Stereo Matching

Srikumar Ramalingam
School of Computing
University of Utah

Reference

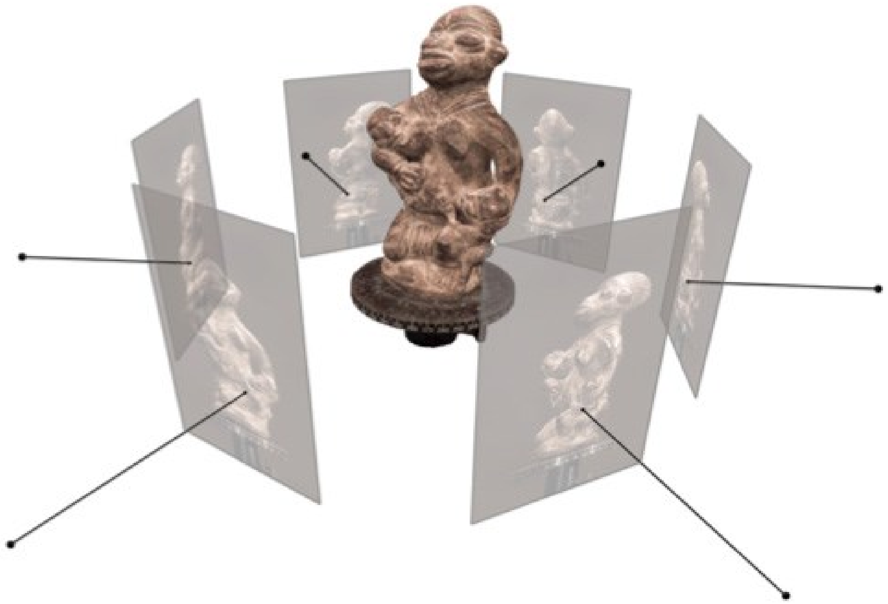
- Daniel Scharstein and Richard Szeliski, A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms, IJCV, 2002.

  http://vision.middlebury.edu/stereo/taxonomy-IJCV.pdf

Dense Correspondence in Computer Vision

Binocular Stereo

Multiview stereo

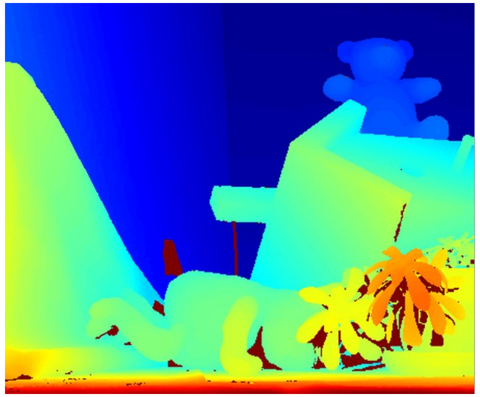


Slide Courtesy: Sudipta Sinha

Stereo Matching


Left (reference)


Right


Left Disparity Map

- Dense pixel correspondence in rectified pairs
- Disparity Map: $D(x, y)$ *(treated as +ve)*

