Lecture 18: Depth estimation

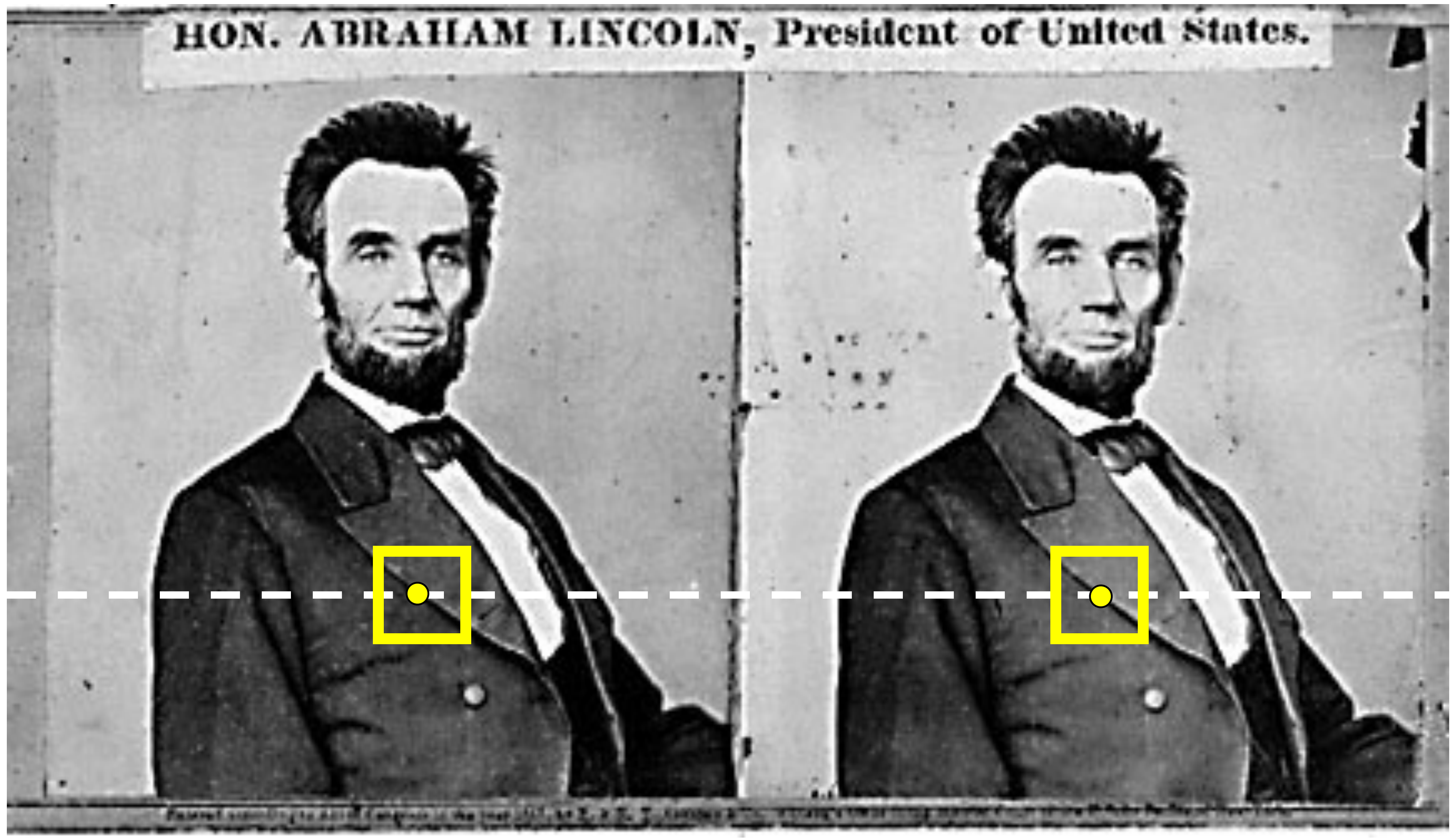
Announcements

- PS9 out tonight: panorama stitching
- New grading policies from UMich (details TBA)
- Final presentation will take place over video chat.
  - We'll send a sign-up sheet next week

Today

- Stereo matching
- Probabilistic graphical models
- Belief propagation
- Learning-based depth estimation

Basic stereo algorithm



For each epipolar line

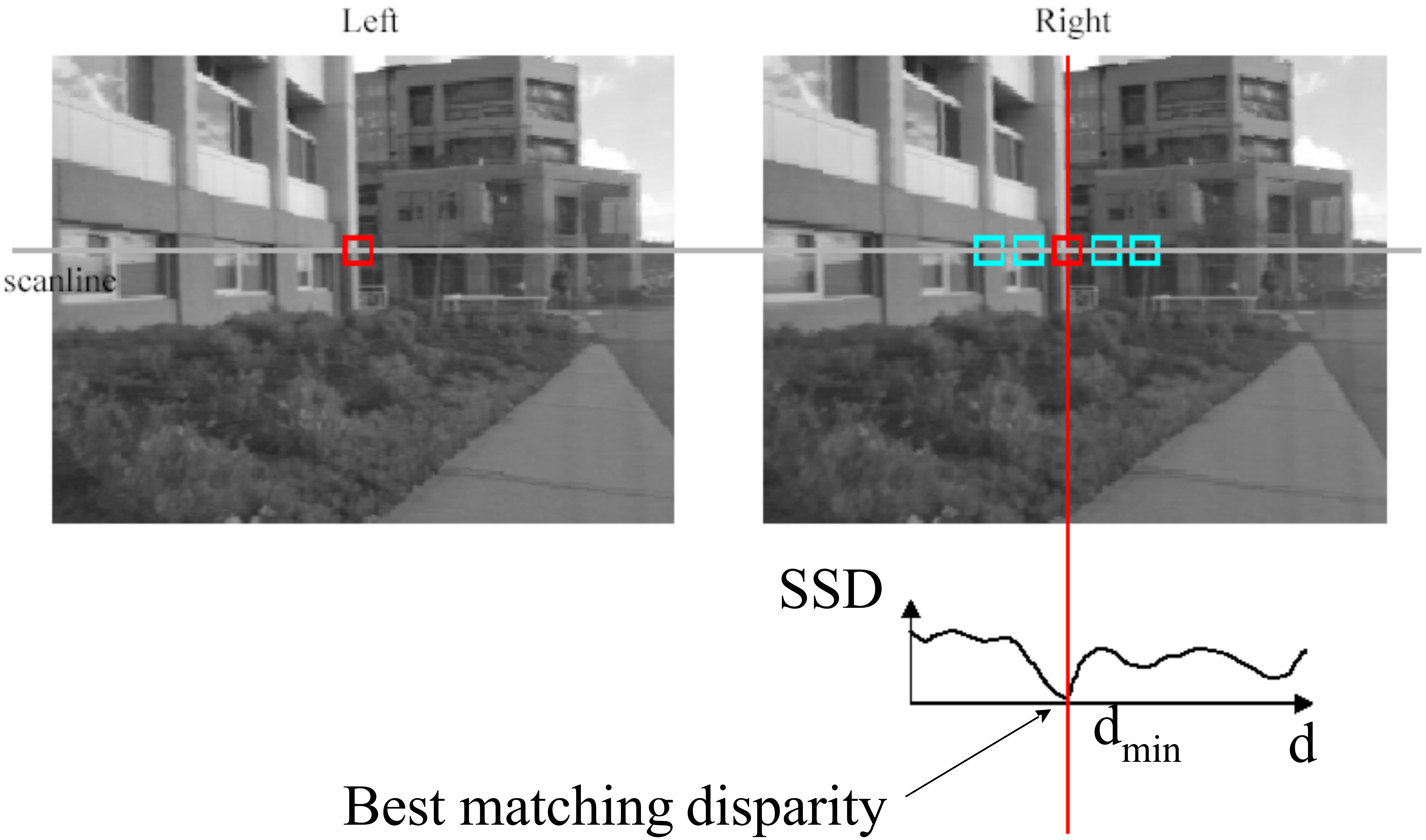
For each pixel in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Improvement: match ***windows***

