Computer Vision 1

HC3b

Motion
Optical Flow
Tracking

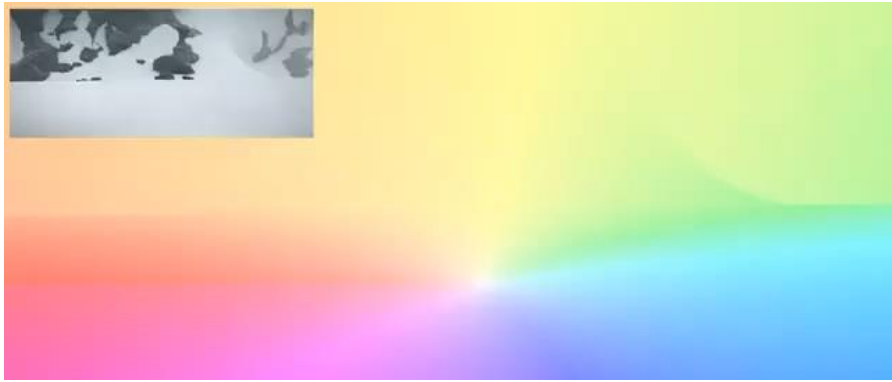
Dr. Martin Oswald, Dr. Dimitris Tzionas, Dr. Arun Mukundan,
(m.r.oswald, d.tzionas, a.mukundan)@uva.nl

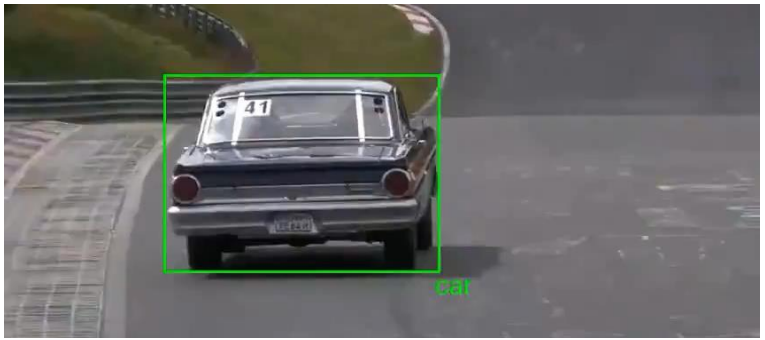
University of Amsterdam

computer vision lab

Outline

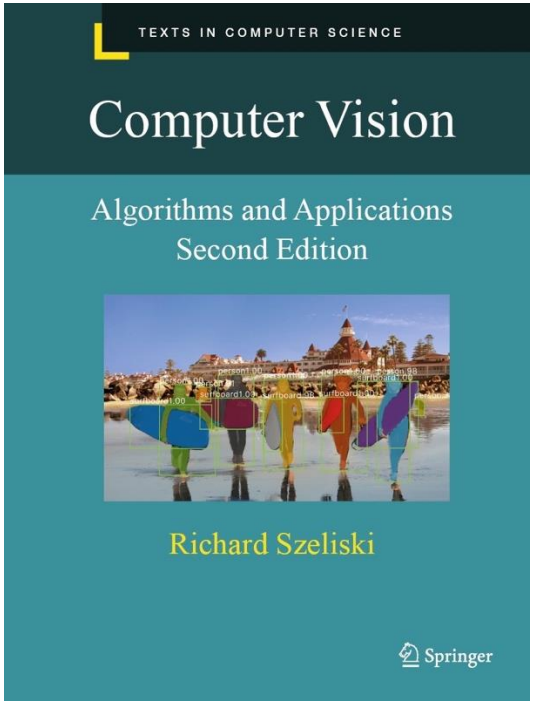
- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking





Textbook

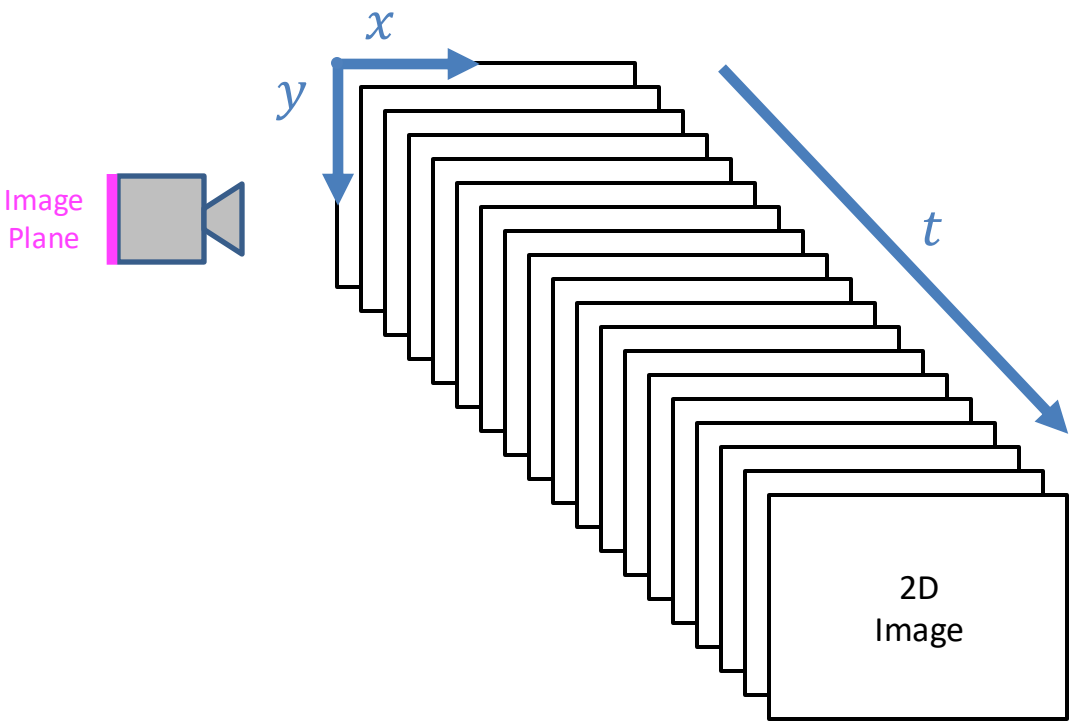
- Sec. 9.1.1
- Sec. 9.1.3
- Sec. 9.3 (9.3.1optional)

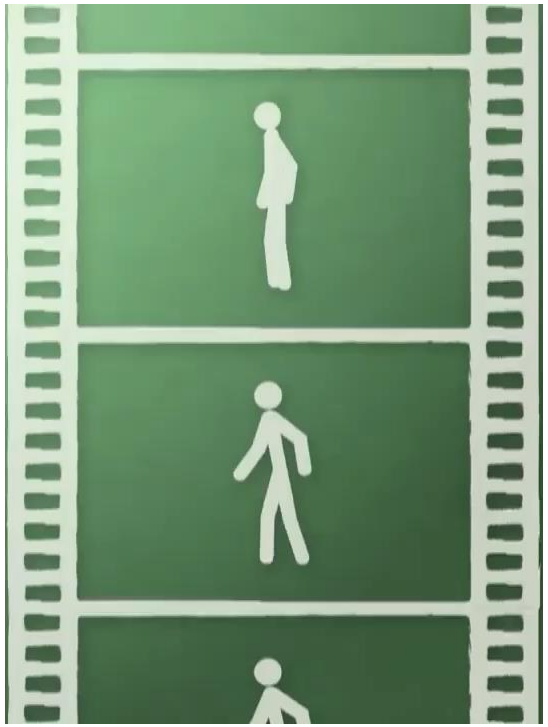


Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

Motion – Video

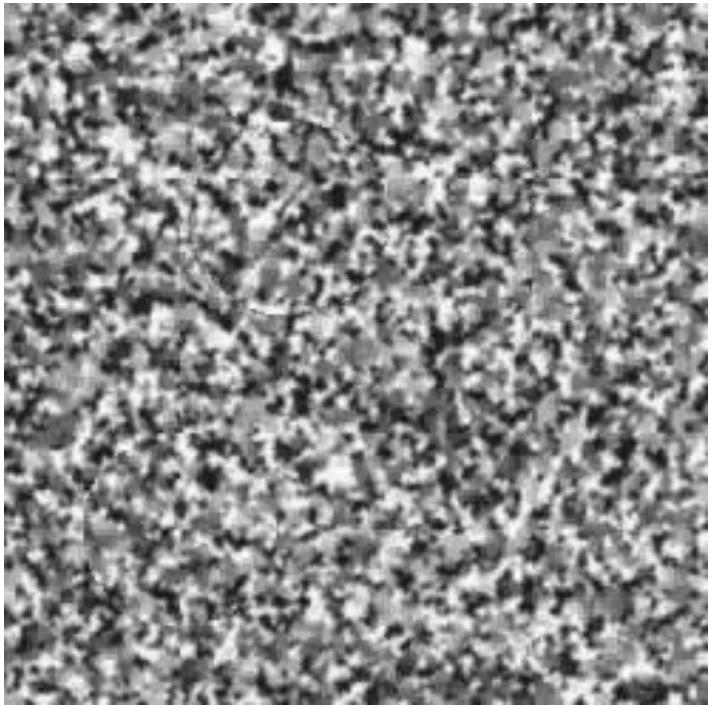




https://youtu.be/AYQAKwCxScc

Motion Perception

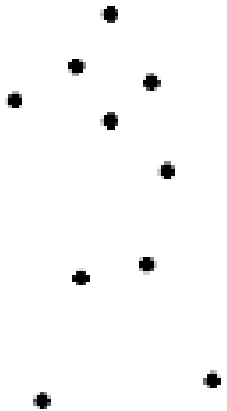
- Motion is a powerful perceptual cue
- Sometimes, it is the only cue



Slide by:
S. Lazebnik

Motion Perception

- Even ‘impoverished’ motion data evoke a strong percept



G. Johansson
‘Visual Perception of Biological Motion and a Model For Its Analysis’
Perception and Psychophysics, 1973.

Making Sense of Video

An often-told story

Aspiring young graduate student

What are the three most important problems in computer vision?

Correspondence, Correspondence, Correspondence!

