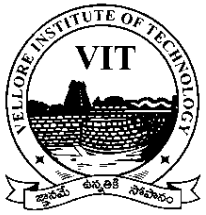


VIT-AP UNIVERSITY

Computer Vision

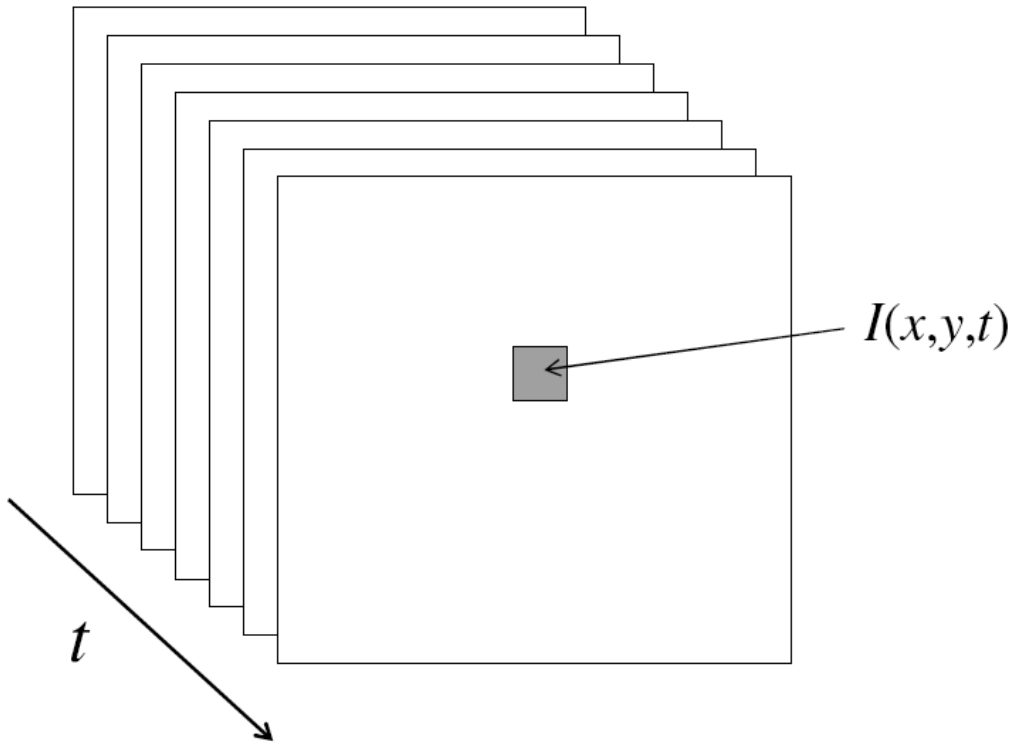
(Course Code: 4047)

Module-3:Lecture-2: Optical Flow

Gundimeda Venugopal, Professor of Practice, SCOPE

Video

- A video is a sequence of frames captured over time
- Now our image data is a function of space (x, y) and time (t)



Optical Flow

Method to estimate apparent motion of scene points from a sequence of images.

Topics:

(1) Motion Field and Optical Flow

(2) Optical Flow Constraint Equation

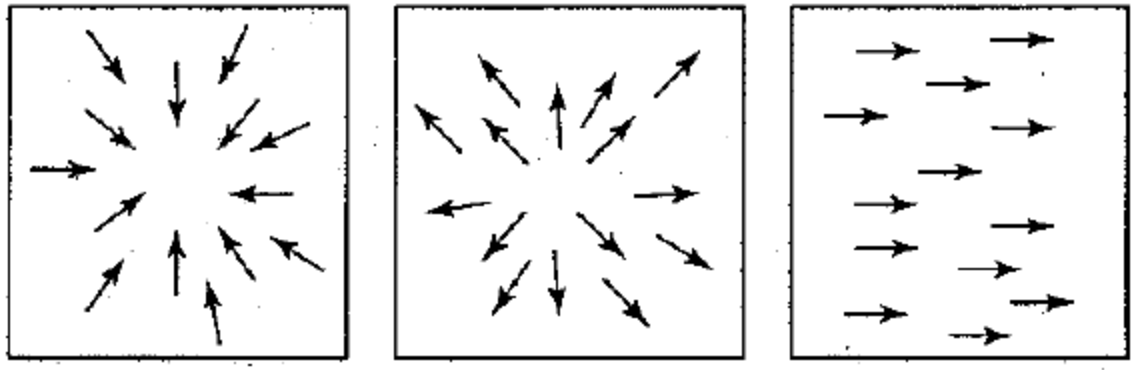
(3) Lucas-Kanade Method

(4) Coarse-to-Fine Flow Estimation

Optical flow

- Definition: optical flow is the *apparent* motion of brightness patterns in the image
- Ideally, optical flow would be the same as the motion field
- Have to be careful: apparent motion can be caused by lighting changes without any actual motion

Motion field + camera motion



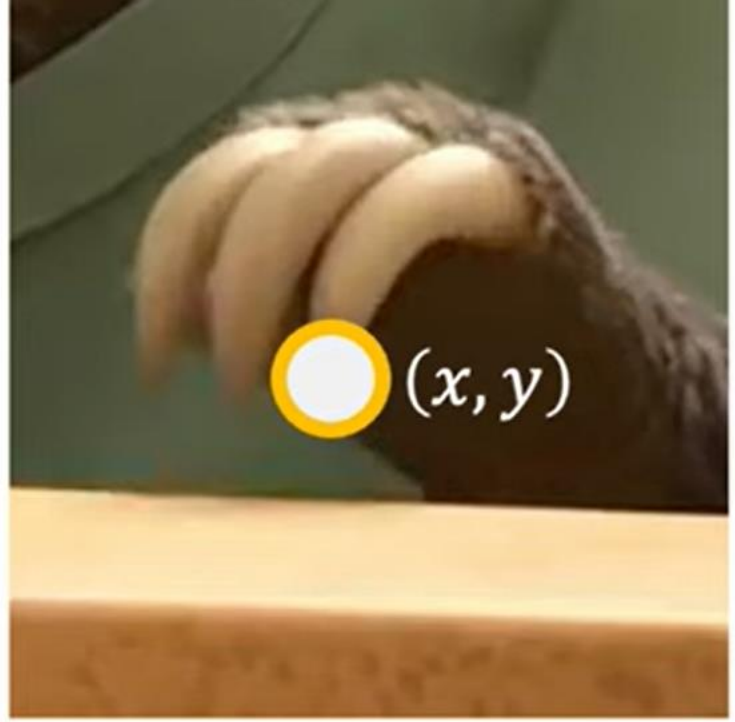
Zoom out **Zoom in** **Pan right to left**

Optical flow

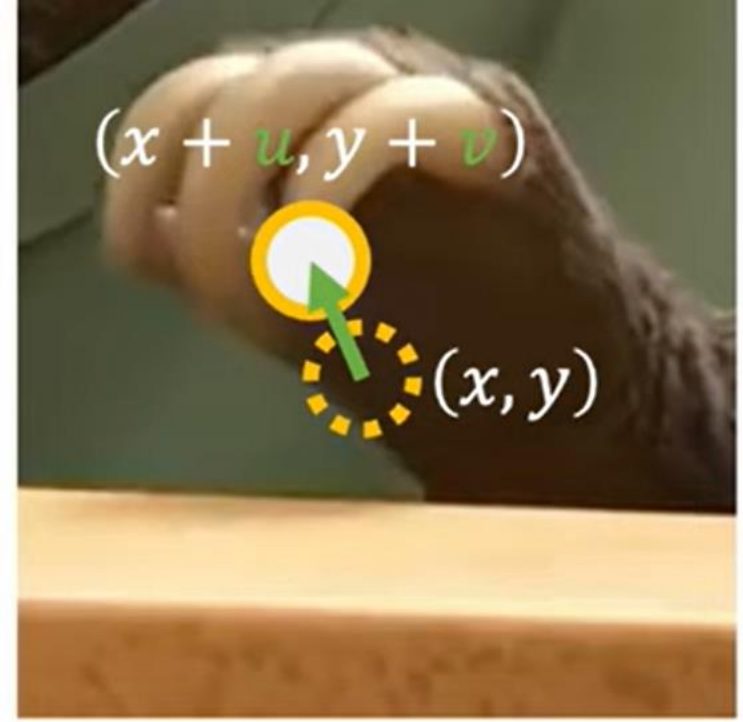
- Definition: optical flow is the *apparent* motion of brightness patterns in the image

Optical flow

Thomas Brox, large displacement optical flow: https://lmb.informatik.uni-freiburg.de/research/opticalflow/



Frame t



Frame $t + 1$

FEATURE TRACKING

Video stabilization

Object tracking

Motion analysis

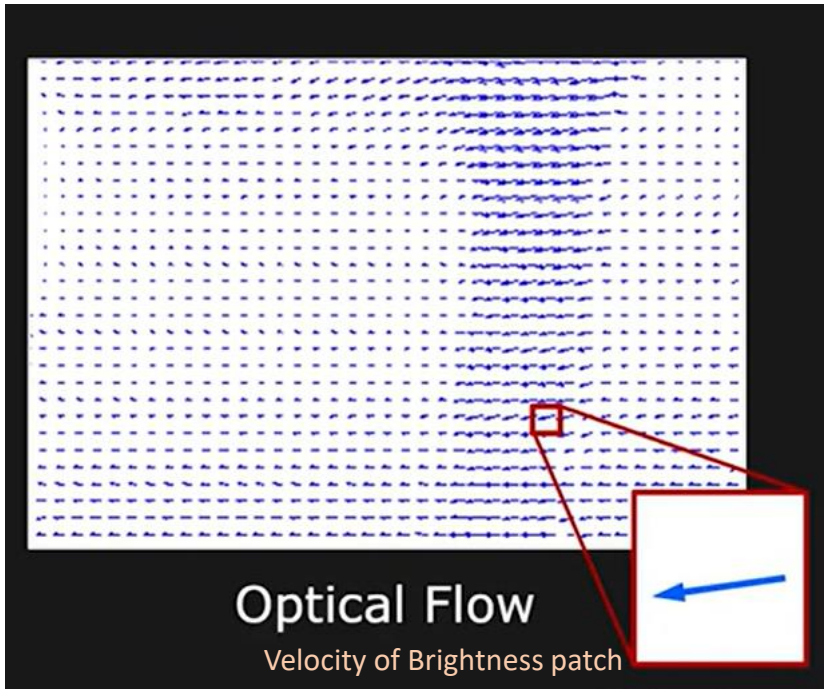
SLAM

Other Uses of motion

- Tracking and Moving Mouse pointer using an Optical Mouse
- Traffic Monitoring
- Video Retiming (Generate slow motion effects through higher frame rate)
- Face Tracking
- Interactive Gaming
- Estimating 3D structure
- Segmenting objects based on motion cues
- Learning dynamical models
- Recognizing events and activities

Optical Flow

Motion of **Brightness patterns** in the image

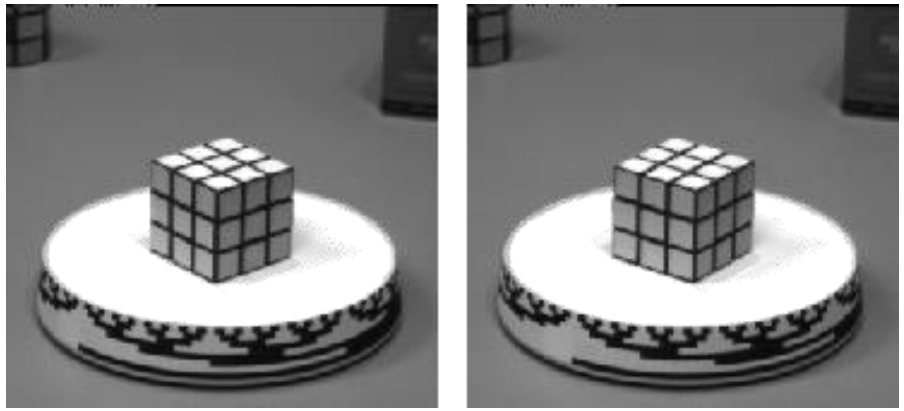


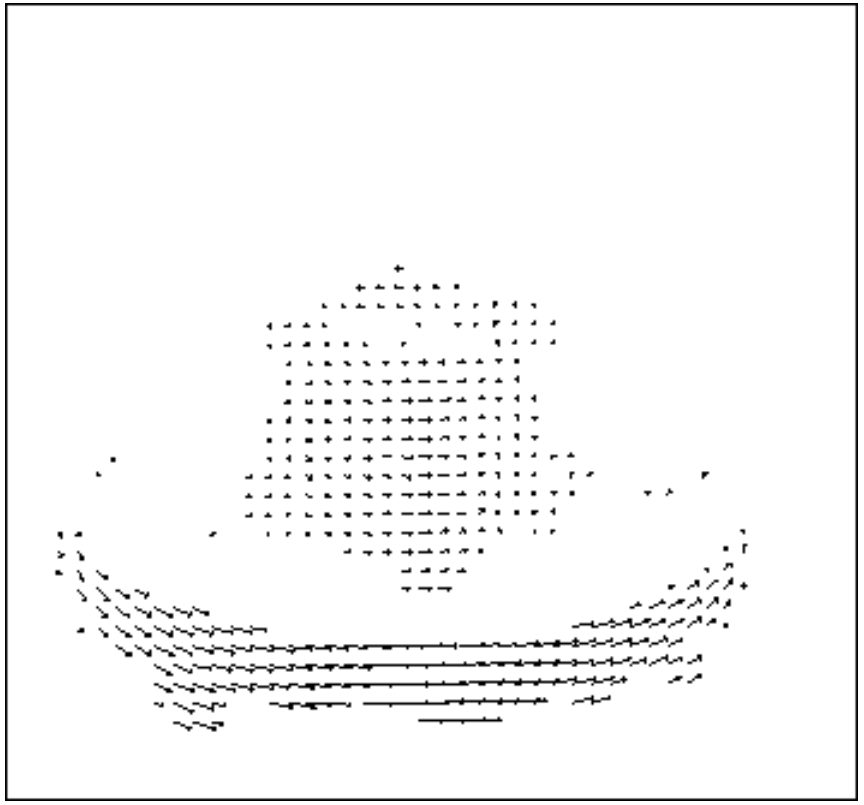
Motion of Brightness patterns in the image

Ideally, **Optical Flow = Motion Field**

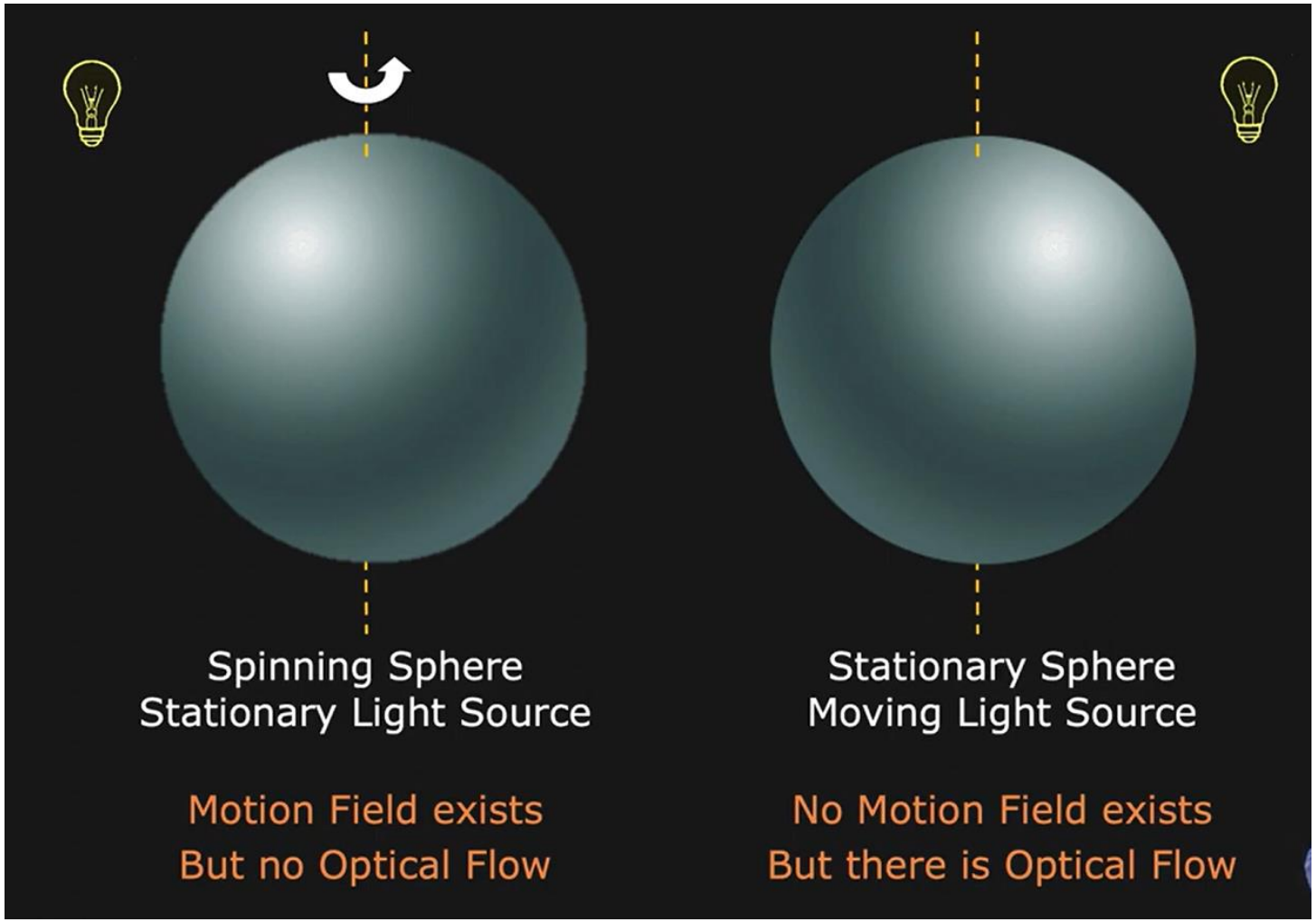
Motion field

- The motion field is the projection of the 3D scene motion into the image





When is Optical Flow ≠ Motion Field



Look at the shading on the sphere

In the first case, shading does not move at all (no optical flow).

In the second case, shading moves with the light source (optical field)

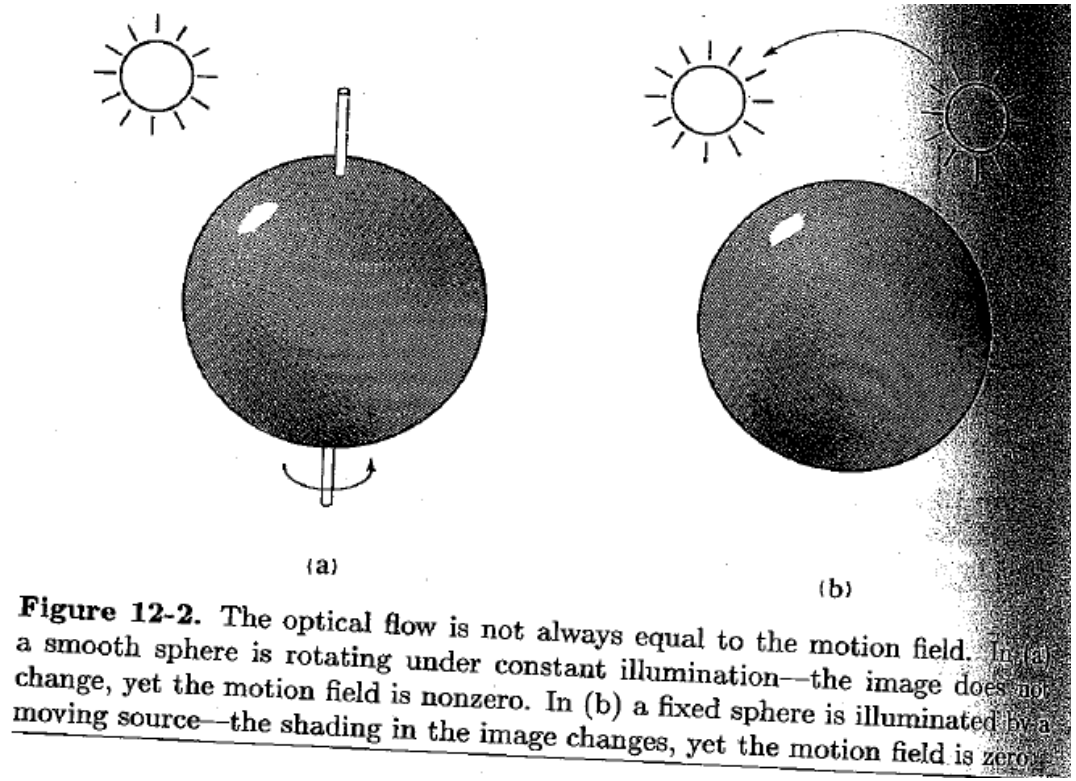


Figure 12-2. The optical flow is not always equal to the motion field. In (a) a smooth sphere is rotating under constant illumination—the image does not change, yet the motion field is nonzero. In (b) a fixed sphere is illuminated by a moving source—the shading in the image changes, yet the motion field is zero.