Source: N. Snavely

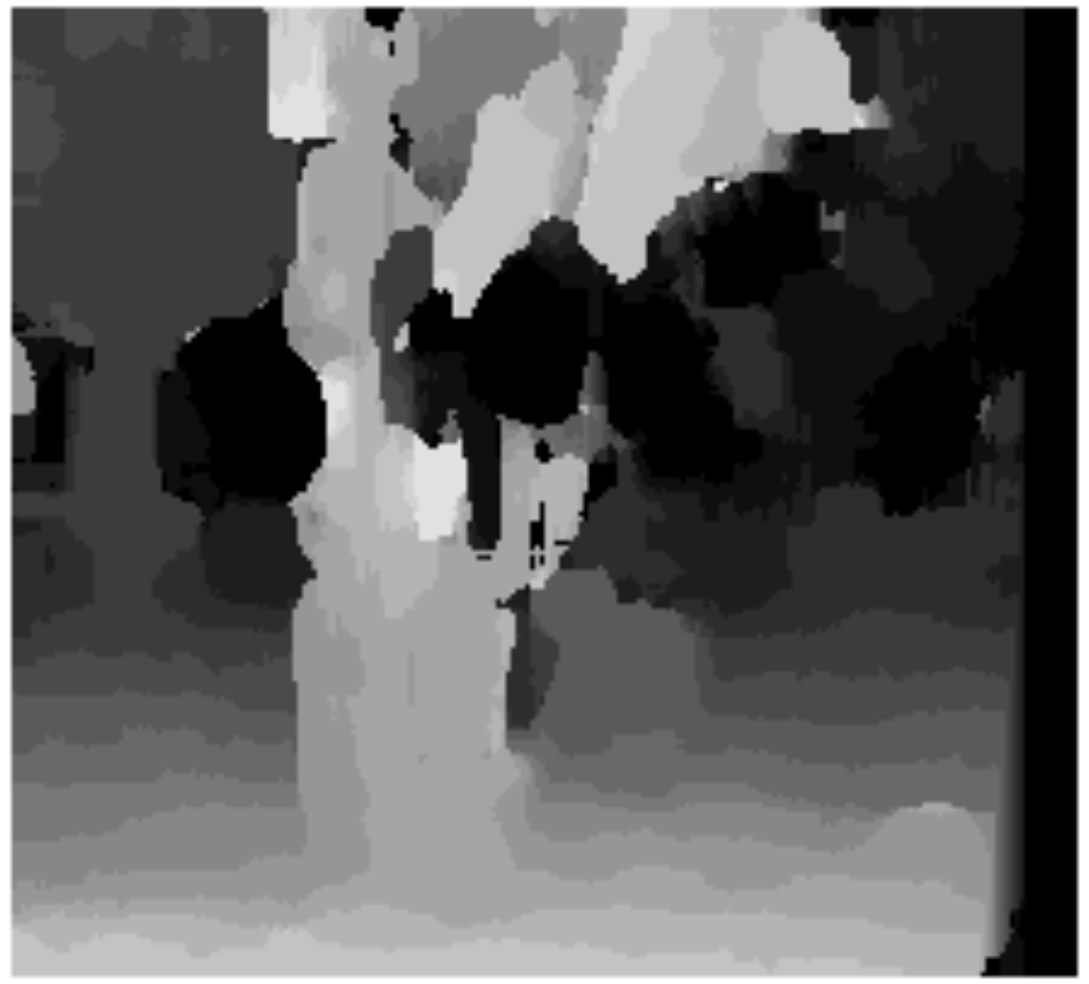
Stereo matching based on SSD



Source: N. Snavely

Window size

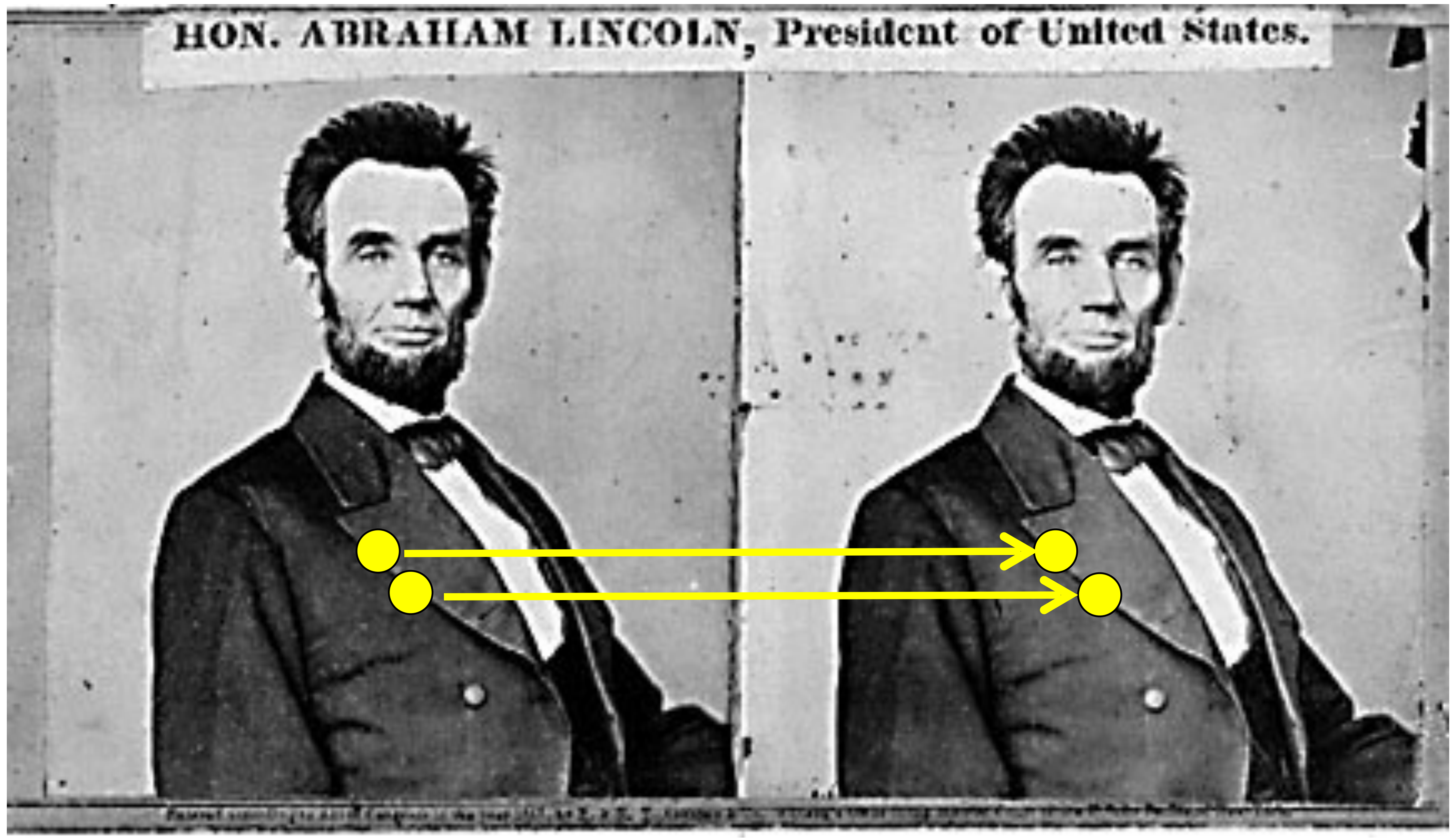
W = 3



W = 20

Source: N. Snavely

Stereo as energy minimization



- What defines a good stereo correspondence?
  1. Match quality
     - Want each pixel to find a good match in the other image
  2. Smoothness
     - If two pixels are adjacent, they should (usually) move about the same amount

Source: N. Snavely

Stereo as energy minimization

- Find disparity map *d* that minimizes an energy function $E(d)$

- Simple pixel / window matching

$$E(d) = \sum_{(x,y) \in I} C(x, y, d(x, y))$$

$C(x, y, d(x, y))$ = Squared distance between windows *I*(*x*, *y*) and *J*(*x* + *d*(*x*,*y*), *y*)

Source: N. Snavely

Stereo as energy minimization

$C(x, y, d)$; the *disparity space image* (DSI)

Source: N. Snavely

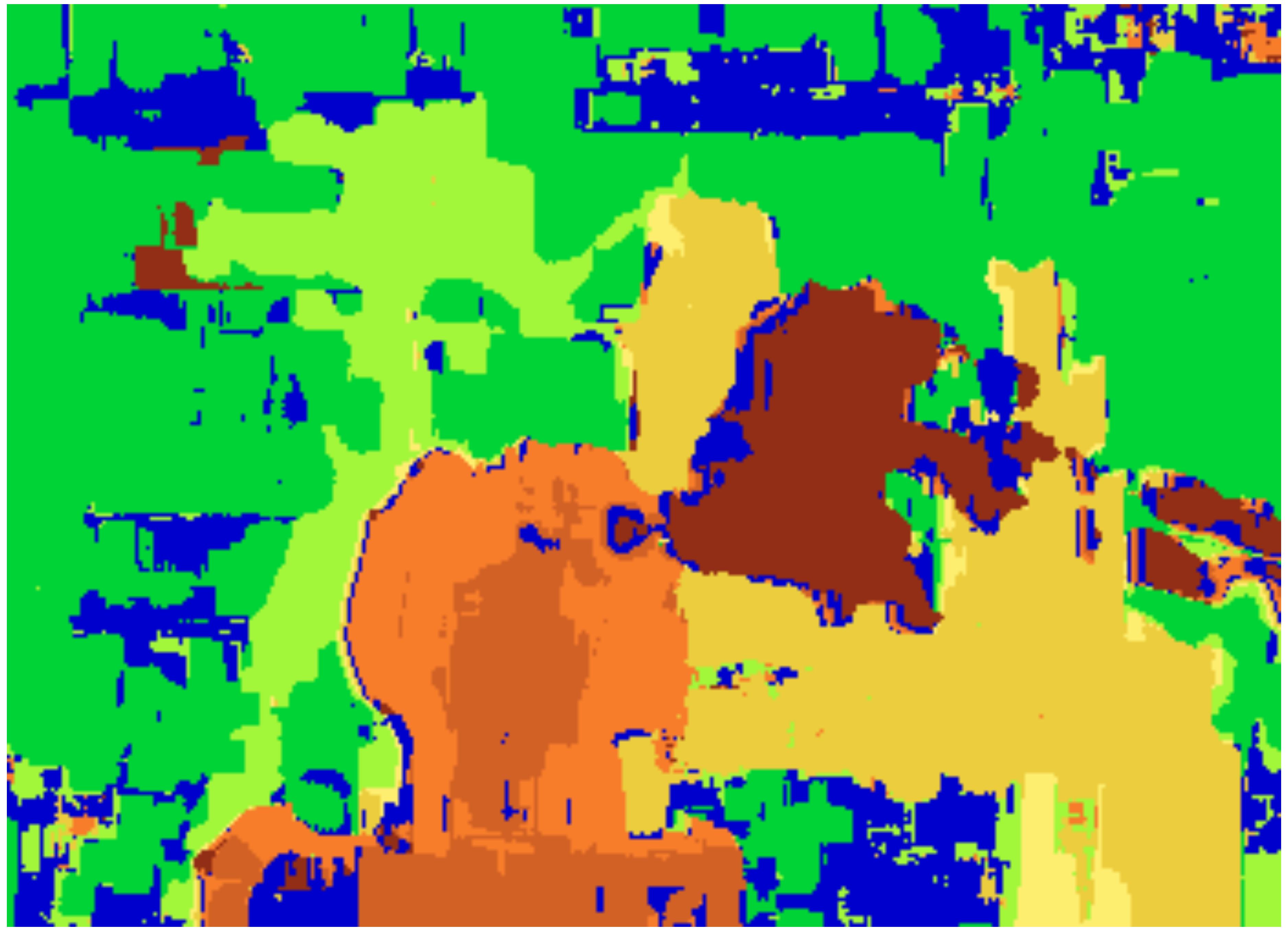
Stereo as energy minimization

Simple pixel / window matching: choose the minimum of each column in the DSI independently:

$$d(x, y) = \arg\min_{d'} \; C(x, y, d')$$

Source: N. Snavely

Greedy selection of best match



Source: N. Snavely

Stereo as energy minimization

- Better objective function

$$E(d) = \underbrace{E_d(d)}_{\text{match cost}} + \lambda \underbrace{E_s(d)}_{\text{smoothness cost}}$$

