اصول پردازش تصویر

Principles of Image Processing

مصطفی کمالی تبریزی

۲۱ مهر ۱۳۹۹

جلسه هشتم

Template Matching

Template Matching

- Goal: find  in image

- Main challenge: What is a good similarity or distance measure between two patches?
  - Cross-Correlation
  - Zero-Mean Cross-Correlation
  - Sum of Squared Differences
  - Normalized Cross-Correlation

Slide Credit: Derek Hoiem

1. Cross-Correlation

- Goal: find  in image
- Method 1: Cross-Correlation

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

Input

Filtered Image (Scaled)

$$g.f = \|g\|\|f\|\cos(\theta) \sim \cos(\theta)$$

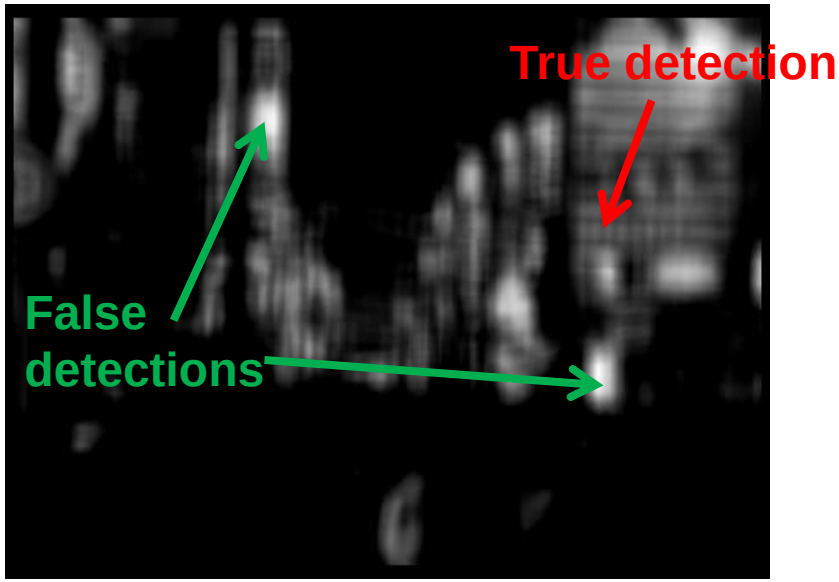
Slide Credit: Derek Hoiem

2. Zero-Mean Cross-Correlation

- Goal: find  in image
- Method 2: Zero-Mean Cross-Correlation

$$h[m,n] = \sum_{k,l} (g[k,l] - \bar{g}) f[m+k, n+l]$$

Input



Filtered Image (Scaled)

What went wrong?

Slide Credit: Derek Hoiem

3. Sum of Squared Differences (SSD)

- Goal: find  in image
- Method 3: SSD

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k, n+l])^2$$

Input

1 - SSD (Scaled)

Slide Credit: Derek Hoiem

3. Sum of Squared Differences (SSD)

Can SSD be implemented with linear filters?

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k, n+l])^2$$

$$= \sum_{k,l} g[k,l]^2 + \sum_{k,l} f[m+k, n+l]^2 - 2 \sum_{k,l} g[k,l] f[m+k, n+l]$$

Slide Credit: Derek Hoiem

3. Sum of Squared Differences (SSD)

Can SSD be implemented with linear filters?

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k, n+l])^2$$

$$= \sum_{k,l} g[k,l]^2 + \sum_{k,l} f[m+k, n+l]^2 - 2 \sum_{k,l} g[k,l] f[m+k, n+l]$$

