Leow Wee Kheng

CS4243 Computer Vision and Pattern Recognition

# Motion Tracking

# Changes are everywhere!

# Illumination change

# Shape change

# Object motion

# Camera motion

# Object & camera motion

# Everything changes!

# Motion analysis is tough in general!

We focus on object / camera motion.

# Change Detection

- Detects any change in two video frames.
- Straightforward method:
  - Compute difference between corresponding pixels:

$$D_t(x, y) = |I(x, y, t+1) - I(x, y, t)|$$

  - $I(x, y, t)$ : intensity / colour at (*x*, *y*) in frame *t*.
  - If $D_t(x, y)$ > threshold, has large difference.

# Any difference?

No

# Any difference?

Yes, illumination change

# Any difference?

Yes,
position change

# Change Detection

- Can detect
  - Illumination change
  - Position change
  - Illumination and position change
- But, cannot distinguish between them.
- Need to detect and measure position change.

# Motion Tracking

- Two approaches
  - Feature-based
  - Intensity gradient-based

# Feature-based Motion Tracking

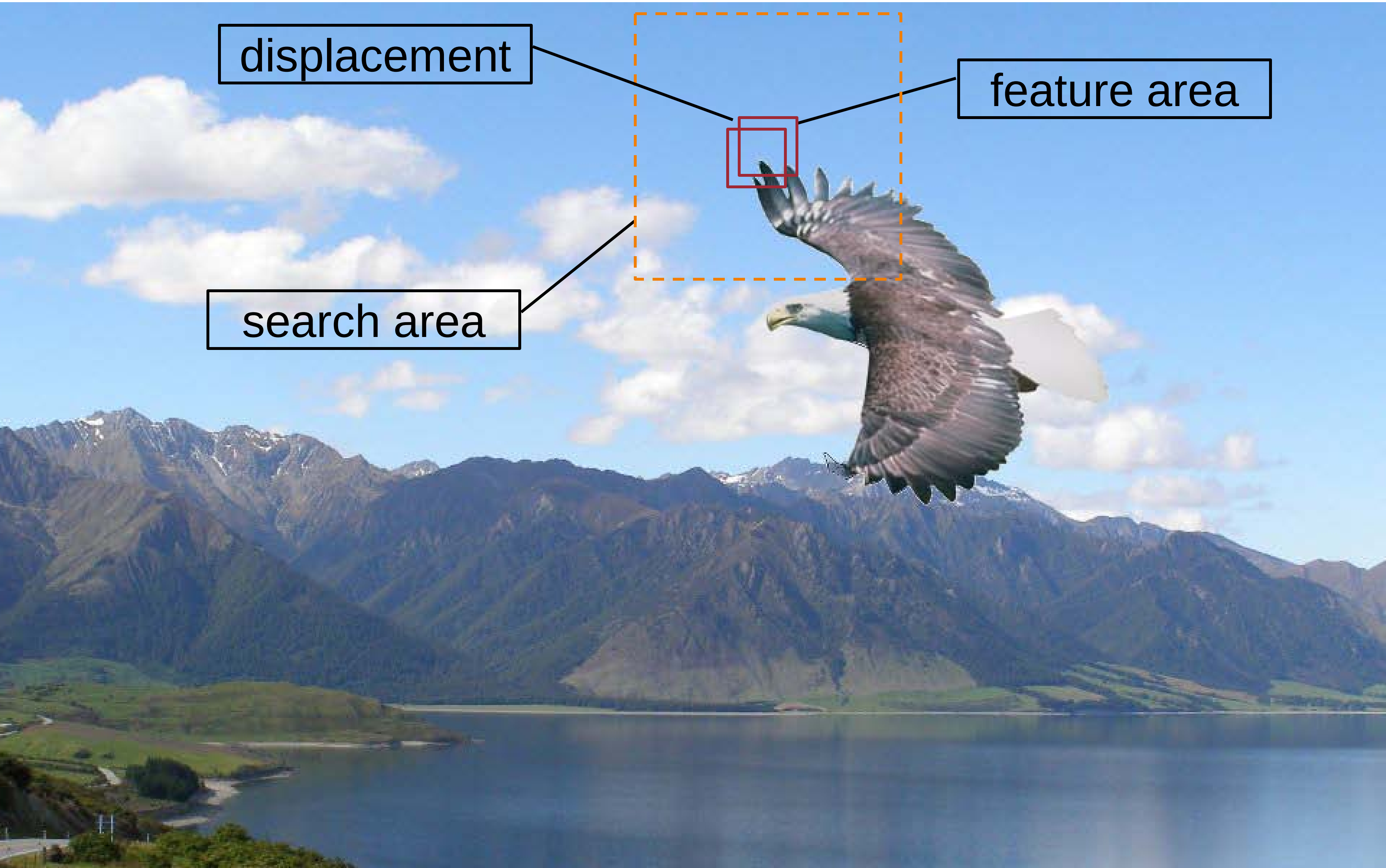
- Look for distinct features that change positions.

  - Eagle's wing tips change positions.
  - Tree tops don't change positions.

# Basic Ideas

1. Look for distinct features in current frame.
2. For each feature
   - Search for matching feature within neighbourhood in next frame.
   - Difference in positions → displacement.
   - Velocity = displacement / time difference.

# Basic Ideas



displacement

feature area

search area

# What feature to use?

- Harris corner
- Tomasi’s feature
- Feature descriptors: SIFT, SURF, GLOH, etc.
- Others

# Summary

- Simple algorithm.
- Can be slow if search area is large.
- Can constrain search area with prior knowledge.

# Gradient-based Motion Tracking

- Two basic assumptions
  - Intensity changes smoothly with position.
  - Intensity of object doesn’t change over time.
- Suppose an object is in motion.
  - Change position (d*x*, d*y*) over time d*t*.
  - Then, from 2nd assumption:

$$I(x+dx, y+dy, t+dt) = I(x, y, t)$$

  - Apply Taylor’s series expansion:

$$I(x+dx, y+dy, t+dt) = I(x, y, t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt + \cdots$$