match cost — Want each pixel to find a good match in the other image

smoothness cost — Adjacent pixels should (usually) move about the same amount

Source: N. Snavely

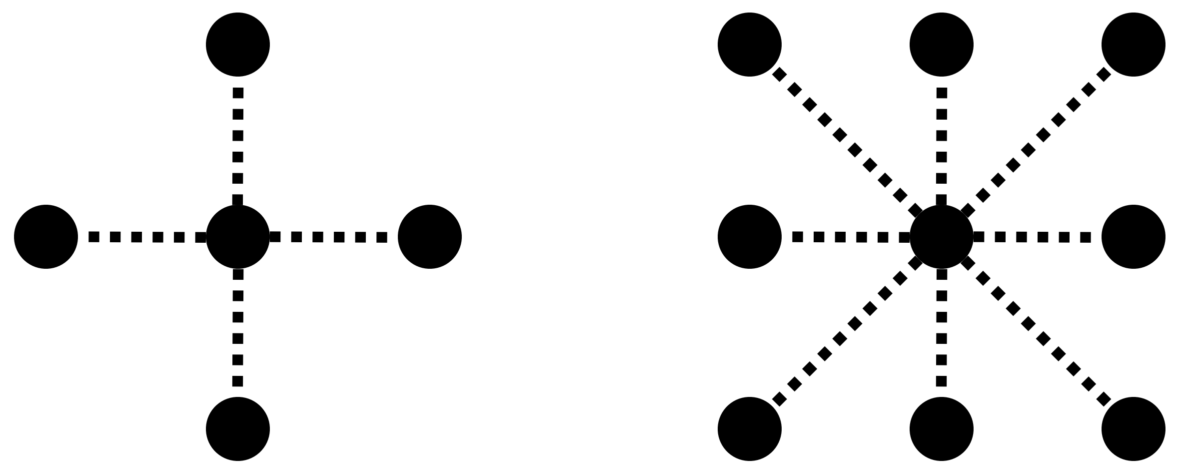
Stereo as energy minimization

$$E(d) = E_d(d) + \lambda E_s(d)$$

match cost: $$E_d(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

smoothness cost: $$E_s(d) = \sum_{(p,q)\in \mathcal{E}} V(d_p, d_q)$$

$\mathcal{E}$: set of neighboring pixels



Source: N. Snavely

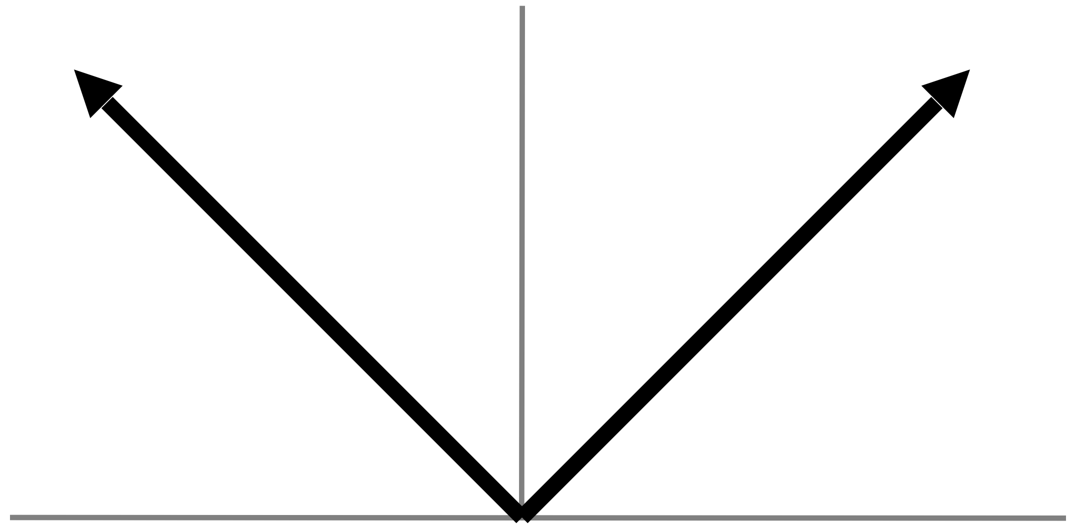
Smoothness cost

$$E_s(d) = \sum_{(p,q)\in\mathcal{E}} V(d_p, d_q)$$

How do we choose *V*?

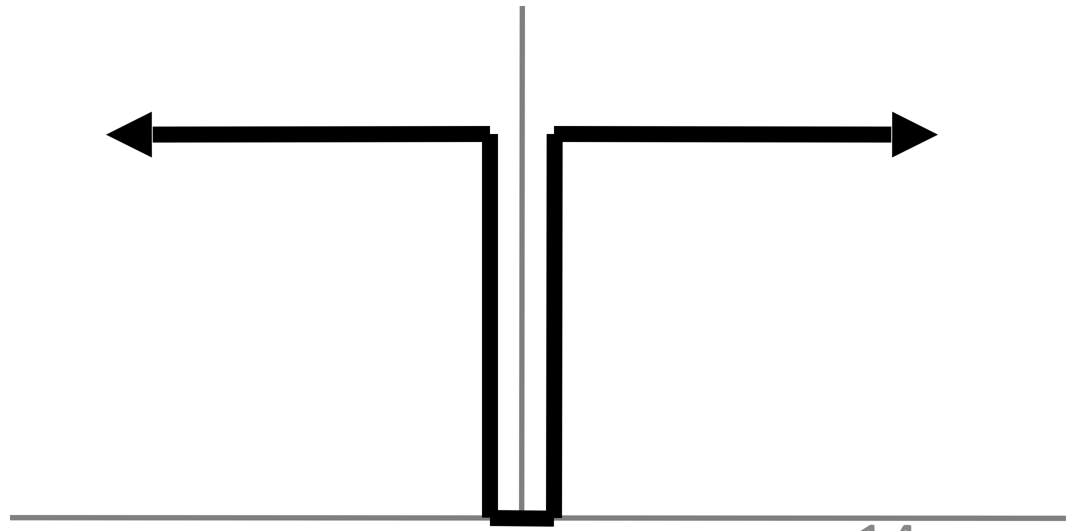
$$V(d_p, d_q) = |d_p - d_q|$$

L_1 distance



$$V(d_p, d_q) = \begin{cases} 0 & \text{if } d_p = d_q \\ 1 & \text{if } d_p \neq d_q \end{cases}$$

"Potts model"



Source: N. Snavely

Probabilistic interpretation

$$E(d) = E_d(d) + \lambda E_s(d)$$

Exponentiate:

$$\exp(E(d)) = \exp(E_d(d) + \lambda E_s(d))$$

Normalize:
(make it sum to 1)

$$\frac{\exp(E(d))}{Z} = \frac{1}{Z} \exp(E_d(d) + \lambda E_s(d))$$

where $Z = \sum_{d'} \exp E(d')$

Rewrite:

$$P(d \mid I) = k \prod_{i} \phi_i(d_i) \prod_{(i,j)} \psi_{ij}(d_i, d_j)$$

Example adapted from Freeman, Torralba, Isola

Probabilistic interpretation

"Local evidence"

How good are the matches?

"Pairwise compatibility"

Is the depth smooth?

$$P(d \mid I) = k \prod_i \phi_i(d_i) \prod_{(i,j)} \psi_{ij}(d_i, d_j)$$

Example adapted from Freeman, Torralba, Isola

Probabilistic interpretation

Local evidence:

$$\phi_i(d_i) = \exp \frac{(I_i - I_{i+d_i})^2}{2\sigma^2}$$

Pairwise compatibility:

$$\psi_{ij}(d_i, d_j) = \begin{cases} \alpha, & \text{if } d_i = d_j \\ \beta, & \text{otherwise} \end{cases}$$

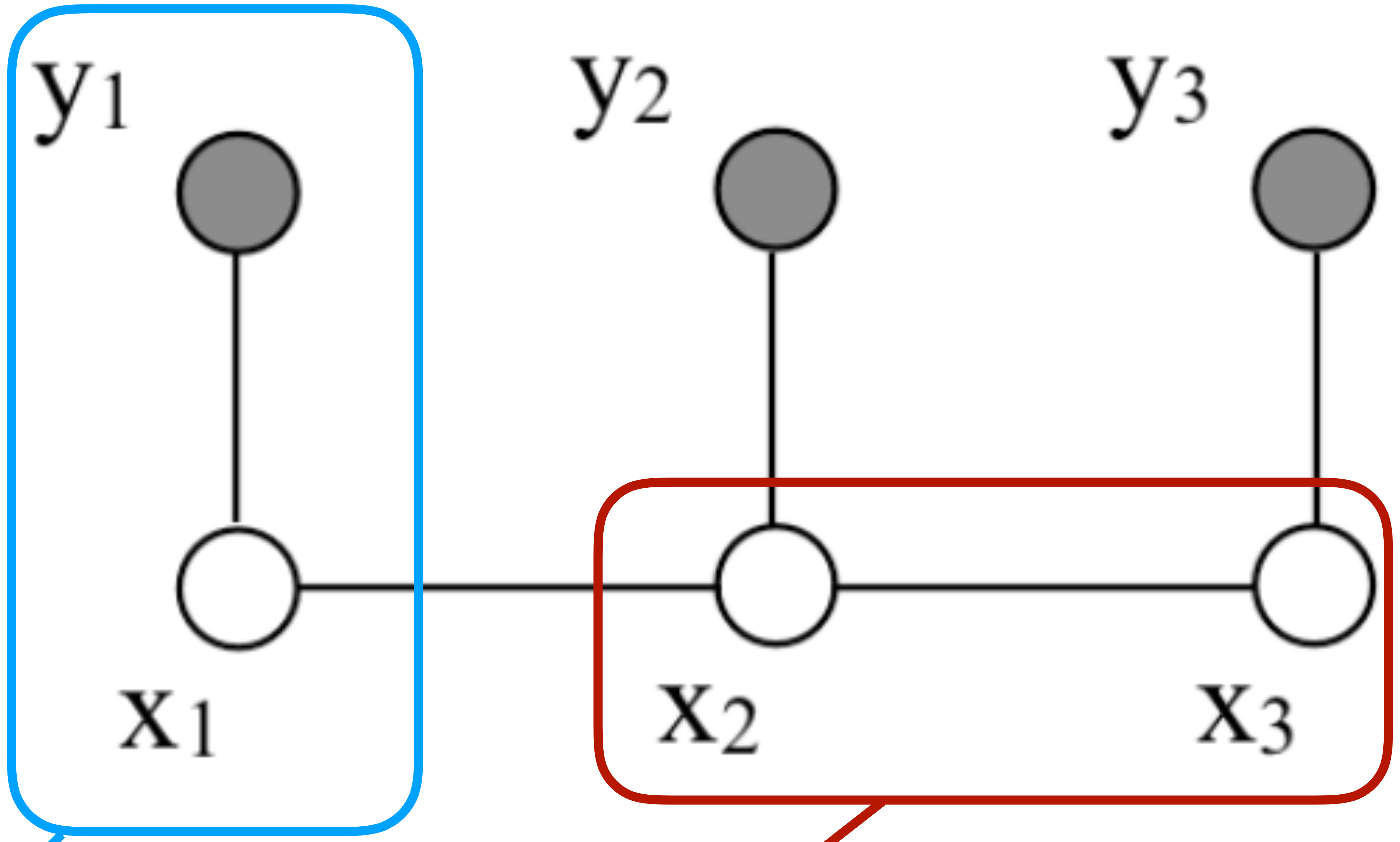
$$P(d \mid I) = k \prod_i \phi_i(d_i) \prod_{(i,j)} \psi_{ij}(d_i, d_j)$$

Example adapted from Freeman, Torralba, Isola

Probabilistic graphical models

Graph structure:

- Open circles for latent variables x_i
  - d_i in our problem
- Filled circle for observations y_i
  - Pixels in our problem
- Edges between interacting variables
  - In general, graph cliques for 3+ variable interactions