$$= C + (eyes * f^2) \; - 2 \, (g * f)$$

Slide Credit: Derek Hoiem

4. Normalized Cross-Correlation

- Goal: find  in image
- Method 4: Normalized Cross-Correlation

mean template

mean image patch

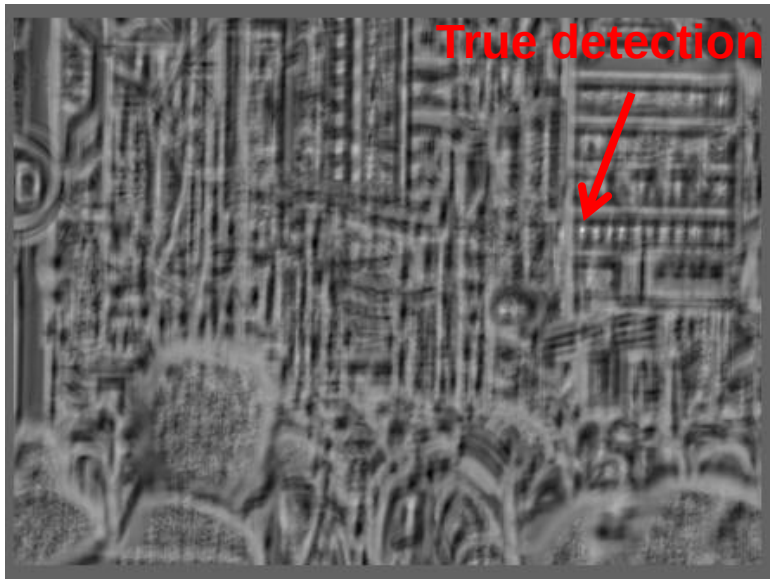
$$h[m,n] = \frac{\sum_{k,l}(g[k,l] - \bar{g})\left(f[m+k,n+l] - \overline{f_{m,n}}\right)}{\left(\sum_{k,l}(g[k,l] - \bar{g})^2 \sum_{k,l}\left(f[m+k,n+l] - \overline{f_{m,n}}\right)^2\right)^{0.5}}$$

Slide Credit: Derek Hoiem

4. Normalized Cross-Correlation

- Goal: find  in image
- Method 4: Normalized Cross-Correlation

Input



Filtered Image (Scaled)

Slide Credit: Derek Hoiem

- *Q: What is the best method to use?*
- *A: Depends!*

- Zero-mean filter: fastest but not a great matcher
- SSD: next fastest, sensitive to overall intensity
- Normalized cross-correlation: slowest, invariant to local average intensity and contrast

Slide Credit: Derek Hoiem

References

- Template Matching
  Gonzalez, section 13.3,

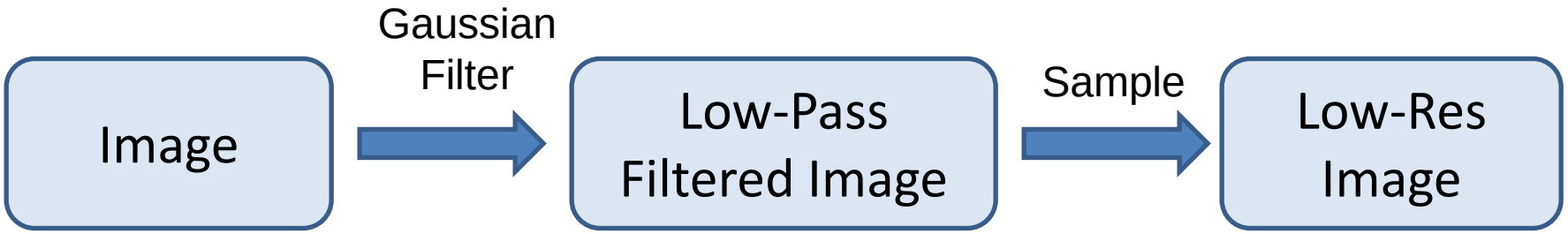
- Translational Alignment
  Szeliski, section 8.1

Image Pyramids

Several slides from Derek Hoiem

Q: What if we want to find larger or smaller templates?

