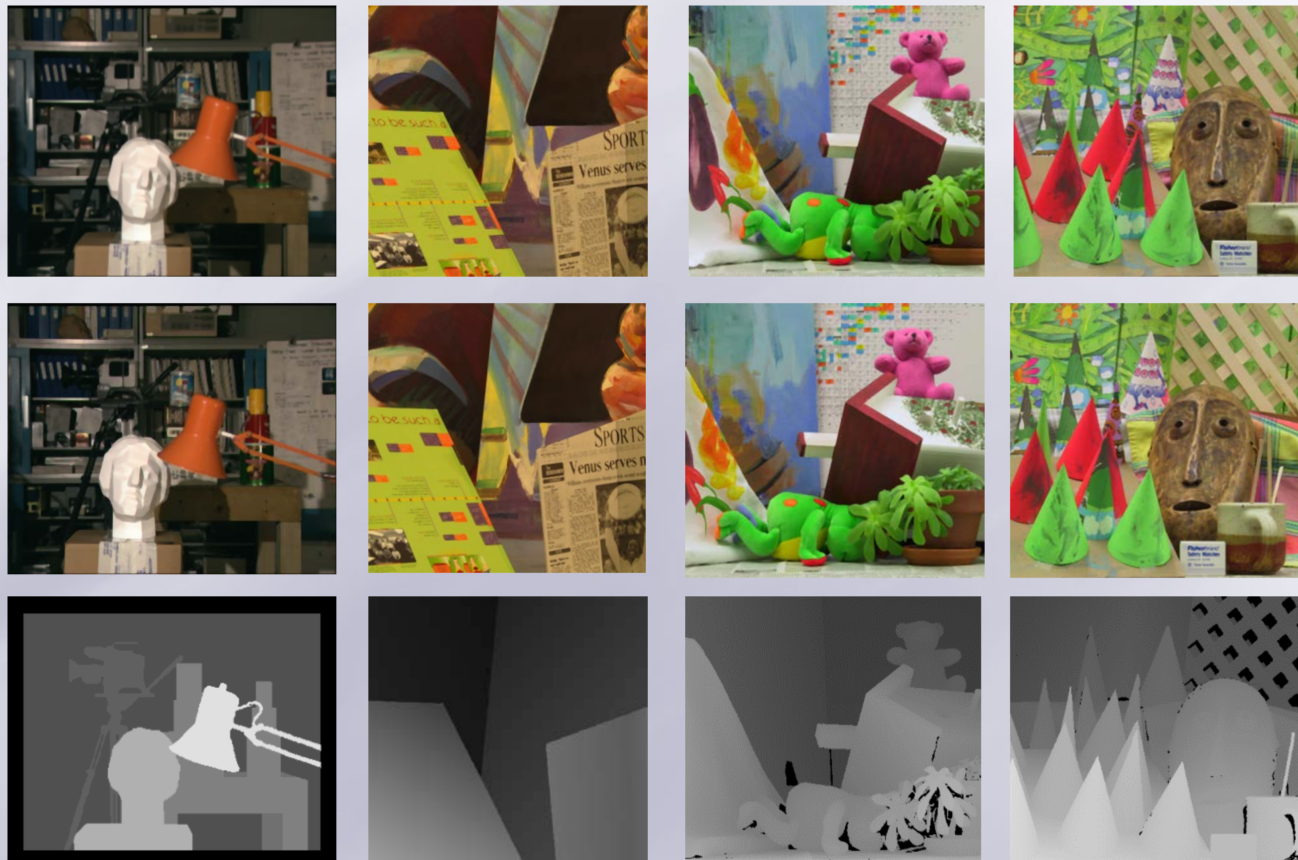
- Evaluation on Middlebury benchmark, using four standard pairs of stereo images. http://vision.middlebury.edu/stereo/

Tsukuba Venus Teddy Cones

Left image

Right image

Ground truth



Experimental results

- Overall comparison of our TF method with four state-of-the-art methods

Algorithm	Rank	tuskuba			venus			teddy			cones			Avg. Error
		nocc	all	disc	nocc	all	disc	nocc	all	disc	nocc	all	disc	
Our TF	15	1.65	1.96	**5.90**	**0.14**	0.31	**1.51**	6.25	**11.8**	**15.1**	**2.49**	8.32	**7.02**	5.21
GF [16]	30	1.51	1.85	7.61	0.20	0.39	2.42	**6.16**	11.8	16.0	2.71	**8.24**	7.66	5.55
GEO [10]	41	1.45	1.83	7.71	0.14	**0.26**	1.90	6.88	13.2	16.1	2.94	8.89	8.32	5.80
BFSeg [21]	66	**1.25**	**1.62**	6.68	0.25	0.64	2.59	8.43	14.2	18.2	3.77	9.87	9.77	6.44
BF [25]	80	1.38	1.85	6.90	0.71	1.19	6.13	7.88	13.3	18.6	3.97	9.79	8.26	6.67

Table 1: The quantitative comparison of our trilateral filter based ASW algorithm and four popular ASW algorithms on the Middlebury benchmark with error threshold 1. Our algorithm outperforms others in most columns, especially in "disc." column.

Figure 2: The term 'disc.' in Table 1 and Table 2 means the regions near depth discontinuities (white areas), occluded and border regions (black), and other regions (gray). In 'disc.' column, errors are only evaluated in the white areas, which mainly contain boundaries.

Experimental results

- Comparison of weight functions
  - Only compare our TF weigh function with BF and GF, fixing other steps (cost computation, disparity optimization, refinement)

Test	Weight	Refinement	Rank	Avg.Error
1	w_{tf}	—	97	7.55
2	w_{bf}	—	105	8.10
3	w_{gf}	—	103	8.30
4	w_{tf}	ours	15	5.21
5	w_{bf}	ours	29	5.45
6	w_{gf}	ours	45	5.96

Table 3: Overall comparison the proposed weight function (w_{tf}) with the bilateral filter weight function (w_{bf}) and the guided filter weight function (w_{gf}). The detail comparison is presented in Table 2. The other steps are set the same and only weight functions are compared.

Conclusion & future work

- A trilateral filter based ASW method is proposed and the experimental results demonstrate its effectiveness.
- We will evaluate the proposed method on a large dataset.
- We will improve the computational efficiency of the proposed method.

Reference

- [5] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In CVPR, pages 1124–1131, 2005.
- [10] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann. Local stereo matching using geodesic support weights. In ICIP, pages 2069–2072, 2009.
- [11] A. Hosni, M. Bleyer, and M. Gelautz. Secrets of adaptive support weight techniques for local stereo matching. CVIU, 117:620–632, 2013.
- [16] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In CVPR, pages 3017–3024, 2011.
- [17] B. Robbins and R. A. Owens. 2d feature detection via local energy. Image and Vision Computing, 15(5):353–368, 1997
- [21] F. Tombari, S. Mattoccia, and L. Di Stefano. Segmentation-based adaptive support for accurate stereo correspondence. In PSIVT, volume 1, pages 427 – 438, 2007.
- [25] K. Yoon and I. Kweon. Adaptive support-weight approach for correspondence search. PAMI, 28(4):650–656, 2006.

Thank you for your attention!