Motion field + camera motion

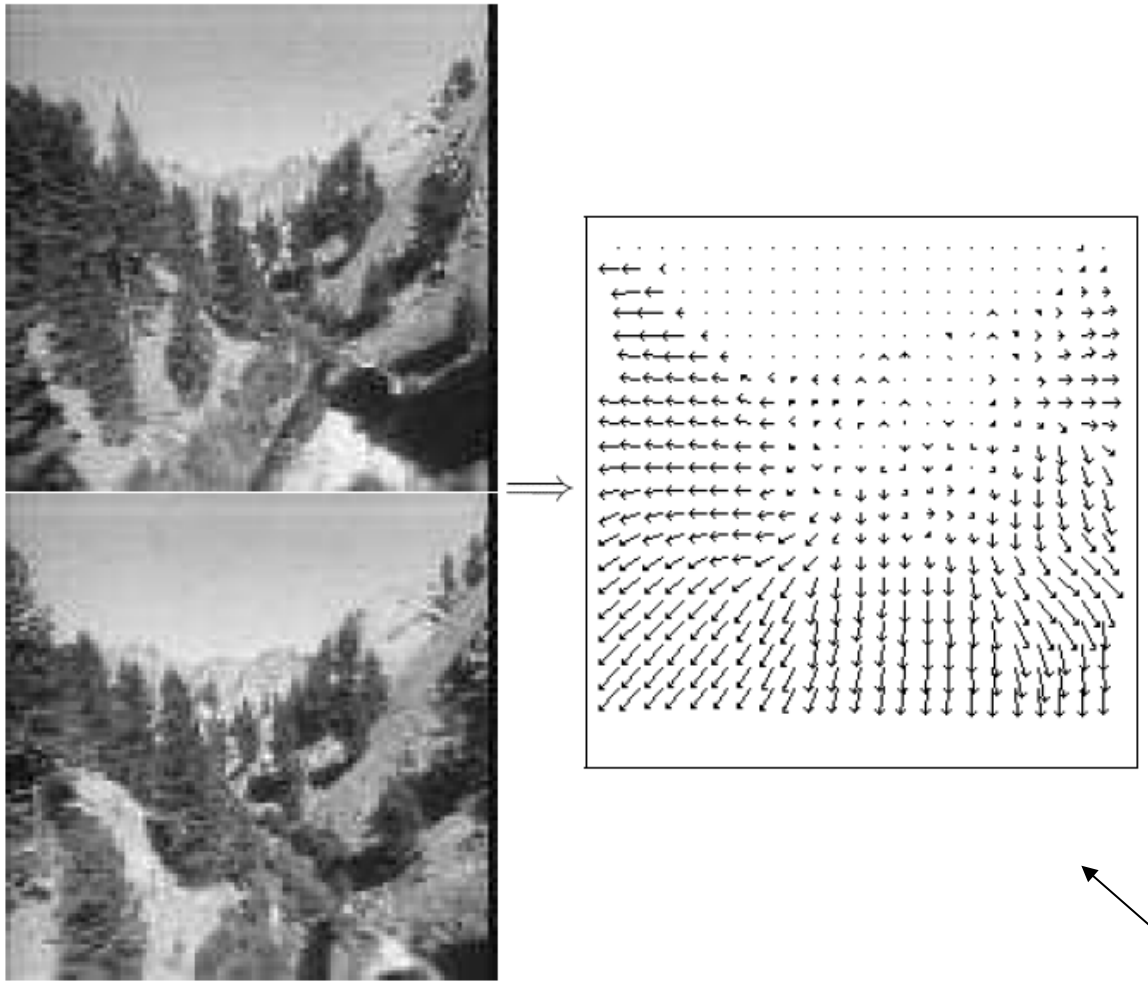


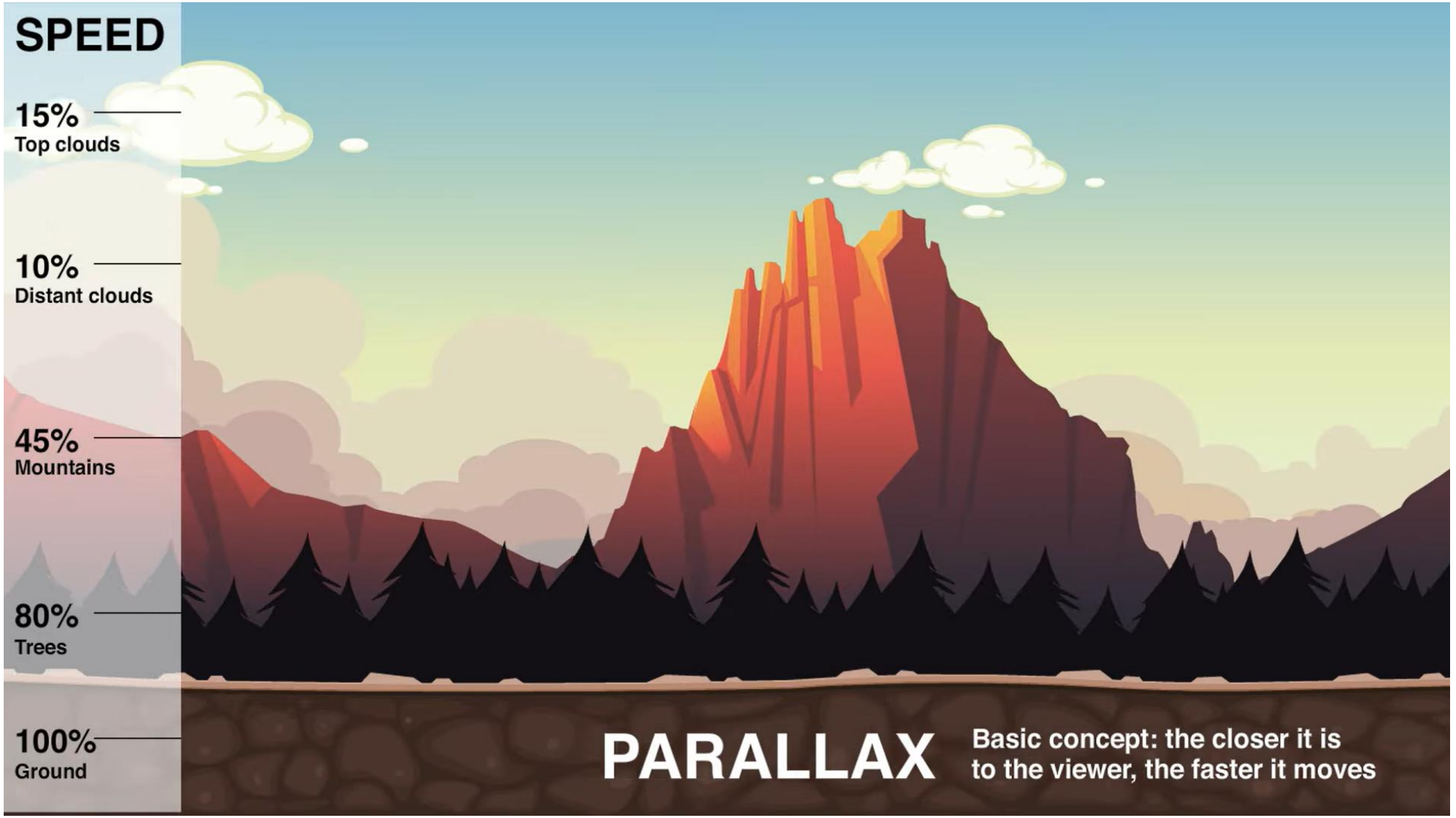
Figure 1.2: Two images taken from a helicopter flying through a canyon and the computed optical flow field.

Figure from Michael Black, Ph.D. Thesis

Length of flow vectors inversely proportional to depth Z of 3d point

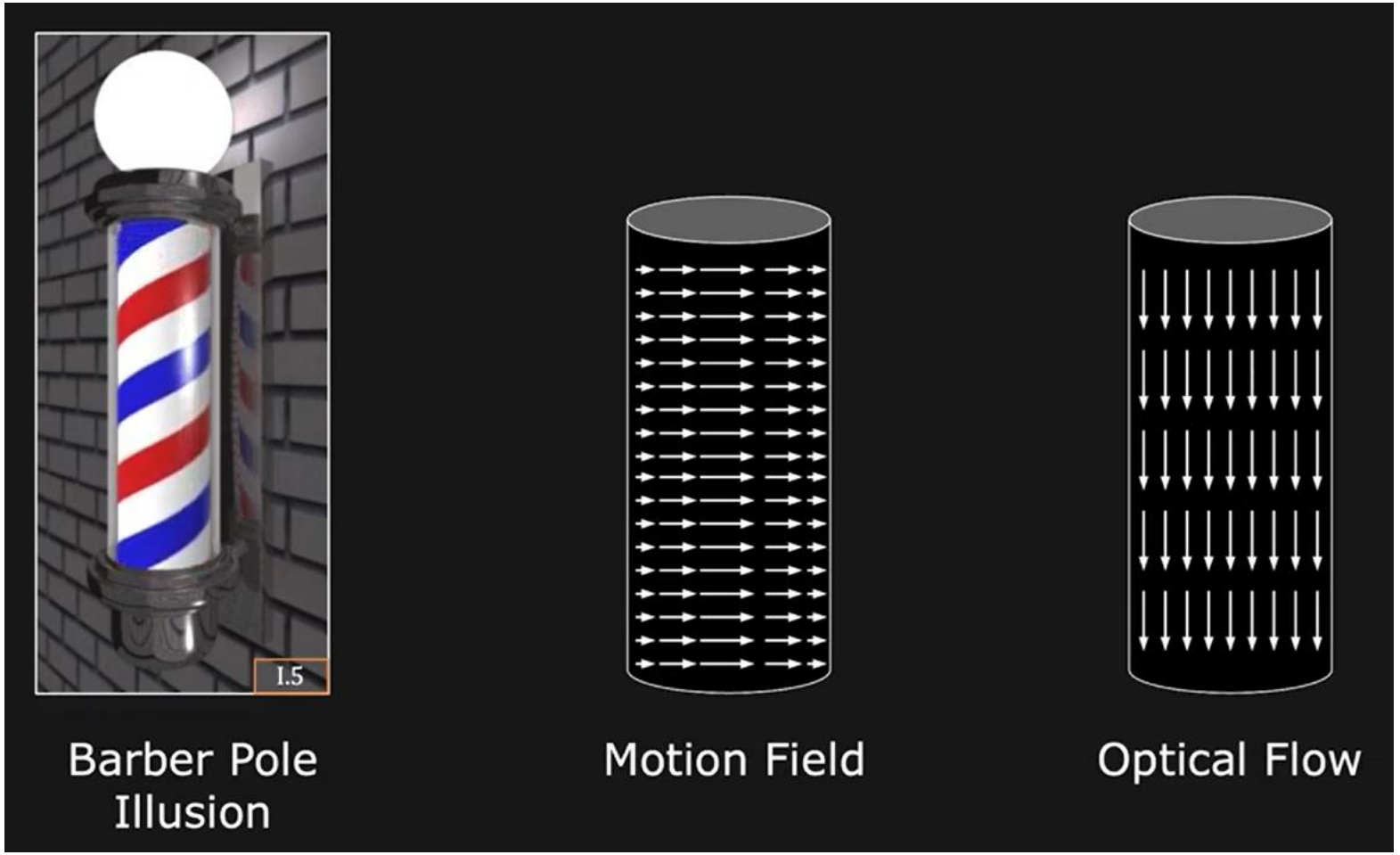
points closer to the camera move more quickly across the image plane

Motion parallax



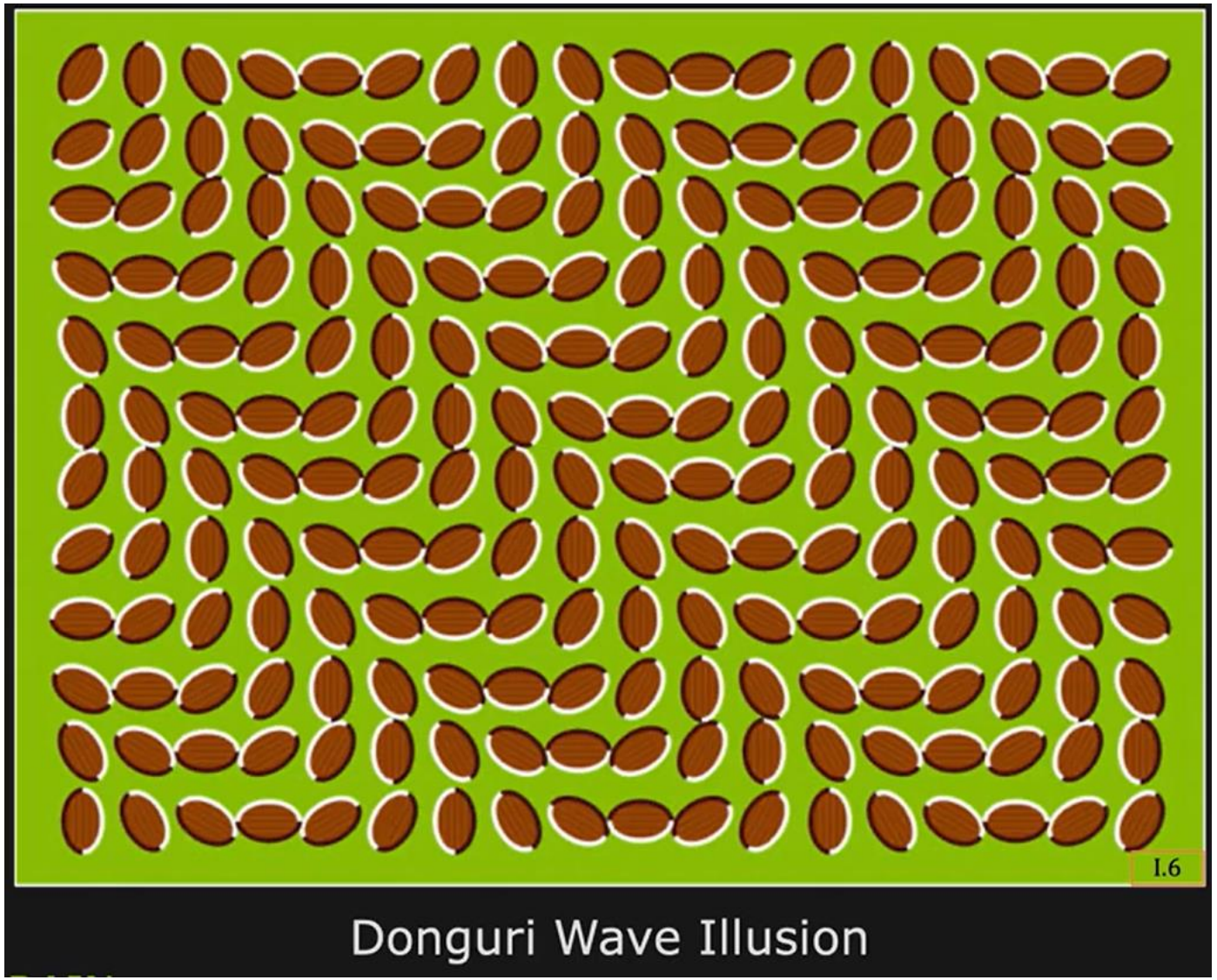
Parallax Scroll Comparison: https://www.youtube.com/watch?v=2z4OTRFuLP8

Barber Poll illusion



http://en.wikipedia.org/wiki/Barberpole_illusion

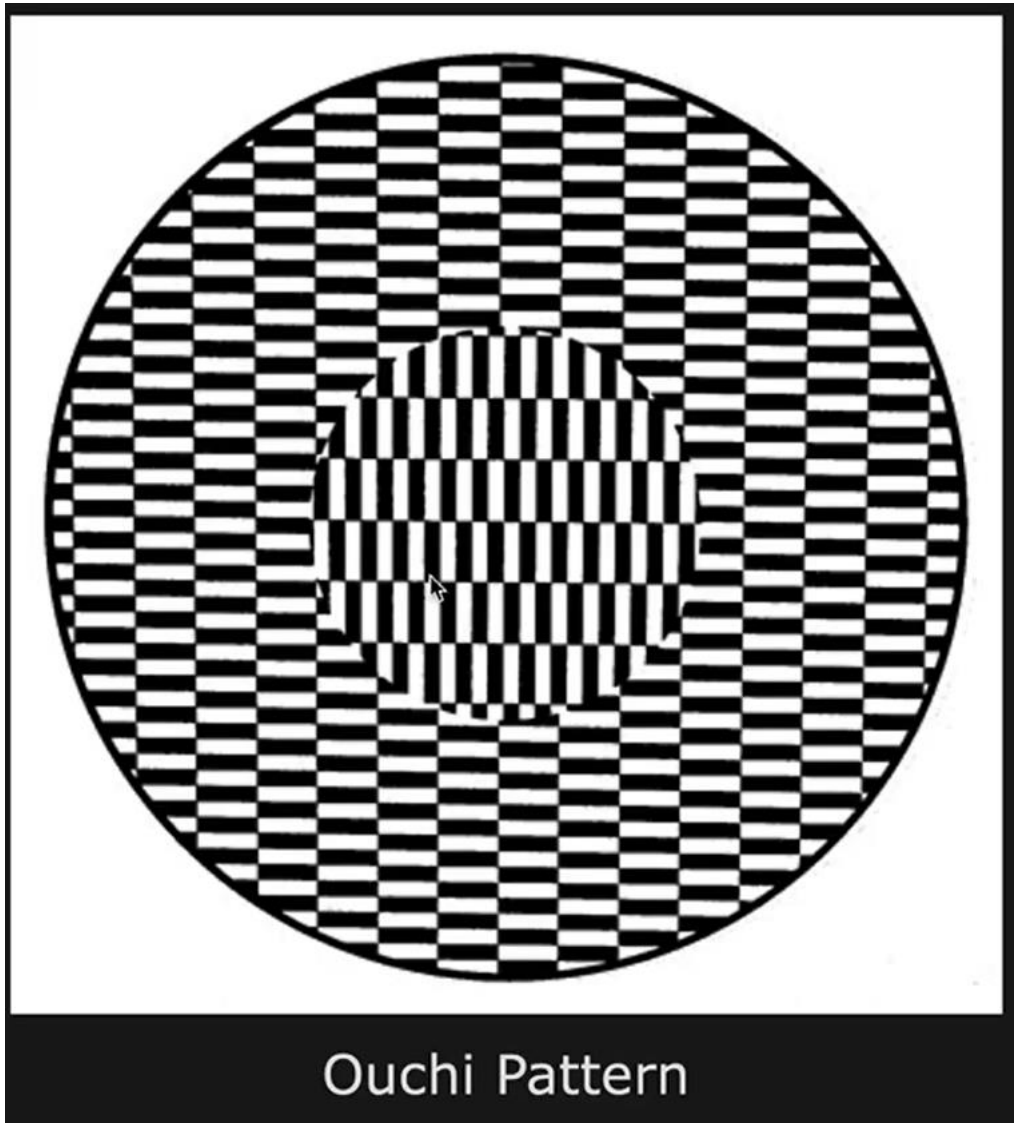
Motion illusions



Move your eyes:

Our visual system sees the optical field

Illusion of Leaves moving



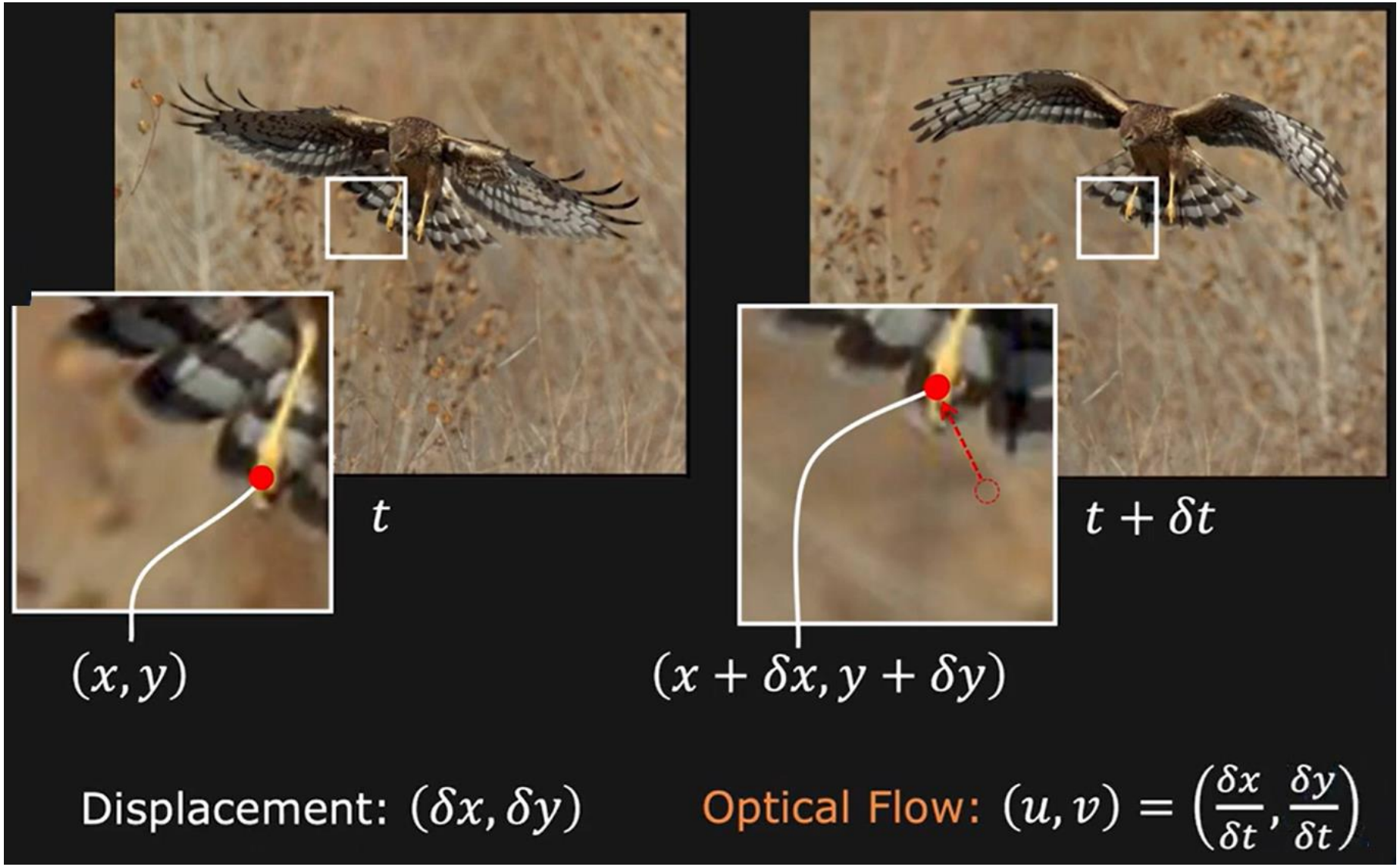
Our visual system perceives the optical field

Illusion of Inner pattern moving

Motion estimation techniques

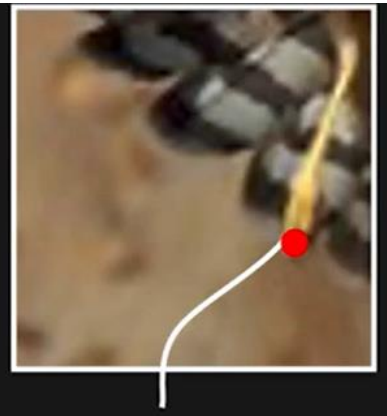
- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small
- Feature-based methods
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)

Optical Flow



An unconstrained problem

Optical Constraint Equation



$I(x, y, t)$

$I(x + \delta x, y + \delta y, t + \delta t)$

Assumption #1:

Brightness of image point remains constant over time

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t)$$

Assumption #2:

Displacement $(\delta x, \delta y)$ and time step δt are small

Taylor Series Expansion

Note: Taylor Series expansion is applicable to any function which is infinitely differentiable which mean all derivatives of the function exist.

Expand a function as an infinite sum of its derivatives

$$f(x+\delta x) = f(x) + \frac{\partial f}{\partial x}\delta x + \frac{\partial^2 f}{\partial x^2}\frac{\delta x^2}{2!} + \cdots + \frac{\partial^n f}{\partial x^n}\frac{\delta x^n}{n!}$$

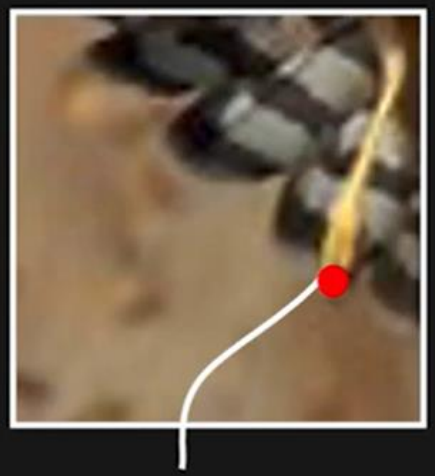
If δx is small:

$$f(x+\delta x) = f(x) + \frac{\partial f}{\partial x}\delta x + \boxed{O(\delta x^2)} \rightarrow \text{Almost Zero}$$

For a function of three variables with small $\delta x, \delta y, \delta t$:

$$f(x+\delta x, y+\delta y, t+\delta t) \approx f(x,y,t) + \frac{\partial f}{\partial x}\delta x + \frac{\partial f}{\partial y}\delta y + \frac{\partial f}{\partial t}\delta t$$

Optical Constraint Equation



$I(x, y, t)$

$I(x + \delta x, y + \delta y, t + \delta t)$

Assumption #2:

Displacement $(\delta x, \delta y)$ and time step δt are small

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t$$

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + I_x \delta x + I_y \delta y + I_t \delta t$$

Optical Constraint Equation

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) \quad \text{----------} (1)$$

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + I_x\delta x + I_y\delta y + I_t\delta t \quad \text{----------} (2)$$

Subtract (1) from (2): $I_x\delta x + I_y\delta y + I_t\delta t = 0$

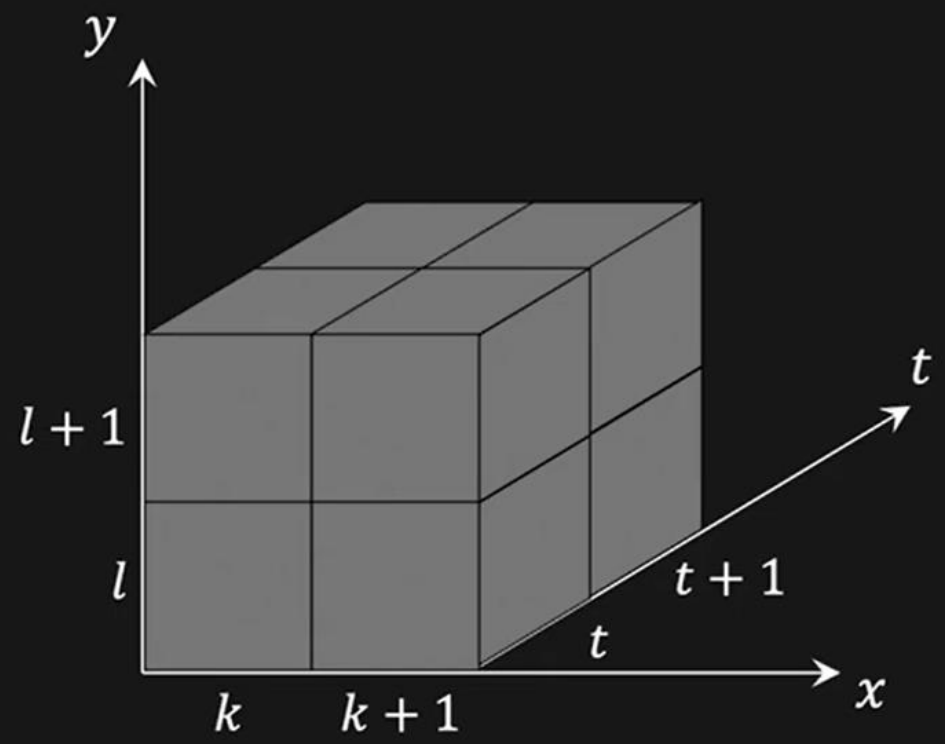
Divide by δt and take limit as $\delta t \to 0$: $I_x\frac{\partial x}{\partial t} + I_y\frac{\partial y}{\partial t} + I_t = 0$

Constraint Equation: $\boxed{I_x u + I_y v + I_t = 0}$ (u, v): Optical Flow

(I_x, I_y, I_t) can be easily computed from two frames

$$0 \approx I_t + \nabla I \cdot [u \ v]$$

Computing Partial Derivatives I_x, I_y, I_t



$$I_x(k,l,t) =$$

$$\frac{1}{4}[I(k+1,l,t) + I(k+1,l,t+1) + I(k+1,l+1,t) + I(k+1,l+1,t+1)]$$

$$-\frac{1}{4}[I(k,l,t) + I(k,l,t+1) + I(k,l+1,t) + I(k,l+1,t+1)]$$

Similarly find $I_y(k,l,t)$ and $I_t(k,l,t)$

Geometric Interpretation

For any point (x, y) in the image, its optical flow (u, v) lies on the line:

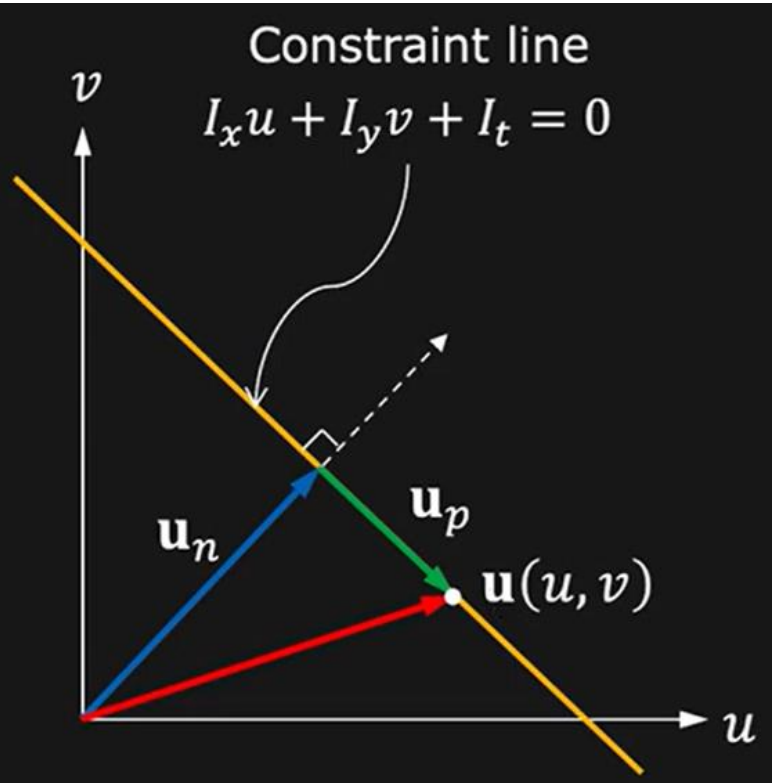
$$I_x u + I_y v + I_t = 0$$

Optical Flow can be split into two components.

$$\mathbf{u} = \mathbf{u}_n + \mathbf{u}_p$$

$\mathbf{u}_n$: Normal Flow

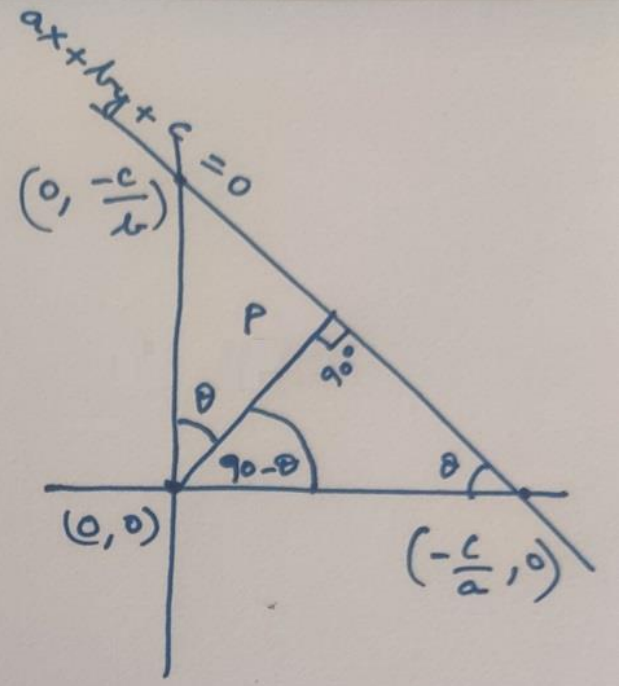
$\mathbf{u}_p$: Parallel Flow



$$ax + by + c = 0$$

$$x = -\frac{b}{a}y - \frac{c}{a}$$

$$y = -\frac{a}{b}x - \frac{c}{b}$$



$$P = -\frac{c}{a}\sin\theta$$

$$P = -\frac{c}{b}\cos\theta$$

$$\sin^2\theta = \frac{a^2p^2}{c^2} \qquad \cos^2\theta = \frac{b^2p^2}{c^2}$$

As $\sin^2\theta + \cos^2\theta = 1$, $\quad \frac{a^2p^2}{c^2} + \frac{b^2p^2}{c^2} = 1$

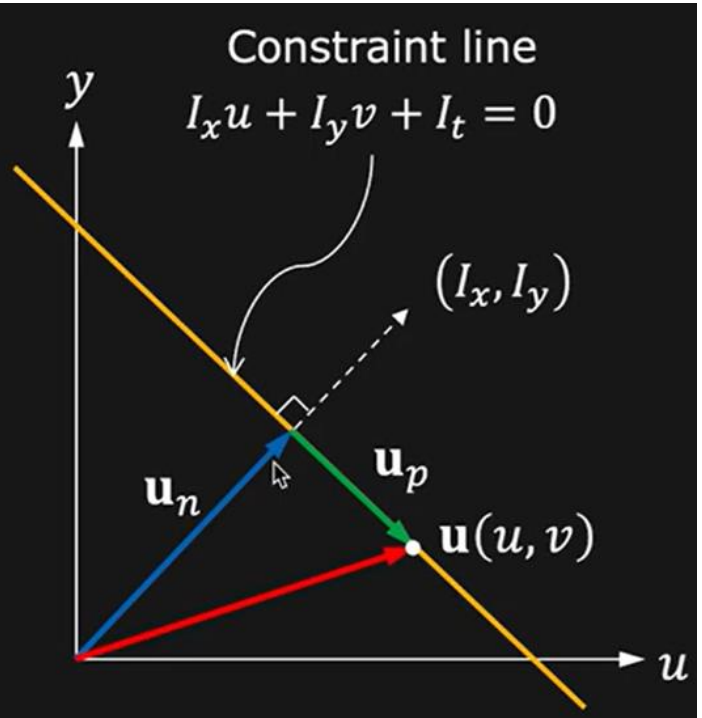
$$p^2 = \frac{c^2}{a^2+b^2} \Rightarrow |p| = \frac{|c|}{|\sqrt{a^2+b^2}|}$$

Normal Flow and Parallel Flow Computation

Direction of Normal Flow:

Unit vector perpendicular to the constraint line:

$$\hat{\mathbf{u}}_n = \frac{(I_x, I_y)}{\sqrt{I_x^2 + I_y^2}}$$



Magnitude of Normal Flow:

Distance of origin from the constraint line:

$$|\mathbf{u}_n| = \frac{|I_t|}{\sqrt{I_x^2 + I_y^2}}$$

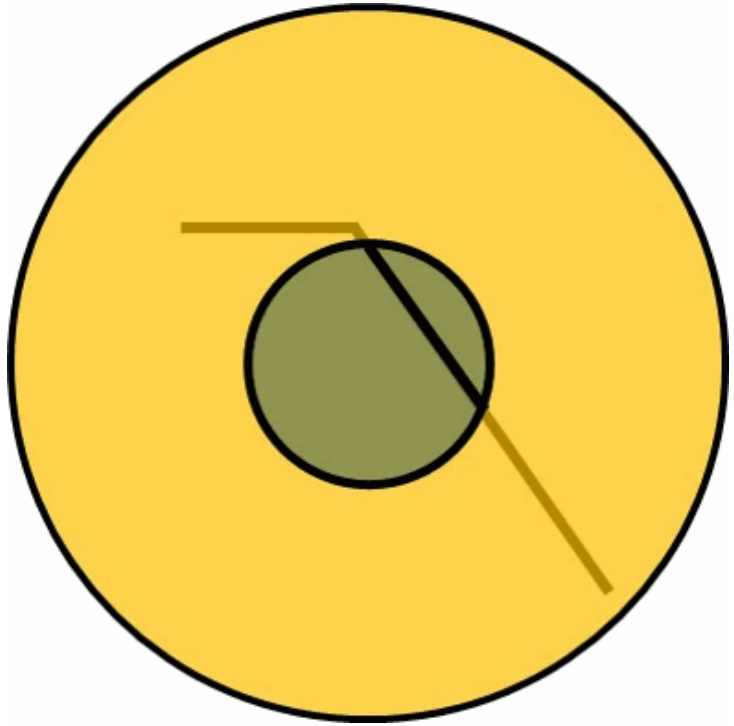
$$\mathbf{u}_n = \frac{|I_t|}{(I_x^2 + I_y^2)}(I_x, I_y)$$

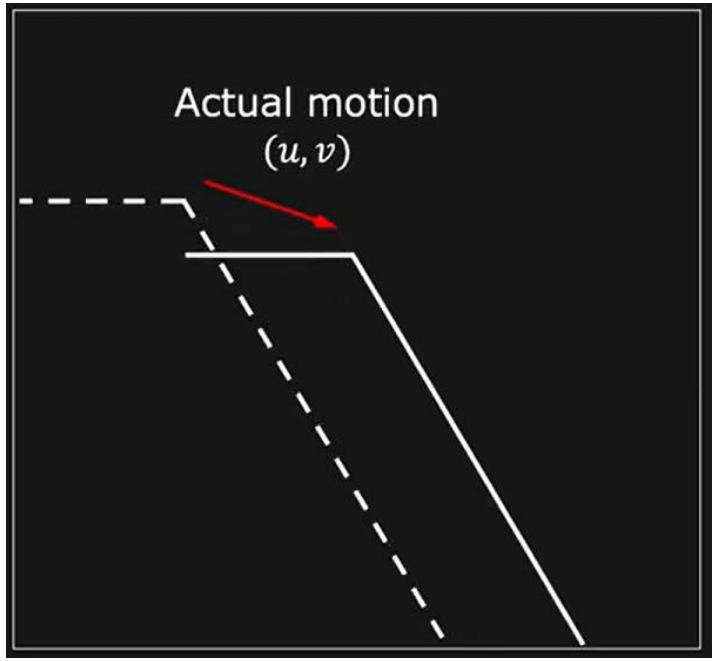
The point u (u, v) can be anywhere on the line

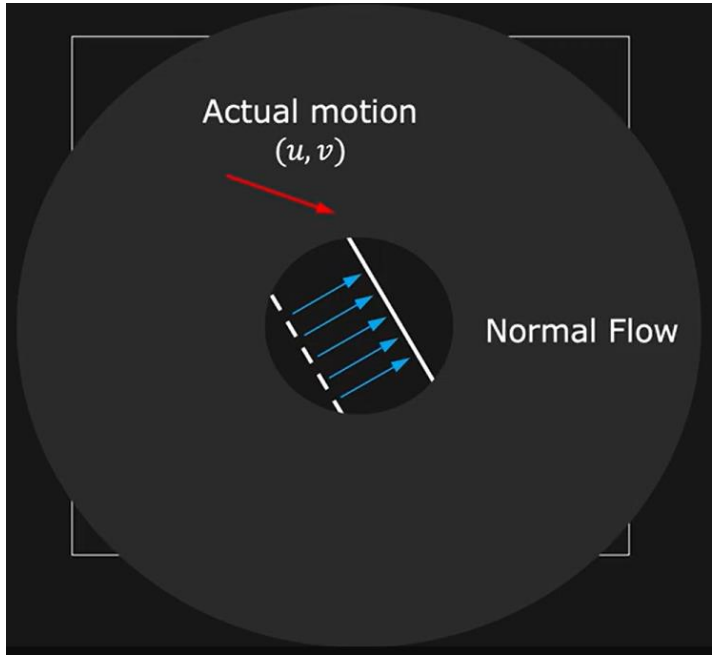
We cannot determine $\mathbf{u}_p$, the optical flow component parallel to the constraint line.