Review of Sampling

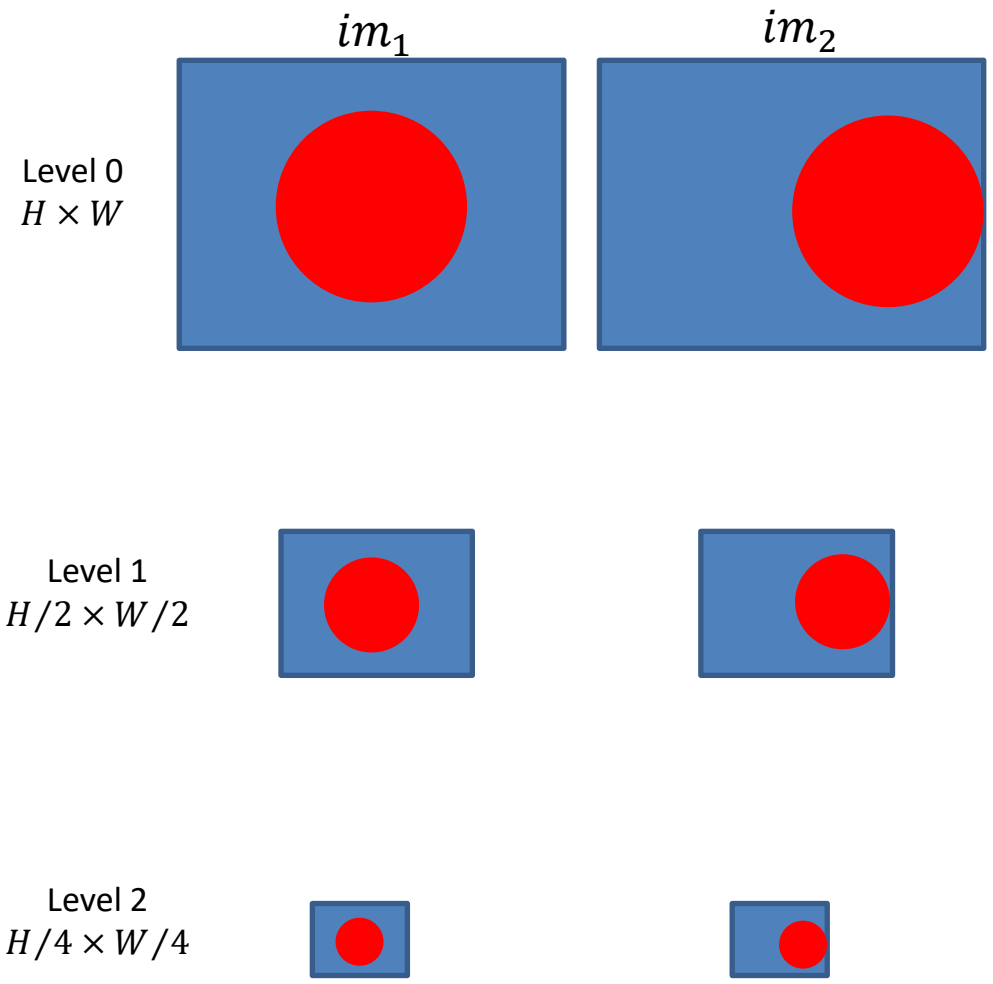


Gaussian Pyramid

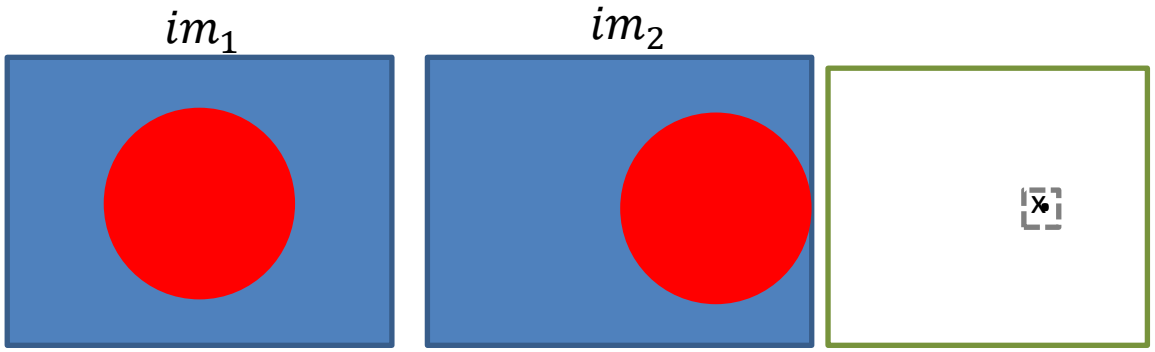
512 256 128 64 32 16 8

Source: Forsyth

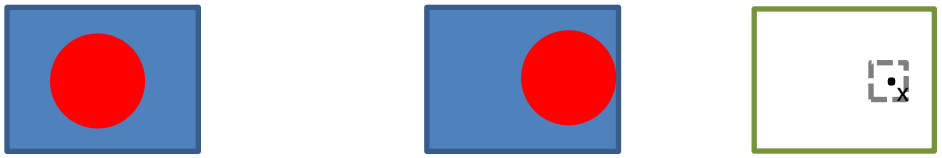
Coarse-to-fine Image Registration



Coarse-to-fine Image Registration



$$(t_{x_0}, t_{y_0}) = \underset{\substack{2t_{x_1}-s \le t_x \le 2t_{x_1}+s \\ 2t_{y_1}-s \le t_y \le 2t_{y_1}+s}}{\operatorname{argmin}} SSD\left(im_1, translate(im_2, t_x, t_y)\right)$$

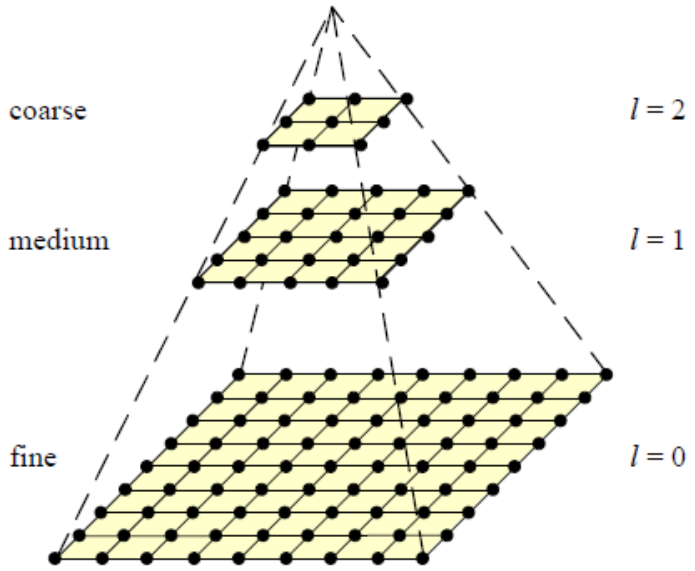