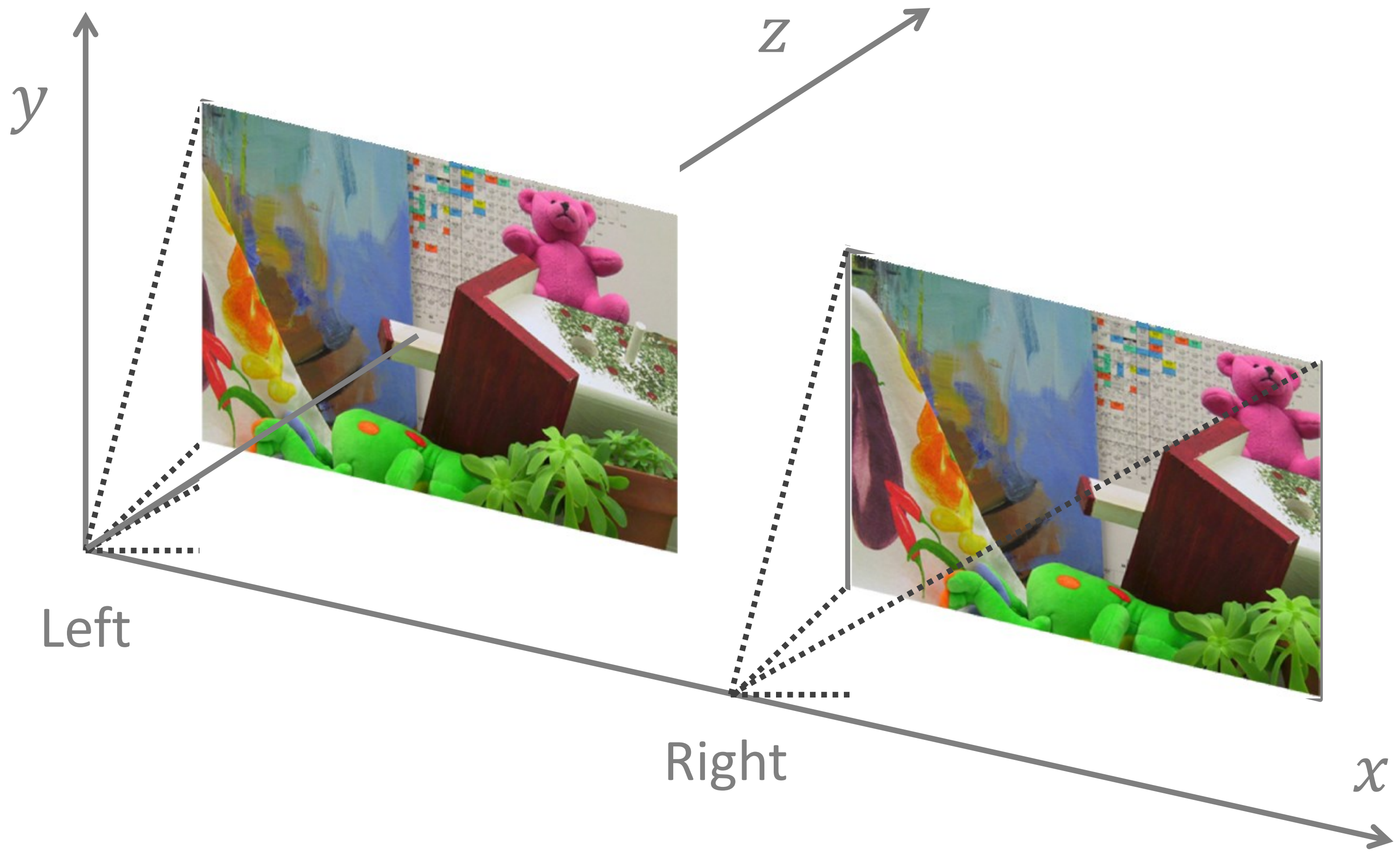
$$x' = x - D(x, y), \quad y' = y$$

- Depth Map: $Z(x, y) = \dfrac{baseline * focal\ length}{D(x,y)}$


Depth Map

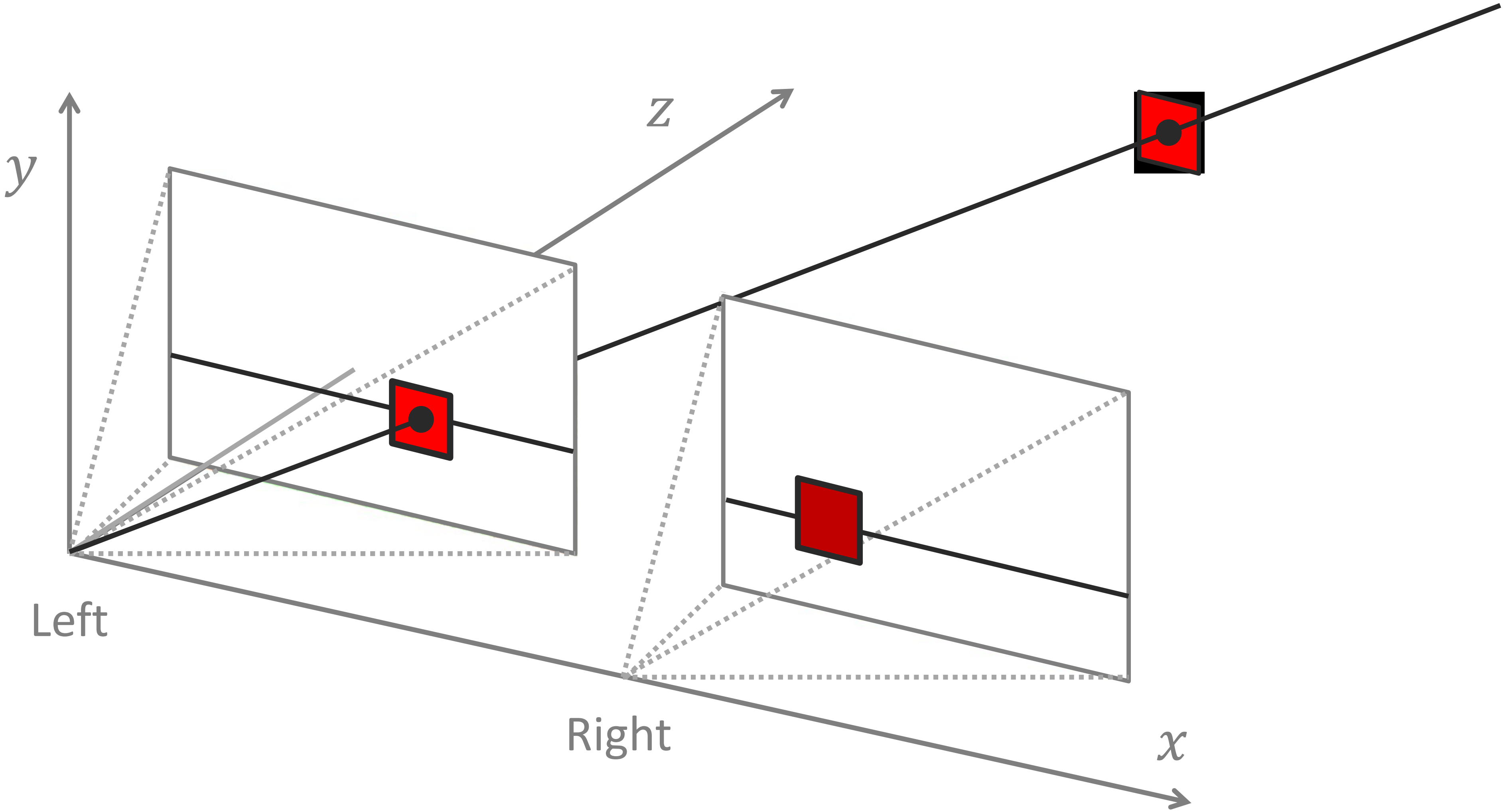
Slide Courtesy: Sudipta Sinha