- Omit higher order terms and divide by d*t*

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

- Denote

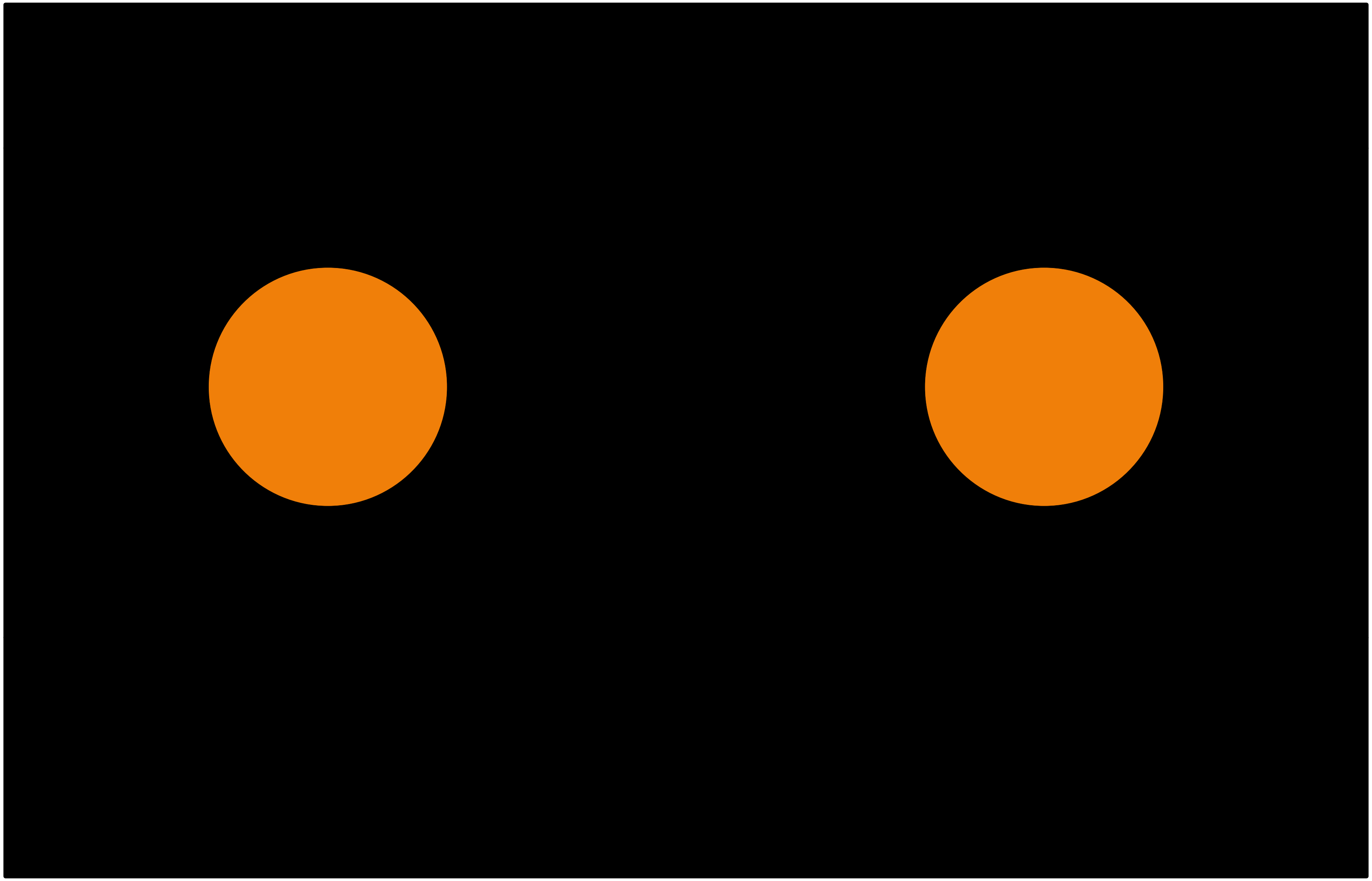
$$u = \frac{dx}{dt} \;,\; v = \frac{dy}{dt} \;,\; I_x = \frac{\partial I}{\partial x} \;,\; I_y = \frac{\partial I}{\partial y} \;,\; I_t = \frac{\partial I}{\partial t} \;.$$

- Then,

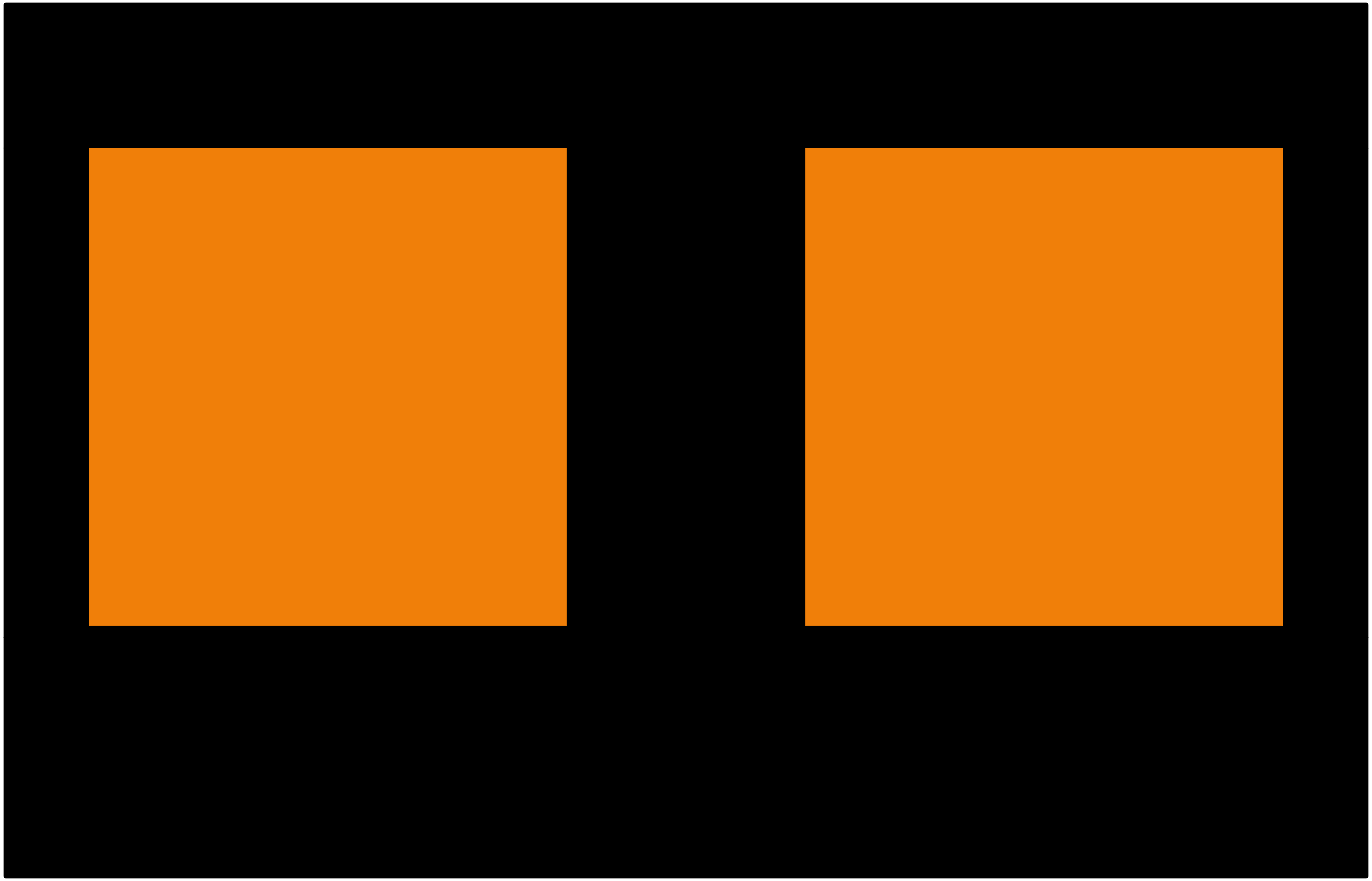
$$I_x u + I_y v + I_t = 0$$

- *u*, *v* are unknown.
- 2 unknowns, 1 equation: can’t solve!