$$(t_{x_1}, t_{y_1}) = \underset{\substack{2t_{x_2}-s \le t_x \le 2t_{x_2}+s \\ 2t_{y_2}-s \le t_y \le 2t_{y_2}+s}}{\operatorname{argmin}} SSD\left(im_1, translate(im_2, t_x, t_y)\right)$$

$$(t_{x_2}, t_{y_2}) = \underset{\substack{-w/8 \le t_x \le w/8 \\ -H/8 \le t_y \le H/8}}{\operatorname{argmin}} SSD\left(im_1, translate(im_2, t_x, t_y)\right)$$

Coarse-to-Fine Image Registration

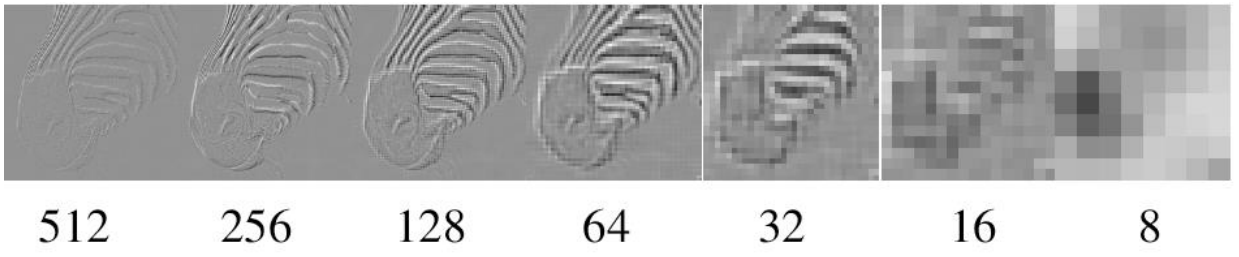
1. Compute Gaussian pyramid
2. Align with coarse pyramid
   - Find minimum SSD position
3. Successively align with finer pyramids
   - Search small range (e.g., 5x5) centered around position determined at coarser scale