Aperture Problem

We can have potentially different optical flow in every local region of the image. That is why look at a small patch (we call it Aperture) to estimate the Optical flow.



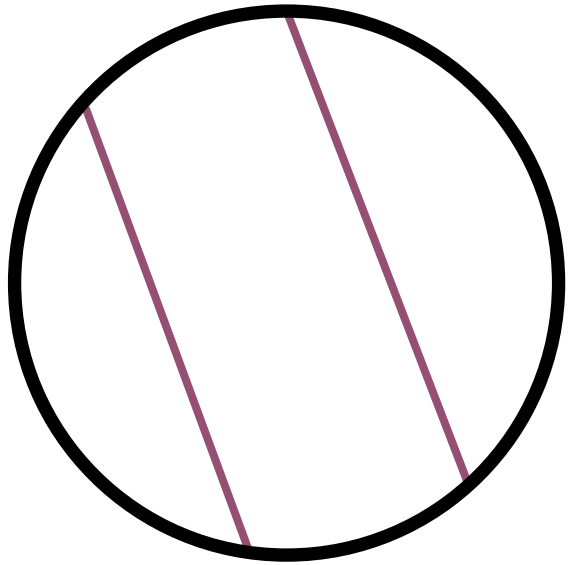




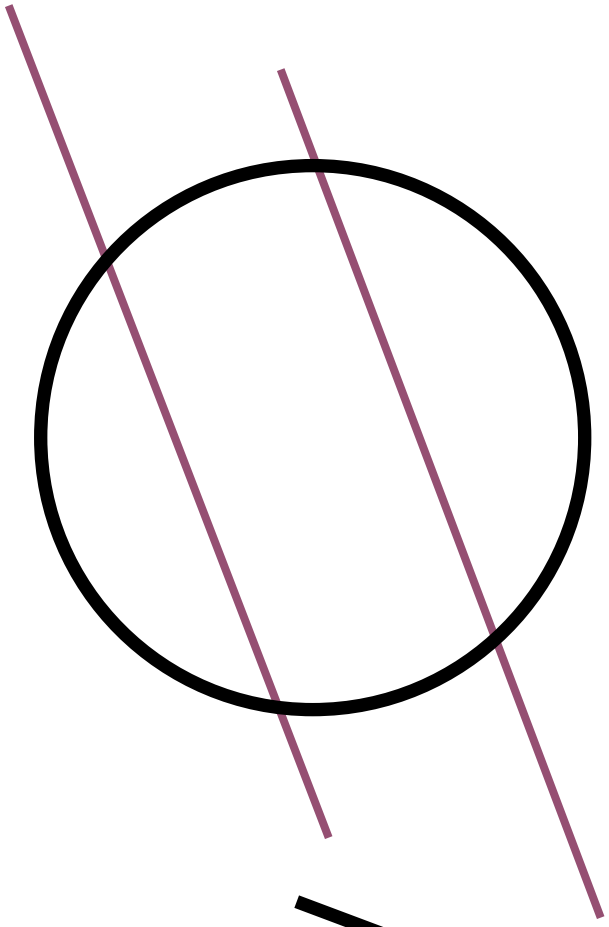
Through our human visual system, if we are observing the movement of the line through a yellow area (through a larger image area or entire image) we can estimate the movement accurately.

When we see through the aperture (a small patch: green patch in this example), we observe (in a wrong way) that the line is moving in the direction

Aperture Problem



Perceived motion



Actual motion

$$\nabla I \cdot (u', v') = 0$$

Optical Flow is Under Constrained

$I_x = \partial I / \partial x$

$I_y = \partial I / \partial y$

$I_t = \partial I / \partial t$

Constraint Equation: $I_x u + I_y v + I_t = 0$

2 unknowns, 1 equation.

Displacement: $(\delta x, \delta y)$

Optical Flow: $(u, v) = \left(\frac{\delta x}{\delta t}, \frac{\delta y}{\delta t}\right)$

Solving the Aperture Problem

- How to get more equations for a pixel?
- **Spatial coherence constraint:** pretend the pixel's neighbors have the same (u,v)

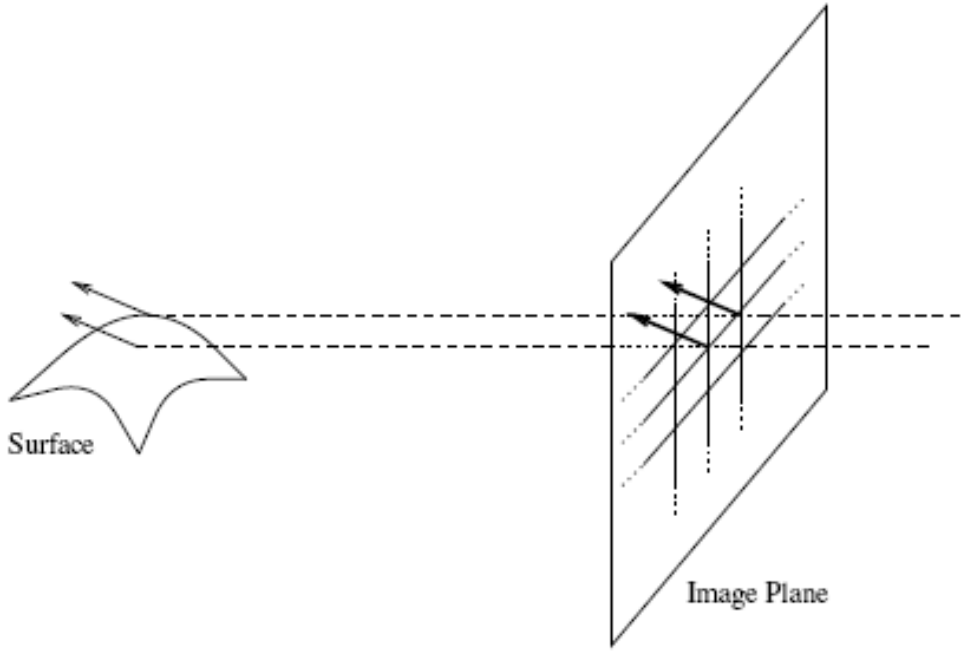


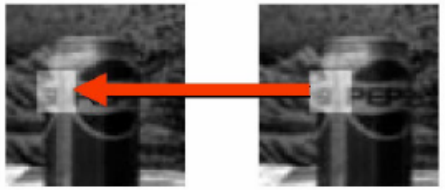
Figure 1.7: Spatial coherence assumption. Neighboring points in the image are assumed to belong to the same surface in the scene.

Figure by Michael Black

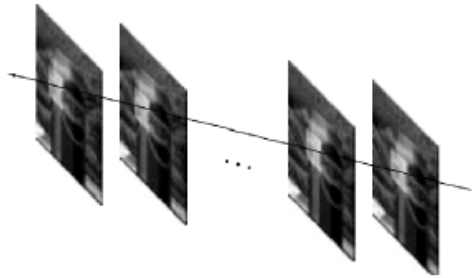
Point Tracking with Differential Methods: Assumptions Summary

Assumptions:

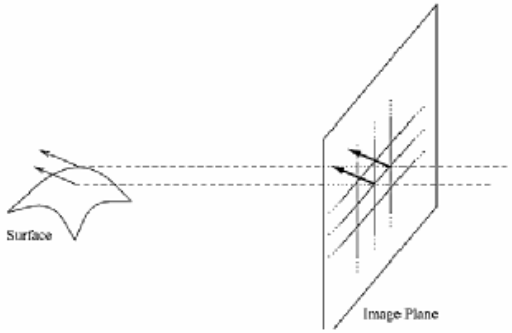
- **Brightness constancy**
  - The intensity of the pixels around the point to track does not change much between the two frames



- **Temporal consistency**
  - The motion displacement is small (1-2 pixels); however, this can be addressed using multi-scale implementations (see later)



- **Spatial coherency**
  - Neighboring pixels undergo similar motion (i.e., they all lay on the same 3D surface, i.e., no depth discontinuity)



Lucas-Kanade Solution

Assumption: For each pixel, assume Motion Field, and hence Optical Flow (u, v), is constant within a small neighborhood W. This means All nearby points move together

That is for all points $(k, l) \in W$:

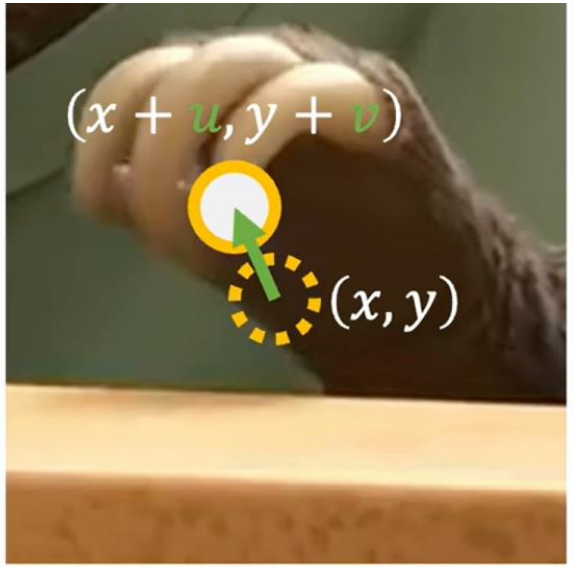
$$I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$$

ee K. Nayar

[Lucas 1981]

Lucas- Kanade Solution: Key constraints

Frame t



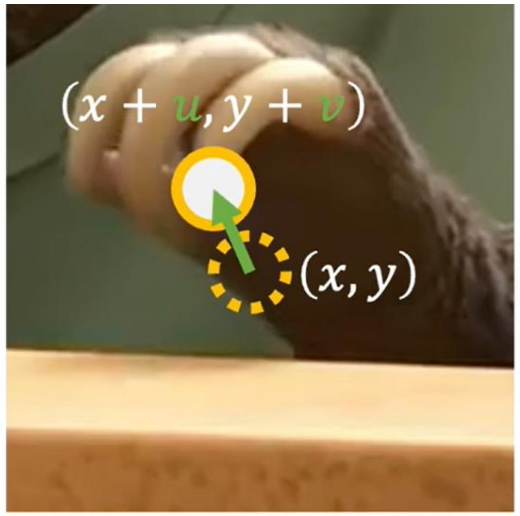
Frame $t+1$

- **Brightness constancy**
- **Small motion**
- **Spatial coherence**

$$uI_x^{(1)} + vI_y^{(1)} = -I_t^{(1)}$$
$$uI_x^{(2)} + vI_y^{(2)} = -I_t^{(2)}$$
$$\vdots$$
$$uI_x^{(N)} + vI_y^{(N)} = -I_t^{(N)}$$

Lucas-Kanade Solution: Constraint Matrices for all points in the window

Frame t



Frame $t+1$

- Brightness constancy
- Small motion
- Spatial coherence

$$\begin{bmatrix} I_x^{(1)} & I_y^{(1)} \\ I_x^{(2)} & I_y^{(2)} \\ \vdots & \vdots \\ I_x^{(N)} & I_y^{(N)} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t^{(1)} \\ I_t^{(2)} \\ \vdots \\ I_t^{(N)} \end{bmatrix}$$

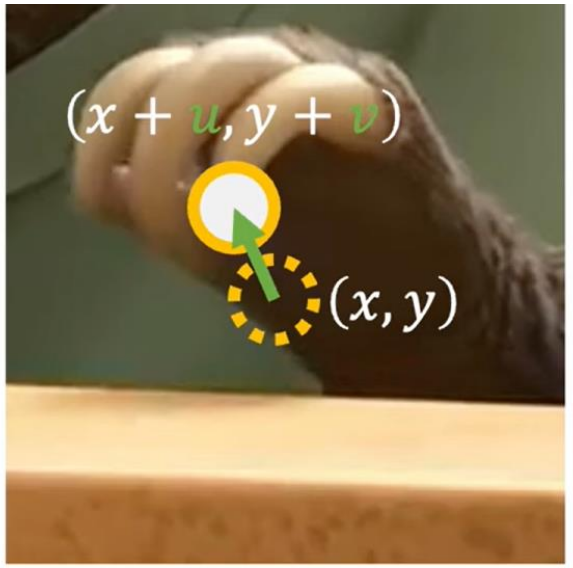
$$A^T A\, d = A^T b$$

All nearby points (points in this window) move together

This is a over deterministic system.

Lucas-Kanade Solution: Calculate Optical Flow using Least Squares Approximation

Frame t



Frame $t+1$

- **Brightness constancy**
- **Small motion**
- **Spatial coherence**

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x^{(i)} I_x^{(i)} & \sum I_x^{(i)} I_y^{(i)} \\ \sum I_x^{(i)} I_y^{(i)} & \sum I_y^{(i)} I_y^{(i)} \end{bmatrix}^{-1} - \begin{bmatrix} \sum I_x^{(i)} I_t^{(i)} \\ \sum I_y^{(i)} I_t^{(i)} \end{bmatrix}$$

$$d \qquad (A^T A)^{-1} \qquad A^T b$$

Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$

Note: This method works if All these equations are not dependent on each other.

This is a over deterministic system.

If you have a patch that has interesting enough texture, I_x and I_y would be different for different pixels within the patch

Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$

A (Known) $n^2 \times 2$ | $\mathbf{u}$ (Unknown) 2×1 | B (Known) $n^2 \times 1$

n^2 Equations, 2 Unknowns: Find Least Squares Solution

Least Squares Solution

Solve linear system: $A\mathbf{u} = B$

$$A^T A\mathbf{u} = A^T B$$ (Least-Squares using Pseudo-Inverse)

In matrix form:

$$\begin{bmatrix} \sum_w I_x I_x & \sum_w I_x I_y \\ \sum_w I_x I_y & \sum_w I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\sum_w I_x I_t \\ -\sum_w I_y I_t \end{bmatrix}$$

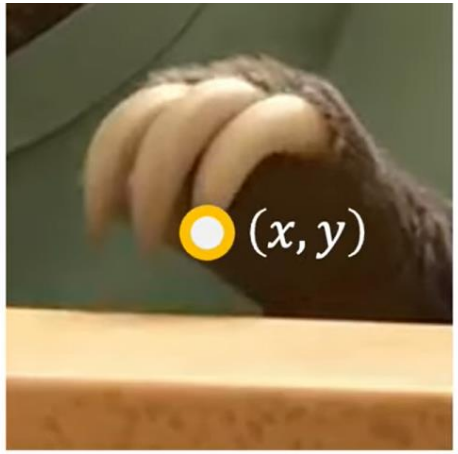
Indices (k, l) not written for simplicity

$A^T A$	$\mathbf{u}$	$A^T B$
(Known)	(Unknown)	(Known)
2×2	2×1	2×1

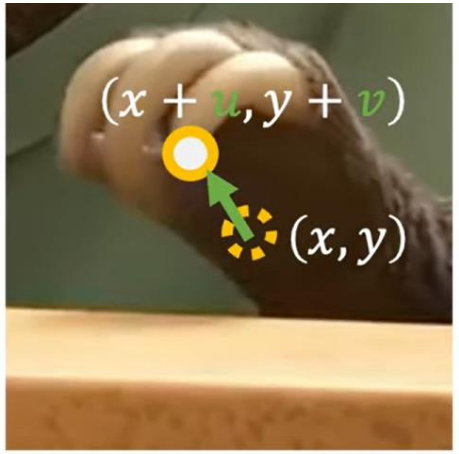
$$\mathbf{u} = (A^T A)^{-1} A^T B$$

Fast and Easy to Solve

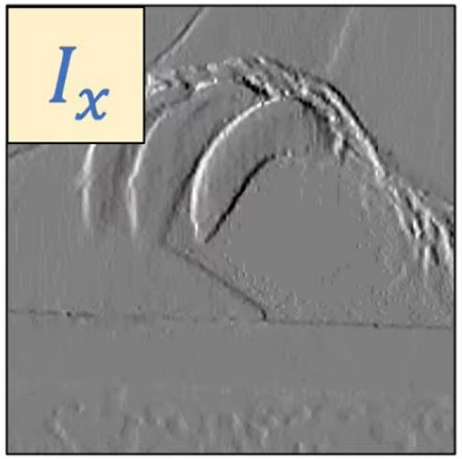
Lucas-Kanade: Compute First Derivatives

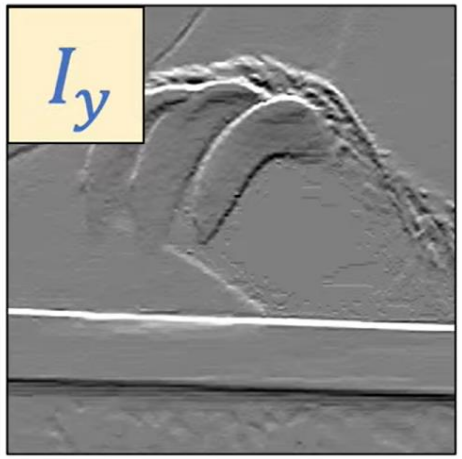


Frame t



Frame $t + 1$

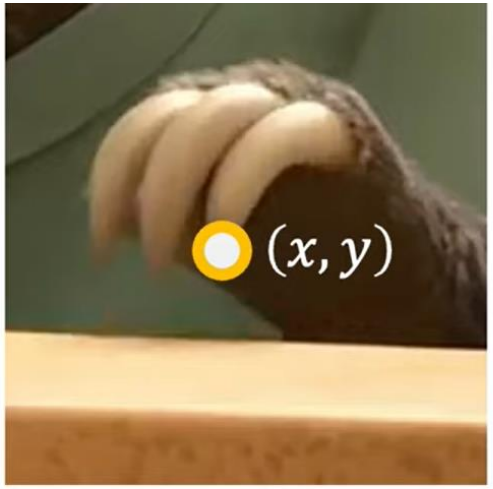




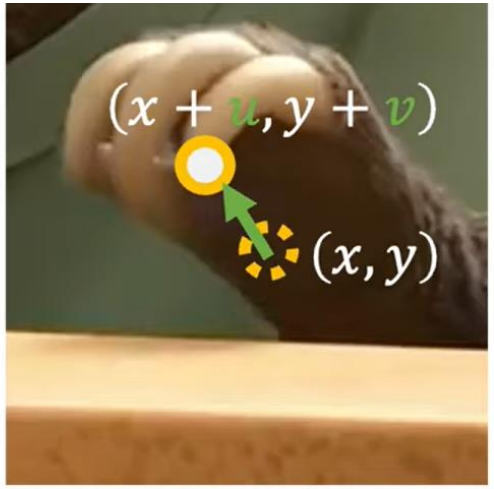
From a frame at time t,

- Compute X and Y derivative using Gradient Filters
- Compute temporal derivative using frame difference