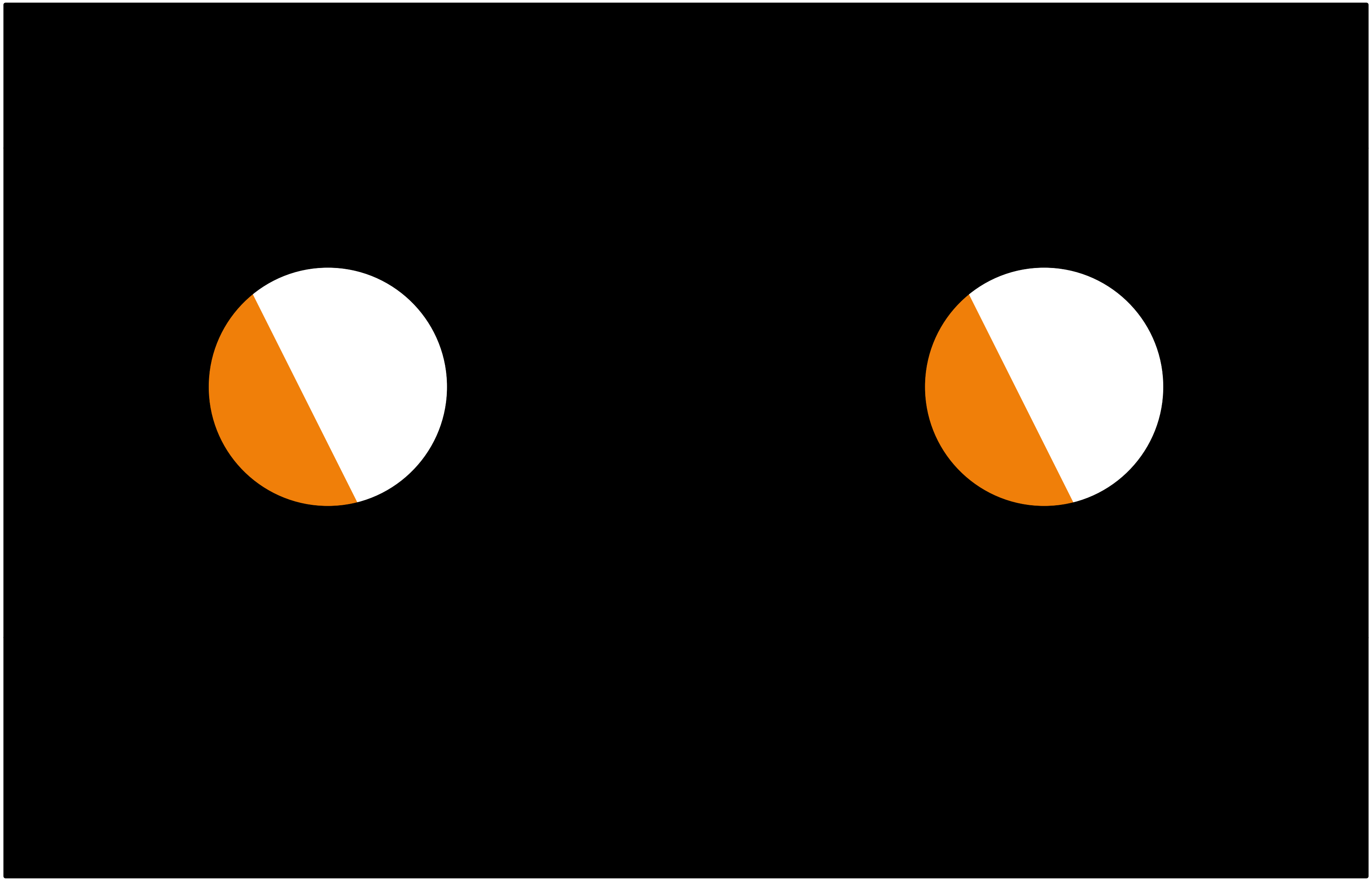
# Any difference in motion?



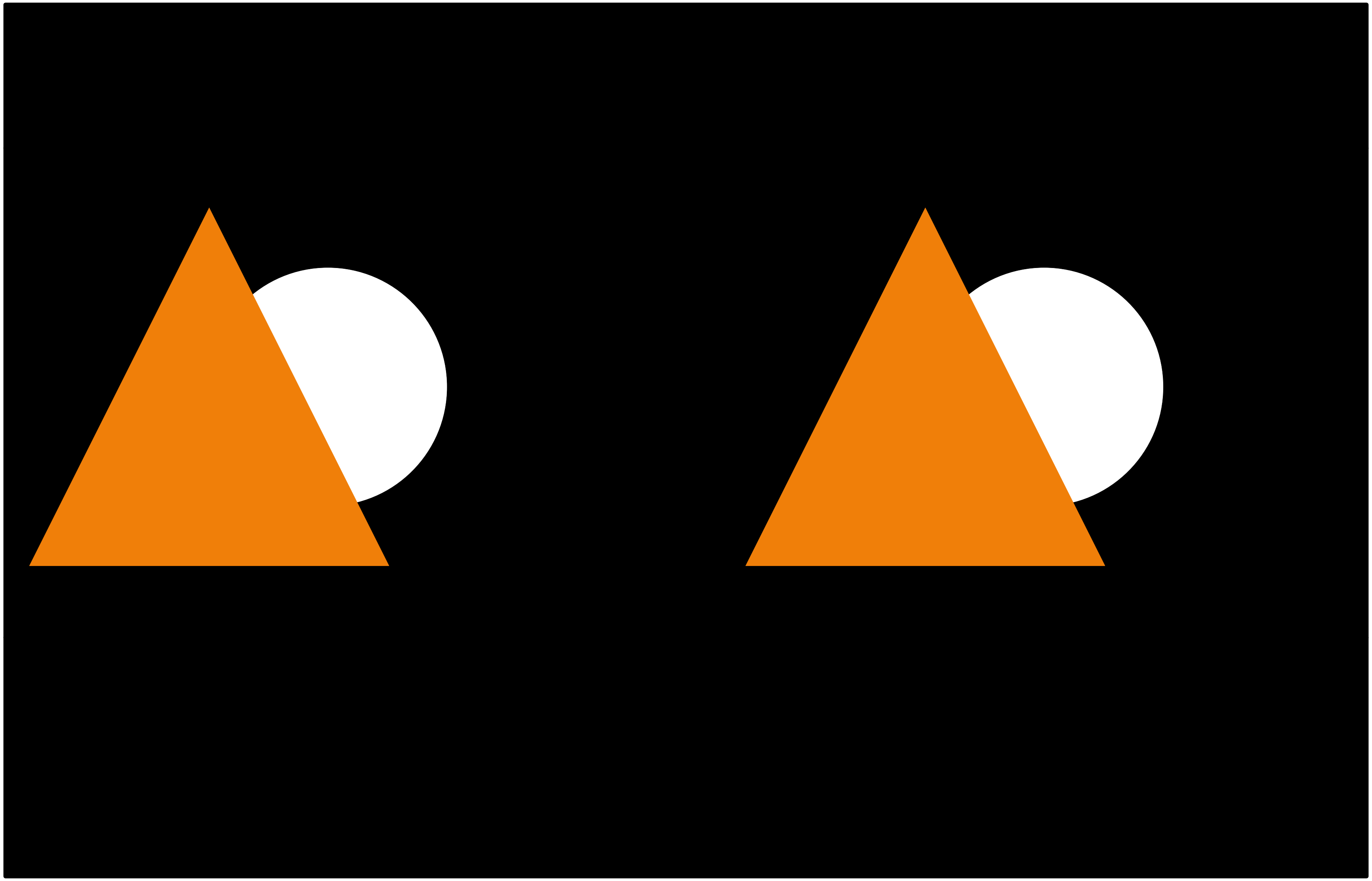
# Any difference in motion?