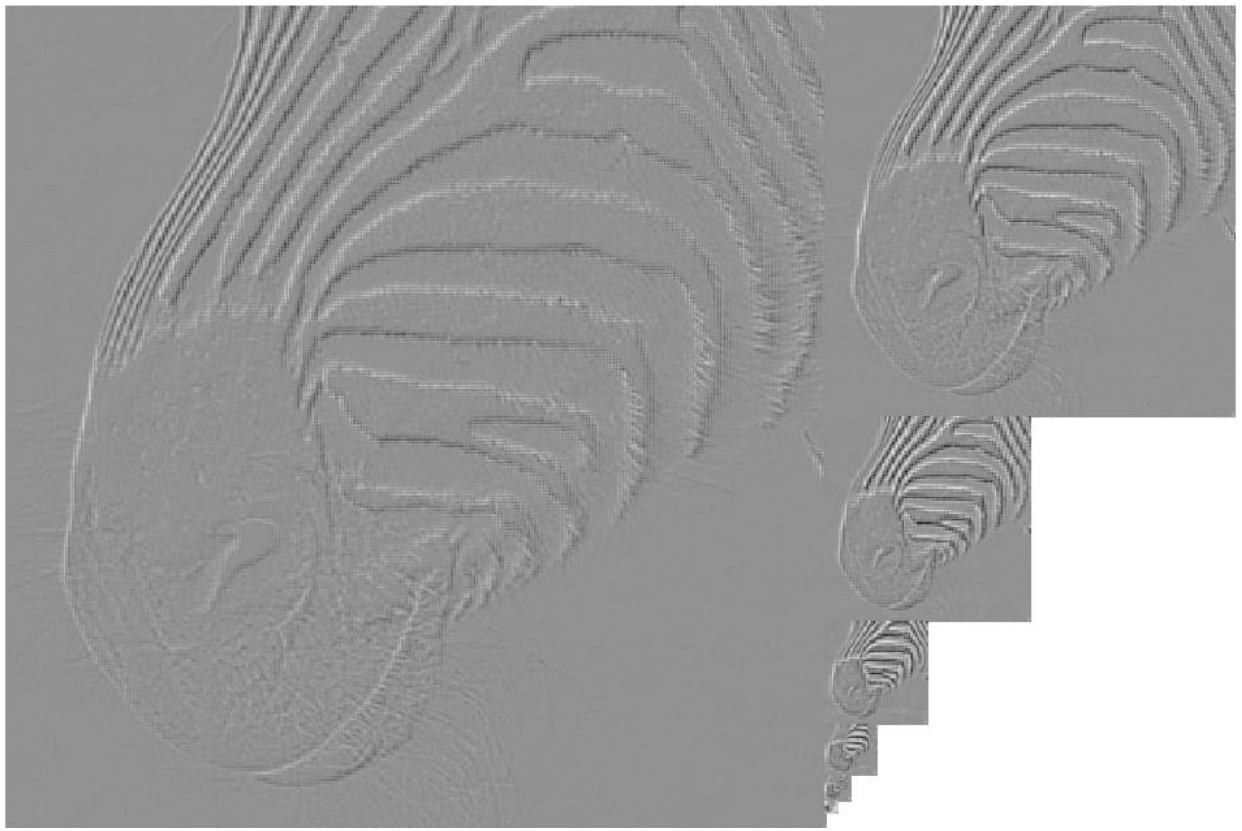
Why is this faster?

Are we guaranteed to get the same result?