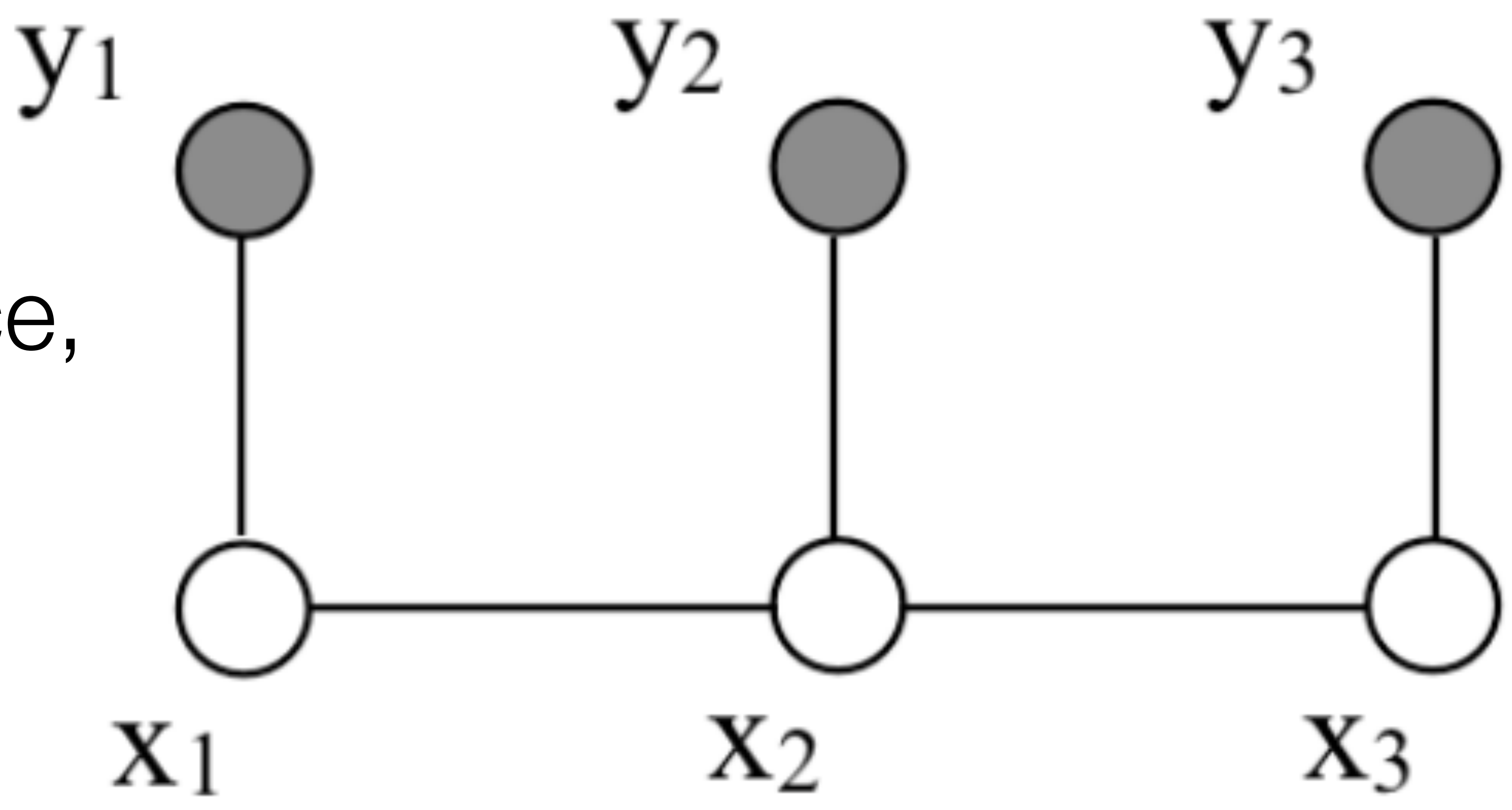
$$P(d \mid I) = k \prod_{i} \phi_i(d_i) \prod_{(i,j)} \psi_{ij}(d_i, d_j)$$

Example adapted from Freeman, Torralba, Isola

Probabilistic graphical models

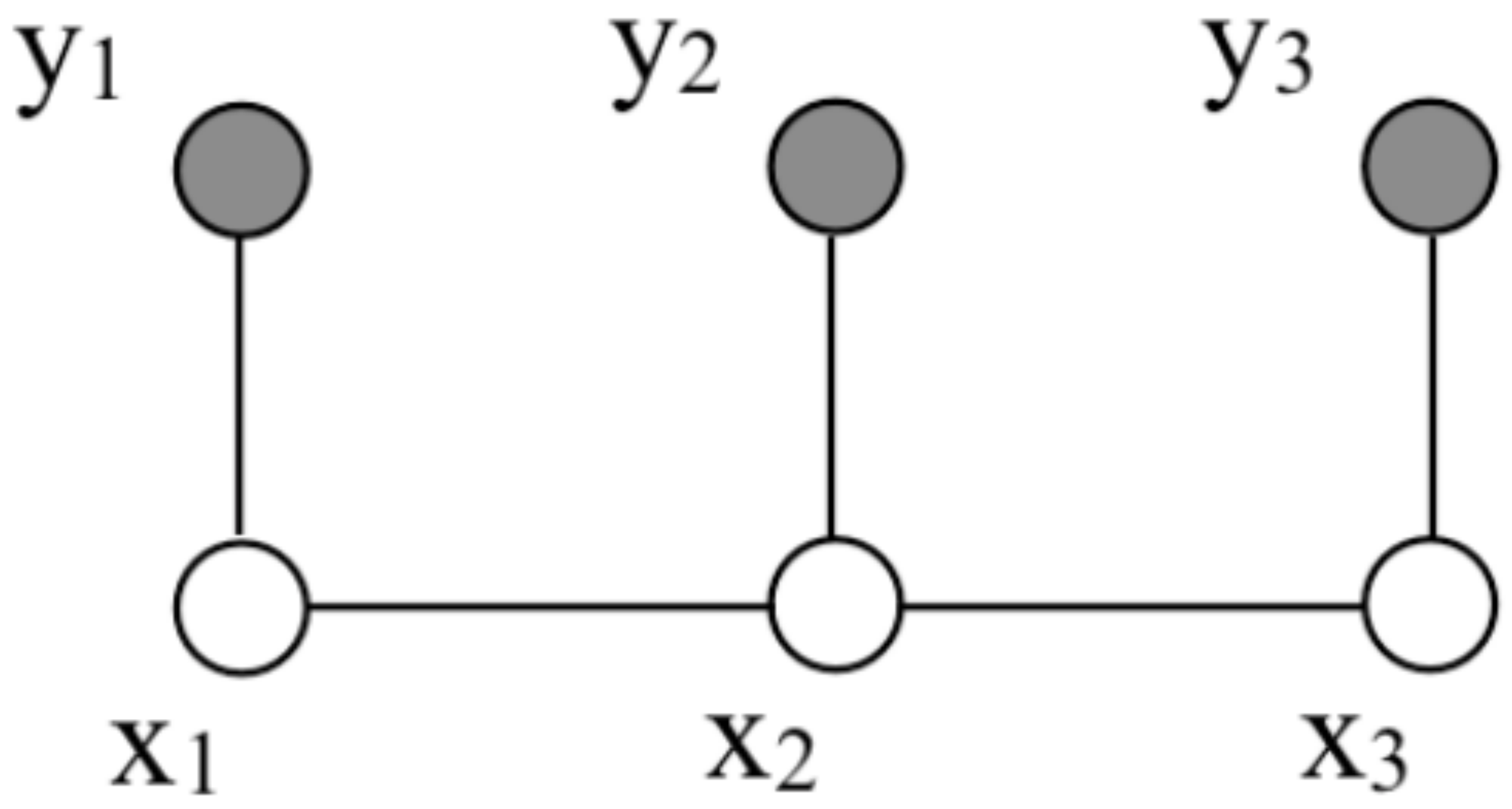
Why formulate it this way?

- Exploit sparse graph structure for fast inference, usually using dynamic programming
- Can use probabilistic inference methods
- Provides framework for learning parameters