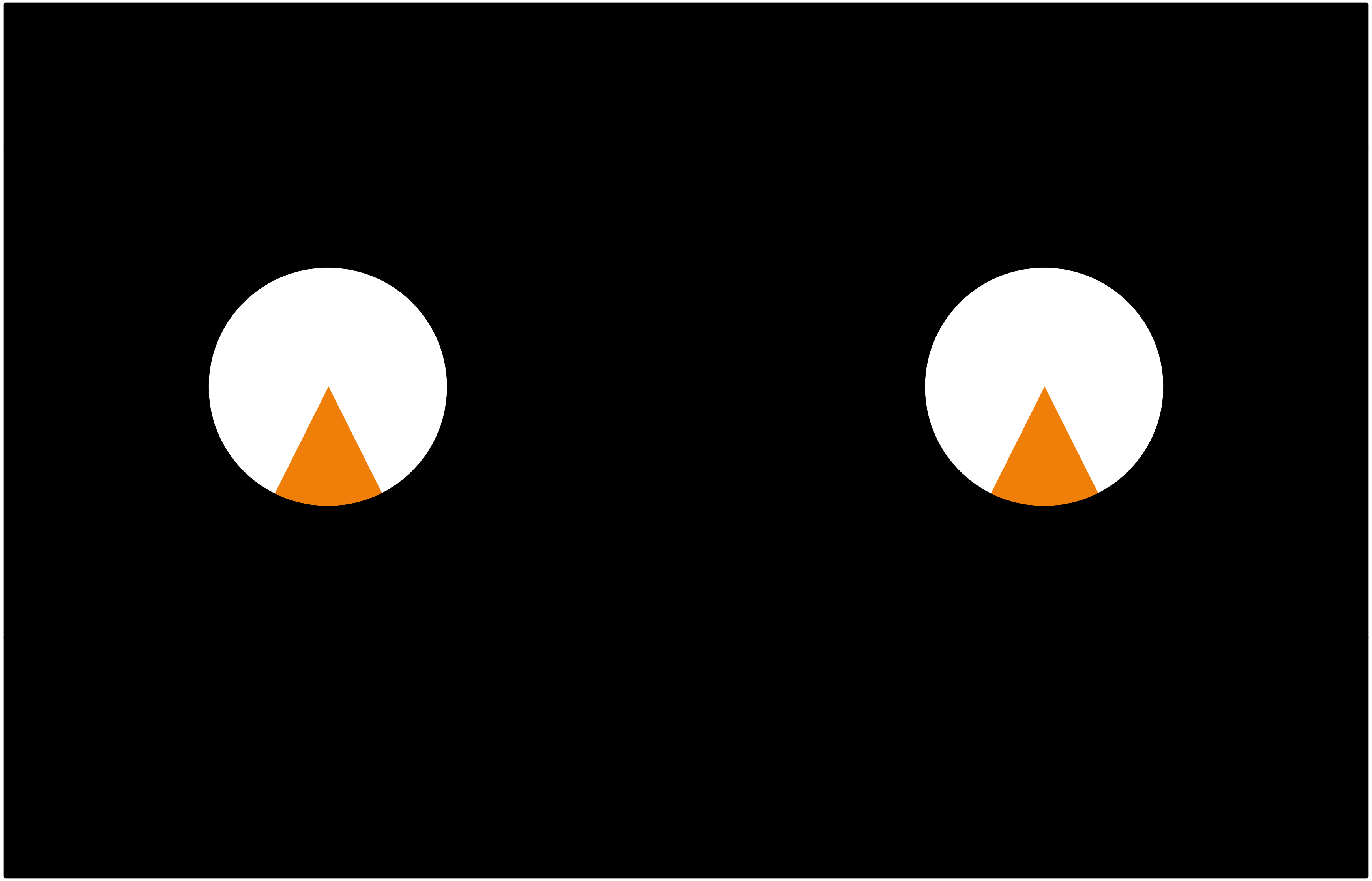
# Any difference in motion?



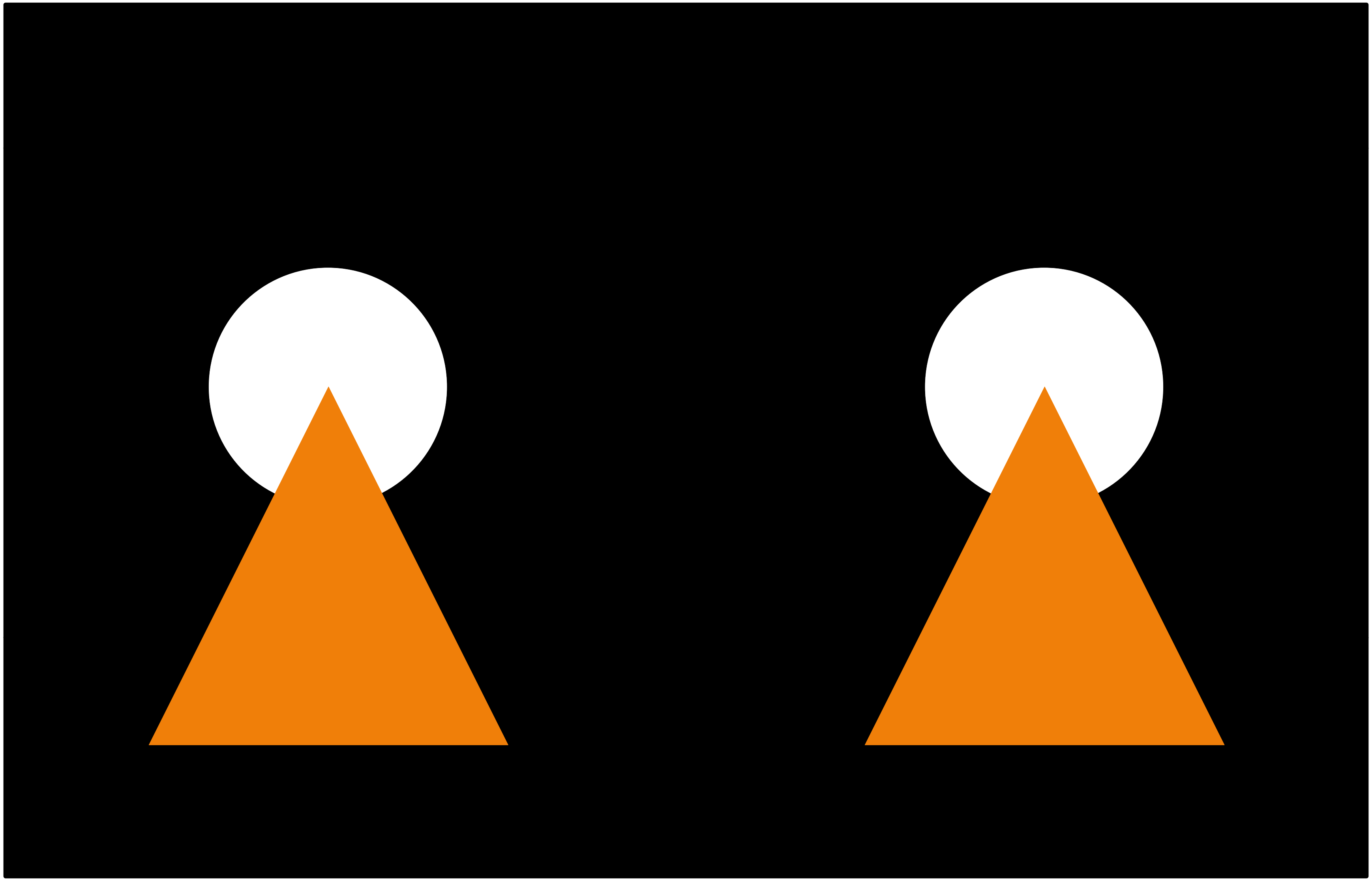
# Any difference in motion?