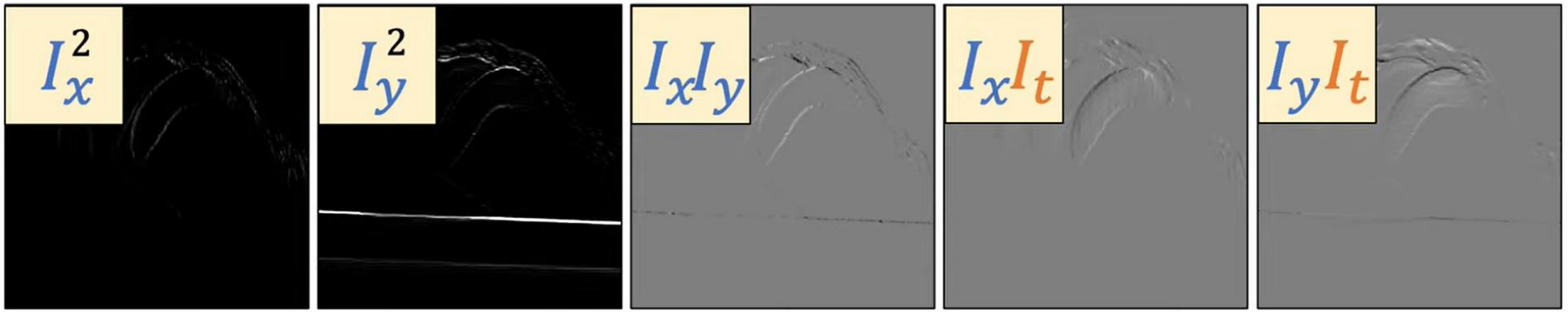
Lucas-Kanade: Compute Matrix terms



Frame t

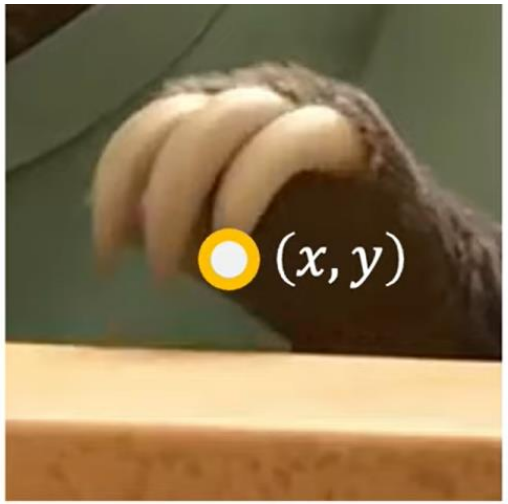


Frame $t + 1$



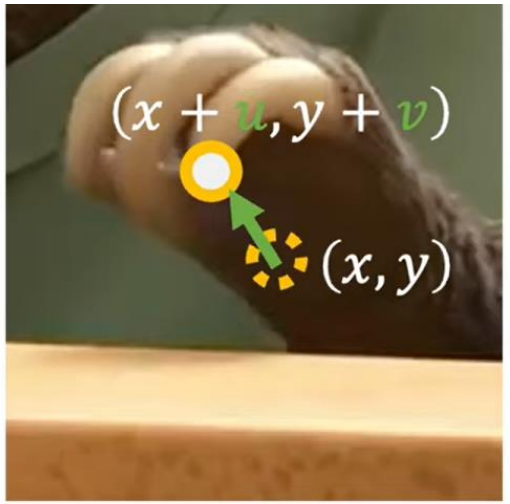
- Compute these terms using Element-wise multiplications

Lucas-Kanade: Compute Matrix terms (weighted Local average)

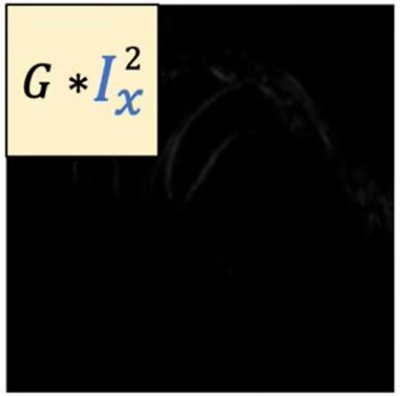


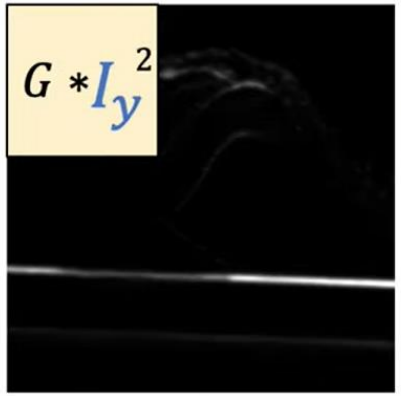
Frame t

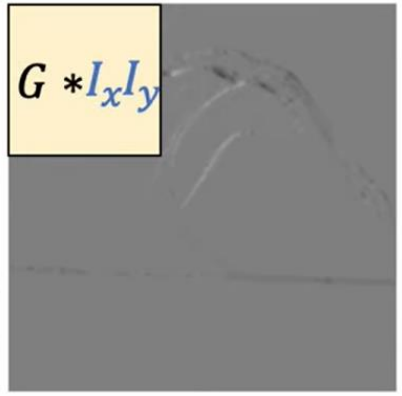
G

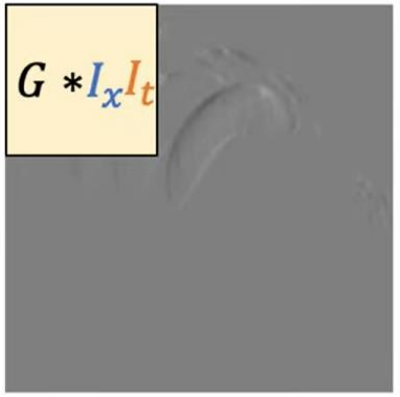


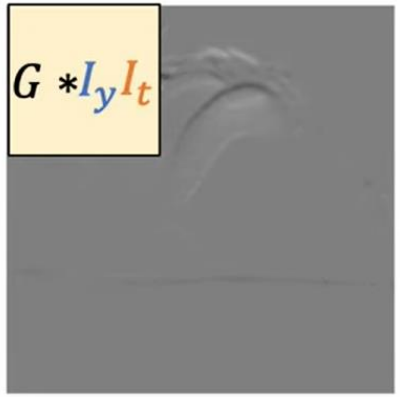
Frame $t + 1$











Compute weighted Local average using Gaussian Filter

When does Optical Flow Estimation work?

$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$

The key assumption: Matrix A^TA is invertible!

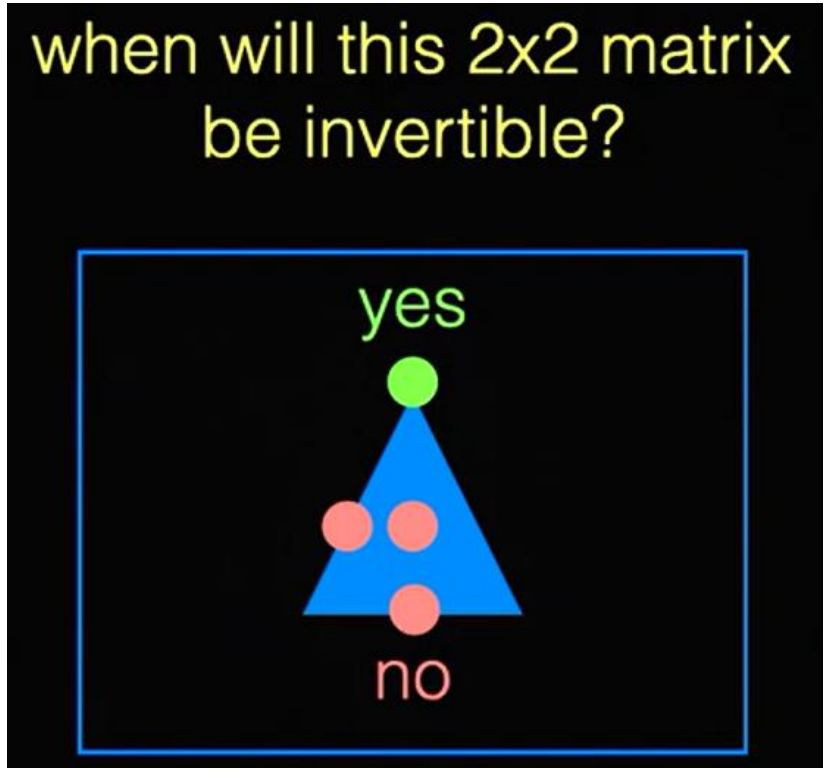
This assumption fails if all gradients in a region are aligned (i.e. a region of constant shading, i.e. of the form I(x,y)=ax+by+c) or if all gradients are null (constant intensity, i.e. I(x,y) = c).

Well conditioned system: A system is one where a significant change in the output for a given change in input. You can take the output and estimate the input.

If you change in the input and nothing changes in the output, then the system is well conditioned.

The eigen values λ_1 and λ_2 should be significant and one should not be significantly larger than the other.

When will this 2X2 matrix be invertible?



$$\begin{bmatrix} \sum_w I_x I_x & \sum_w I_x I_y \\ \sum_w I_x I_y & \sum_w I_y I_y \end{bmatrix}$$

Harris Corner matrix

Uniform Patch (centre red dot): [$I_x = 0$] and [$I_y = 0$]

The Matrix would have all zeros.

Rank deficient matrix. Not invertible. Cannot estimate motion.

Line (bottom red dot): [$I_x = 0$]

3 Cells of matrix would have zero values.

Rank deficient matrix. Not invertible. Cannot estimate motion.

Line / Any oriented edge (Left red dot):

Rate of change in one direction.

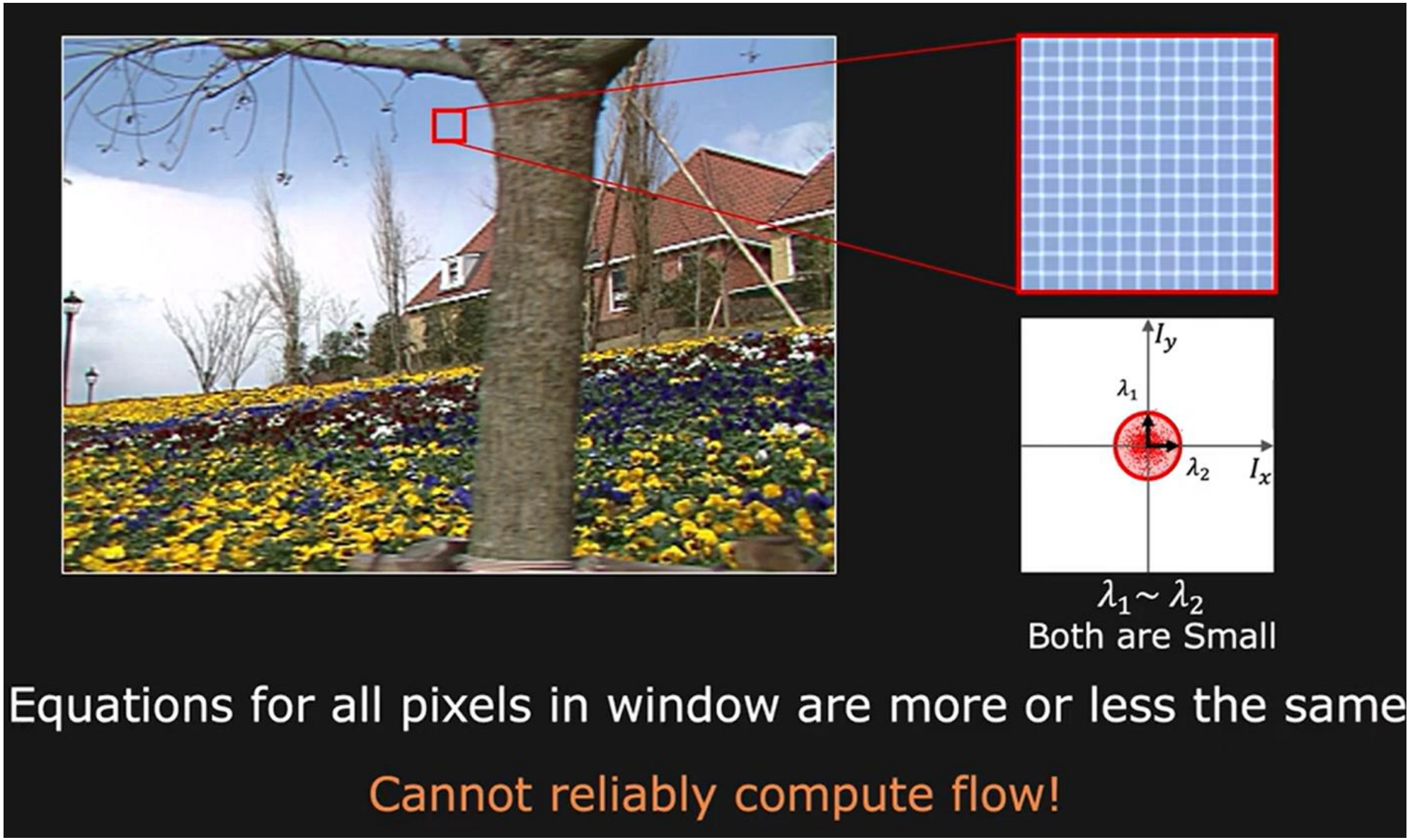
Cannot estimate motion. Aperture problem

Corner [Green red dot]: [$I_x \neq 0$ and $I_y \neq 0$]

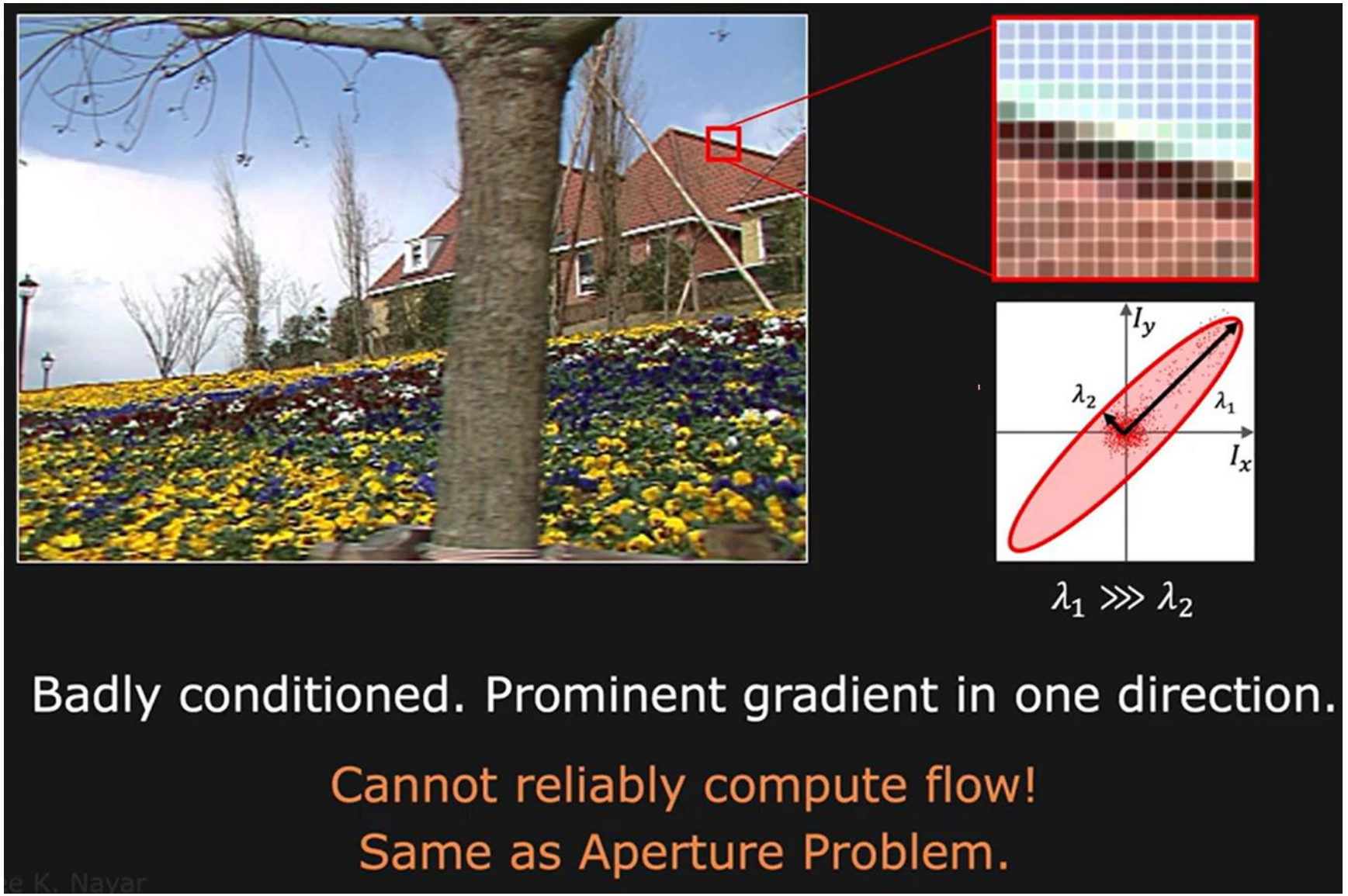
Distinctive pattern in both x-direction and y-direction

Invertible. Can estimate motion

Smooth Regions (bad regions to compute Optical Flow)

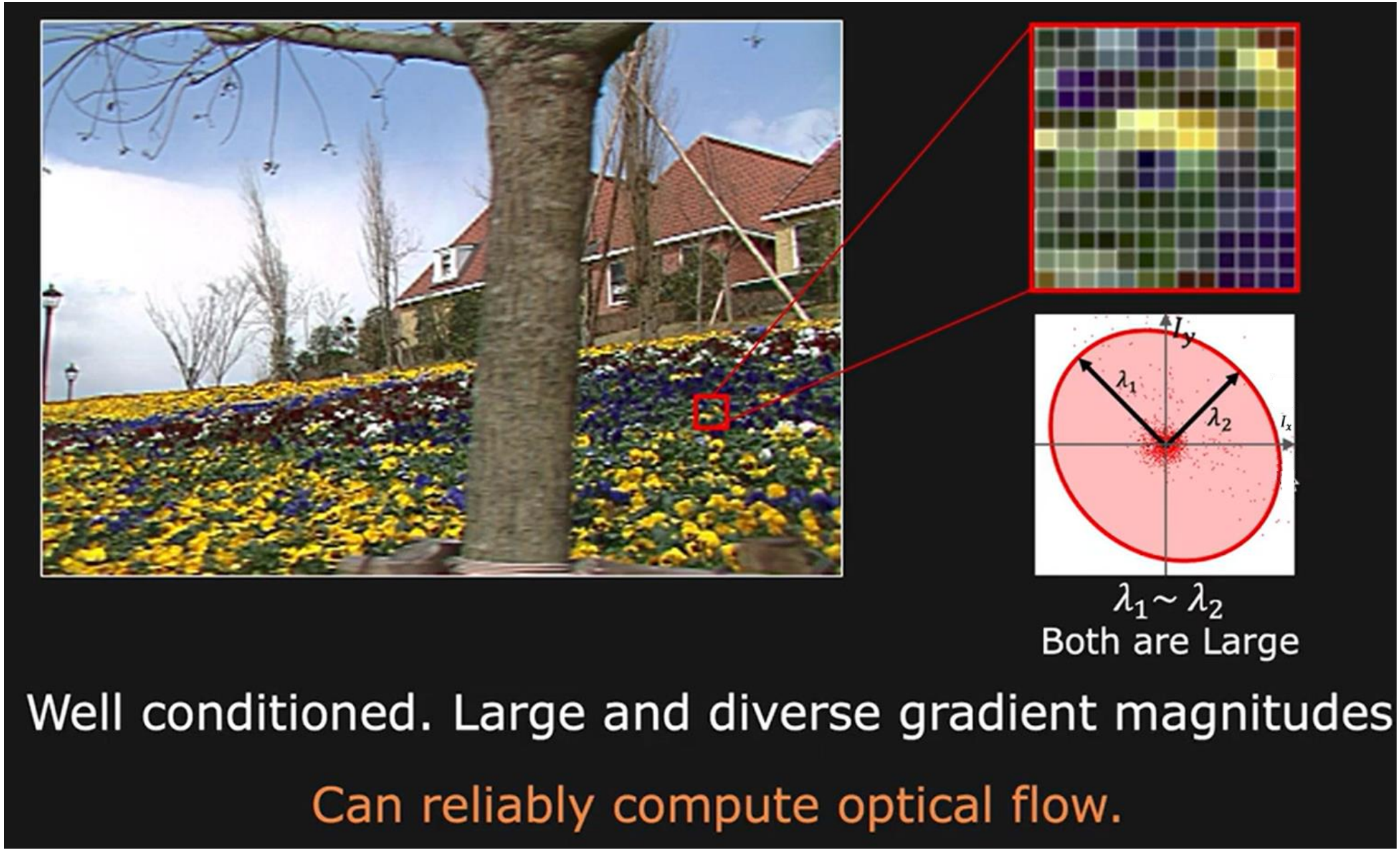


Edges (bad regions to compute Optical Flow)