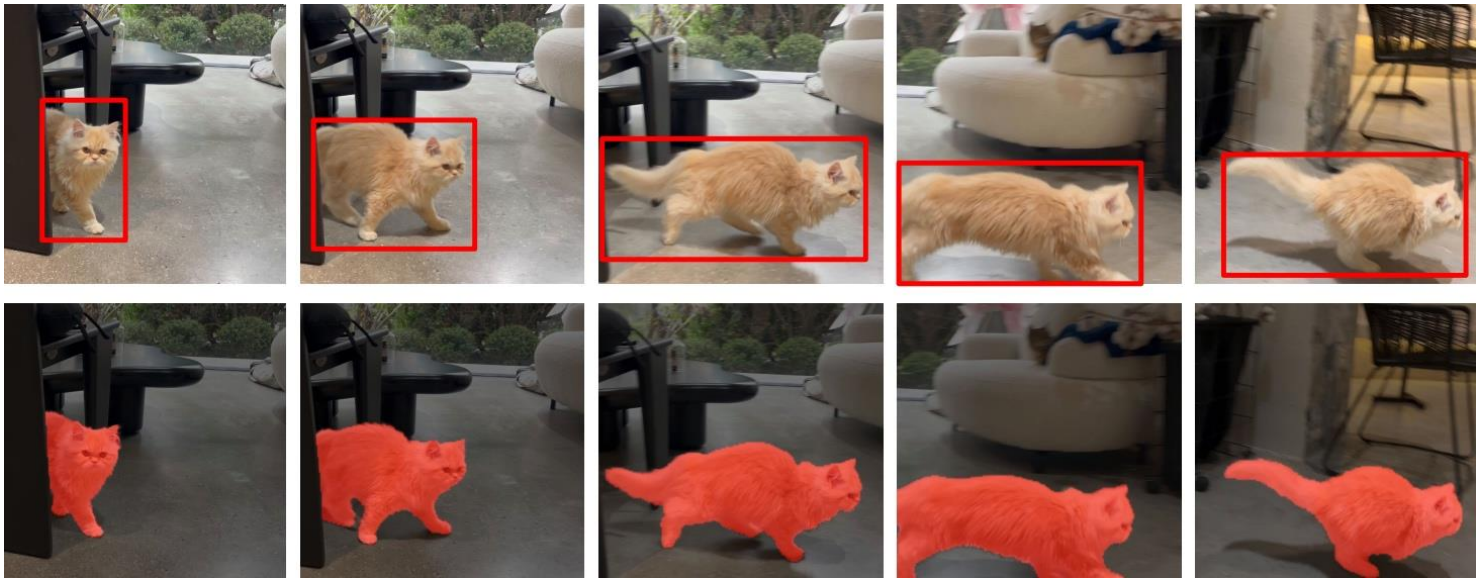
Takeo Kanade

Slide: J. Tompkin

Many Forms of Correspondence

Bounding box

Segmentation



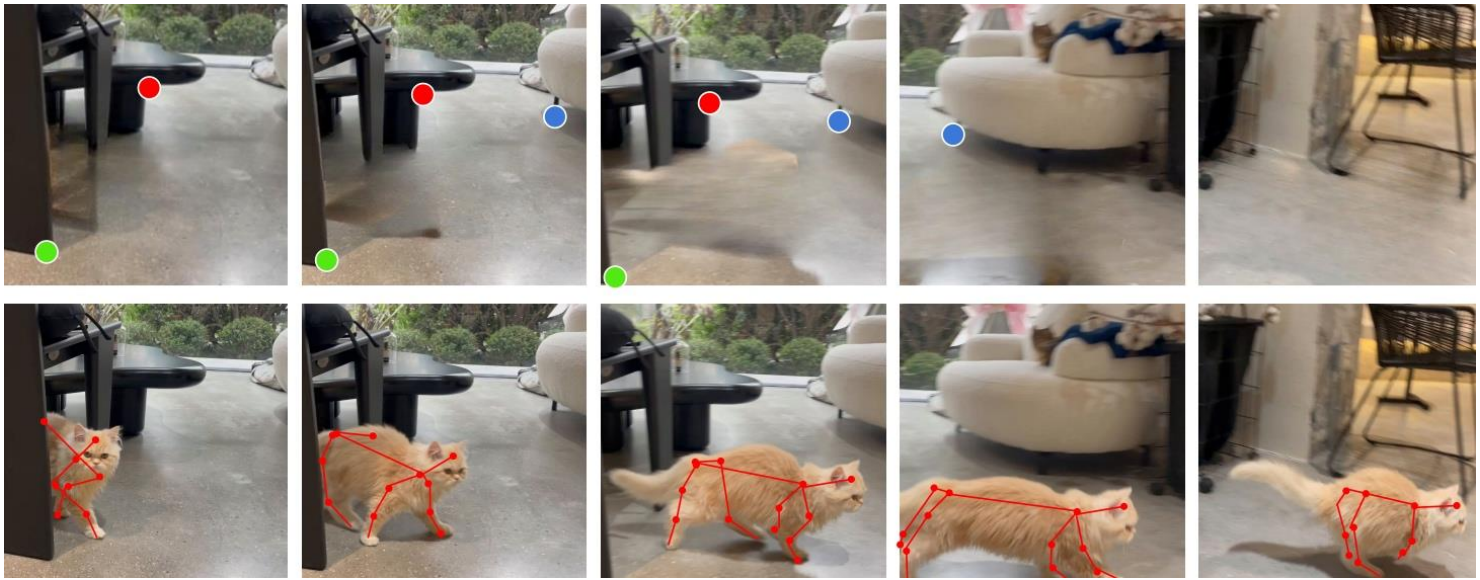
Slide:
J. Tompkin

Image credit:
Doersch *et al.* NeurIPS 2022

Many Forms of Correspondence

Keypoints

Pose/semantic keypoints



Slide:
J. Tompkin

Image credit:
Doersch *et al.* NeurIPS 2022

Dense Correspondences

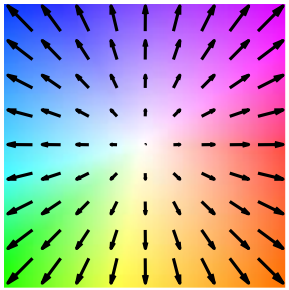
Optical Flow (today's topic)

Input

[Liu *et al.* CVPR'08]

Optical Flow

(2D motion vector)



Color key

[Baker *et al.* IJCV'11]

Slide:
J. Tompkin

Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

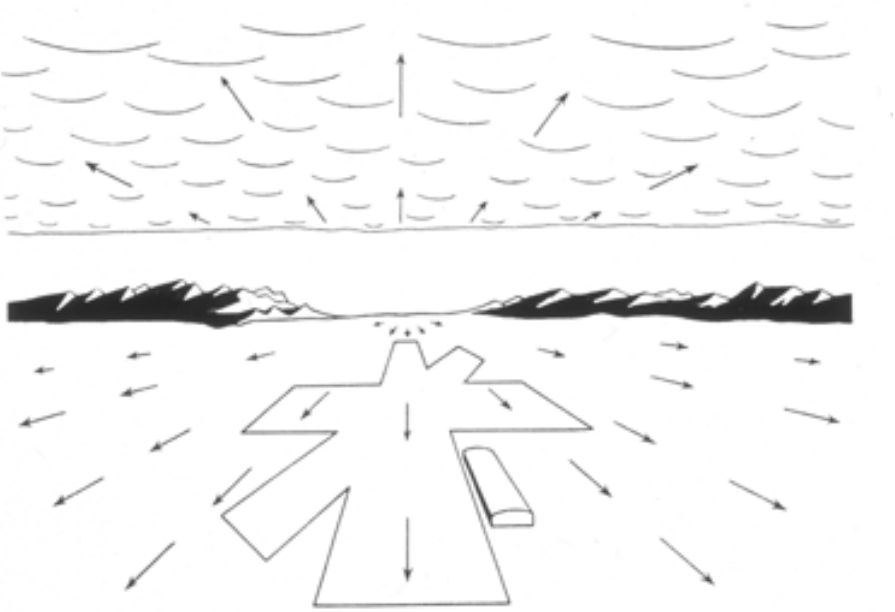
Motion Perception

Animal moves

↓

Changing luminance pattern
on 'optic array' [Gibson]
or camera (img plane)

Optical Flow:
'Motion' of this pattern
('apparent motion')



Assumption:
Gives *direct access* to properties of the world that are important for organism to *survive*.

- Rabbit runs to hide in hole
- Lion jumps to eat you

Adapted slide
of M. J. Black

J. J. Gibson, *The Ecological Approach to Visual Perception*, 1979

https://yzhu.io/courses/core/reading/02.gibson.pdf

Motion Field (MF) & Optical Flow (OF)

2D Motion Field (MF): Describes ***projection*** of the **3D motion** of **3D scene points** *onto 2D image plane*

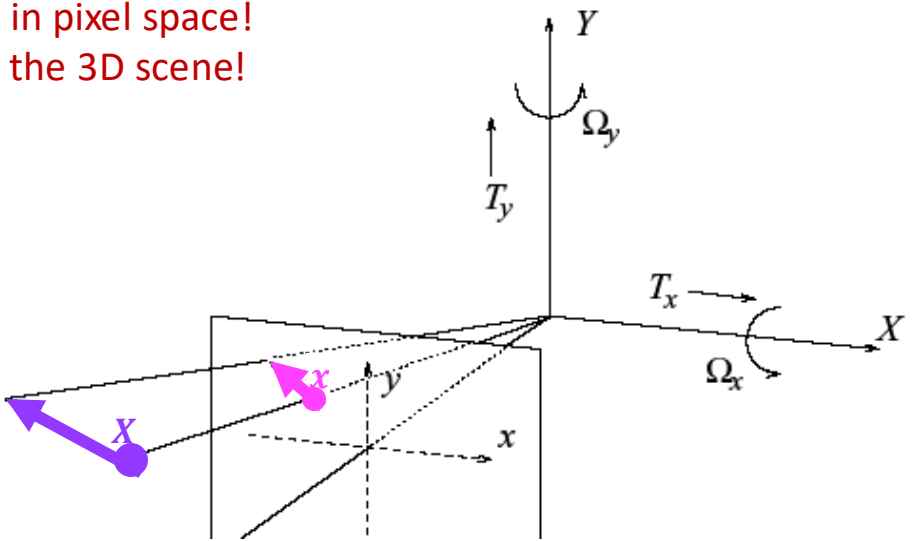
Note: Knows the 3D scene!

2D Optical-Flow Field (OF): Describes ***apparent motion*** in *images*

Note: Reasoning only in pixel space!
Does not know the 3D scene!

$\boldsymbol{x} = (x, y)$

$\boldsymbol{X} = (X, Y, Z)$



Adapted slide of M. J. Black

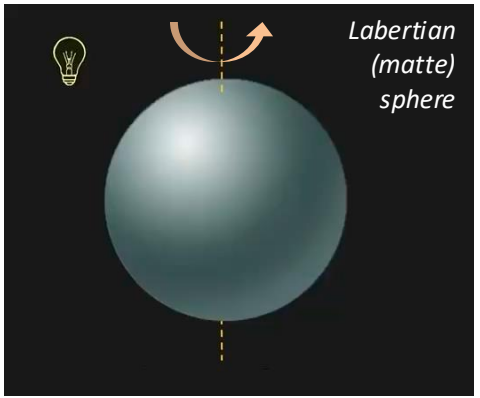
MF & OF – Thought Experiment #1

2D Motion Field (MF): Describes ***projection*** of the **3D motion** of **3D scene points** *onto 2D image plane*

Note: Knows the 3D scene!

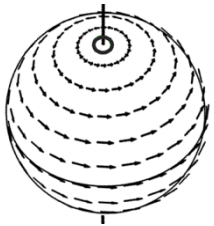
2D Optical-Flow Field (OF): Describes ***apparent motion*** in *images*

Note: Reasoning only in pixel space!
Does not know the 3D scene!



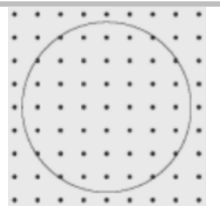
Example: Rotating **Labertian** ('matte') sphere

Motion Field → We know how the 3D sphere points **rotate** ('3D Scene Flow')
Project the 3D motion vectors onto 2D image plane & get:
MF == **Non-zero 2D vectors** w rough direction **left→right**



https://en.wikipedia.org/wiki/Optical_flow

Optical Flow → We do **not** know how the 3D sphere rotates (3D Scene Flow)
We **just** judge by **looking at pixels**
OF == **Constant** (no apparent motion) == Zero-length 2D vectors



http://www.cs.toronto.edu/~jepson/csc2503/opticalFlow.pdf

Adapted slide of M. J. Black

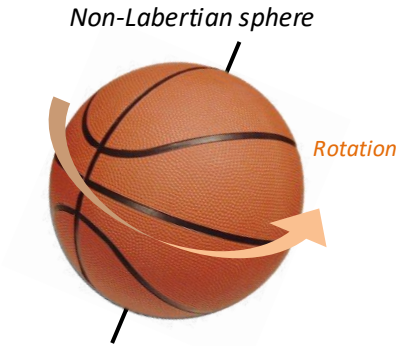
MF & OF – Thought Experiment #2

2D Motion Field (MF): Describes ***projection*** of the **3D motion** of **3D scene points** *onto 2D image plane*

Note: Knows the 3D scene!

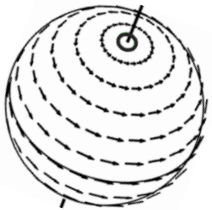
2D Optical-Flow Field (OF): Describes ***apparent motion*** in *images*

Note: Reasoning only in pixel space!
Does not know the 3D scene!



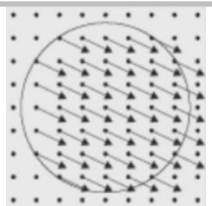
Example: Rotating **Non-Labertian** sphere

Motion Field → We know how the 3D sphere points **rotate** ('3D Scene Flow')
Project the 3D motion vectors onto 2D image plane & get:
MF == **Non-zero 2D vectors** w rough direction **left→right**



https://en.wikipedia.org/wiki/Optical_flow

Optical Flow → We do **not** know how the 3D sphere rotates (3D Scene Flow)
We **just** judge by **looking at pixels**
OF == **Non-zero 2D vectors** w rough direction **left→right**



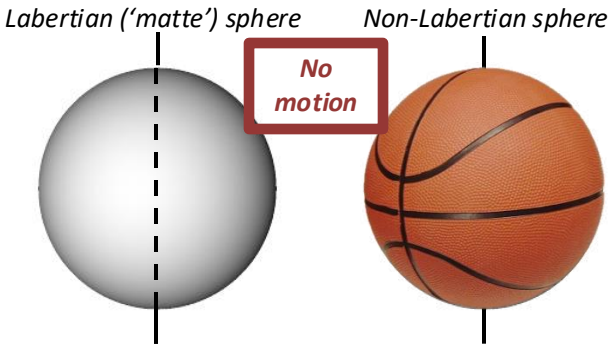
http://www.cs.toronto.edu/~jepson/csc2503/opticalFlow.pdf

Adapted slide of M. J. Black

MF & OF – Thought Experiment #3

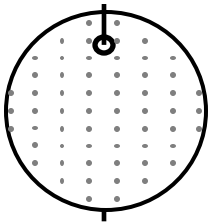
2D Motion Field (MF): Describes ***projection*** of the **3D motion** of **3D scene points** *onto 2D image plane*
Note: Knows the 3D scene!

2D Optical-Flow Field (OF): Describes ***apparent motion*** in *images*
Note: Reasoning only in pixel space!
Does not know the 3D scene!