# Any difference in motion?



# Any difference in motion?



# Aperture Problem

- Homogeneous region

$$I_x u + I_y v + I_t = 0$$

  - $I_x = I_y = I_t = 0$.
  - No change in local region.
  - Cannot detect motion.

# Aperture Problem

- Edge

$$I_x u + I_y v + I_t = 0$$

  - $I_x$ and $I_y$ are zero along edge
  - Cannot measure motion tangential to edge
  - $I_x$ and $I_y$ are non-zero normal to edge
  - Can measure motion normal to edge
  - So, cannot measure actual motion

# Aperture Problem

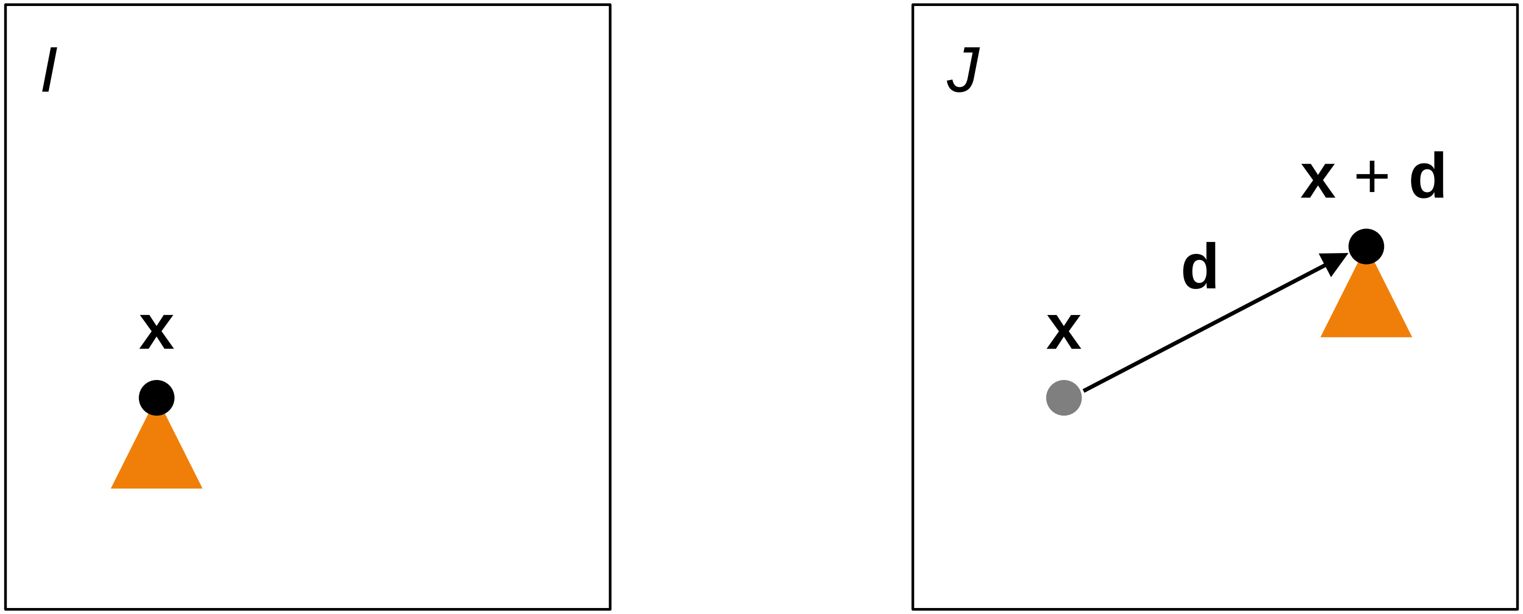
- Corner

$$I_x u + I_y v + I_t = 0$$

  - $I_x$ and $I_y$ are non-zero in two perpendicular directions
  - 2 unknowns, 2 equations
  - Can measure actual motion

# Lucas-Kanade Method

- Consider two consecutive image frames *I* and *J*:



  - Object moves from $\mathbf{x} = (x, y)^T$ to $\mathbf{x} + \mathbf{d}$.
  - $\mathbf{d} = (u, v)^T$

- So,

$$J(\mathbf{x}+\mathbf{d}) = I(\mathbf{x})$$

- Or

$$J(\mathbf{x}) = I(\mathbf{x}-\mathbf{d})$$

- Due to noise, there's an error at position **x**:

$$e(\mathbf{x}) = I(\mathbf{x}-\mathbf{d}) - J(\mathbf{x})$$

- Sum error over small window *W* at position **x**:

$$E(\mathbf{x}) = \sum_{\mathbf{x}\in W} w(\mathbf{x})\,[I(\mathbf{x}-\mathbf{d}) - J(\mathbf{x})]^2$$

Similar to template matching

weight

- If *E* is small, patterns in *I* and *J* match well.
- So, find **d** that minimises *E*:
  - Set ∂*E* / ∂**d** = 0, compute **d** that minimises *E*.
  - First, expand *I*(**x** – **d**) by Taylor's series expansion:

$$I(x-u, y-v) = I(x, y) - u\, I_x(x, y) - v\, I_y(x, y) + \cdots$$

  - Omit higher order terms:

$$I(x-u, y-v) = I(x, y) - u\, I_x(x, y) - v\, I_y(x, y)$$

  - Write in matrix form:

$$I(\mathbf{x}-\mathbf{d}) = I(\mathbf{x}) - \mathbf{d}^\top \mathbf{g}(\mathbf{x}) \qquad \mathbf{g}(\mathbf{x}) = \begin{bmatrix} I_x(\mathbf{x}) \\ I_y(\mathbf{x}) \end{bmatrix}$$

Intensity gradient

- Now, error *E* at position **x** is:

$$E(\mathbf{x}) = \sum_{\mathbf{x}\in W} w(\mathbf{x}) \left[ I(\mathbf{x}) - J(\mathbf{x}) - \mathbf{d}^\top \mathbf{g}(\mathbf{x}) \right]^2$$

- Now, differentiate *E* with respect to **d** (exercise):

$$\frac{\partial E}{\partial \mathbf{d}} = -2 \sum_{\mathbf{x}\in W} w(\mathbf{x}) \left[ I(\mathbf{x}) - J(\mathbf{x}) - \mathbf{d}^\top \mathbf{g}(\mathbf{x}) \right] \mathbf{g}(\mathbf{x})$$