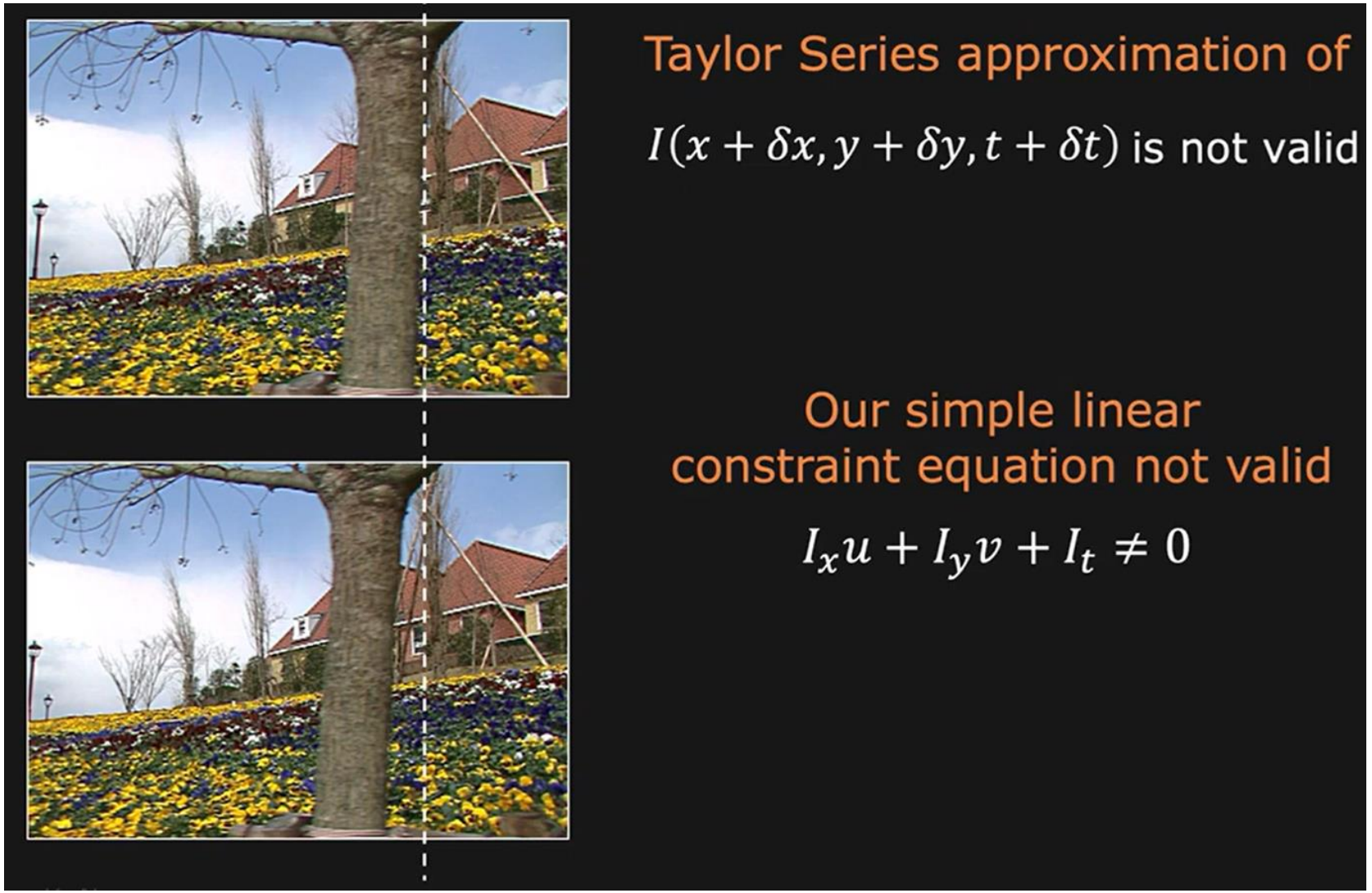
This corresponds to the aperture problem we have seen before

Textured Regions (Good)



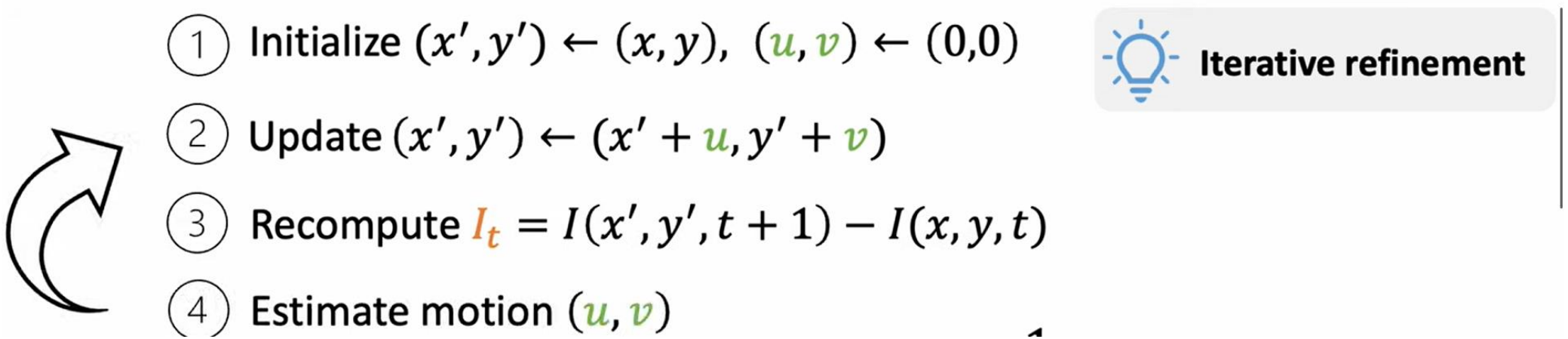
What if we have Large Motion?

To track objects with large motion or objects with higher speed of motion



Idea 1: Iterative Refinement

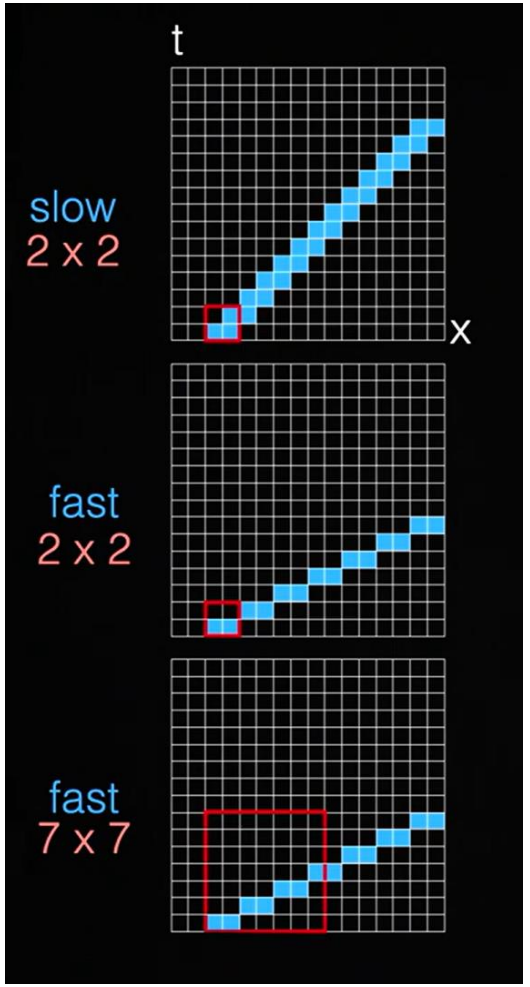
Iterative Refinement



This idea works for small motion but still not sufficient for large motion

Motion Estimation: Multi-scale implementation – few practical ideas

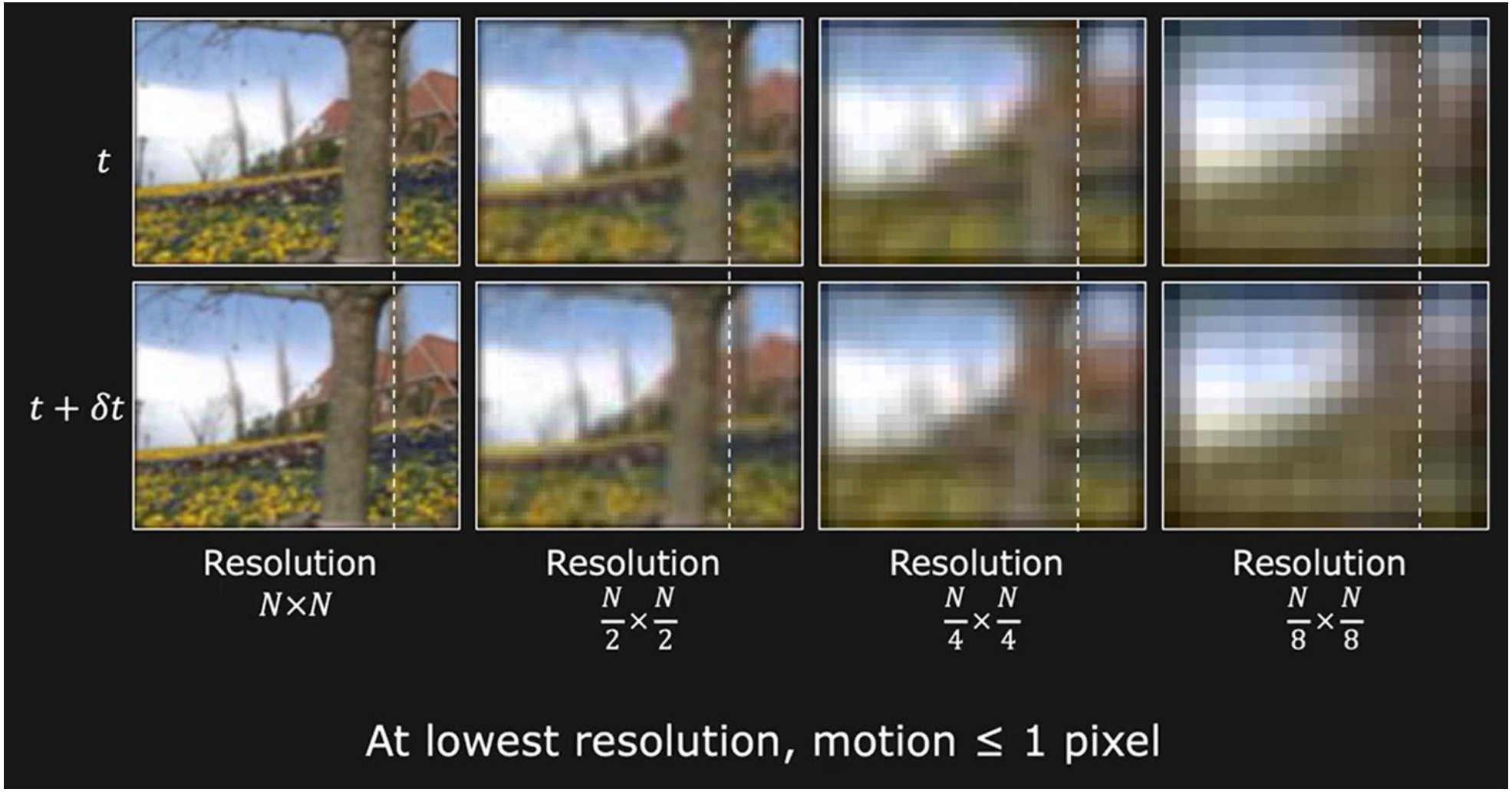
- Derivative Filter
  - Small filter 2 X2: we will miss pixels for x and y derivative calculation to support faster motion
  - Use bigger filters 3X3, 4X4, 5X5, 7X7 etc to account for faster motion
- Use of multiple frames
  - Two frames: apply Average pre-filter in x-direction, time derivative in x-direction for y-derivative and vice versa.
  - Multiple frames: Apply average filter 3X3, 4X4, 5X5, 7X7 etc to account for faster motion



Farid & Simoncelli. Differentiation of Discrete Multi-Dimensional Signals. *IEEE Transactions on Image Processing*, 13(4):496-508, 2004

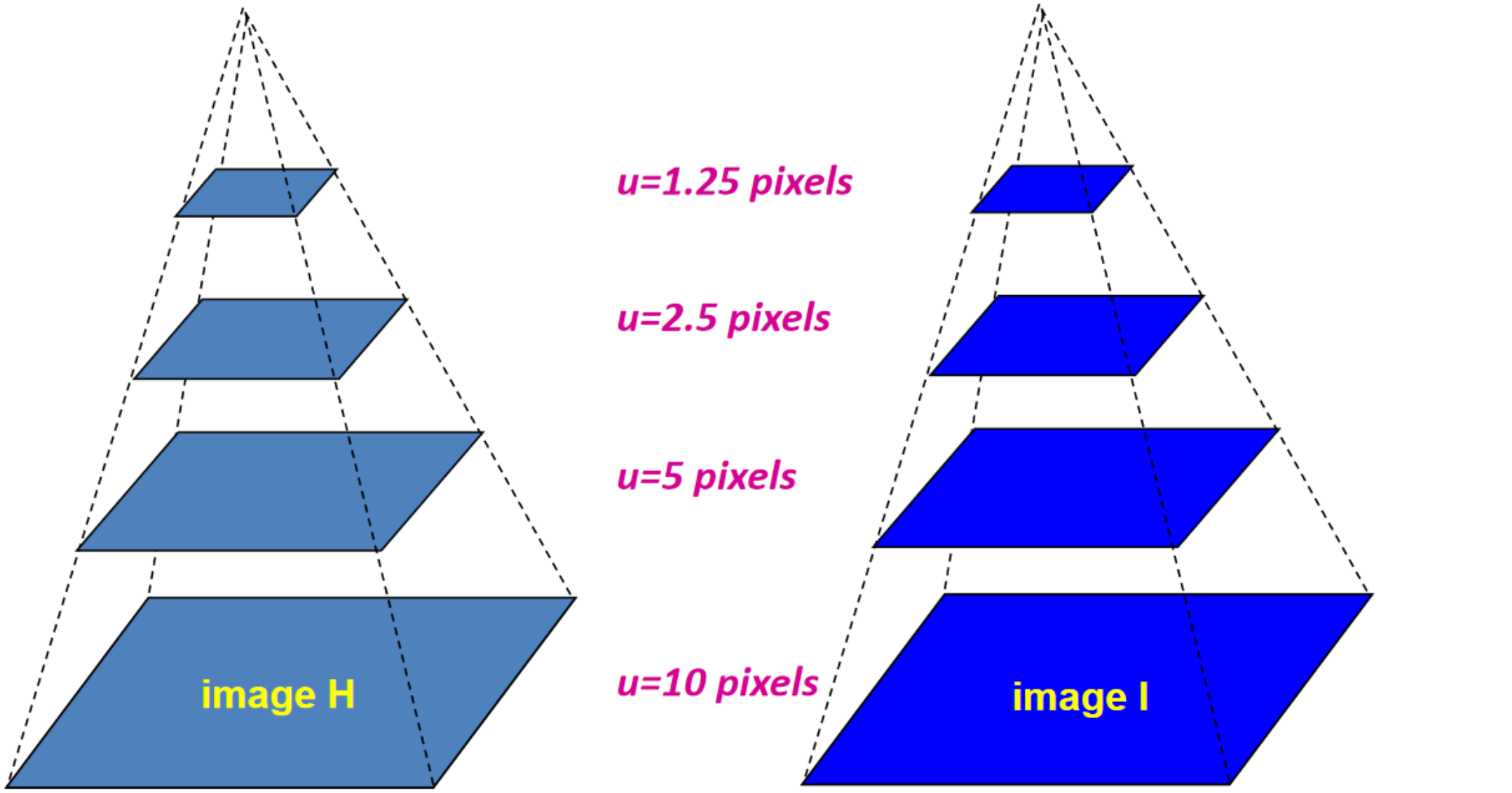
Idea 2: Coarse-to-Fine Optical Flow Estimation (Multi-scale differential motion estimation)

Large Motion: Coarse-to-Fine Optical Flow Estimation



Resolution Reduction:
For all pixel grids:
Take a 2X2 pixel grid and replace it with the average intensity of the four pixels