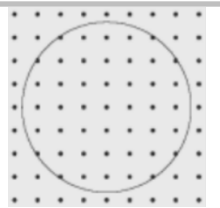
Example: No sphere rotation

Motion Field → We know that the 3D sphere points **don't rotate**
Project zero-length 3D motion vectors on 2D and get…
MF == **Constant** motion == Zero-length 2D vectors



https://en.wikipedia.org/wiki/Optical_flow

Optical Flow → We do **not** know how the 3D sphere rotates (3D Scene Flow)
We **just** judge by **looking at pixels**
OF == **Constant** (no apparent motion) == Zero-length 2D vectors



Adapted slide of M. J. Black

http://www.cs.toronto.edu/~jepson/csc2503/opticalFlow.pdf

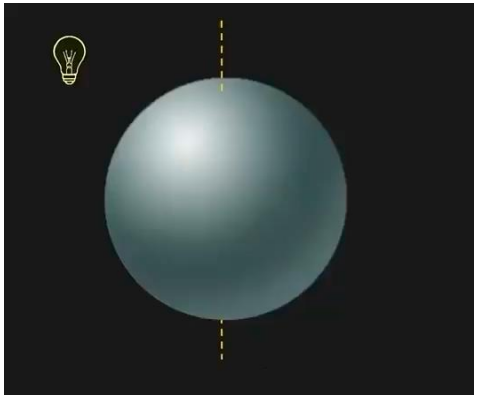
MF & OF – Thought Experiment #4

2D Motion Field (MF): Describes ***projection*** of the **3D motion** of **3D scene points** *onto 2D image plane*

Note: Knows the 3D scene!

2D Optical-Flow Field (OF): Describes ***apparent motion*** in *images*

Note: Reasoning only in pixel space!
Does not know the 3D scene!



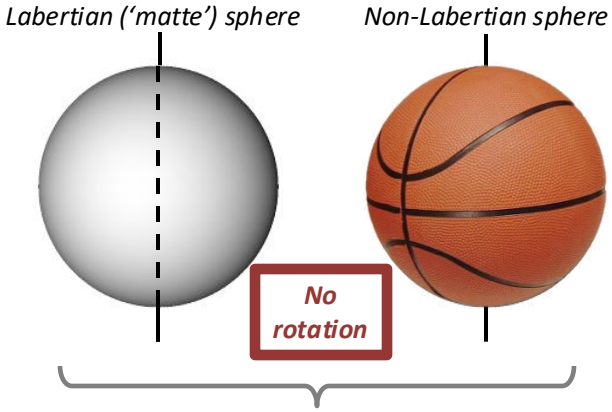
Example: No sphere rotation – But light moves!

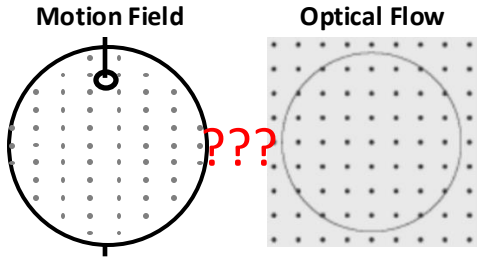
MF == Constant motion
== Zero-length 2D vectors

OF == Non-zero 2D vectors
direction depends on **apparent motion**
due to light motion

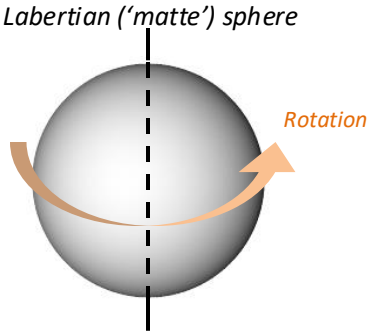
Animation by
S. Nayar

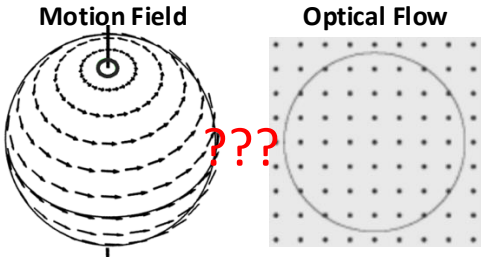
MF & OF – Thought Experiments – Summary



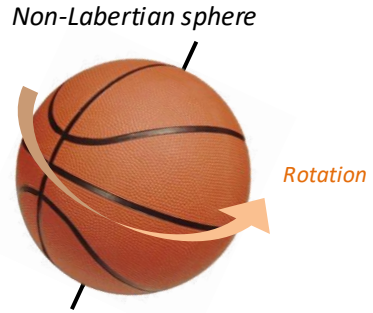


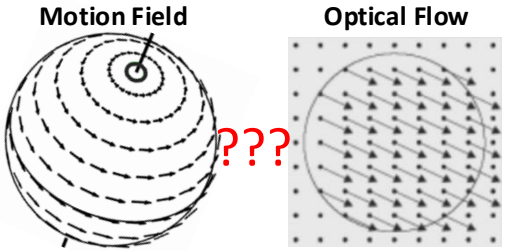
www.cs.toronto.edu/~jepson/csc2503/opticalFlow.pdf





www.cs.toronto.edu/~jepson/csc2503/opticalFlow.pdf



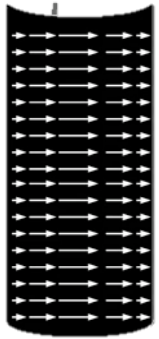


www.cs.toronto.edu/~jepson/csc2503/opticalFlow.pdf

Adapted slide of M. J. Black

Optical Flow != Motion Field

Barber Pole



Motion Field

(3D aware)

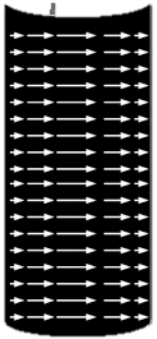
Optical Flow

(only 2D aware)

Slide adapted from S. Nayar

Optical Flow != Motion Field

‘Peephole’
(explains Optical Flow)



Motion Field

(3D aware)

Optical Flow

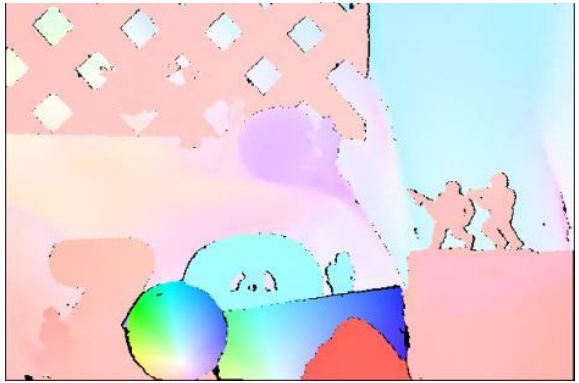
(only 2D aware)

Slide adapted from S. Nayar

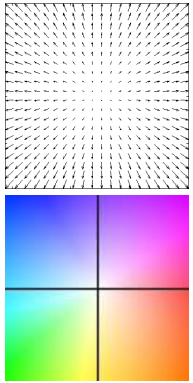
Optical Flow

Apparent Motion (displacement) of Pixels

https://vision.middlebury.edu/flow/data
https://eval-color-allframes.zip
Seq. 'Army', frames: 11 & 12



Optical Flow
(apparent displacement)



Color-coding

$I(x, y, t)$ Image intensity (pixel value) at time t & location $x = (x, y)$

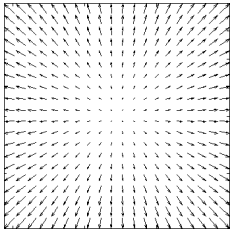
$u(x, y)$ Horizontal component

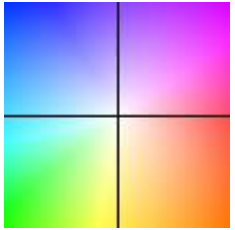
$v(x, y)$ Vertical component

Color → Direction
Saturation → Magnitude

Adapted slide of M. J. Black

Optical Flow – Goal





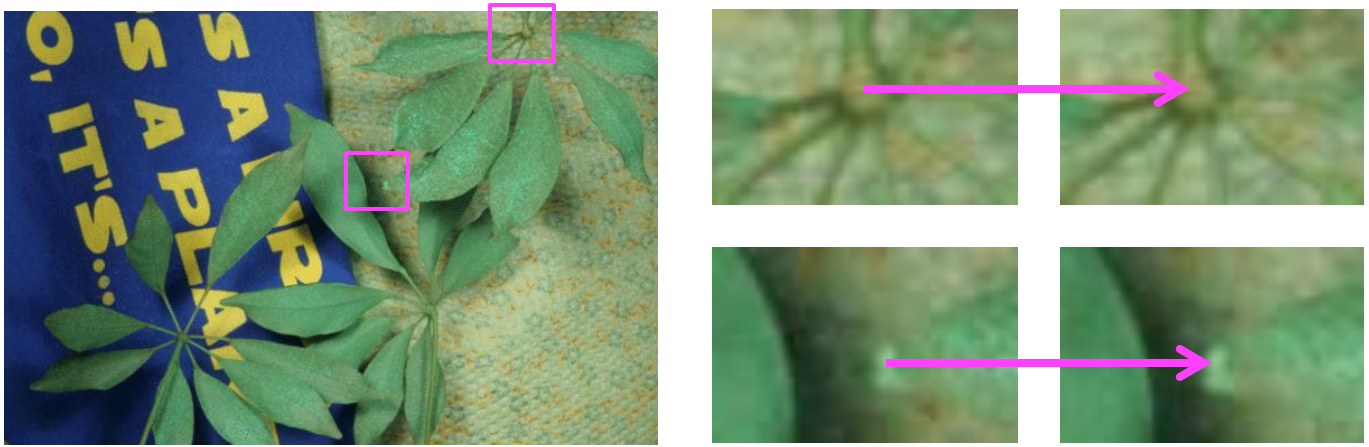
The ***apparent*** 2D image ***motion of pixels***
from one image frame to the next one
in a video sequence

Adapted slide
of M. J. Black

OF – Assumption #1 – Brightness Constancy

The most foundational assumption for OF

Assumption #1: Brightness Constancy



$$I(x+u, y+v, t+1) = I(x, y, t)$$

Displacement @ next frame

Optical Flow

$$I(x + udt, t + dt) = I(x, t)$$

Alternative notation

Adapted slide of M. J. Black

OF – Assumption #2 – Small Motion

Assumption #2: Small motions

- Estimate OF u, v → Take these out of function I
- Small motion == Small Flow
- Image: differentiable function

Use **Taylor's expansion** →

$$I(x+u, y+v, t+1) = I(x, y, t)$$

$dx = u$ $dy = v$ $dt = 1$ *

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

$$u\frac{\partial}{\partial x}I(x,y,t) + v\frac{\partial}{\partial y}I(x,y,t) + \frac{\partial}{\partial t}I(x,y,t) = 0$$

$$uI_x + vI_y + I_t = 0$$

Can be approx. computed

2 unknowns in **1 equation**

* To simplify our notation
In theory: dt is infinitesimally small
In practice: $dt = \frac{1}{FPS}$ of camera

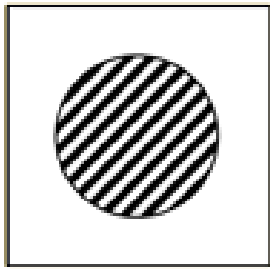
Estimating 2D Optical Flow for 1 point is ill posed!

Adapted slide of M. J. Black

Outline

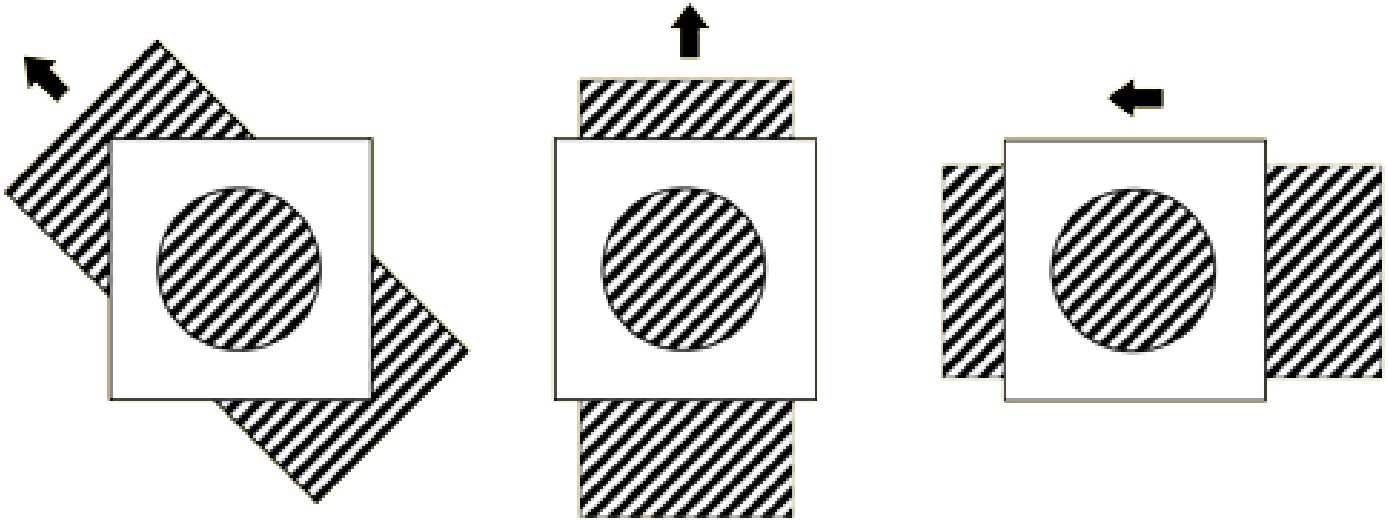
- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