- Setting ∂*E* / ∂**d** = 0 gives **b**

the only unknown

$$\boxed{\sum_{\mathbf{x}\in W} w(\mathbf{x}) \left[ I(\mathbf{x}) - J(\mathbf{x}) \right] \mathbf{g}(\mathbf{x})}$$

$$= \sum_{\mathbf{x}\in W} w(\mathbf{x})\, \mathbf{d}^\top \mathbf{g}(\mathbf{x})\, \mathbf{g}(\mathbf{x}) = \overset{\mathbf{Z}}{\boxed{\sum_{\mathbf{x}\in W} w(\mathbf{x})\, \mathbf{g}(\mathbf{x})\, \mathbf{g}^\top(\mathbf{x})}}\, \mathbf{d}$$

- So, we get

$$\mathbf{Z}\,\mathbf{d} = \mathbf{b}$$

$$\mathbf{Z} = \begin{bmatrix} \sum_{\mathbf{x}\in W} w\, I_x^2 & \sum_{\mathbf{x}\in W} w\, I_x\, I_y \\ \sum_{\mathbf{x}\in W} w\, I_x\, I_y & \sum_{\mathbf{x}\in W} w\, I_y^2 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} \sum_{\mathbf{x}\in W} w\,(I - J)\, I_x \\ \sum_{\mathbf{x}\in W} w\,(I - J)\, I_y \end{bmatrix}$$

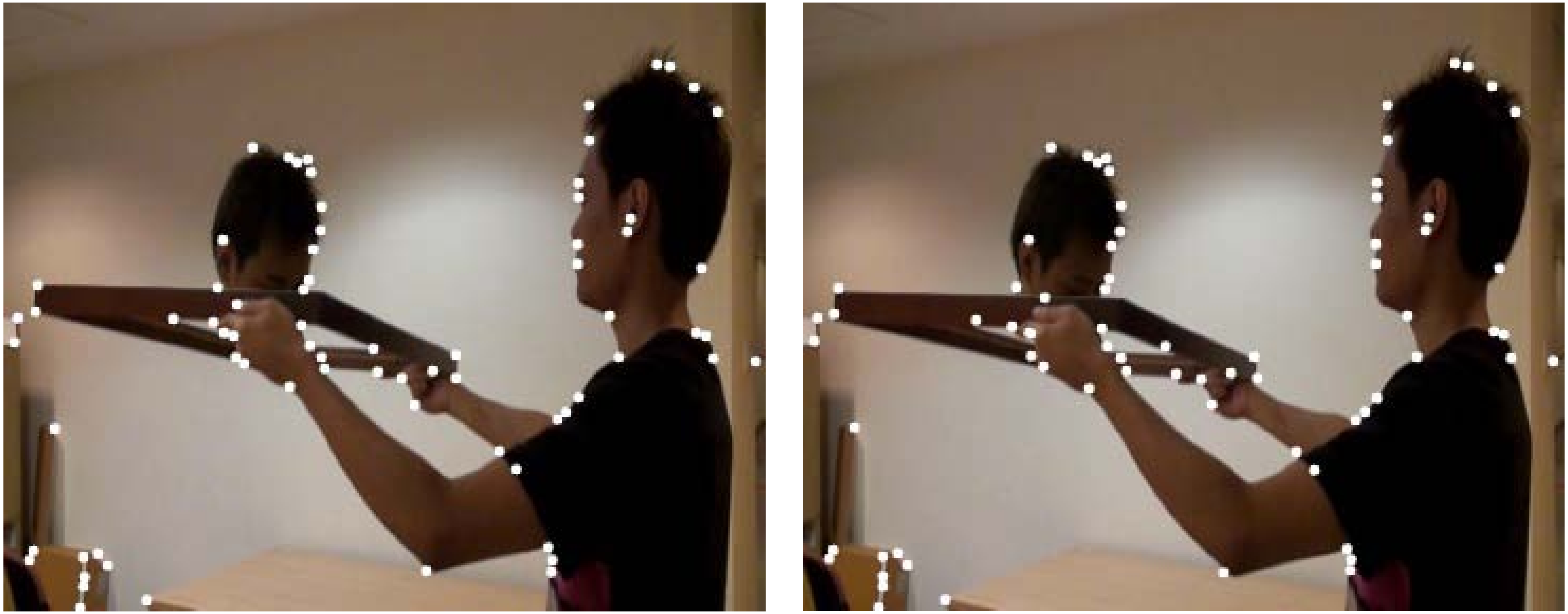
- 2 unknowns, 2 equations. Can solve for **d** = (*u*, *v*).

- What happen to aperture problem?
  Did it disappear?

# Lucas-Kanade + Tomasi

- Lucas-Kanade algorithm is often used with Tomasi's feature
  - Apply Tomasi's method to detect good features.
  - Apply LK method to compute **d** for each pixel.
  - Accept **d** only for good features.

# Example



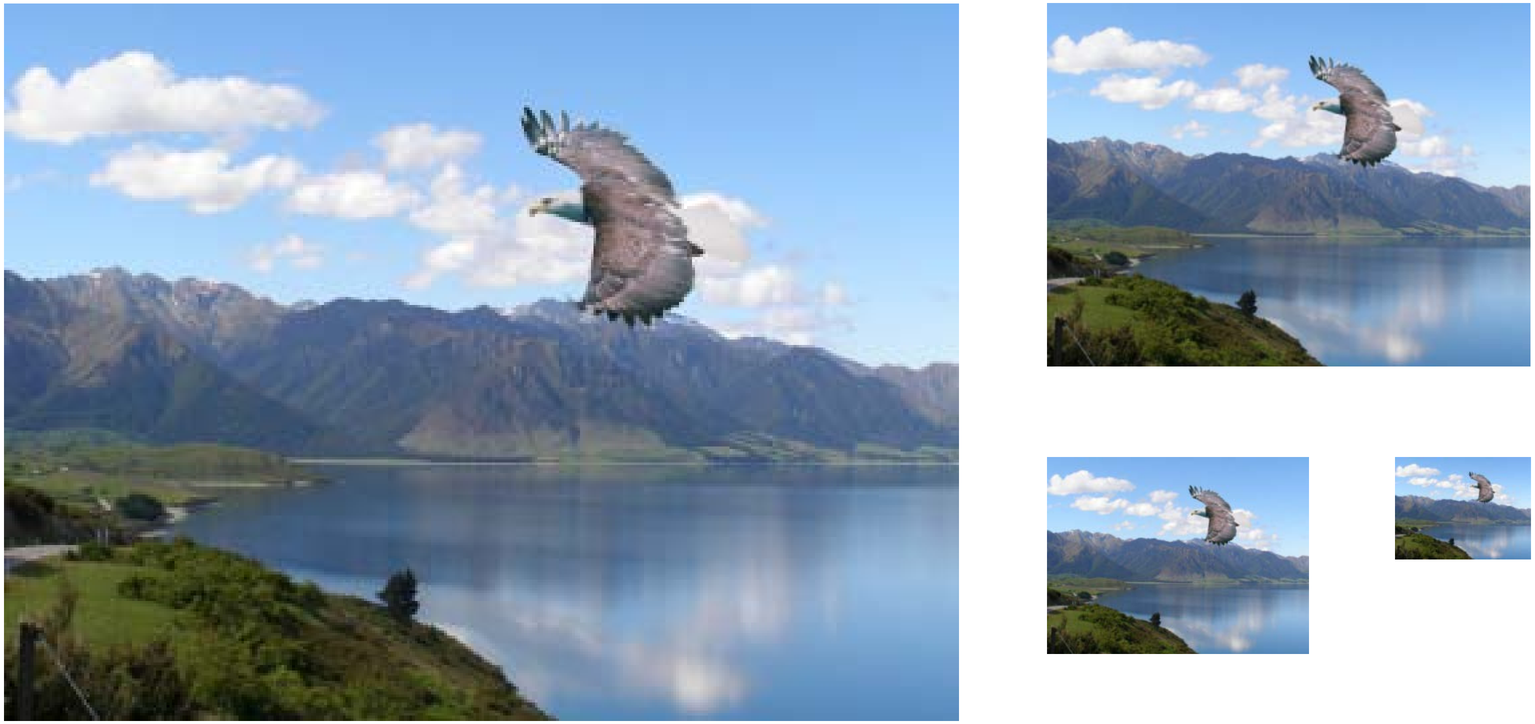
Can you spot tracking errors?