Binocular Stereo Matching



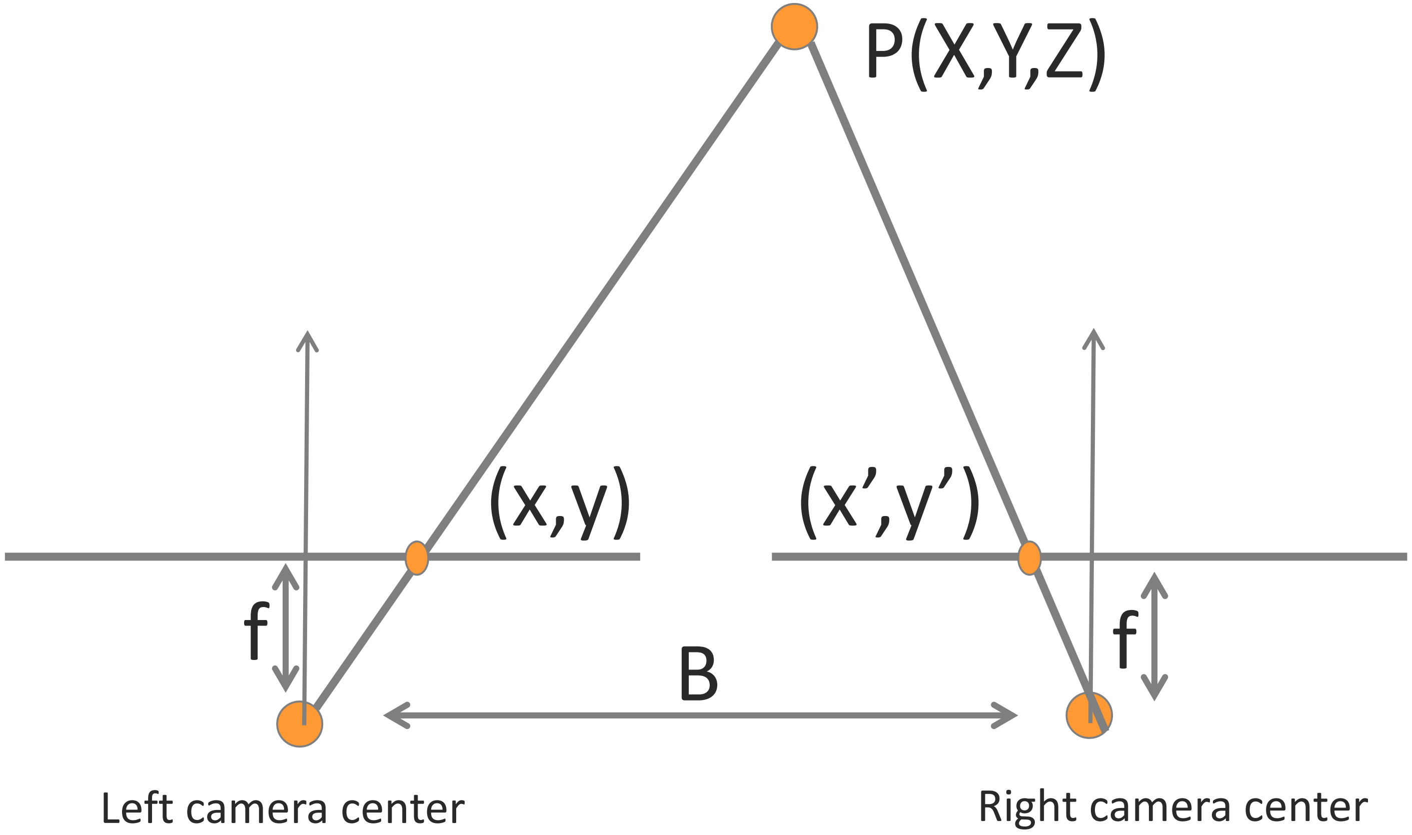
Slide Courtesy: Sudipta Sinha

Binocular Stereo Matching



Slide Courtesy: Sudipta Sinha

Binocular stereo matching (Top View)