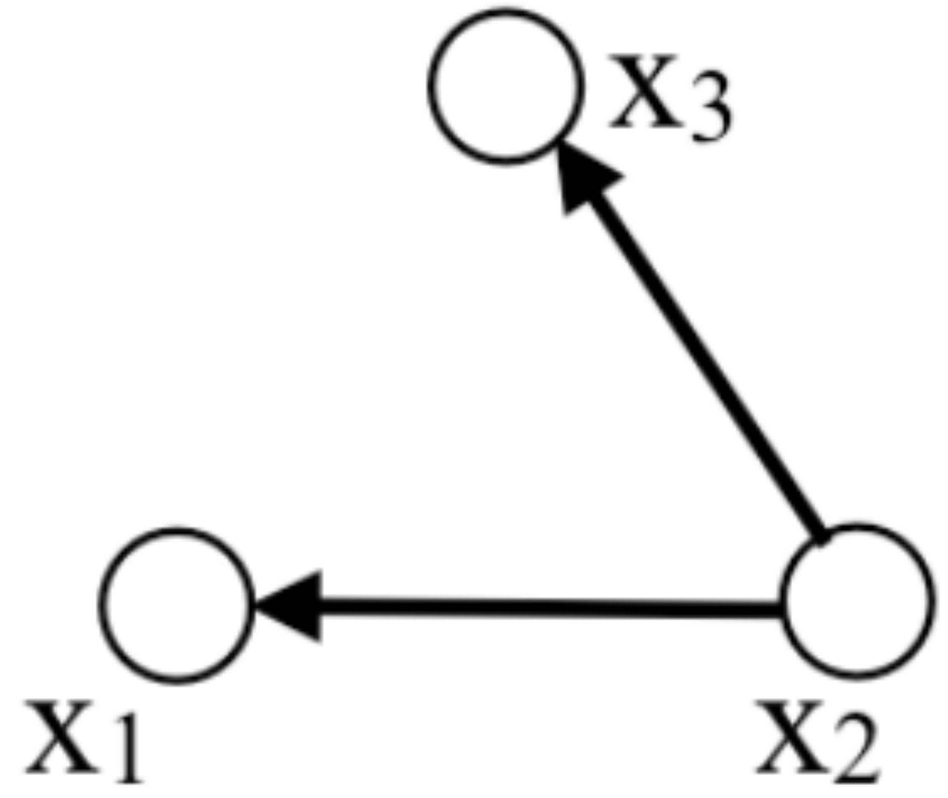


$$P(d \mid I) = k \prod_{i} \phi_i(d_i) \prod_{(i,j)} \psi_{ij}(d_i, d_j)$$

Probabilistic graphical models



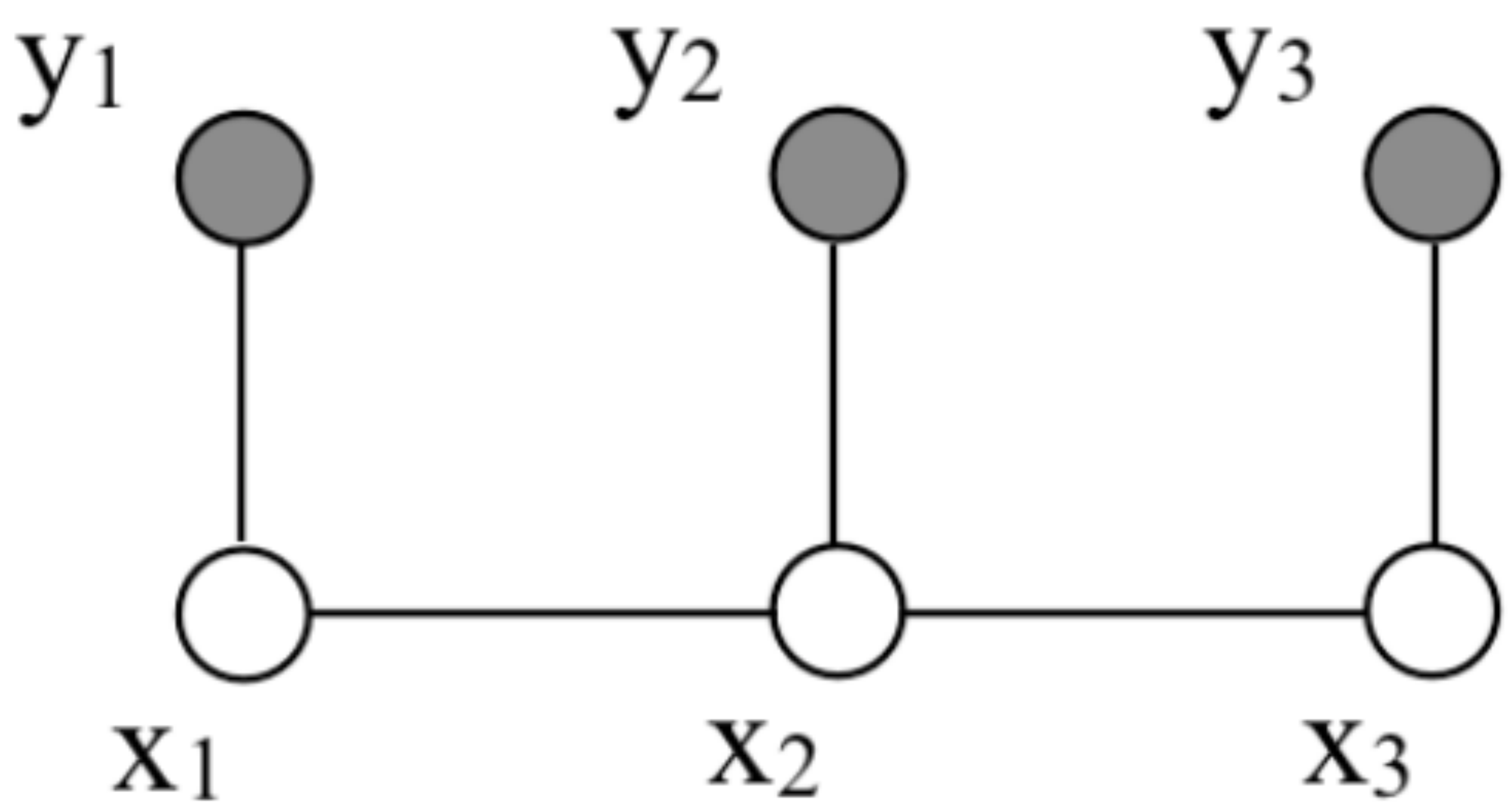
Undirected graphical model.
Also known as Markov Random Field (MRF).



$$P(x_1, x_2, x_3) = P(x_2)P(x_1|x_2)P(x_3|x_2)$$

Directed graphical model
Also know as Bayesian network
(Not covered in this course)

Marginalization



What's the marginal distribution for x_1?
i.e. what's the probability of x_1 being in a particular state?

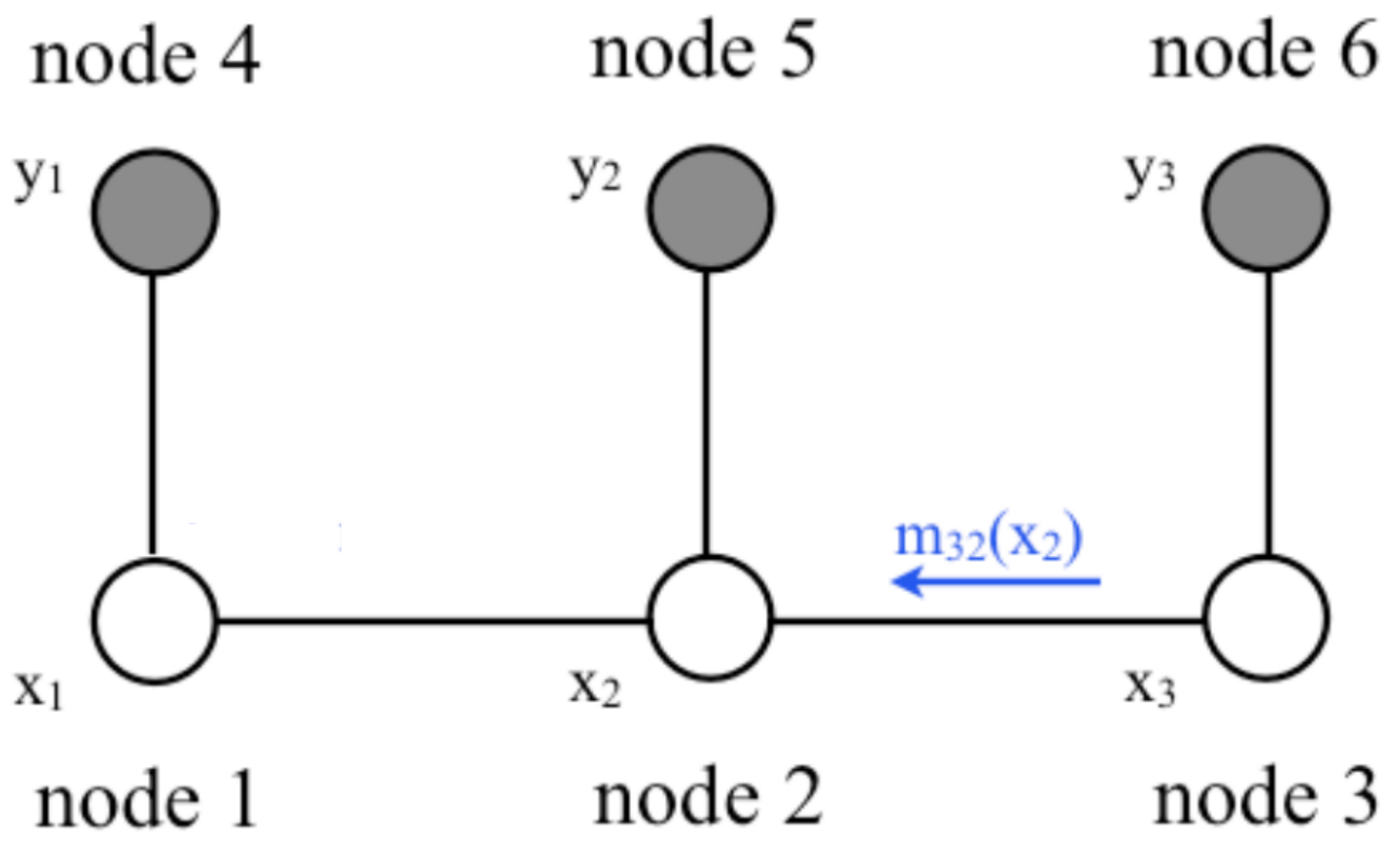
$$P(x_1|\vec{y}) = \sum_{x_2} \sum_{x_3} P(x_1, x_2, x_3|\vec{y})$$

- But this is expensive: $O(|L|\text{^}N)$
- Exploit graph structure!

Marginalization

$$
\begin{aligned}
P(x_1|\vec{y}) &= \frac{1}{P(\vec{y})} \sum_{x_2} \sum_{x_3} \phi_{12}(x_1, x_2)\phi_{23}(x_2, x_3)\psi_1(y_1, x_1)\psi_2(y_2, x_2)\psi_3(y_3, x_3) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1, x_1) \sum_{x_2} \phi_{12}(x_1, x_2)\psi_2(y_2, x_2) \sum_{x_3} \phi_{23}(x_2, x_3)\psi_3(y_3, x_3) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1, x_1) \sum_{x_2} \phi_{12}(x_1, x_2)\psi_2(y_2, x_2)m_{32}(x_2) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1, x_1)m_{21}(x_1)
\end{aligned}
$$

Message passing



Can think of "local evidence" message passing

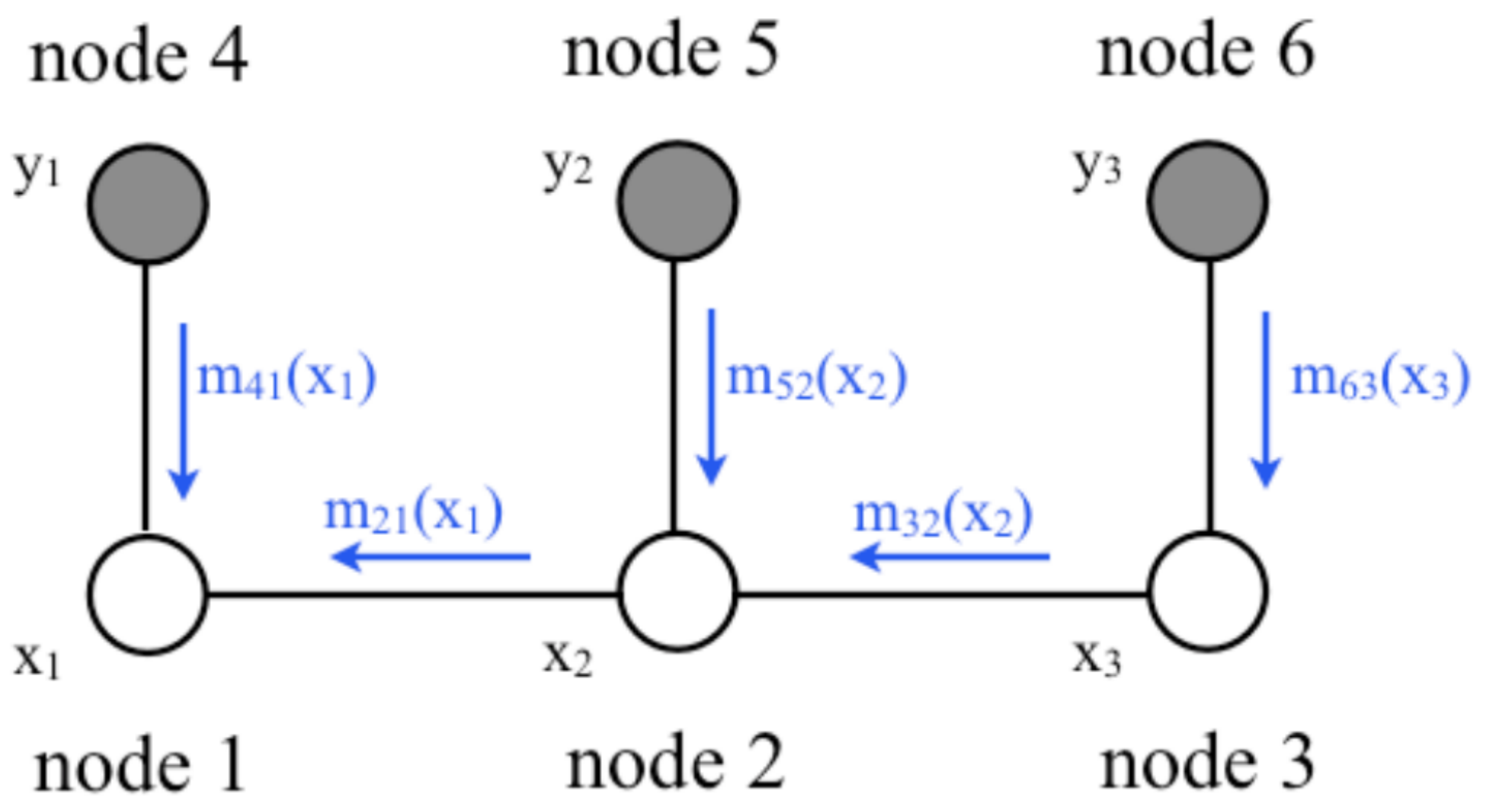
Message that node x_3 sends to node x_2

$$= \frac{1}{P(\vec{y})} \psi_1(y_1, x_1) \sum_{x_2} \phi_{12}(x_1, x_2) \psi_2(y_2, x_2) m_{32}(x_2)$$