# Constraints

- Math of LK tracker assumes **d** is small.
- In implementation, *W* is also small.
- LK tracker is good only for small displacement.

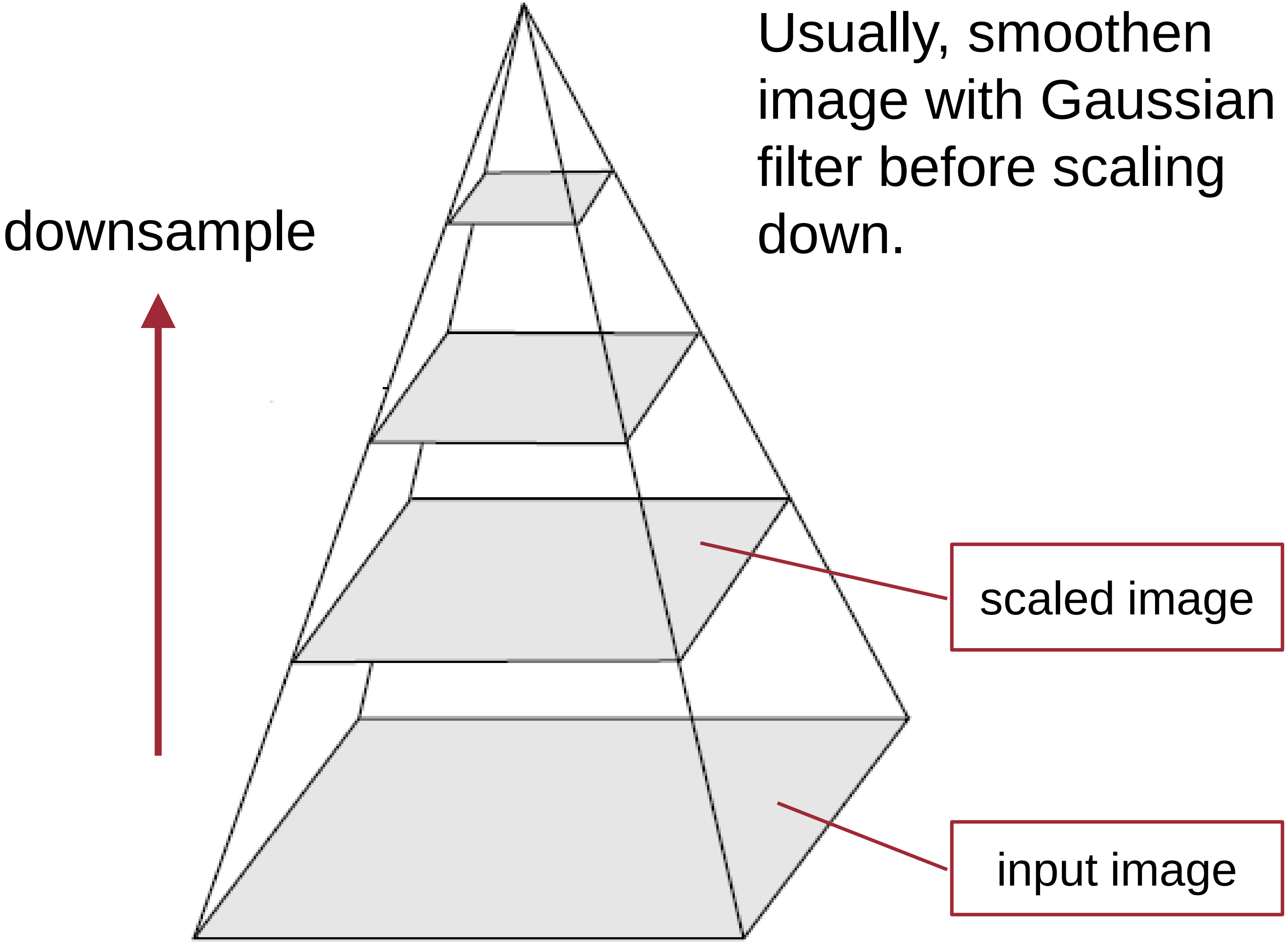
# How to handle large displacement?

- What if we scale down images?
  - Displacements are smaller!