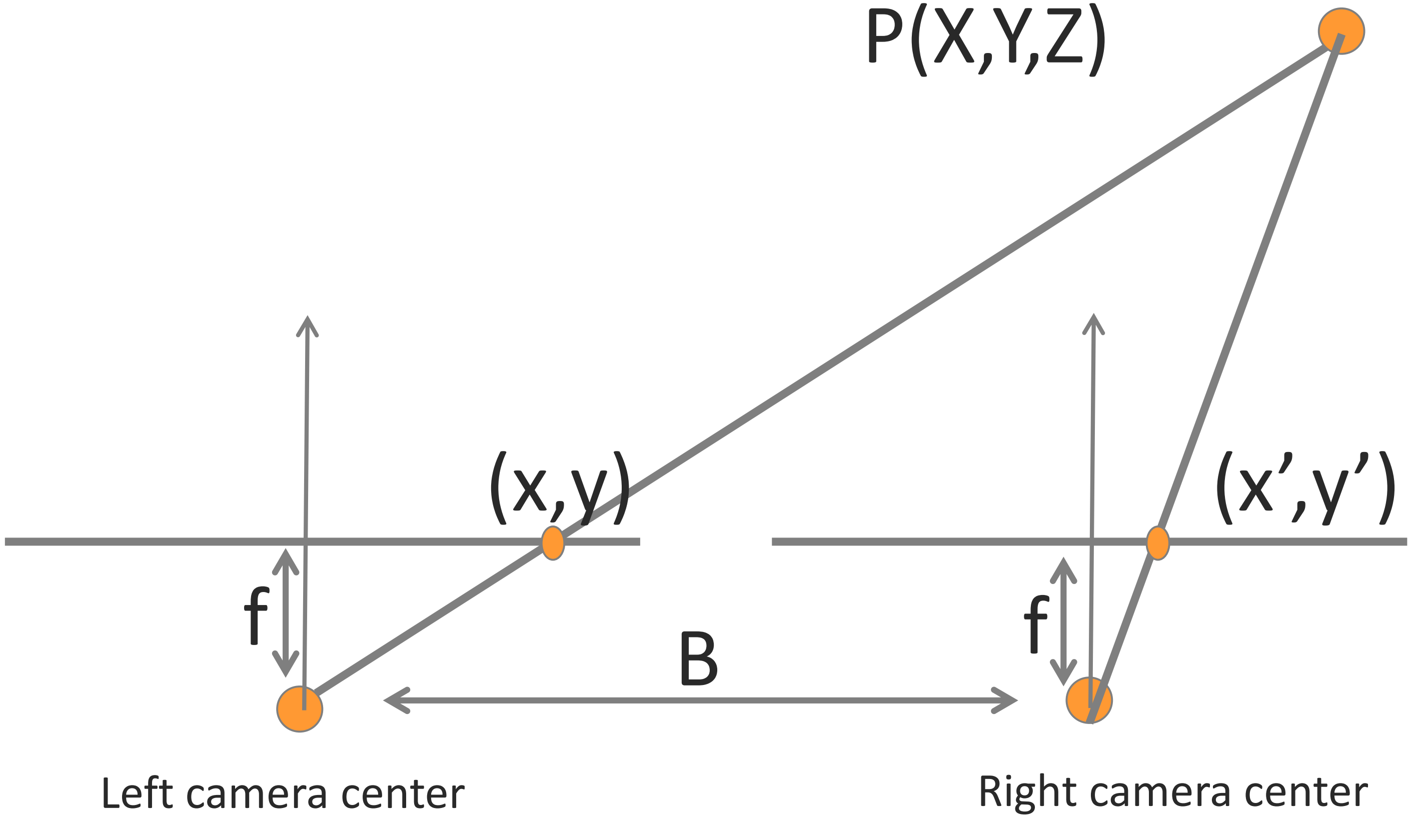
D(x,y) = x-x'

Z=bf/Disparity

X=Zx/f

Y=Zy/f

Binocular stereo matching (Top View)



$D(x,y) = x - x'$

$Z = bf / \text{Disparity}$

$X = Zx / f$

$Y = Zy / f$

Matching costs C(x,y,d)

- Find pairs of pixels (or local patches) with similar appearance
- Minimize *matching cost (maximize photo-consistency)*
  - **Patch**-based (parametric vs non-parametric)
    - *Sum of Absolute Difference (SAD),*
    - *Sum of Squared Difference (SSD),*
    - *Normalized Cross Correlation (ZNCC)*
  - **Descriptor**-based
    - *(hand-crafted features) SIFT, DAISY, ..*
    - *(learnt features) Deep learning (revisit later)*