Optical Flow – Aperture Problem



What do we see through the hole?
(“point”)

Optical Flow – Aperture Problem



https://cdn.sinauer.com/wolfe4e/wa08.02.html

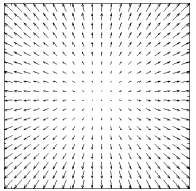
Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

OF – Assumption #3: Spatial Smoothness

For method of **Lucas-Kanade**

Assumption #3: Spatial Smoothness



Neighboring pixels in the image are likely to belong to the same surface

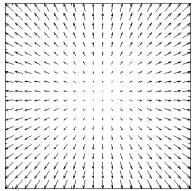
Surfaces are mostly smooth

Neighboring pixels will have similar flow

OF – Assumption #3: Spatial Smoothness

For method of **Lucas-Kanade**

Assumption #3: Spatial Smoothness



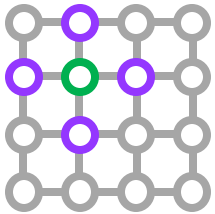
Neighboring *points* have the **same** optical flow

$$\begin{cases} u_x = u_{x_i} \\ v_x = v_{x_i} \end{cases}$$

i.e., the spatial derivative of optical flow is zero

$$x_i \in G(x)$$

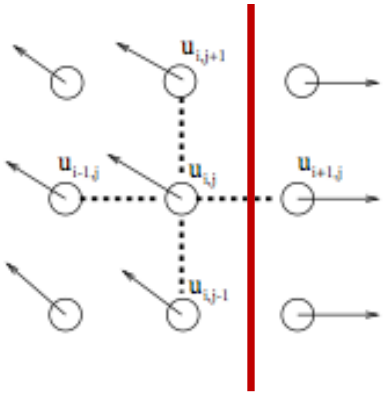
Graph neighborhood of 'central' point $x = (x, y)$



OF – Assumption #3: Spatial Smoothness

For method of **Lucas-Kanade**

Assumption #3: Spatial Smoothness



Smoothness: **Neighboring** pixels ***usually*** belong to ***same*** **surface**

Except when they ***don't*** → **Different objects** separated by **motion boundaries/discontinuities**

Optical Flow – Lucas-Kanade Method

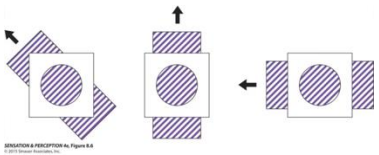
For 1 point:

$$uI_x + vI_y + I_t = 0$$

2 unknowns
1 equation

Problem

Estimating 2D Optical Flow for 1 point is ill posed …
(aperture problem)



Solution

so, let's consider **more points** $n \geq 2$ and assume that the…

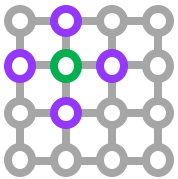
- central point $\boldsymbol{x} = (x, y)$ and
- surrounding points $\boldsymbol{x}_i = (x_i, y_i)$

have **same Optical Flow** $\boldsymbol{u} = (u, v)$

$$\boldsymbol{u}_{\boldsymbol{x}} = \boldsymbol{u}_{\boldsymbol{x}_i}$$

$$\boldsymbol{x}_i \in G(\boldsymbol{x})$$

$$i = 1, \dots, n$$



For all $i = 1, \dots, n$:

$$uI_{x_i} + vI_{y_i} = -I_t$$

$$uI_x(\boldsymbol{x}_i, t) + vI_y(\boldsymbol{x}_i, t) = -I_t(\boldsymbol{x}_i, t)$$

Stacking equations for $i = 1, \dots, n$:

$$\begin{bmatrix} I_x(\boldsymbol{x}_1, t) & I_y(\boldsymbol{x}_1, t) \\ I_x(\boldsymbol{x}_2, t) & I_y(\boldsymbol{x}_2, t) \\ \vdots & \vdots \\ I_x(\boldsymbol{x}_n, t) & I_y(\boldsymbol{x}_n, t) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\boldsymbol{x}_1, t) \\ I_t(\boldsymbol{x}_2, t) \\ \vdots \\ I_t(\boldsymbol{x}_n, t) \end{bmatrix}$$

$$\mathrm{A}\boldsymbol{u} = -b$$

Least-squares solution

$$\boldsymbol{u} = -(\mathrm{A}^{\mathrm{T}}\mathrm{A})^{-1}\mathrm{A}^{\mathrm{T}}b$$

Structure tensor

$$\mathrm{A}^{\mathrm{T}}\mathrm{A} = \begin{bmatrix} \sum_{i=1}^{n} I_x^2(\boldsymbol{x}_i, t) & \sum_{i=1}^{n} I_x(\boldsymbol{x}_i, t) I_y(\boldsymbol{x}_i, t) \\ \sum_{i=1}^{n} I_x(\boldsymbol{x}_i, t) I_y(\boldsymbol{x}_i, t) & \sum_{i=1}^{n} I_y^2(\boldsymbol{x}_i, t) \end{bmatrix}$$

Depend on the image neighborhood we focus on

Consider:

- Which neighborhood to choose
- Whether $\boldsymbol{x}_i$ should be weighted in least-squares error
- Works **only** for ***small*** **displacements** (eg max ~2 pixels)
- For ***big displacements*** … ??? (see next slide)

Outline

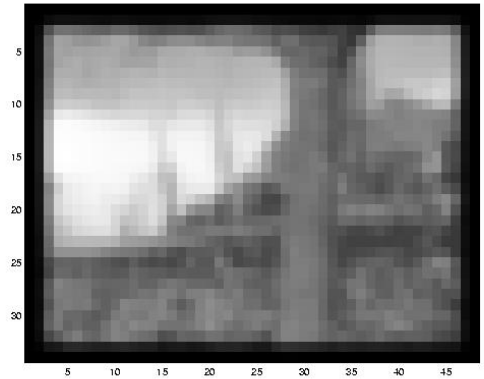
- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

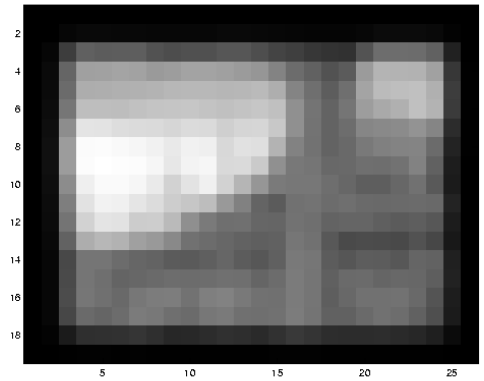
Revisiting Small Motion

- Is this motion small enough?
  - Probably not—it's much larger than 1 pixel
  - How might we solve this problem?

Slide by:
S. Lazebnik

Reduce the resolution!

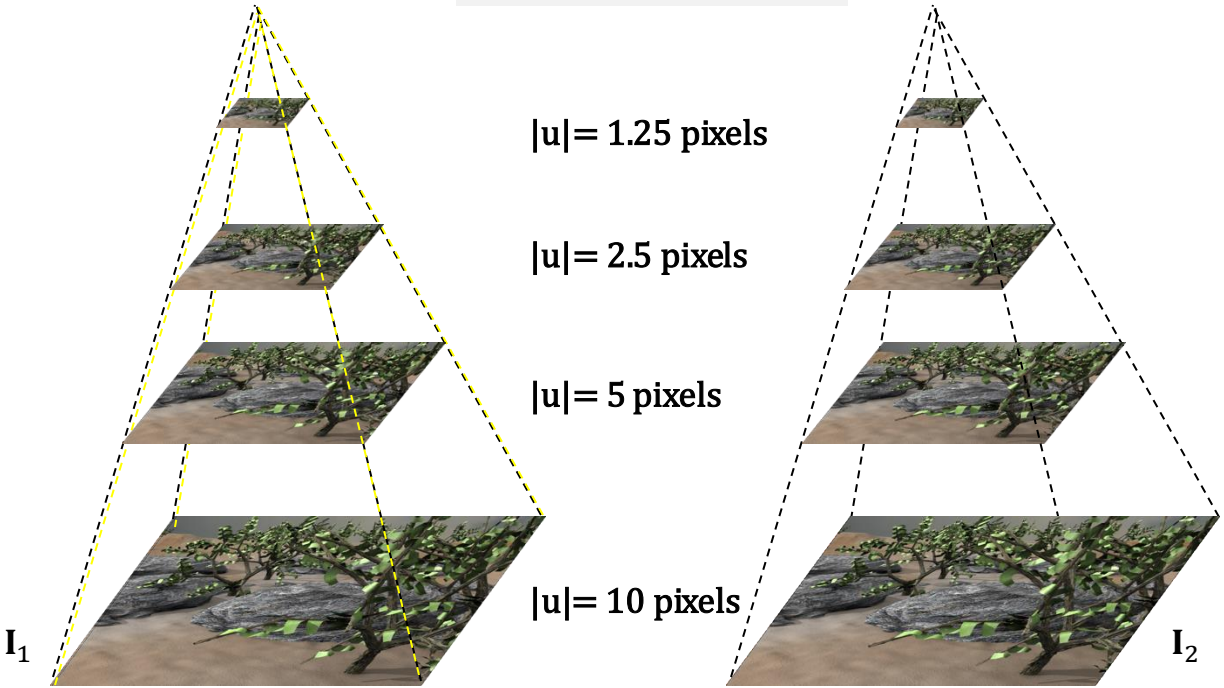




Slide by:
S. Lazebnik

Optical Flow – Coarse-to-fine Estimation

Image Pyramid
Levels of varying resolution



Motions smaller at higher levels

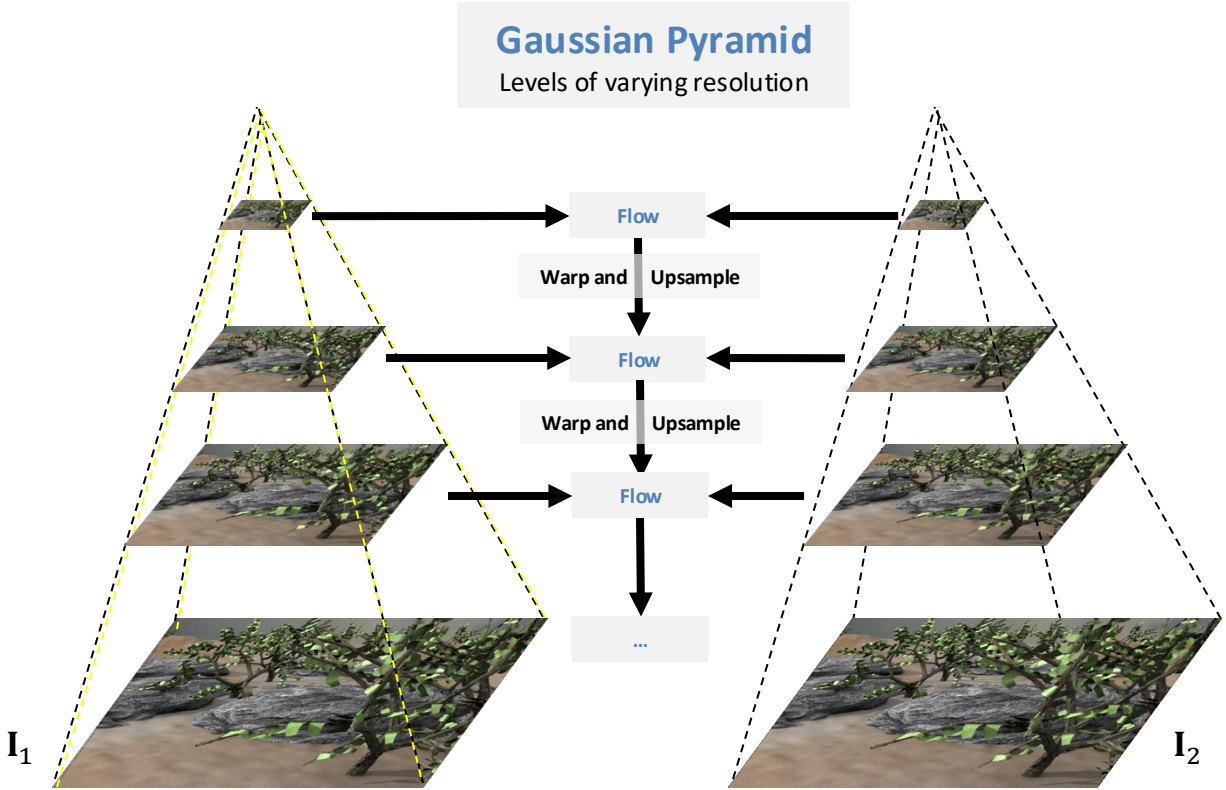
So, first order Taylor series is fine

'Initialize' OF solution of each level with the one of the previous level

Small things 'disappear' at top of pyramid and never 'come back'

Bergen et al., *Hierarchical Model-Based Motion Estimation*, ECCV 1992

Optical Flow – Coarse-to-fine Estimation



Motions smaller at higher levels

So, first order Taylor series is fine

'Initialize' OF solution of each level with the one of the previous level

Small things 'disappear' at top of pyramid and never 'come back'

Bergen et al., *Hierarchical Model-Based Motion Estimation*, ECCV 1992

Optical Flow – Coarse-to-fine Estimation

Warping Operation

1st frame and **2nd frame**

Big difference

1st frame and **warped 2nd frame**

Smaller difference!