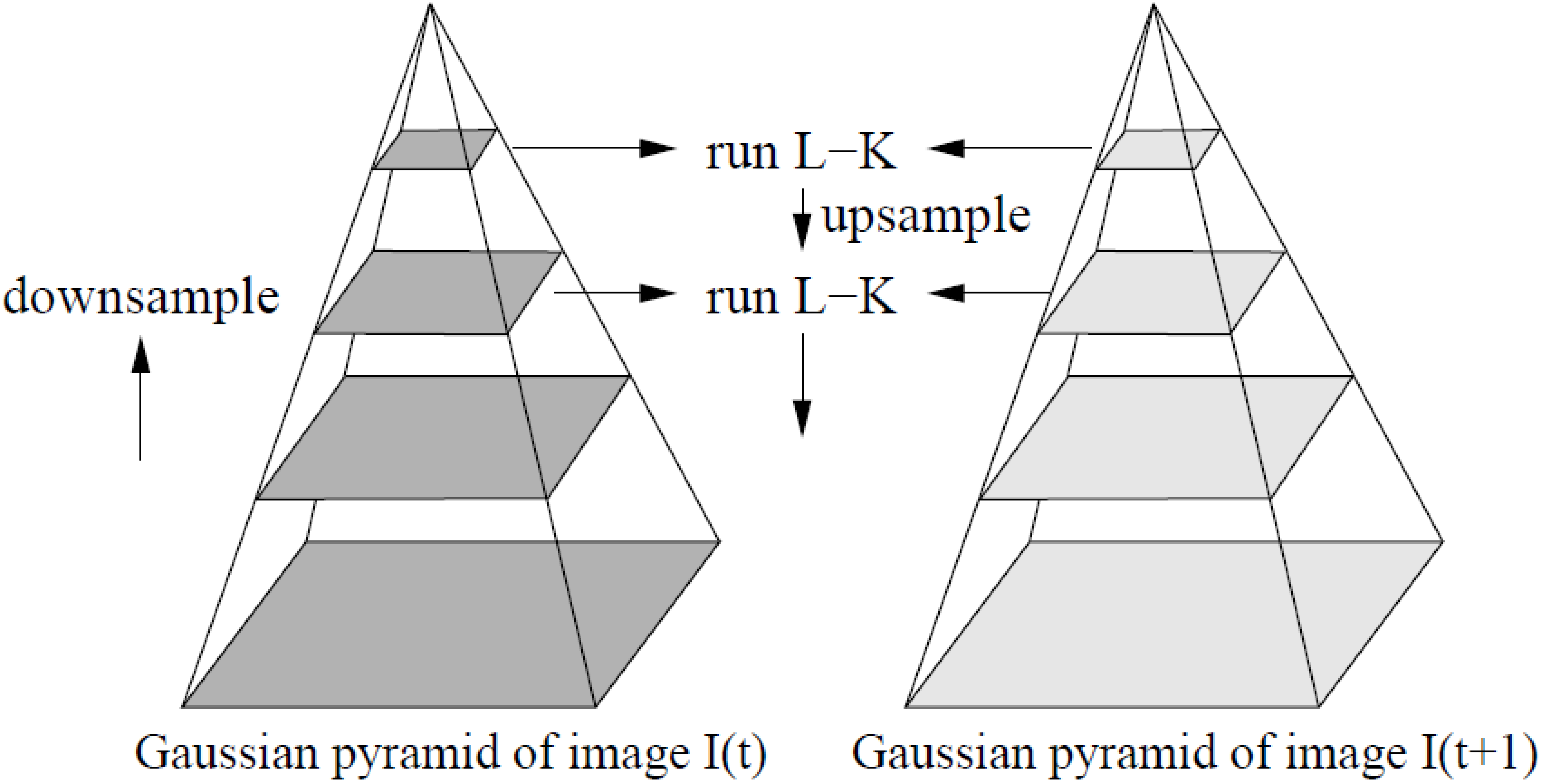


# Image Pyramid

Usually, smoothen image with Gaussian filter before scaling down.

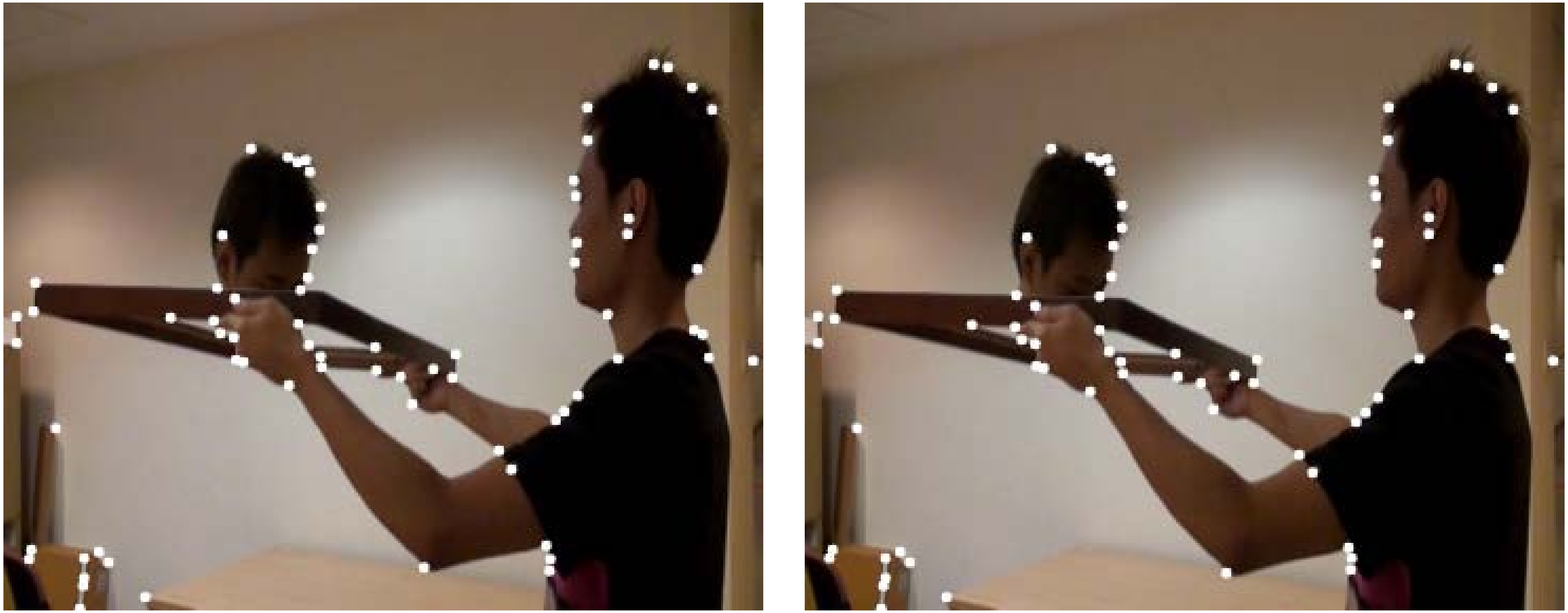


# LK Tracking with Image Pyramid



- Construct image pyramids.
- Apply LK tracker to low-resolution images.
- Propagate results to higher-resolution images.
- Apply LK tracker to higher-resolution images.

# Example



Tracking results are more accurate.

# Summary

- Efficient algorithm, no explicit search.
- Has aperture problem;
  track good features only
- LK tracker can't track large displacement.
- Use LK + image pyramid for large displacement.

# Software

- OpenCV supports LK and LK with pyramid.
- [Bir] offers LK with Tomasi’s features & pyramid.

# Appendix

- Calculation of $I_x$, $I_y$, $I_t$
  - Use finite difference method
  - Forward difference

$$I_x = I(x + 1, y, t) - I(x, y, t)$$
$$I_y = I(x, y + 1, t) - I(x, y, t)$$
$$I_t = I(x, y, t + 1) - I(x, y, t)$$

  - Backward difference

$$I_x = I(x, y, t) - I(x - 1, y, t)$$
$$I_y = I(x, y, t) - I(x, y - 1, t)$$
$$I_t = I(x, y, t) - I(x, y, t - 1)$$

# Further Readings

- Lucas-Kanade tracking with pyramid: [BK08] Chapter 10.
- Optical flow: [Sze10] Section 8.4.
- Hierarchical motion estimation (with image pyramid): [Sze10] Section 8.1.1.

# References

- [Bir] S. Birchfield. *KLT: An implementation of the Kanade-Lucas-Tomasi feature tracker*. http://vision.stanford.edu/~birch/klt/.
- [BK08] Bradski and Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, 2008.
- [LK81] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *In Proceedings of 7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.
- [ST94] J. Shi and C. Tomasi. Good features to track. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.
- [Sze10] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.

# References

- [TK91] C. Tomasi and T. Kanade. *Detection and tracking of point features*. Technical Report CMU-CS-91-132, School of Computer Science, Carnegie Mellon University, 1991.