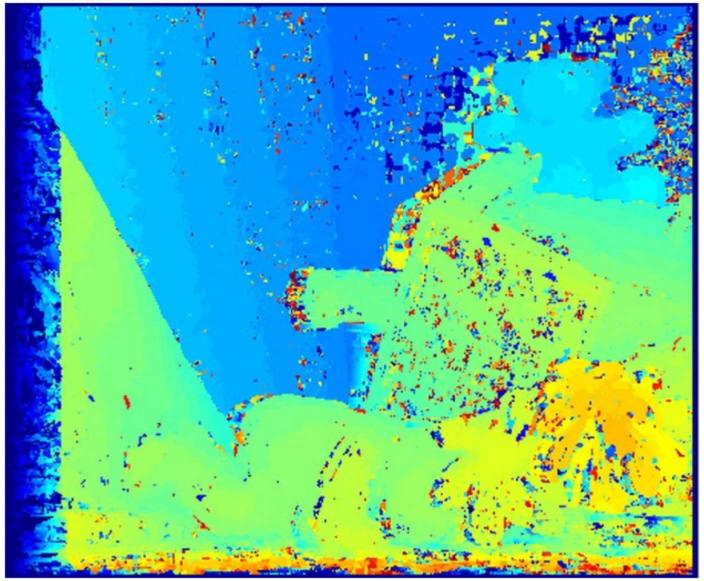
$$C_{SAD}(\mathbf{p},\mathbf{d}) = \sum_{\mathbf{q}\in N_{\mathbf{p}}} |I_L(\mathbf{q}) - I_R(\mathbf{q}-\mathbf{d})|$$

$$C_{ZNCC}(\mathbf{p},\mathbf{d}) = \frac{\sum_{\mathbf{q}\in N_{\mathbf{p}}}(I_L(\mathbf{q}) - \bar{I}_L(\mathbf{p}))(I_R(\mathbf{q}-\mathbf{d}) - \bar{I}_R(\mathbf{p}-\mathbf{d}))}{\sqrt{\sum_{\mathbf{q}\in N_{\mathbf{p}}}(I_L(\mathbf{q}) - \bar{I}_L(\mathbf{p}))^2 \sum_{\mathbf{q}\in N_{\mathbf{p}}}(I_R(\mathbf{q}-\mathbf{d}) - \bar{I}_R(\mathbf{p}-\mathbf{d}))^2}}$$

Slide Courtesy: Sudipta Sinha

Local Optimization

- Minimize matching cost at each pixel in the left image <u>independently</u>
- Winner-take-all (WTA)



Slide Courtesy: Sudipta Sinha

Local evidence not enough …

- Photometric Variations
- Reflections
- Transparent surfaces
- Texture-less Areas
- Non-Lambertian Surfaces
- Repetitive patterns
- Complex Occlusions

(Image Source: Lectures on stereo matching, Christian Unger and Nassir Navab, TU Munchen)
http://campar.in.tum.de/twiki/pub/Chair/TeachingWs09Cv2/3D_CV2_WS_2009_Stereo.pdf

Local evidence not enough …

- Photometric Variations
- Reflections ➡
- Transparent surfaces
- Texture-less Areas
- Non-Lambertian Surfaces
- Repetitive patterns
- Complex Occlusions

(Image Source: Lectures on stereo matching, Christian Unger and Nassir Navab, TU Munchen)
http://campar.in.tum.de/twiki/pub/Chair/TeachingWs09Cv2/3D_CV2_WS_2009_Stereo.pdf

Local evidence not enough …

- Photometric Variations
- Reflections
- Transparent surfaces
- Texture-less Areas
- Non-Lambertian Surfaces
- Repetitive patterns
- Complex Occlusions

(Image Source: Lectures on stereo matching, Christian Unger and Nassir Navab, TU Munchen)
http://campar.in.tum.de/twiki/pub/Chair/TeachingWs09Cv2/3D_CV2_WS_2009_Stereo.pdf

Global Optimization

- Solve for all disparities simultaneously …
- Solve a pixel labeling problem
- Labels are discrete (ordered), $d \in L_D$

$$L_{\mathrm{D}} = [d_{min}, d_{max}]$$

- Incorporate regularization into objective

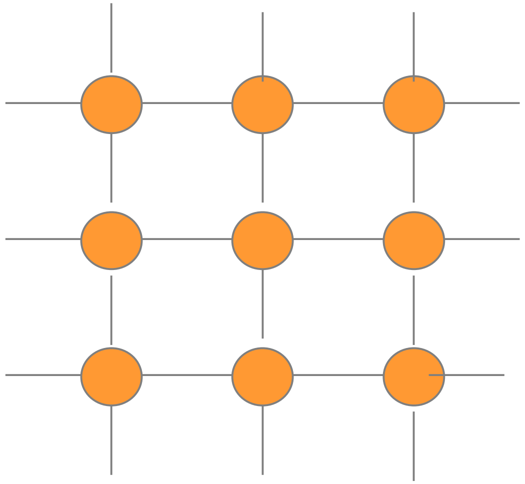
$$E(d) = E_{data}(d) + \lambda E_{smooth}(d)$$