Message that x_2 sends to x_1

$$= \frac{1}{P(\vec{y})} \psi_1(y_1, x_1) m_{21}(x_1)$$

Message passing



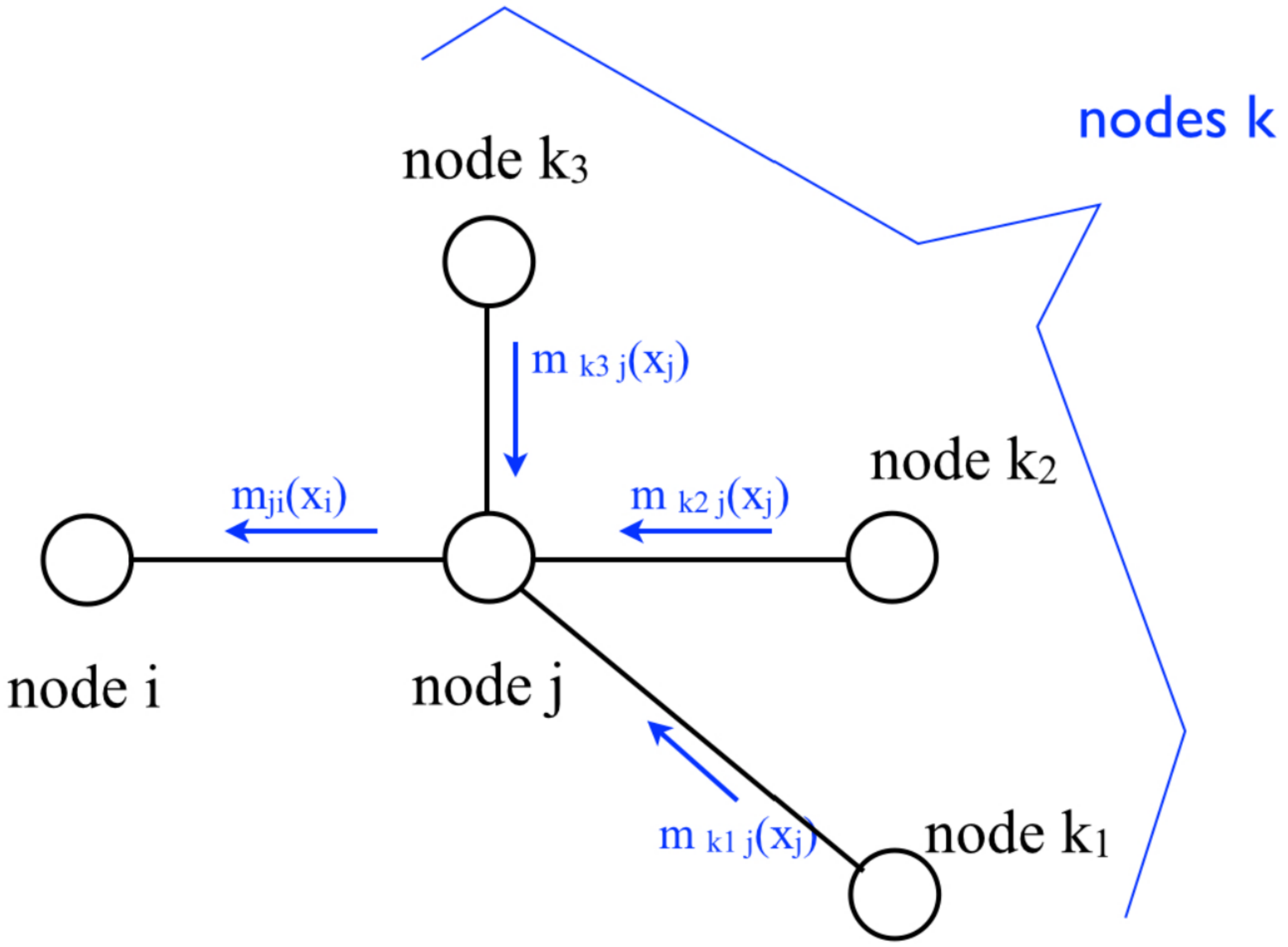
- Message $\mathbf{m_{ij}}$ is the sum over *all states of all nodes* in the subtree leaving node i at node j
- It summarizes what this node “believes”.
  - E.g. if you have label x_2, what’s the probability of my subgraph?
- Shared computation! E.g. could reuse m_{32} to help estimate $p(x_2 \mid y)$.

Belief propagation

- Estimate all marginals $p(x_i \mid y)$ at once! [Pearl 1982]
- Given a tree-structured graph, send messages in topological order

Sending message from j to i:

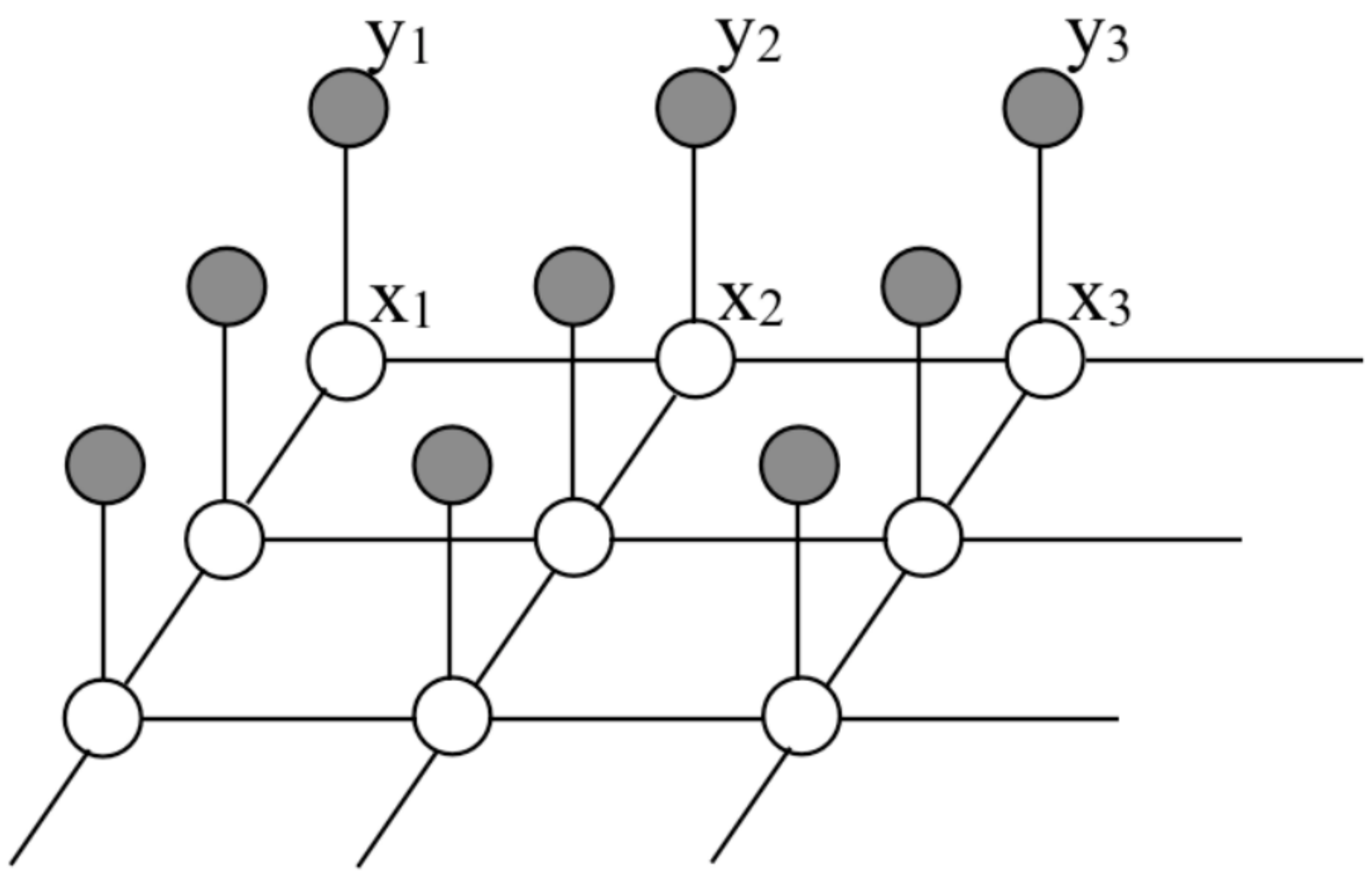
1. Multiply all incoming messages (except for the one from i)
2. Multiply the pairwise compatibility
3. Marginalize over x_j



$$m_{ji}(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in \eta(j) \setminus i} m_{kj}(x_j)$$