Optical Flow – Coarse-to-fine Estimation

Original Pair

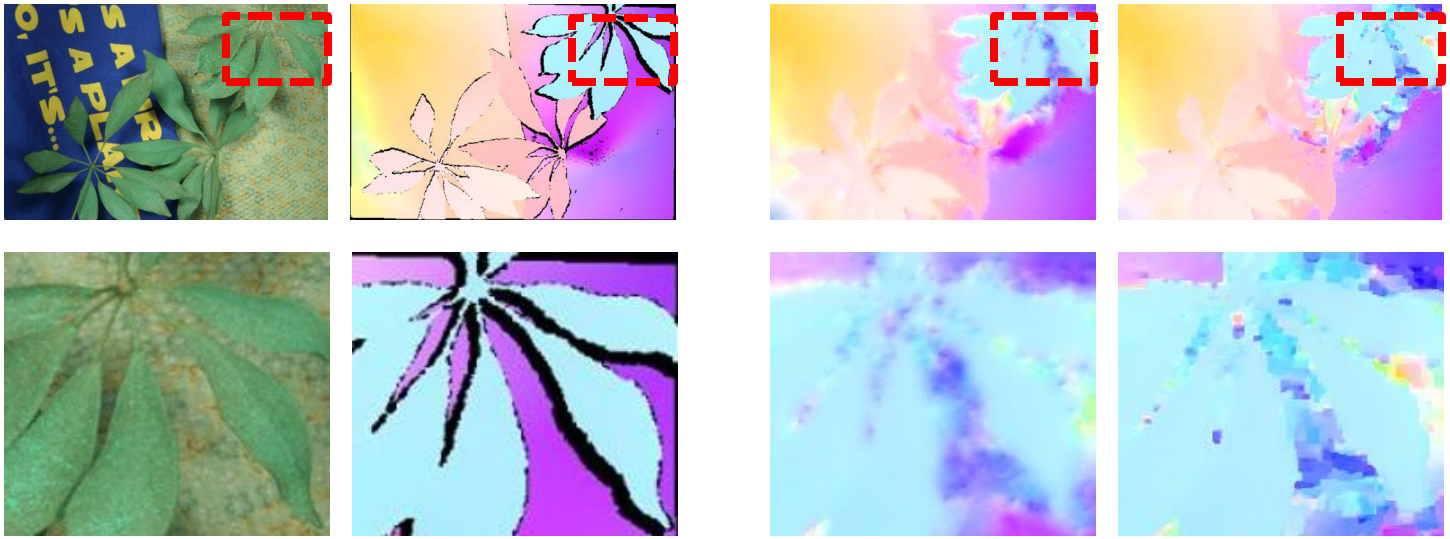
Flow Aligned Pair

Optical Flow – Coarse-to-fine Estimation

Original Pair

Flow Aligned Pair

Optical Flow – Challenges



* Some 'classic' methods

Occluded regions in black

- **Fine Structures**
- **Motion boundaries**
- **Occlusions**

Challenging for **Optical** Flow estimation

Adapted slide of M. J. Black

Optical Flow – Challenges

Large motion, motion blur

Texture-less regions

Occlusions

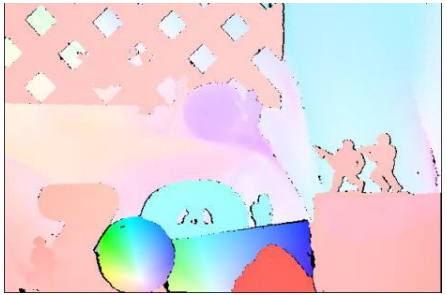
Lighting changes, noise …

Slide:
J. Tompkin

Optical Flow – Two Classic Methods

Color Images

https://vision.middlebury.edu/flow/data
Seq. 'Army'

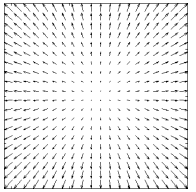


Ground truth

Horn & Schunck, 1981 (HS) *

Black & Anandan, 1996 (BA) *

* Some 'classic' methods
(pre-Deep Learning)

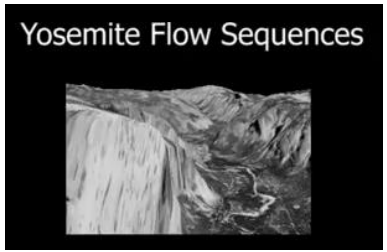


Color-coding

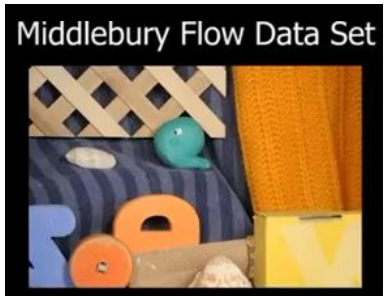
Adapted slide
of M. J. Black

Optical Flow – Datasets

Small-scale Datasets
(Evaluation)



Performance of optical flow techniques
John L. Barron, David J. Fleet, Steven S. Beauchemin
IJCV 1994



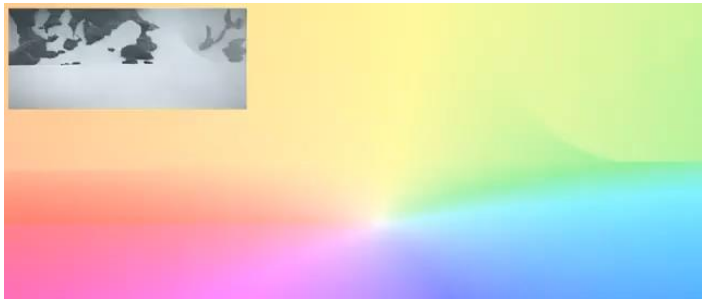
A Database and Evaluation Methodology for Optical Flow
Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, Michael J. Black, Richard Szeliski
IJCV 2011

Large-scale Datasets
(Evaluation + Machine Learning)

'**Sintel**' `dataset

A naturalistic open source movie for optical flow evaluation
Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, Michael J. Black
ECCV 2012

http://sintel.is.tue.mpg.de



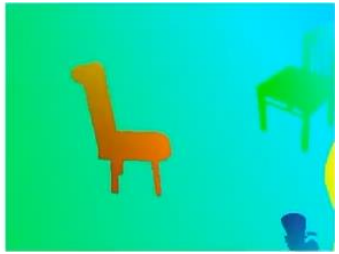
Sintel - Open Movie by Blender Foundation

https://youtu.be/eRsGyueVLvQ

'**Flying Chairs**' dataset

FlowNet: Learning optical flow with convolutional networks
Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, Thomas Brox
ICCV 2015

https://lmb.informatik.uni-freiburg.de/resources/datasets/FlyingChairs.en.html



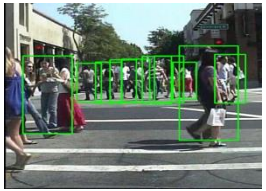
Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

Optical Flow – Applications

Image processing

[Liu & Freeman 2010]



[Walk et al. 2010]

Video denoising

Video super resolution

Video compression

Pedestrian detection

…

Computer Graphics

[Hays & Essa 2004]

[Wang et al. 2010]

Painterly animation

Motion magnification

Video retargeting

…

Games

Plant growth

Biology: cell motion,

heart wall motion, …

Ocean currents, Meteorology, …

Particle image velocimetry, …

…

Adapted slide
of M. J. Black

Optical Flow – Applications

Video Super-resolution

[Liu & Sun CVPR 2011,TPAMI 2014]

Optical Flow – Applications

Video Stabilization

Slide by:
Linda Shapiro

Optical Flow – Applications

(Ego-Pose) Tracking

Optical Flow – Applications

Motion Interpolation

Super SloMo

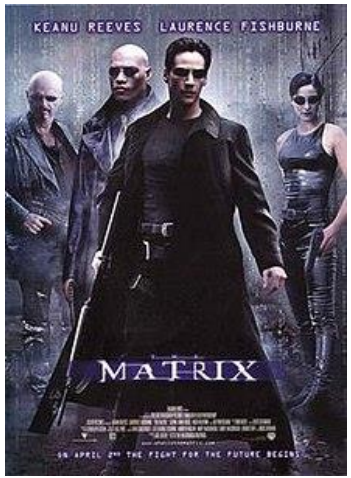
Jiang, Sun, *et al.* CVPR 2018 Spotlight]

Slide:
J. Tompkin

Incorporated into **NVIDIA NGX** SDK for the Turing GPU

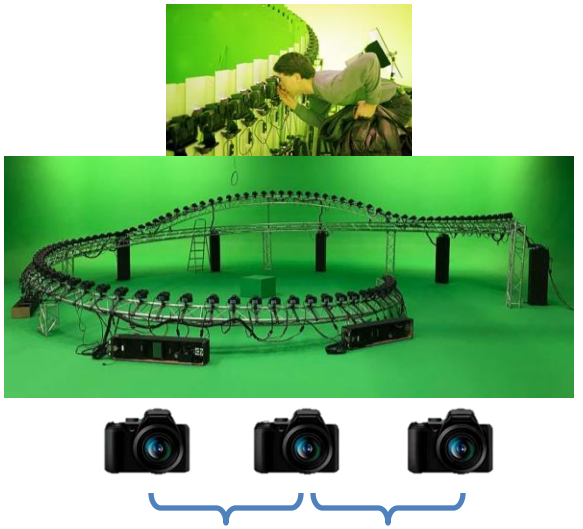
Optical Flow – Applications

Pose Interpolation



The Maatrix, 1998
https://www.imdb.com/title/tt0133093

Academy award, 2000
Best Visual Effects



Optical Flow → Correspondence between camera views

Smooth interpolation (virtual camera) between views

Adapted slide
of M. J. Black

Optical Flow – Applications

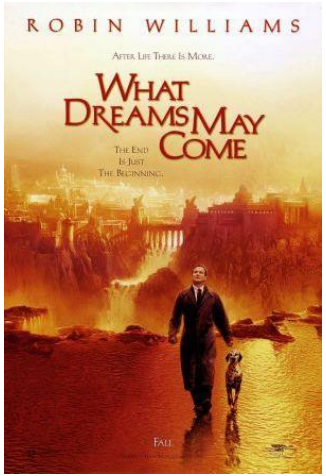
Appearance Transfer

The final shot was enabled with extensive development of tracking techniques, optical flow and a specialized particles tool to produce the painterly effects

Adapted slide of M. J. Black

Optical Flow – Applications

Appearance Transfer



What Dreams May Come, 1998
https://www.imdb.com/title/tt0120889

Academy award, 1999
Best Visual Effects

Impressionist effect

Transfer: Real-world motion ↔ Painting

Same principle for propagating annotations across video

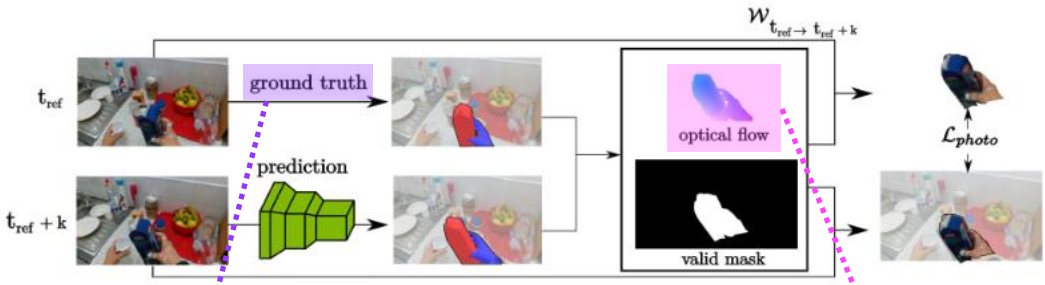
Adapted slide
of M. J. Black

Optical Flow – Applications

Weak/Self-Supervision

LASR: Learning Articulated Shape Reconstruction from a Monocular Video [Yang et al., CVPR'21]

*'Why should motion be treated as a first-class citizen? Besides that **optical flow** naturally encodes correspondences, it provides more fine-grained information than keypoints as well as semantic parts. […] can be obtained more reliably over two consecutive frames.'*



Leveraging Photometric Consistency over Time for Sparsely Supervised Hand-Object Reconstruction [Hasson et al. CVPR'20]

*'[…] **annotations** are only available for a sparse subset of frames in a video'*

*'Given our estimated reconstructions, we differentiably render the **optical flow** between pairs of adjacent images and use it within the network to warp one frame to another. We then apply a **self-supervised photometric loss** that relies on the **visual consistency between nearby images**.'*

*'The loss effectively **propagates information** from a **few annotated frames** to the **rest of the video**.'*

Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking


1

Tracking Demo

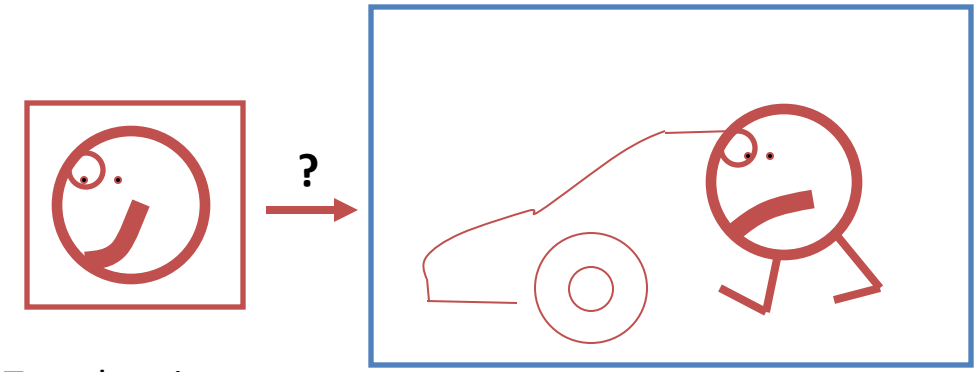
https://www.youtube.com/watch?v=EwxtyHHo3ql

Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

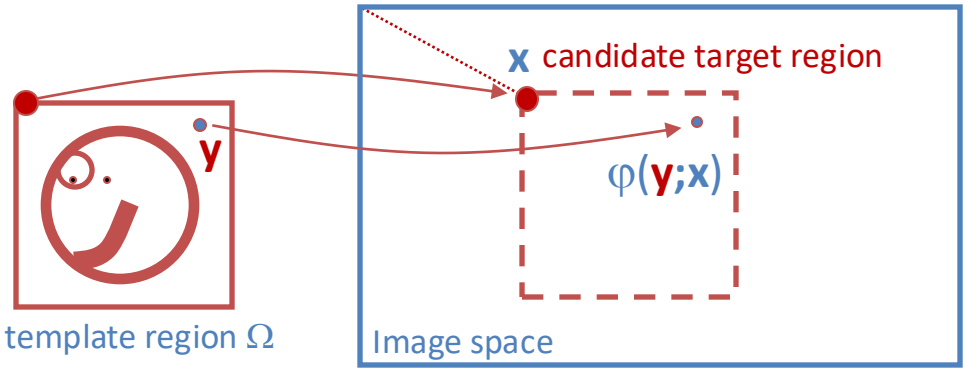
Template-based Tracking

- Given a **template** image $T(\mathbf{x})$ that shows a target object
- Search for the target object in an image frame $I(\mathbf{x},t)$ …
- … by comparing the **template** image $T(\mathbf{x})$ with image areas
- Assume that the template is fixed and given in advance



Template-to-Image Alignment

- 'Align' template into candidate target region in image via transformation: $\varphi(\mathbf{y})$
- This transformation depends on the running position **x** of candidate region
- So, we write $\varphi(\mathbf{y}; \mathbf{x})$

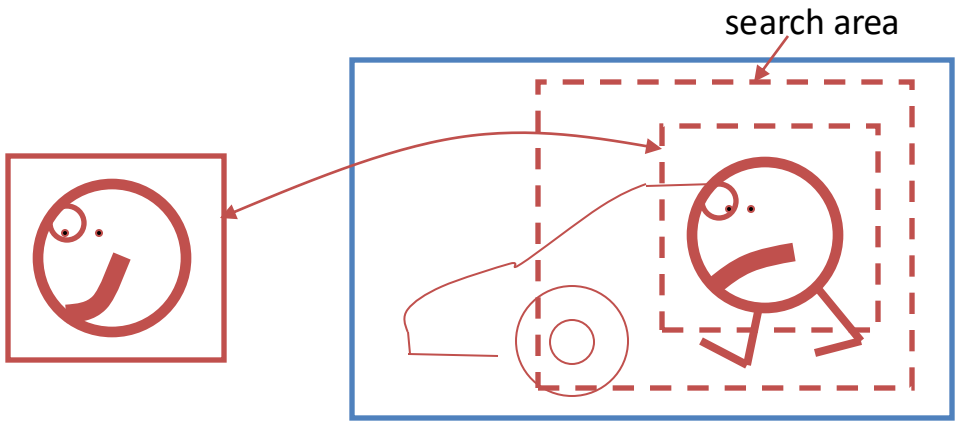


Example

```
y = [2, 10]
x = [4, 06]
φ(y;x) = [6, 16]
```

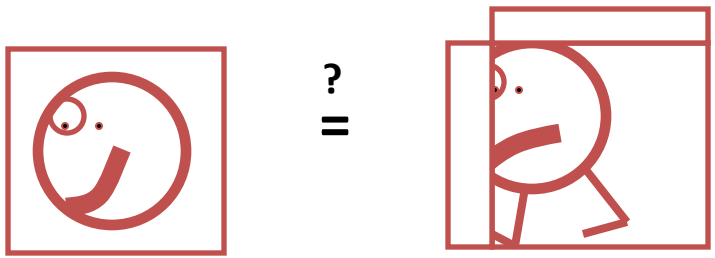
Search Space

- ‘Align’ the template with every possible candidate region (i.e. every possible position **y**) in the image
- Find the **most similar candidate** according to a **similarity measure**
- Search only in area around previous position (assume: object won’t have moved far)



Similarity Measure

- Need a measure of **how similar** the template and cand date are
- The similarity measure can be based on:
  - Pixelwise intensity/color difference → **SSD** and **Correlation** trackers
  - Histogram difference → **Mean-shift** tracker



SSD and Correlation

- **SSD** is short for '**Sum of Squared Differences**':

$$SSD(\mathbf{y}) = \sum_{\mathbf{x}\in\Omega} \|I(\mathbf{x}+\mathbf{y}) - T(\mathbf{x})\|^2 \longrightarrow y^* = \min_{\mathbf{y}} SSD(\mathbf{y})$$

- Simpler measure → **Cross-Correlation** (unnormalized):

$$CC(\mathbf{y}) = \sum_{\mathbf{x}\in\Omega} I(\mathbf{x}+\mathbf{y})T(\mathbf{x}) \longrightarrow y^* = \max_{\mathbf{y}} CC(\mathbf{y})$$

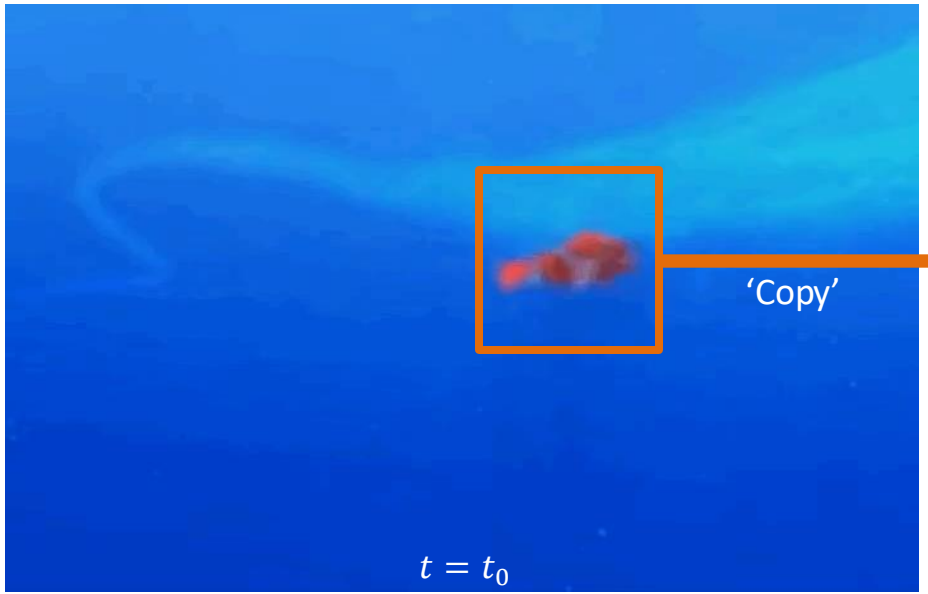
Exhaustive Search

- Calculate **SSD** or **CC** for every candidate position **y** in a search window
- Choose the position with the **smallest SSD** or the **biggest CC**

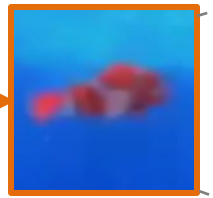
- **Strengths**:
  - Robustness
  - Simplicity in implementation

- **Weaknesses**:
  - Computations time-consuming for large search window
  - Only suitable for translation

Toy Example – Define Template



'Copy'



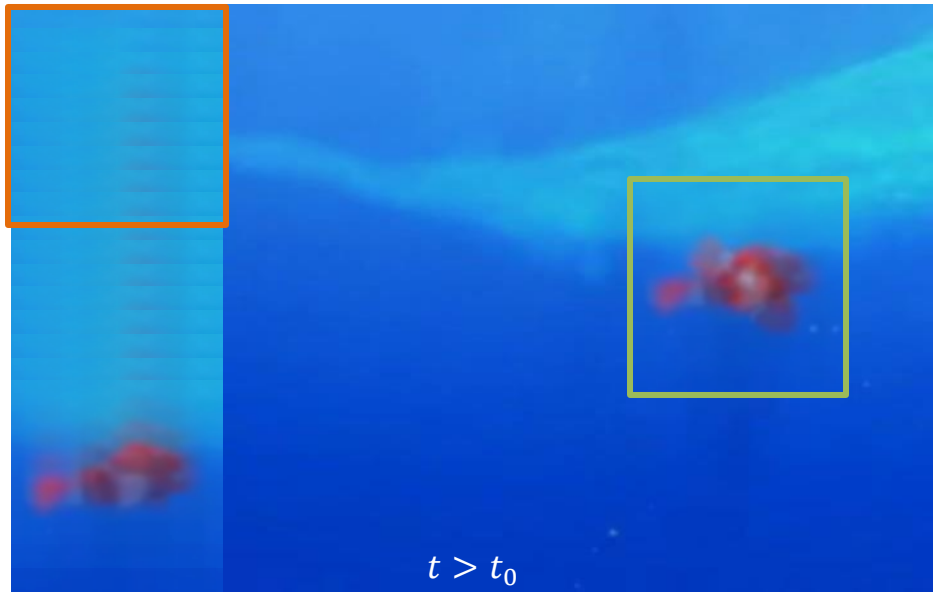
Template

Toy numbers:
- Upper water → values ~120
- Lower water → values ~085
- Fish → values ~220

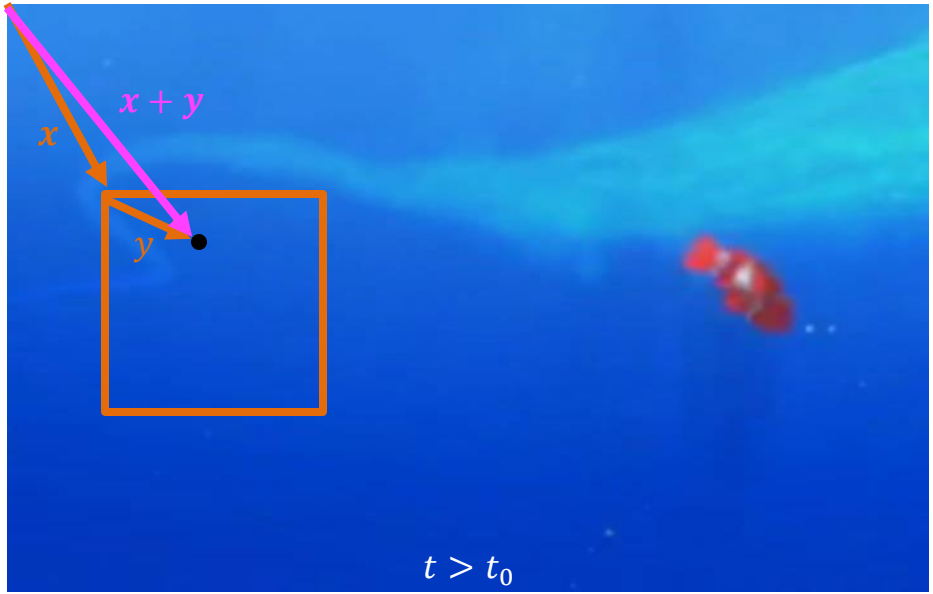
'Zoom in'

090	122	118	221	222	220	092
085	221	225	223	224	221	091
084	220	222	220	219	218	085
085	087	085	086	085	086	085
088	085	088	085	086	085	086
085	086	085	088	088	088	086
087	085	087	088	085	086	085
086	085	087	086	087	086	088