- Data term encodes matching costs
- Smoothness term encodes priors
  - Encourage adjacent pixels to take similar disparities

Slide Courtesy: Sudipta Sinha

Disparity Computation

Let the disparity map be given by d, where d(x,y) denotes the disparity for a pixel in the left image at location (x,y).
Let all the edges in the image graph be given by E by treating the image as a 4-connected or 8-connected graph:



$$E(d) = E_{data}(d) + \lambda E_{smooth}(d)$$

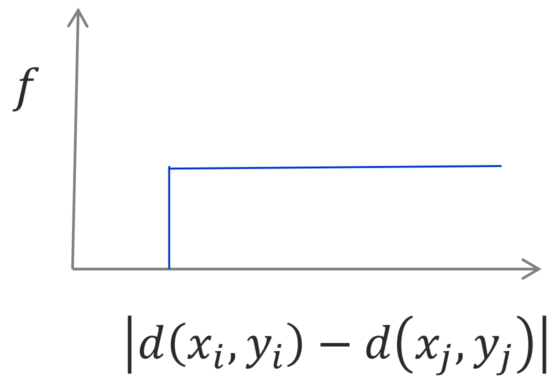
λ is the constant term that can be manually fixed or learned from data.

Data term:

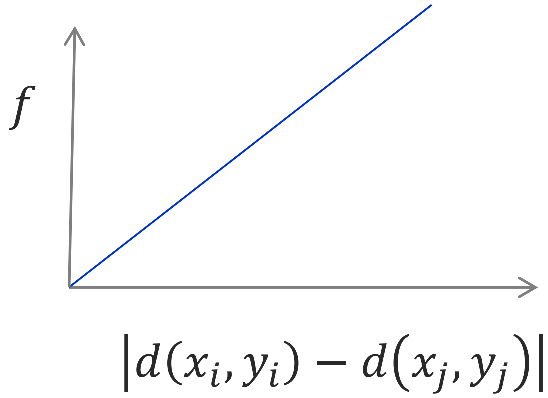
$$E_{data}(d) = \sum_{(x,y)} C(x, y, d(x, y))$$

Different smoothness terms

$$E_{smooth}(d) = \sum_{(i,j)\in E} f(d(x_i, y_i), d(x_j, y_j))$$



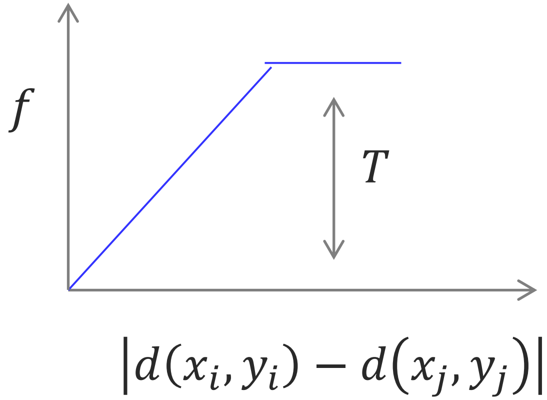
$$diff = |d(x_i, y_i) - d(x_j, y_j)|$$



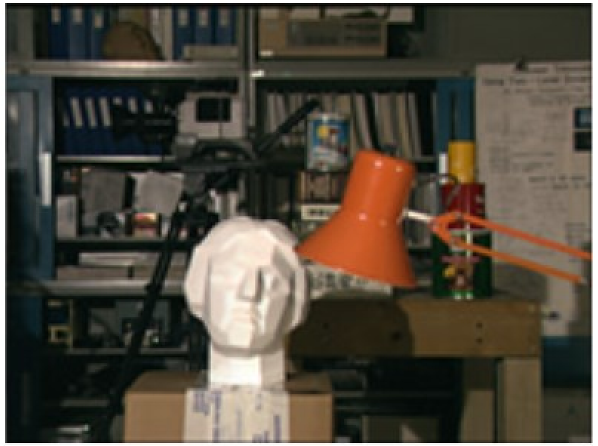
Potts: $f = 1 \quad if \quad diff > 0 \; else \; f = 0$ (Robust)

Linear: $f = diff$ (Not Robust)

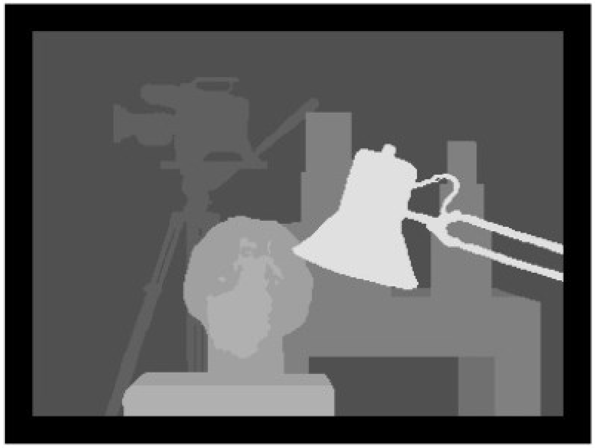
Truncated model: $f = diff \quad if \quad diff < T \; else \; f = T$ (Robust)