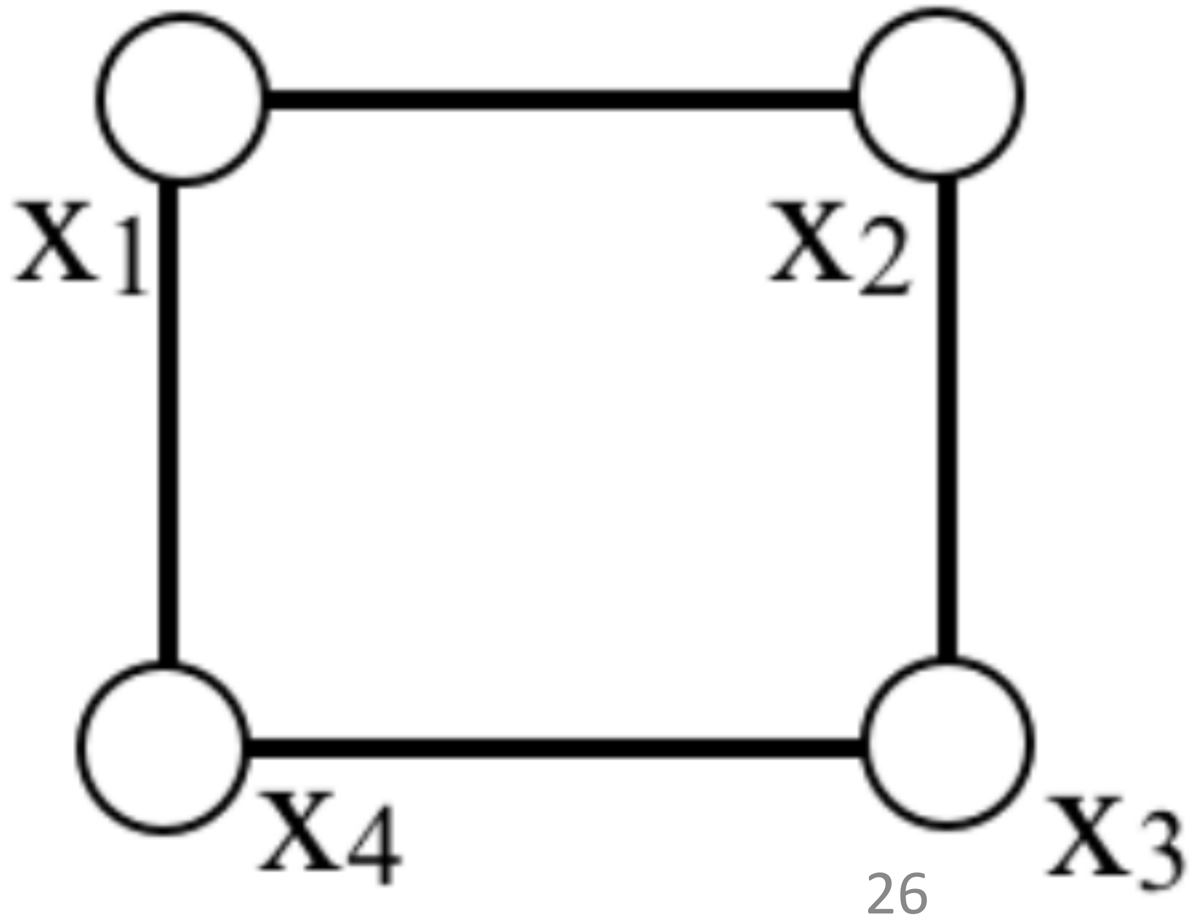
General graphs

- Vision problems often are often on grid graphs
- Pretend the graph is tree-structured and do belief propagation iteratively!
- Can also have consistency with N > 2 variables
  - But complexity is exponential in N!



Loopy belief propagation:

1. Initialize all messages to 1
2. Walk through the edges in an arbitrary order (e.g. random)
3. Apply the messages updates



Finding best labels

Often want to find the labels that jointly maximize probability:

$$\underset{x_1, x_2, x_3}{\text{argmax}} \; P(x_1, x_2, x_3 | \vec{y})$$

This is called *maximum a posteriori* estimation (MAP estimation).

Marginal:

$$P(x_1 | \vec{y}) = \sum_{x_2} \sum_{x_3} P(x_1, x_2, x_3 | \vec{y})$$

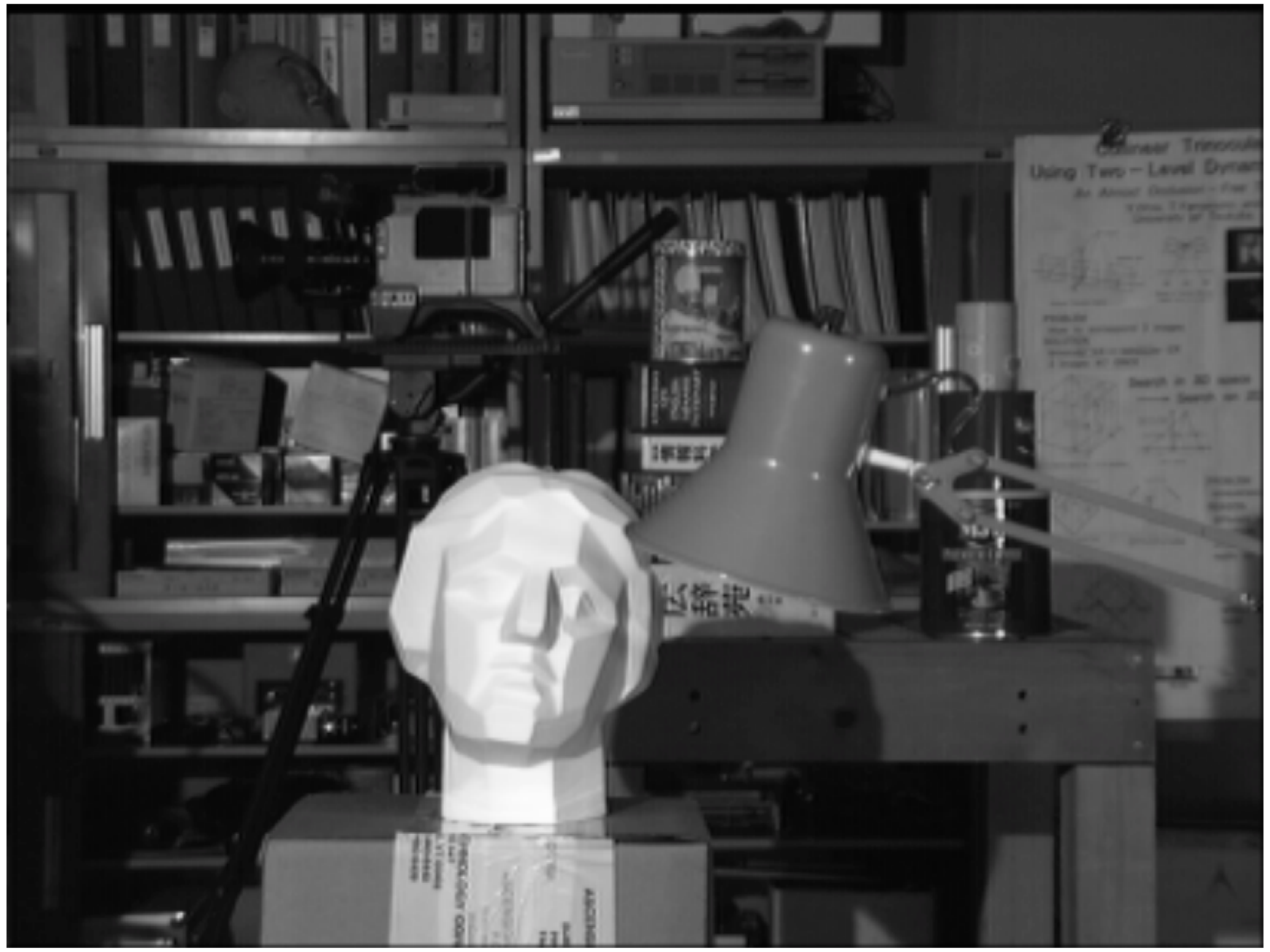
$$m_{ji}(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in \eta(j) \setminus i} m_{kj}(x_j)$$

"Max marginal" instead:

$$b(x_1 \mid \vec{y}) = \max_{x_2, x_3} P(x_1, x_2, x_3 | \vec{y})$$

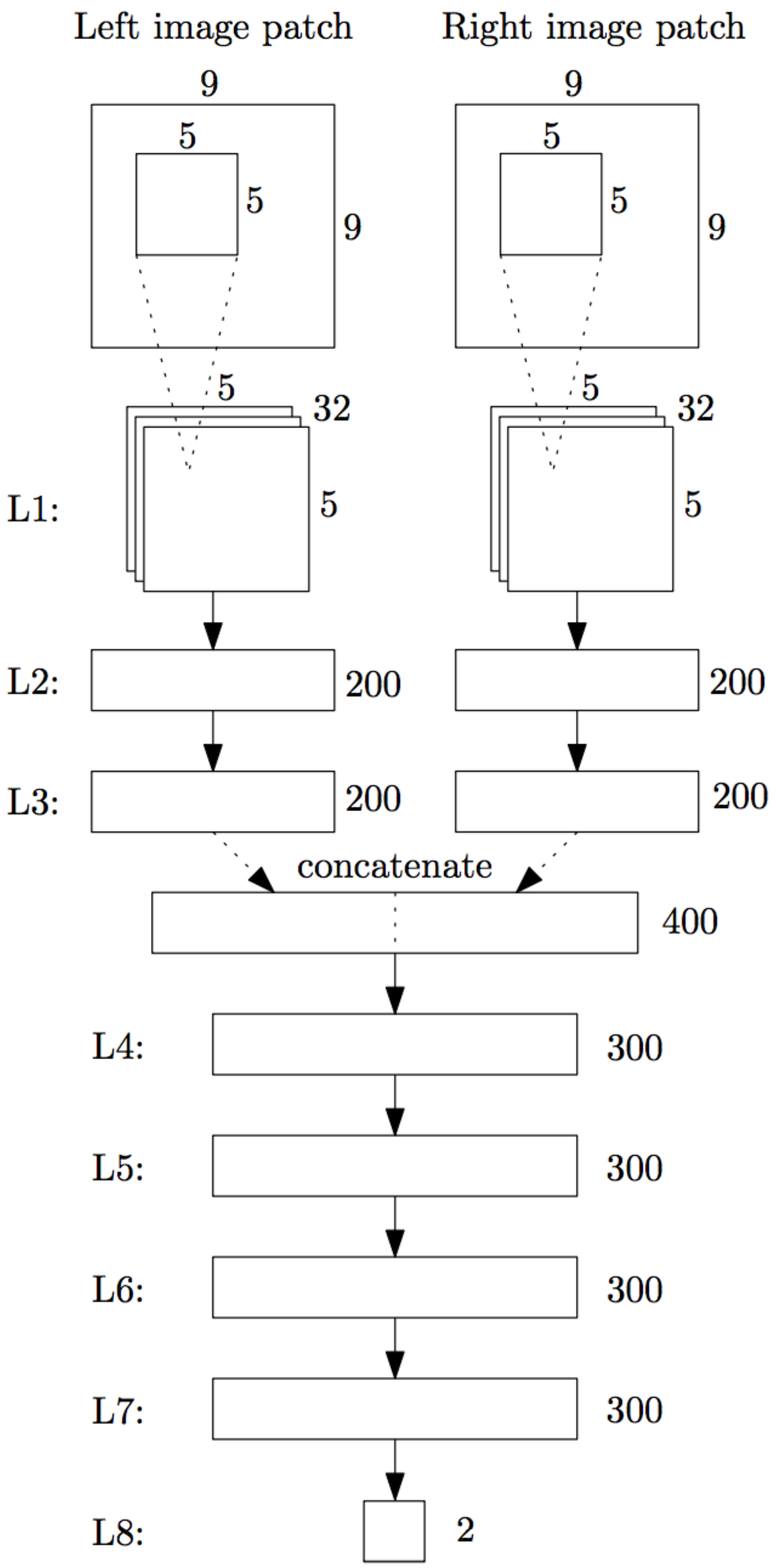
$$m_{ji}(x_i) = \max_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in \eta(j) \setminus i} m_{kj}(x_j)$$