Toy Example – ‘Swipe’ Template over Image

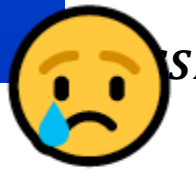


Toy Example – Similarity Metric



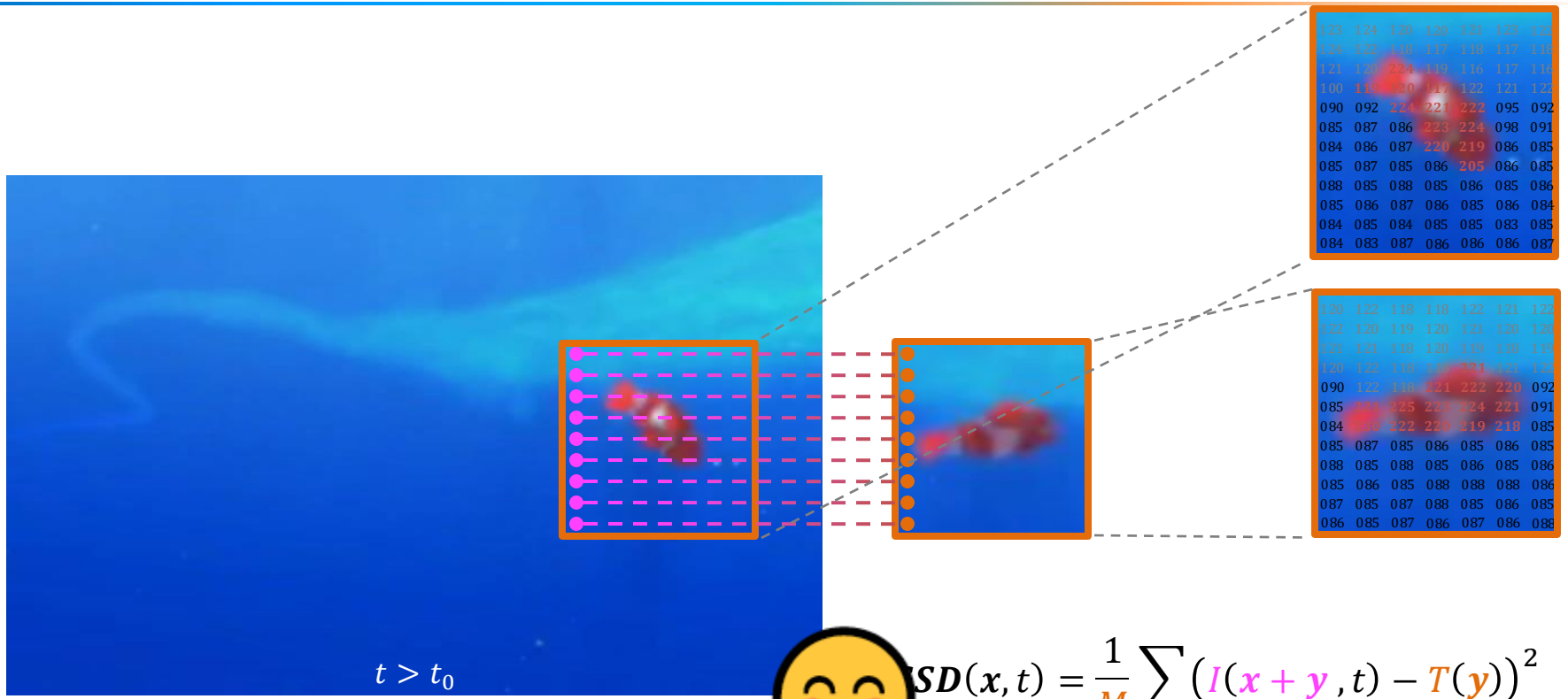
$$\boldsymbol{SSD}(\boldsymbol{x}, t) = \frac{1}{M} \sum_{\boldsymbol{y} \in \Omega} \left(I(\boldsymbol{x} + \boldsymbol{y}, t) - T(\boldsymbol{y}) \right)^2$$

Toy Example – Similarity Metric – Pixel-wise



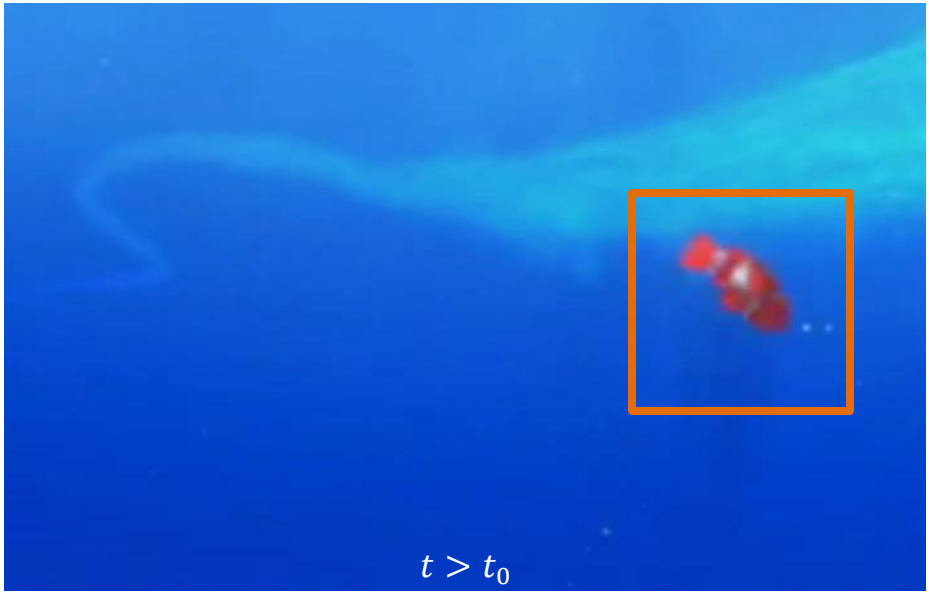
$$\boldsymbol{SD}(\boldsymbol{x},t) = \frac{1}{M}\sum_{\boldsymbol{y}\in\Omega}\left(I(\boldsymbol{x}+\boldsymbol{y},t) - T(\boldsymbol{y})\right)^2$$

Toy Example – Similarity Metric – Pixel-wise



$$SD(\boldsymbol{x}, t) = \frac{1}{M} \sum_{\boldsymbol{y} \in \Omega} \left(I(\boldsymbol{x} + \boldsymbol{y}, t) - T(\boldsymbol{y}) \right)^2$$

Toy Example – SSD



Small SSD == Small difference

$$x^{\bigstar}(t) = arg\min_{x} SSD(\boldsymbol{x}, t)$$

★ optimal value

Or **maximize** the **Correlation** metric

Toy Example – Street Smart

'Global' search works *well*
but is computationally *expensive*

If we know the motion history of the fish
(i.e. we have tracked the fish for a few frames) ...?