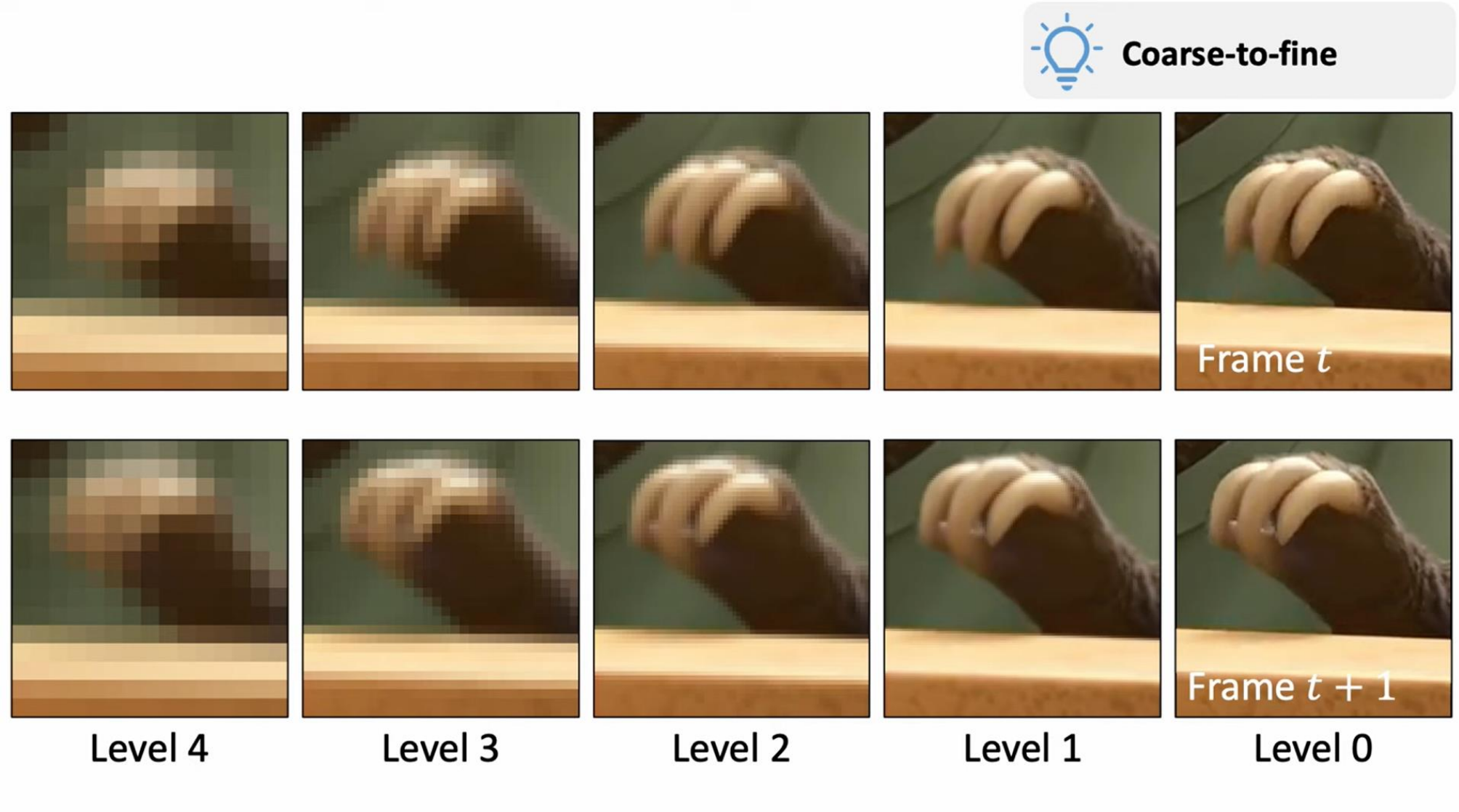
Coarse-to-Fine Optical Flow Estimation



Gaussian pyramid of image H at **t**

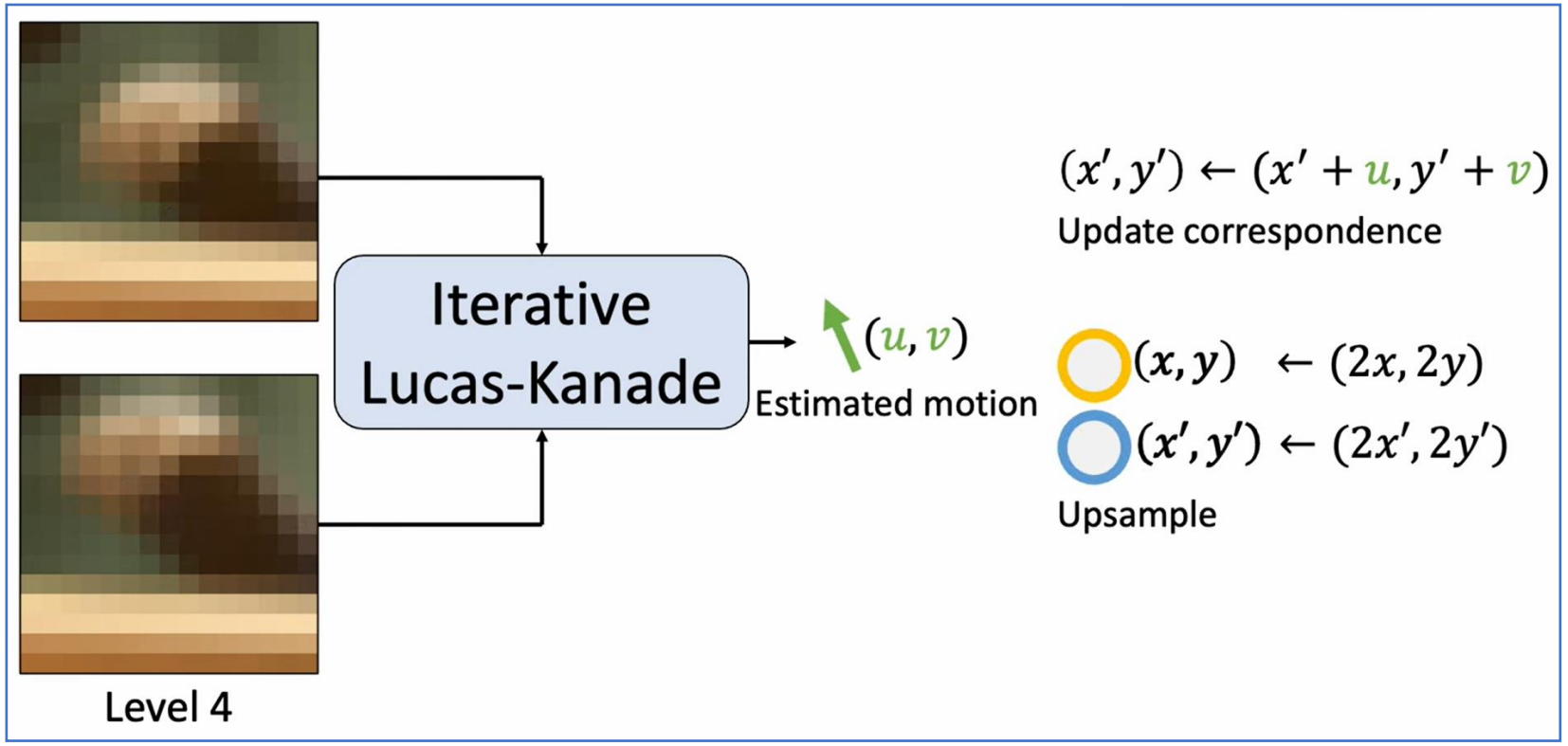
Gaussian pyramid of image I at **t + δt**

Build a Gaussian Pyramid at multiple levels



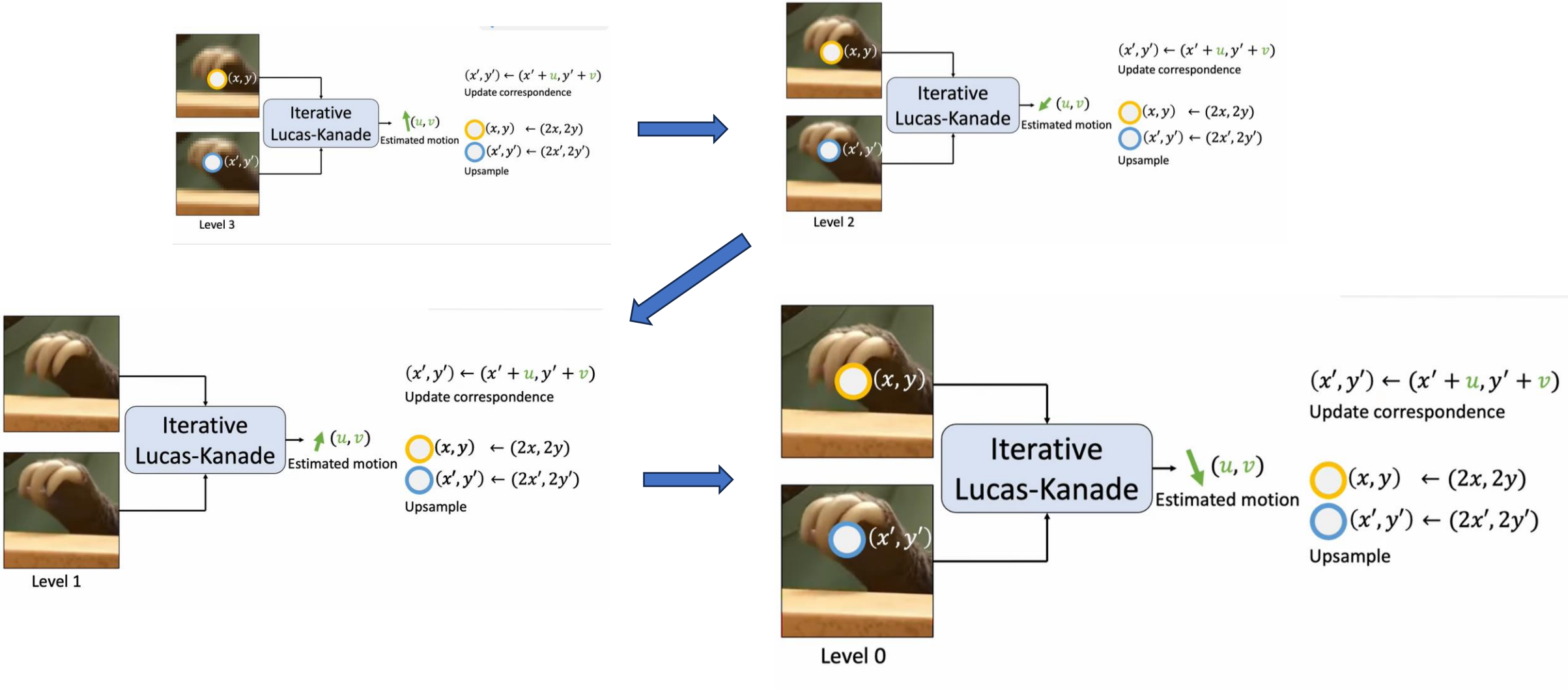
Build a Gaussian Pyramid of multiple levels of each image

Coarse-to-Fine Optical Flow Estimation Algorithm



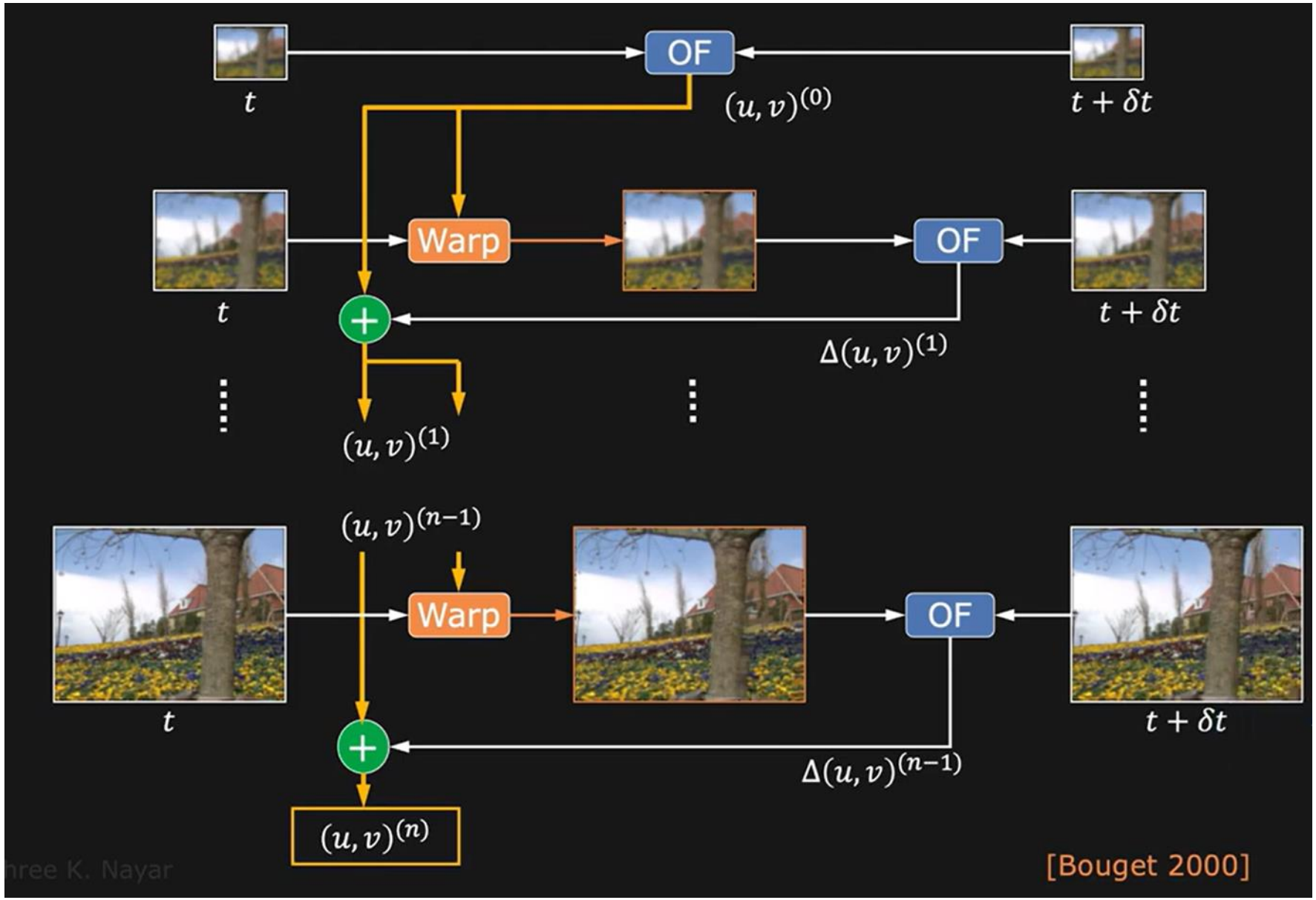
- Large Motion in Original resolution is a tiny motion in low resolution images
- Start with the lower image resolution and estimate the motion using Iterative Lucas-Kanade algorithm
- Update correspondence using the estimated motion and upscale them to the next level
- This provides an accurate initialization for the next level

Coarse-to-Fine Optical Flow Estimation

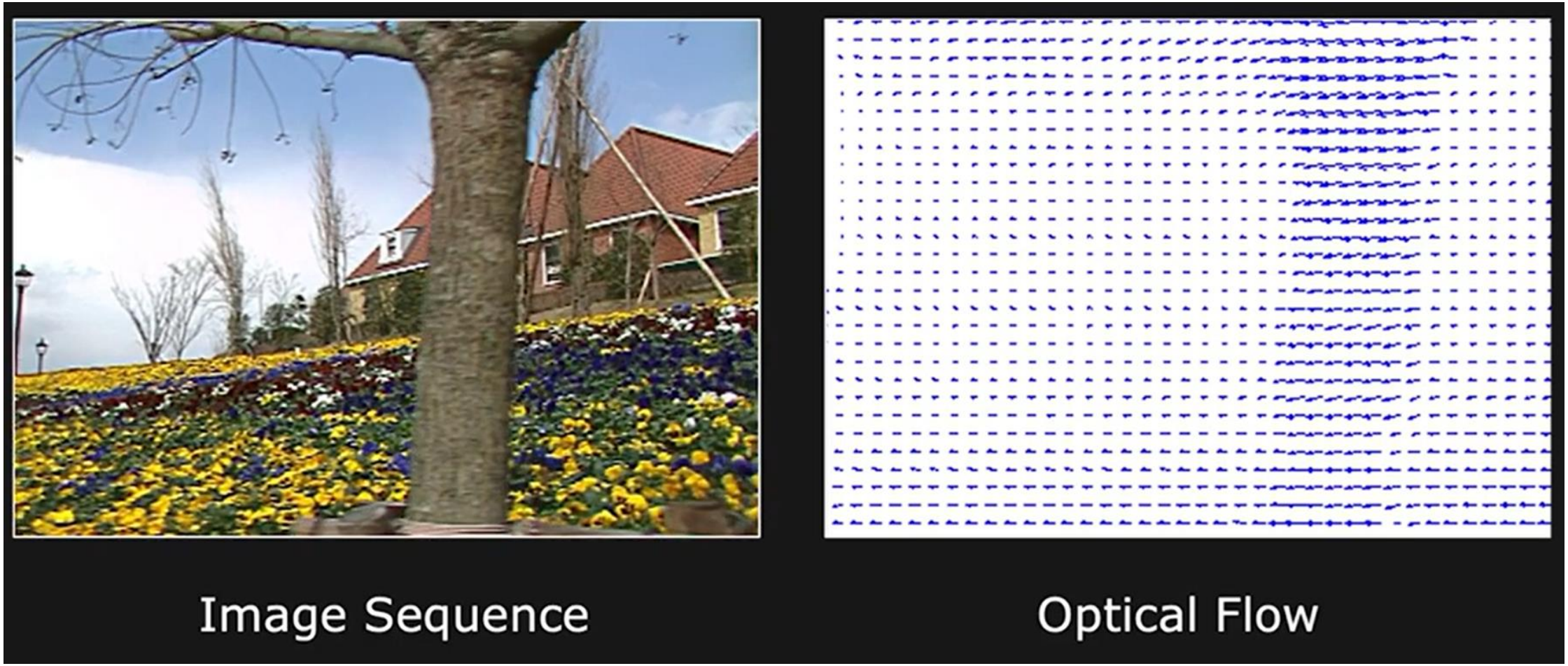


Repeat this process until we reach the original resolution

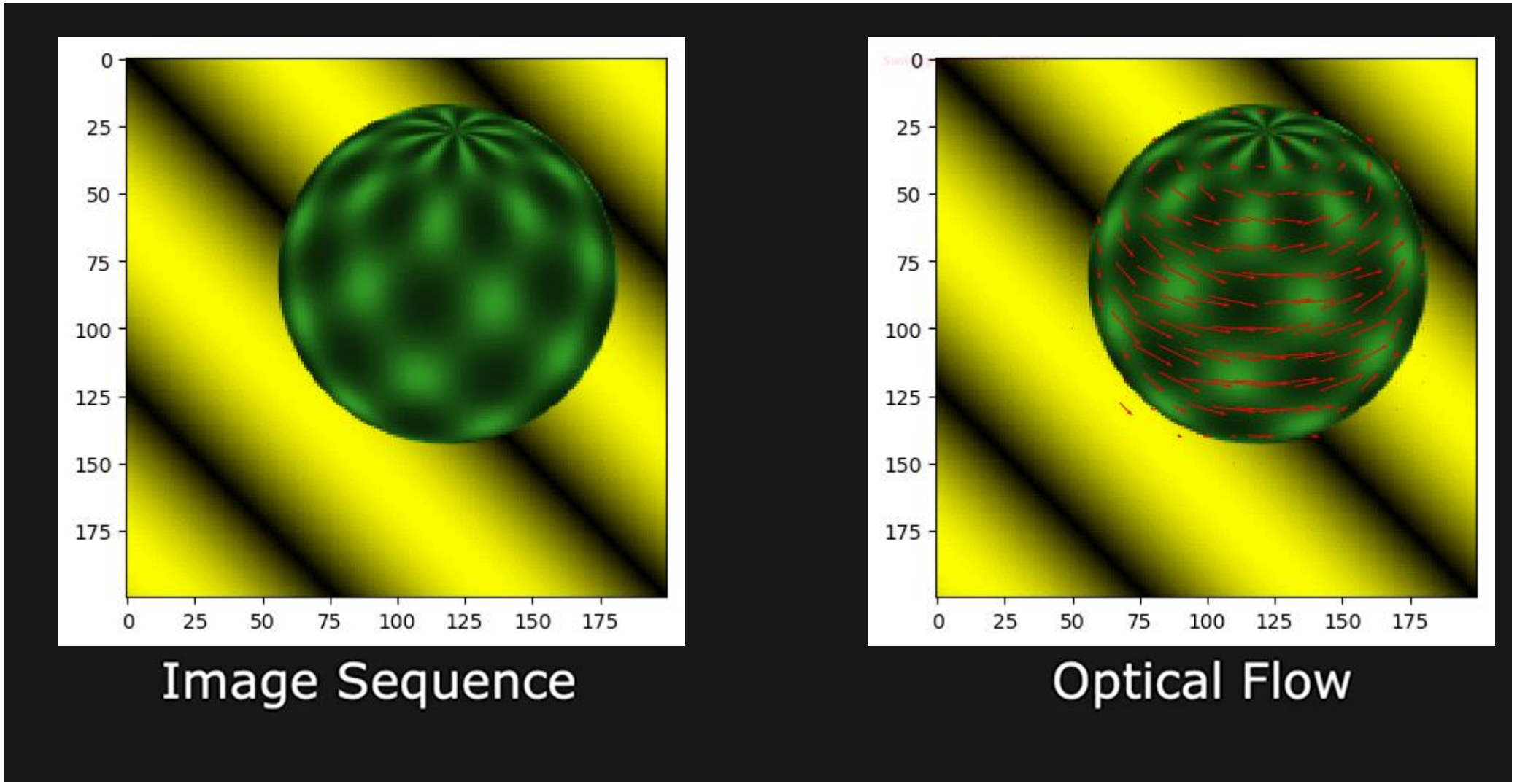
Coarse-to-Fine Optical Flow Estimation Algorithm