Laplacian Pyramid

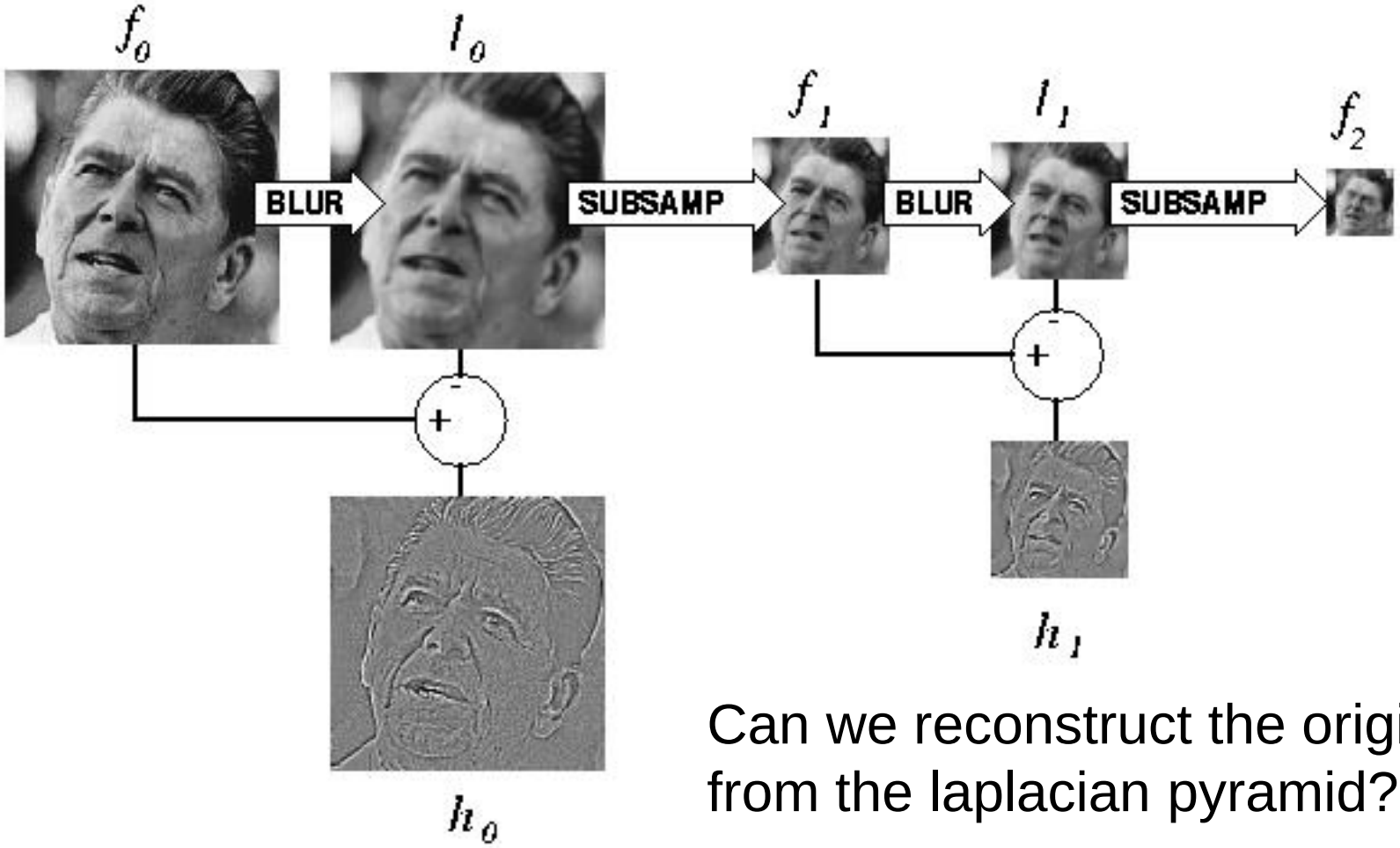


512 256 128 64 32 16 8



Source: Forsyth

Computing Gaussian/Laplacian Pyramid



Can we reconstruct the original from the laplacian pyramid?

Slide: Derek Hoiem