Stereo Image with Groundtruth

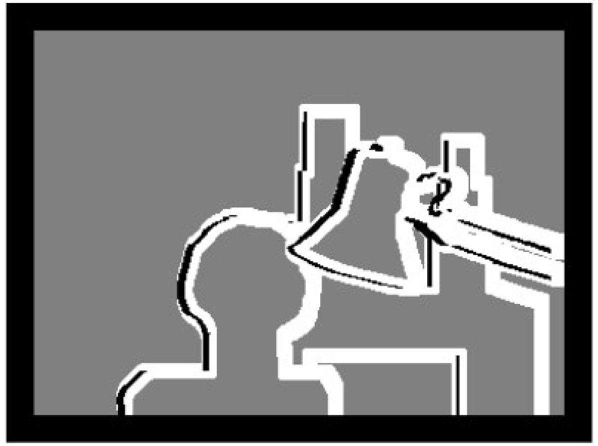


Left image from orig. stereo pair



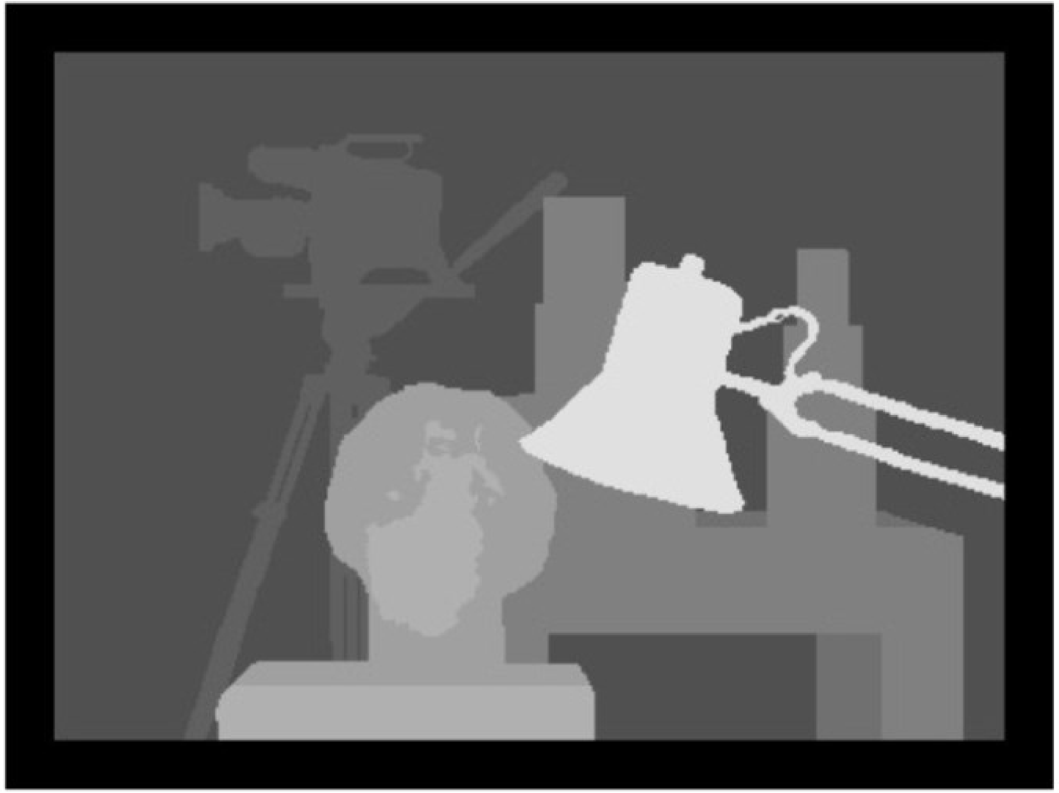
Groundtruth disparity

Texture-less regions



Depth discontinuities (white) and Occlusion (Black)

BP Results



True disparities

Belief propagation

Sample Problem (not the same as real setup)

You are given two images: I1 (2x2 pixel grid) and I2 (2x3 pixel grid) as shown in Figure 1. Find the match for every pixel in the first image I1. Every pixel $p(x, y)$ in I1 can be matched to a pixel $p'(x, y)$ or $p'(x+1, y)$ in I2. In other words, every pixel in I1 can have only two disparity states $[0, 1]$: 0 when $p(x, y)$ is matched to $p'(x, y)$, and 1 when $p(x, y)$ is matched to $p'(x+1, y)$. The unary for a pixel $p(x, y)$ (cost function that depends only on a single pixel in I1) is given by:

$U(0) = |p(x, y) - p'(x, y)|,\ U(1) = |p(x, y) - p'(x+1, y)|$

The pairwise function depends on the states of two nearby pixels (in the image I1) and is given by:

$P(0, 0) = 0,\ P(0, 1) = 10,\ P(1, 0) = 10,\ P(1, 1) = 0$

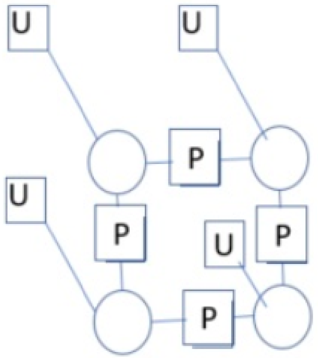
Use Belief propagation to solve the matching problem. Please show the messages in each iteration till the algorithm terminates.