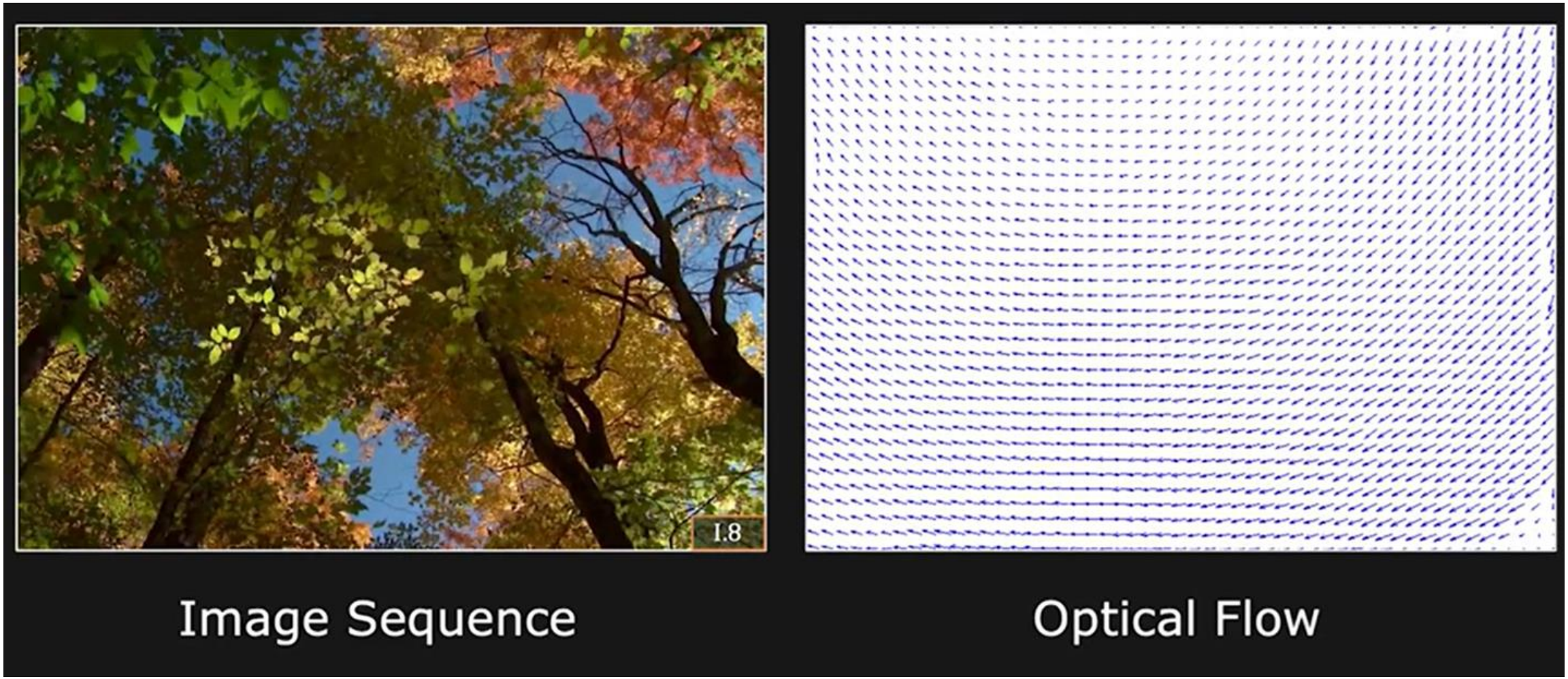
Results: Tree Sequence



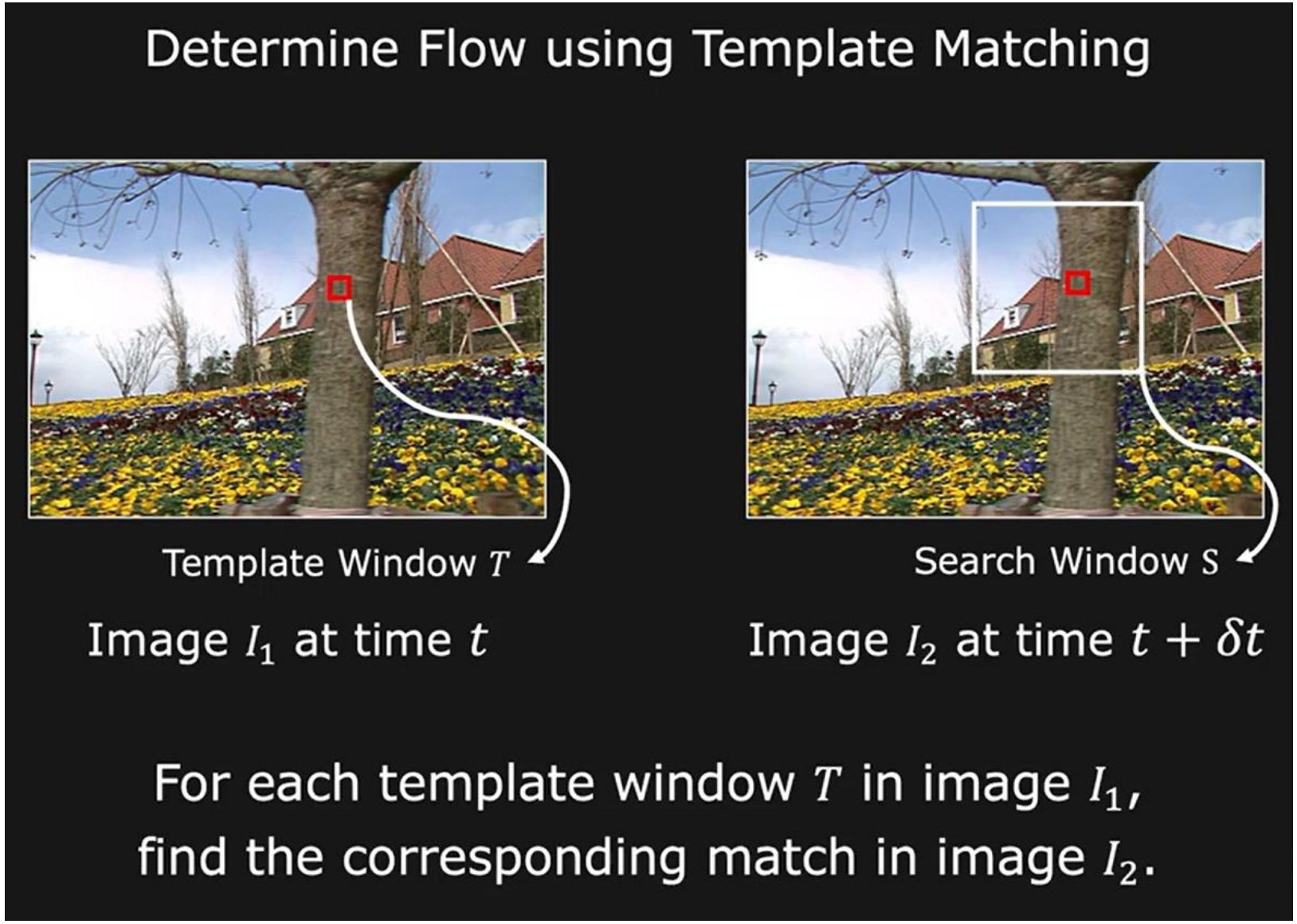
Results: Rotating Ball



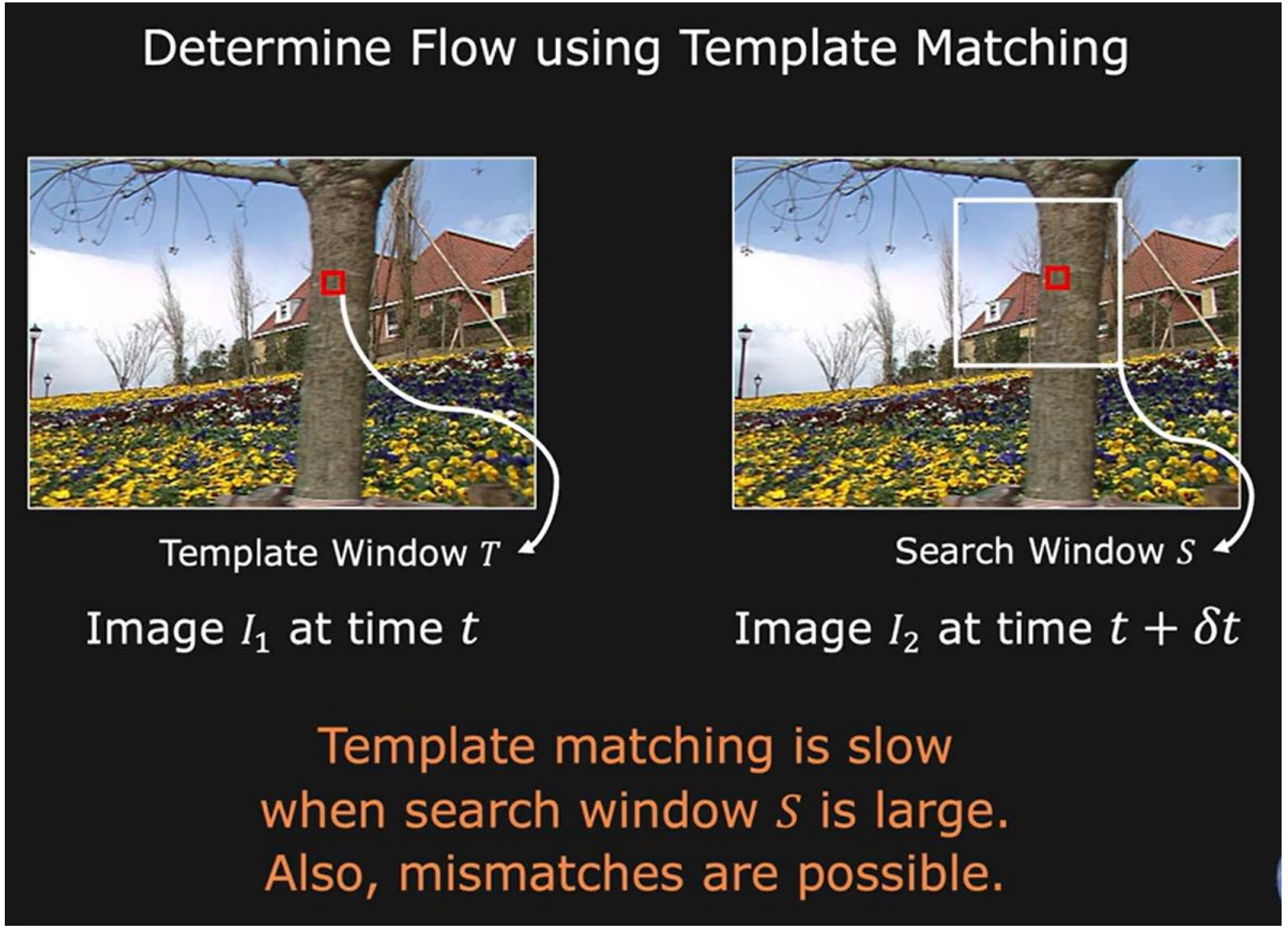
Results: Rotating camera



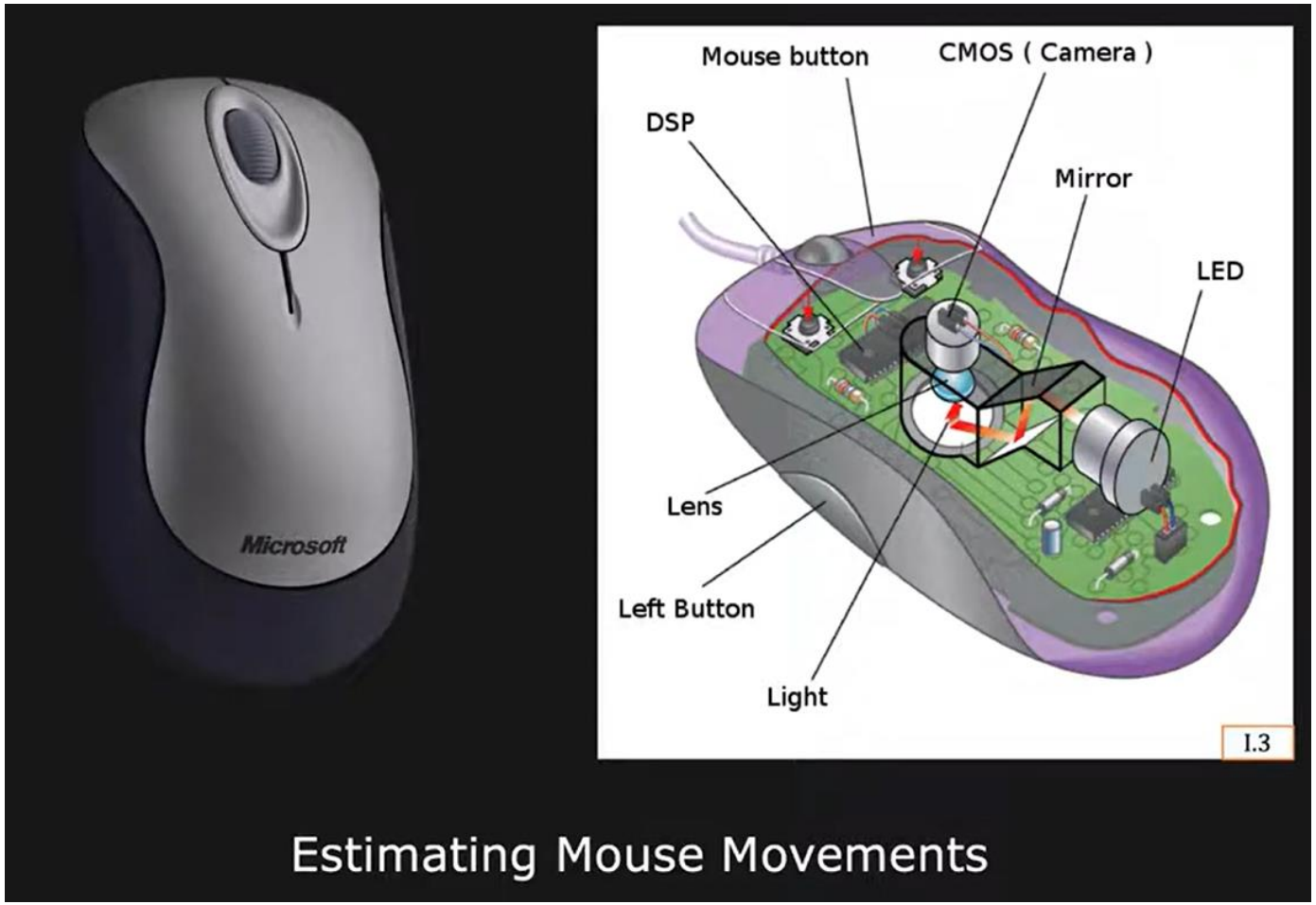
Alternative Approach: Template Matching



Large Motion: Template Matching



Optical Mouse



Optical mouse has a computer vision system inside it.

It has a light source that illuminates the surface it is sitting on.

It has a very low resolution camera (64 X 64 pixels).

Because of the low resolution, pixels are made really large and very sensitive to light. This enables the mouse system to produce video at a high framerate (1000s of frames/sec).

The images are captured and compared with each other (that is optical flow is computed using derivates or correlation) and that gives the motion of the mouse from one frame to next w.r.t the surface it is sitting on.

This information is used to control the cursor on the computer screen.

Traffic Monitoring

Static Stationary camera looking down at a highway

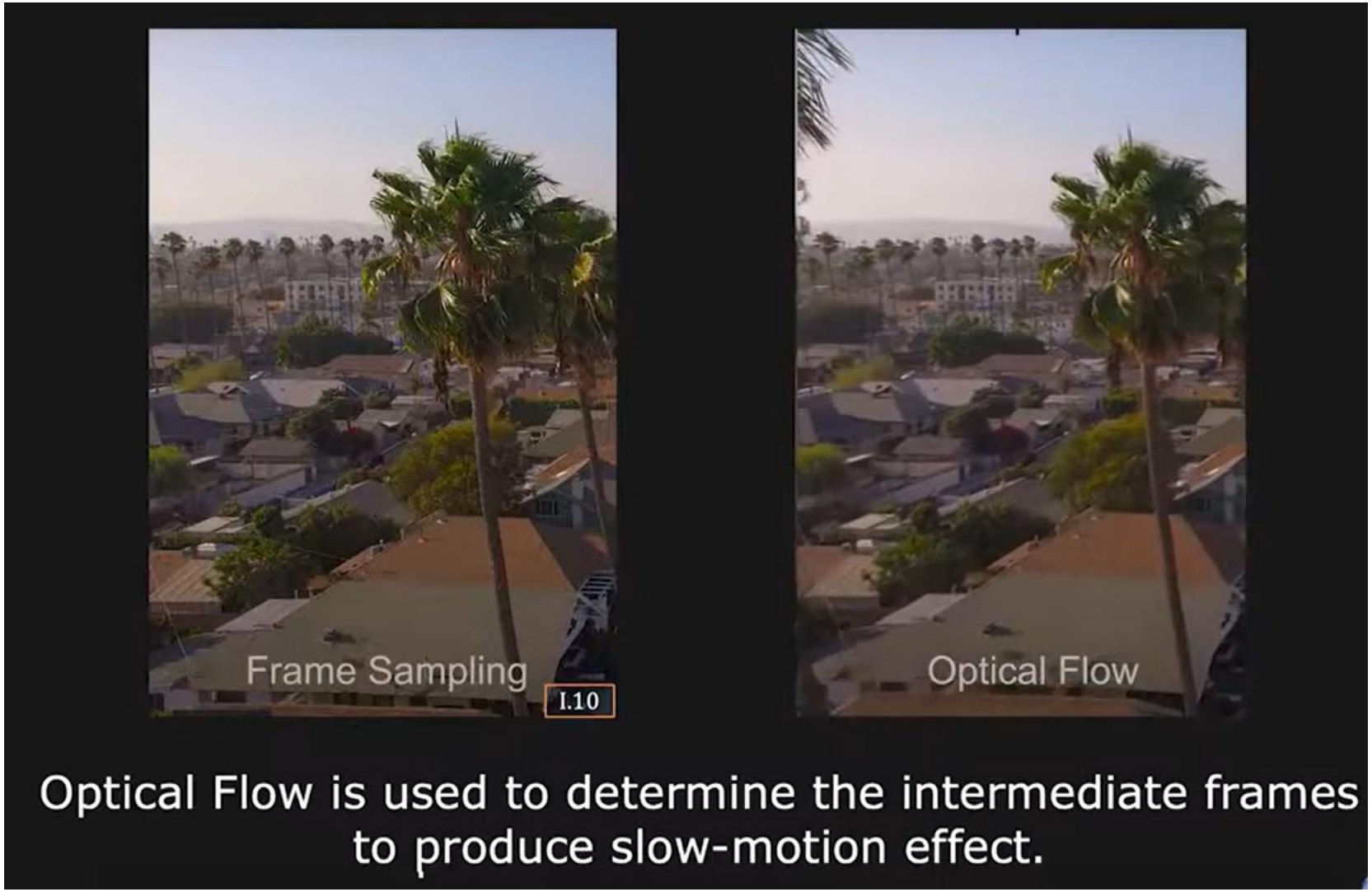
3D structure of the scene: Know Orientation and distance of the highway from the camera. Know the ground level

Calculate Optical Flow of the vehicle(s) on the highway

From the flow vector, estimate the speed of the vehicles

Flag vehicles that are speeding, identify licence plates and issue/send tickets to home

Video Retiming



Increase Frame rate

Image Stabilization

- Optical Flow is computed from frame to frame
- Look for dominant flow (That is way most points are moving)
- That usually corresponds to the motion of the background of the video (e.g., house in video)
- That flow is compensated for to remove the shake of the camera.
- What you are left with is the remaining objects and their motion in the video.
- After applying image stabilization, you end up with a stabilized video.

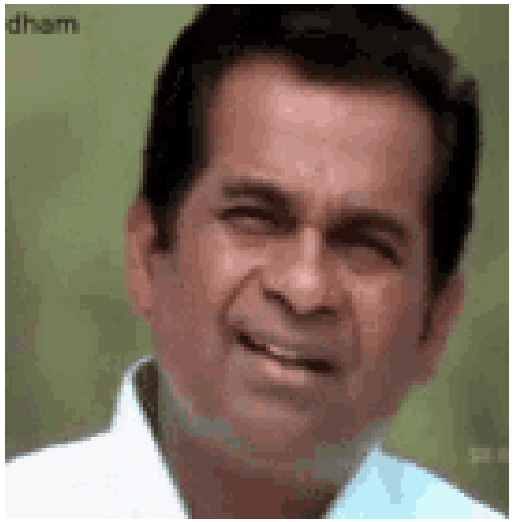
Face Tracking

- Optical Flow of points on a Face is computed from frame to frame.
- Look at Face Mesh: Every vertex on this mesh is tracked from frame to frame
- Figure out when the Person is blinking, How lips are moving, what is the expression of the person etc

Optical Flow for Gaming

- The user has a video of himself. In this video, a virtual ball is shown
- Use can Interact with the ball by moving user's hands.
- Optical flow vectors are computed and they are used to guide the ball to move it through space and time

Facial animation



http://www.fxguide.com/article333.html

Motion Paint

Use optical flow to track brush strokes, in order to animate them to follow underlying scene motion.

http://www.fxguide.com/article333.html

Motion estimation techniques

- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small

- Feature-based methods
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)

References

- Motion Field Optical Flow
- How Do Computers See Motion? Lucas-Kanade Method Explained
- https://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2017/OpticalFlow.pdf
- https://sandipanweb.wordpress.com/2018/02/25/implementing-lucas-kanade-optical-flow-algorithm-in-python/