Application to stereo

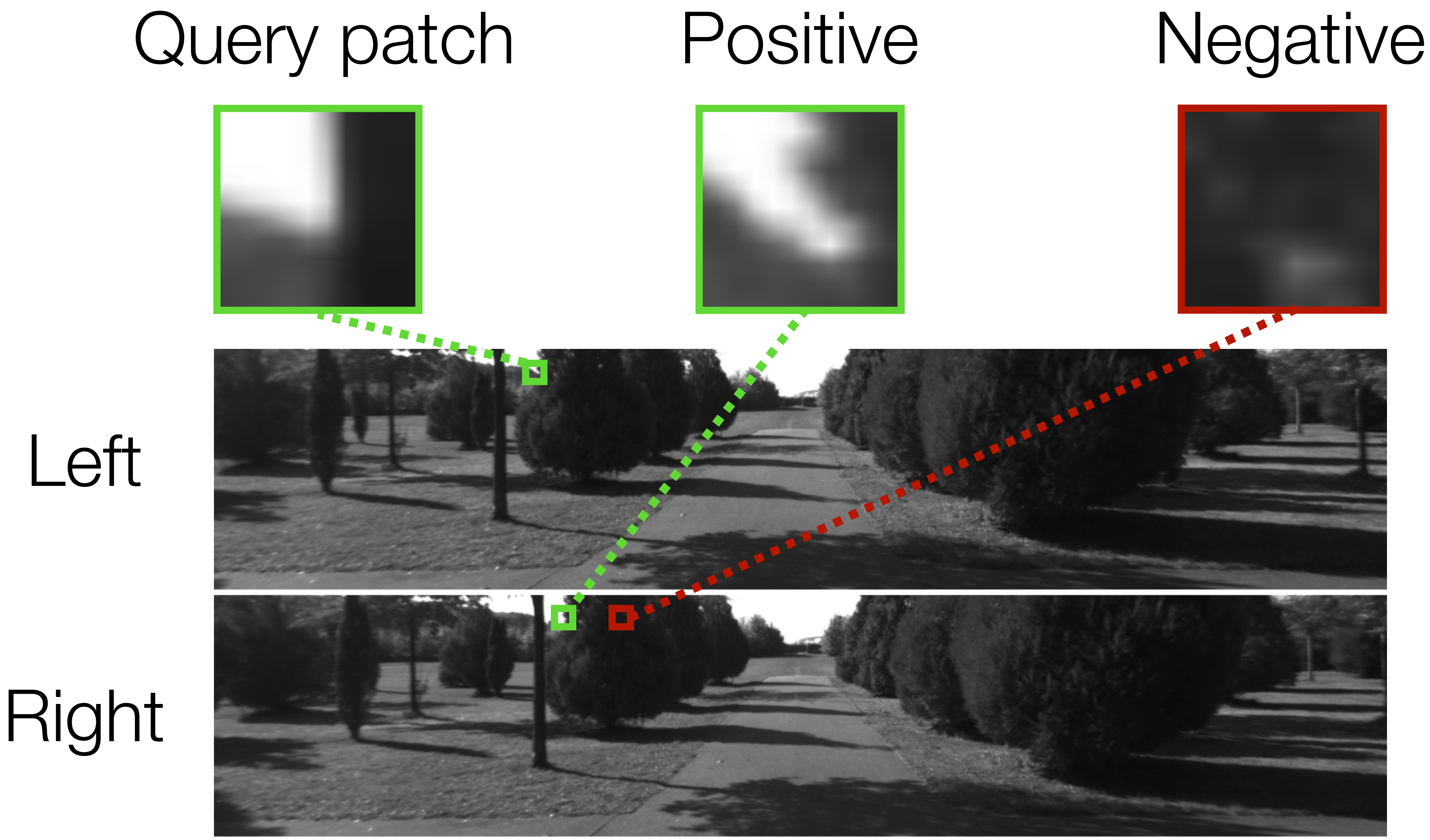


[Felzenzwalb & Huttenlocher, “Efficient Belief Propagation for Early Vision”, 2006]

Deep learning + MRF refinement



[Zbontar & LeCun, 2015]

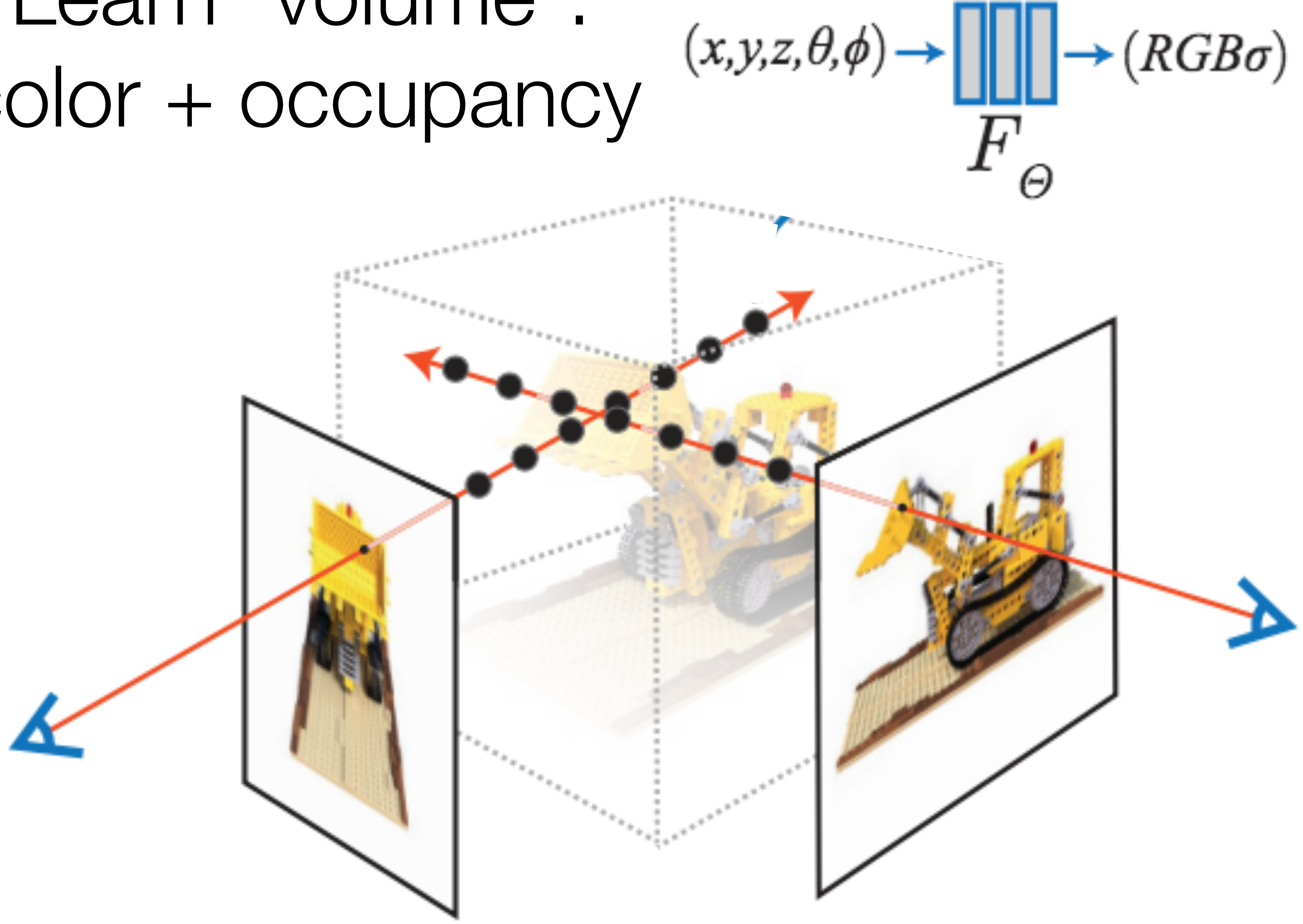


CNN-based matching + MRF refinement

Learning to estimate depth without ground truth

3D scene

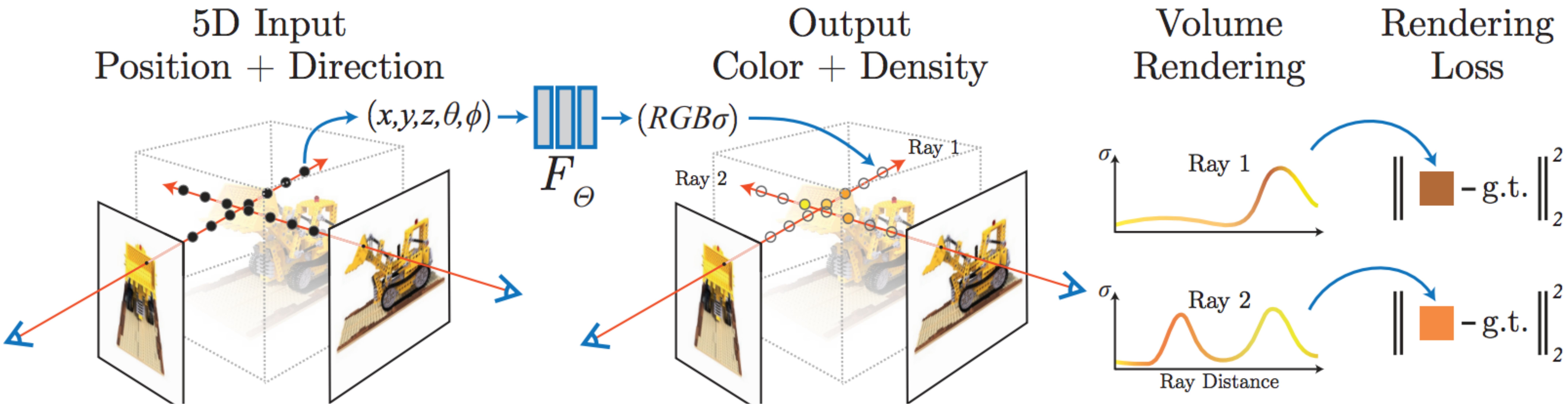
Learn “volume”:
color + occupancy



Viewpoints

[Mildenhall*, Srinivasan*, Tanick*, et al., Neural radiance fields, 2020]

Learning to estimate depth without ground truth



A good volume should reconstruct the input views

[Mildenhall*, Srinivasan*, Tanick*, et al. 2020]

Learning to estimate depth without ground truth

[Mildenhall*, Srinivasan*, Tanick*, et al. 2020]

Learning to estimate depth without ground truth

Inserting virtual objects

View synthesis

[Mildenhall*, Srinivasan*, Tanick*, et al. 2020]

Next class: motion