50	0
50	0

I1

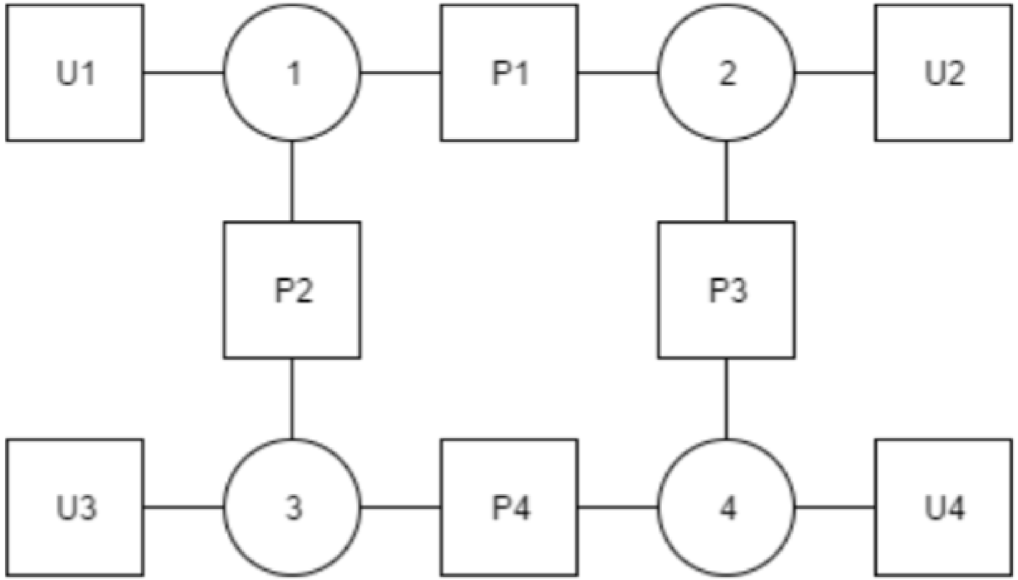
50	0	0
0	50	0

I2



Factor graph

Solution



u1	0	p1	0	10
	50		10	0
u2	0	p2	0	10
	0		10	0
u3	50	p3	0	10
	0		10	0
u4	50	p4	0	10
	0		10	0

Solution: Kyle Pierson

Solution

First Iteration

$m_{1\to u1}$	0 0	$m_{1\to p1}$	0 0	$m_{1\to p2}$	0 0
$m_{2\to u2}$	0 0	$m_{2\to p1}$	0 0	$m_{2\to p3}$	0 0
$m_{3\to u3}$	0 0	$m_{3\to p2}$	0 0	$m_{3\to p4}$	0 0
$m_{4\to u4}$	0 0	$m_{4\to p3}$	0 0	$m_{4\to p4}$	0 0

$m_{u1\to 1}$	0 50	$m_{p1\to 1}$	0 0	$m_{p2\to 1}$	0 0
$m_{u2\to 2}$	0 0	$m_{p1\to 2}$	0 0	$m_{p3\to 2}$	0 0
$m_{u3\to 3}$	50 0	$m_{p2\to 3}$	0 0	$m_{p4\to 3}$	0 0
$m_{u4\to 4}$	50 0	$m_{p3\to 4}$	0 0	$m_{p4\to 4}$	0 0

b_1	0 50	b_2	0 0	b_3	50 0	b_4	50 0
X1	0	X2	0	X3	1	X4	1

Second Iteration:

$m_{1\to u1}$	0 0	$m_{1\to p1}$	0 50	$m_{1\to p2}$	0 50
$m_{2\to u2}$	0 0	$m_{2\to p1}$	0 0	$m_{2\to p3}$	0 0
$m_{3\to u3}$	0 0	$m_{3\to p2}$	50 0	$m_{3\to p4}$	50 0
$m_{4\to u4}$	0 0	$m_{4\to p3}$	50 0	$m_{4\to p4}$	50 0

$m_{u1\to 1}$	0 50	$m_{p1\to 1}$	0 0	$m_{p2\to 1}$	10 0
$m_{u2\to 2}$	0 0	$m_{p1\to 2}$	0 10	$m_{p3\to 2}$	10 0
$m_{u3\to 3}$	50 0	$m_{p2\to 3}$	0 10	$m_{p4\to 3}$	10 0
$m_{u4\to 4}$	50 0	$m_{p3\to 4}$	0 0	$m_{p4\to 4}$	10 0

b_1	10 50	b_2	10 10	b_3	60 10	b_4	60 0
X1	0	X2	0	X3	1	X4	1