- ***Predict*** future motion &
- ***Restrict*** search in *'local'* area

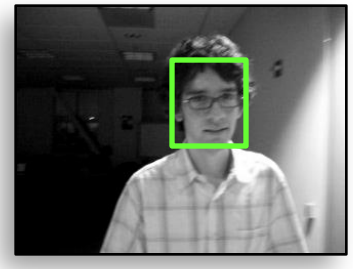
$t > t_0$

Outline

- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

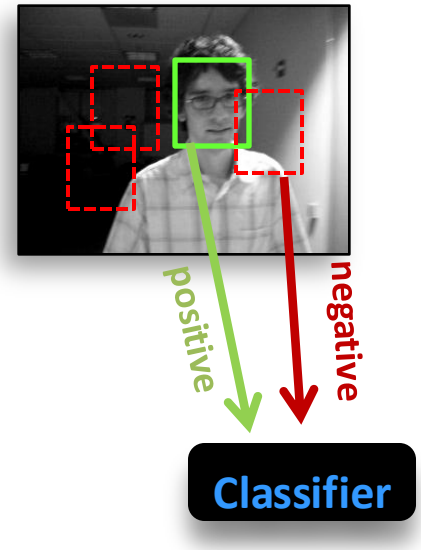
Tracking by Detection

First frame is labeled



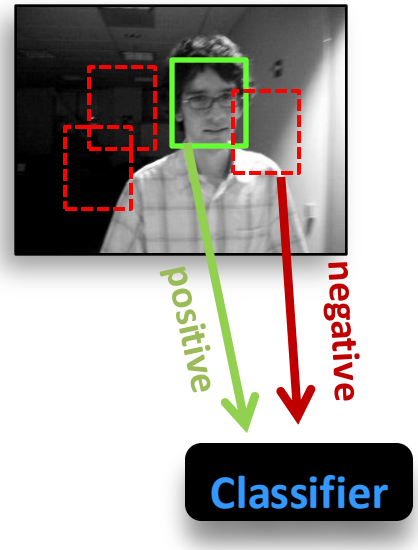
Tracking by Detection

Train/Update model w. 1 **positive** & some **negative** patch



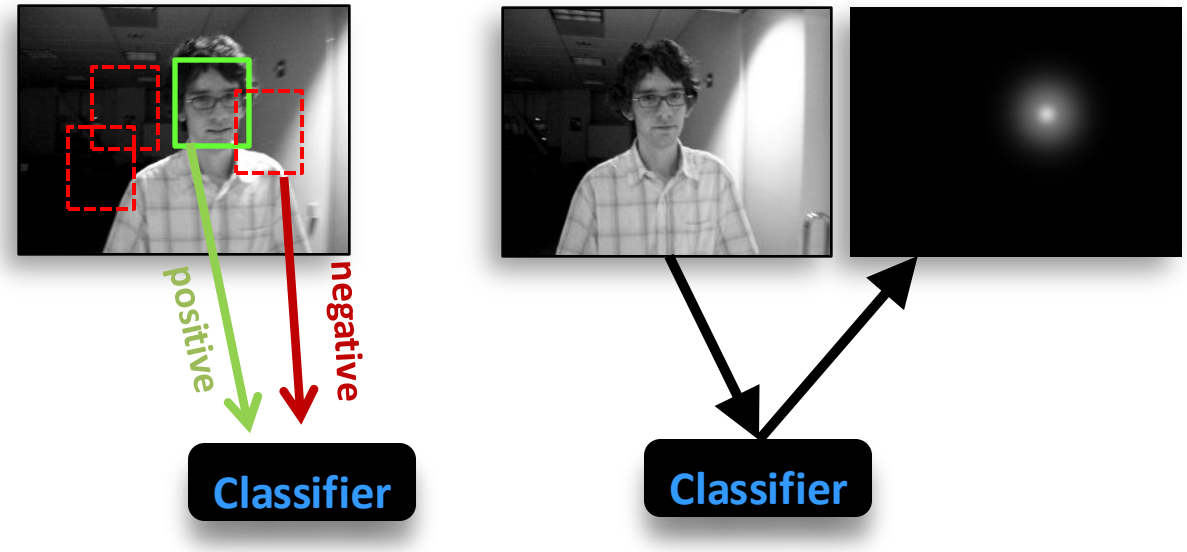
Tracking by Detection

Get next frame



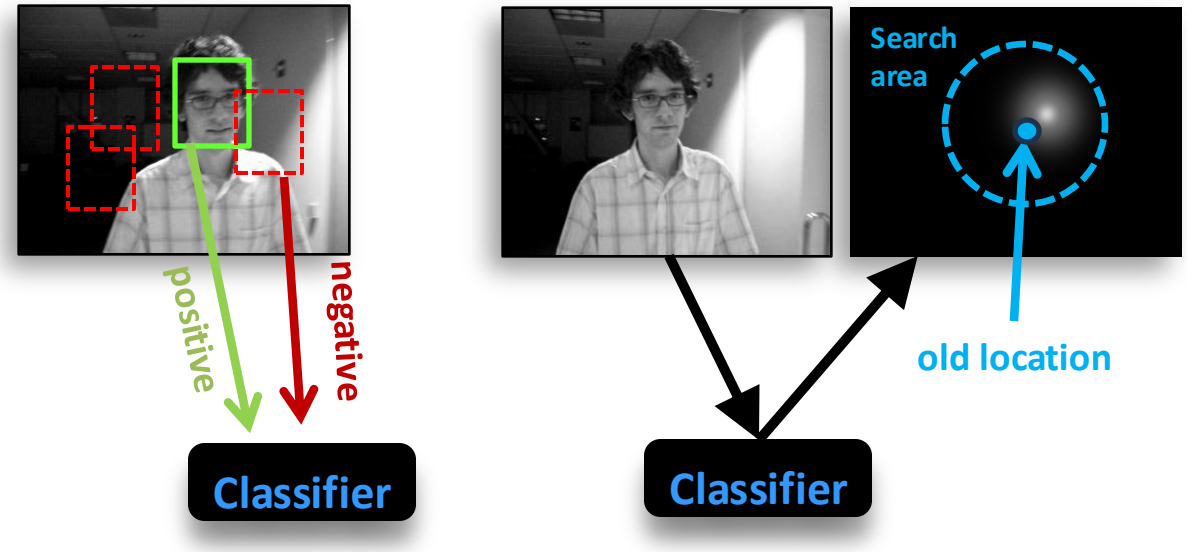
Tracking by Detection

Evaluate classifier in some search window



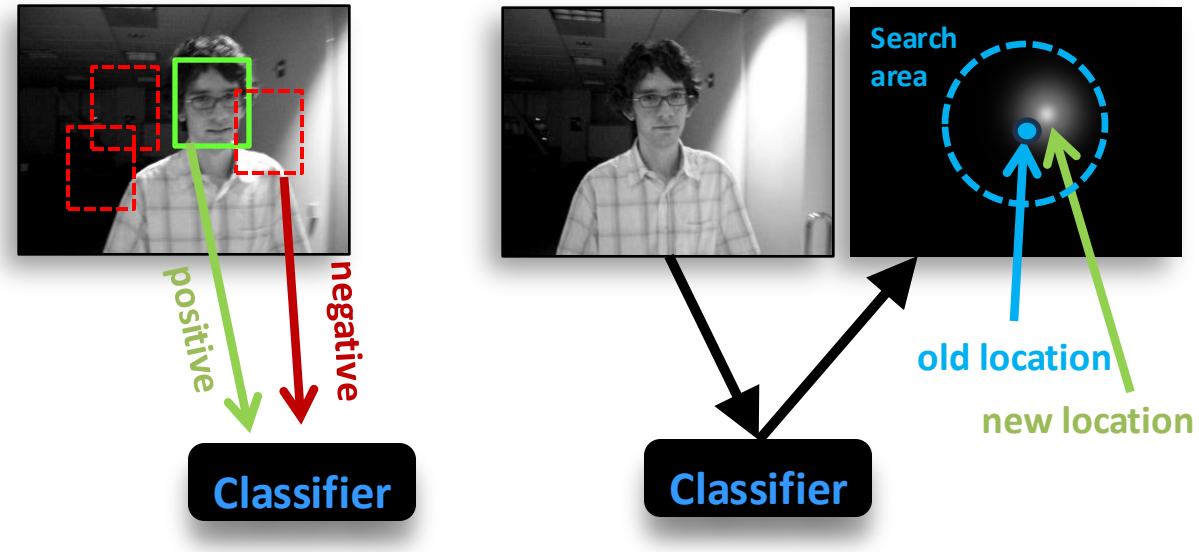
Tracking by Detection

Evaluate classifier in some search window



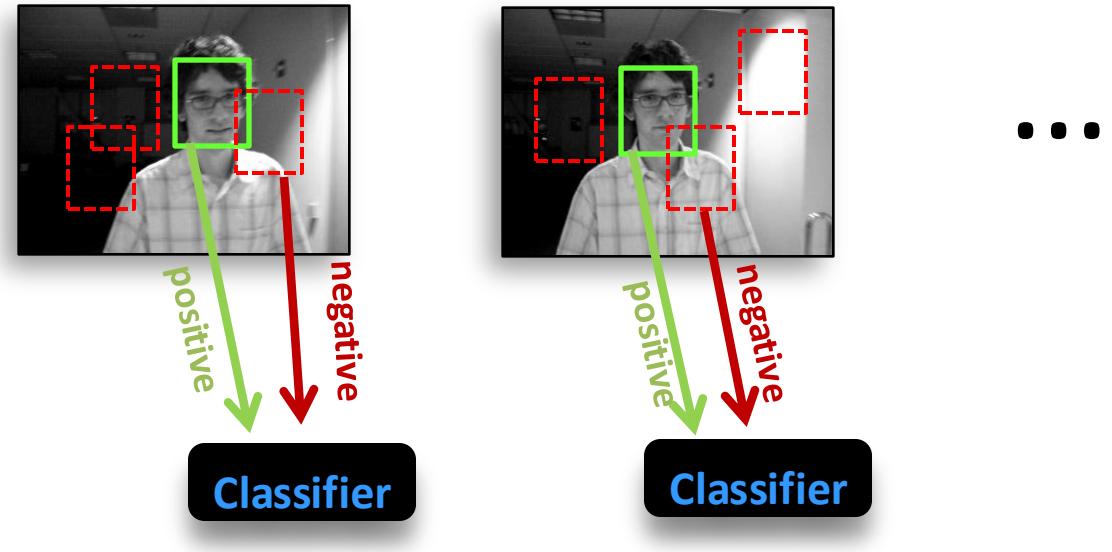
Tracking by Detection

Find max response



Tracking by Detection

… Repeat …

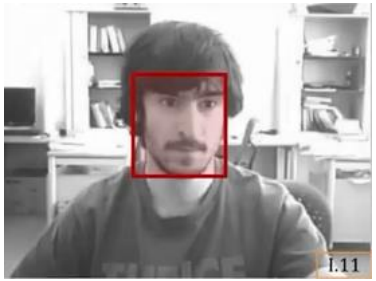


Tracking by Detection

Most tracking works:

- Focus on appearance model
- Borrow techniques from object detection (see: next week)
  - ‘Slide’ a discriminative classifier around image
- **Adaptive** appearance model

Tracking by Detection – SIFT

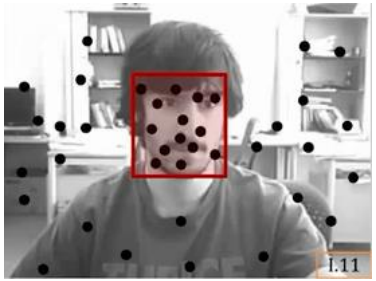


Frame at **t=0** with
Bounding Box

Slide by
S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT

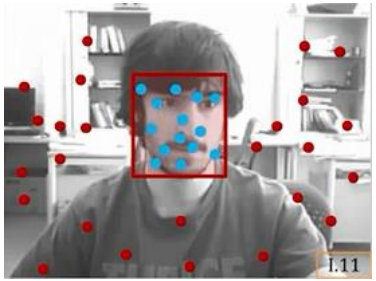


Frame at **t=0** with
Bounding Box
+ **SIFT** Features

Slide by
S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT



Frame at **t=0** with
Bounding Box
+ **SIFT** Features

Initialize

Background Model

Initialize

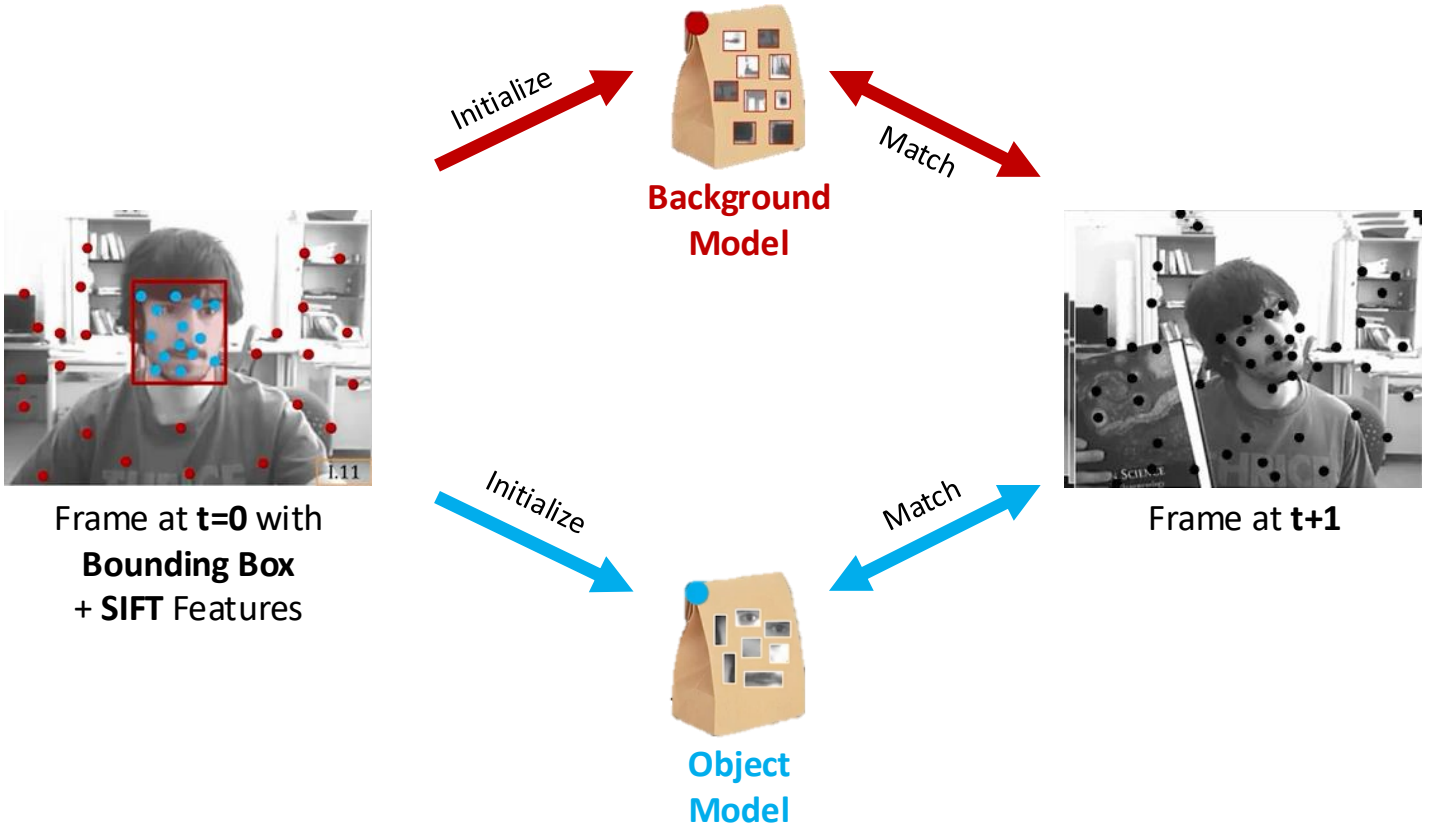
Object Model

Frame at **t+1**

Slide by
S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

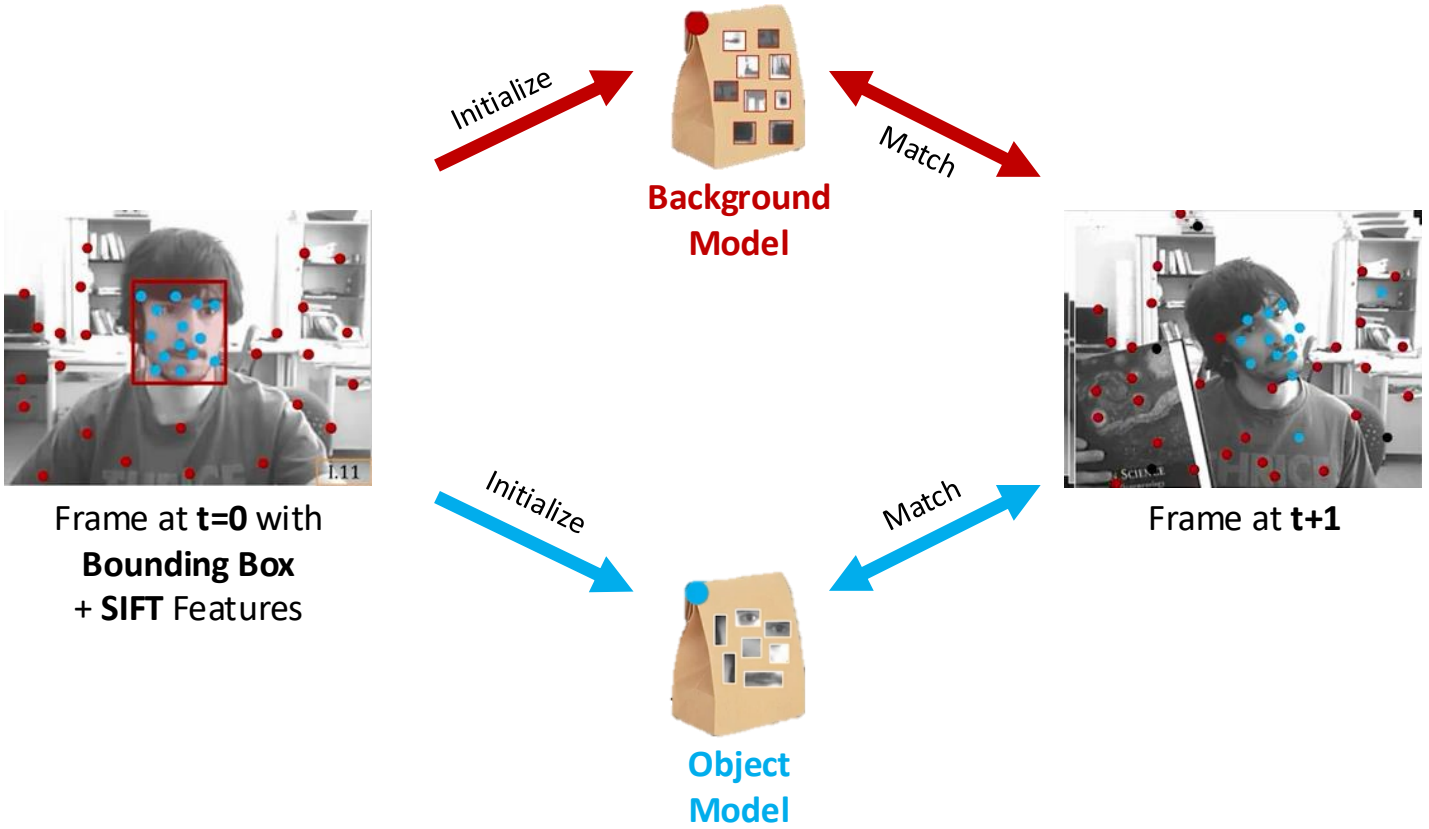
Tracking by Detection – SIFT



Slide by S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

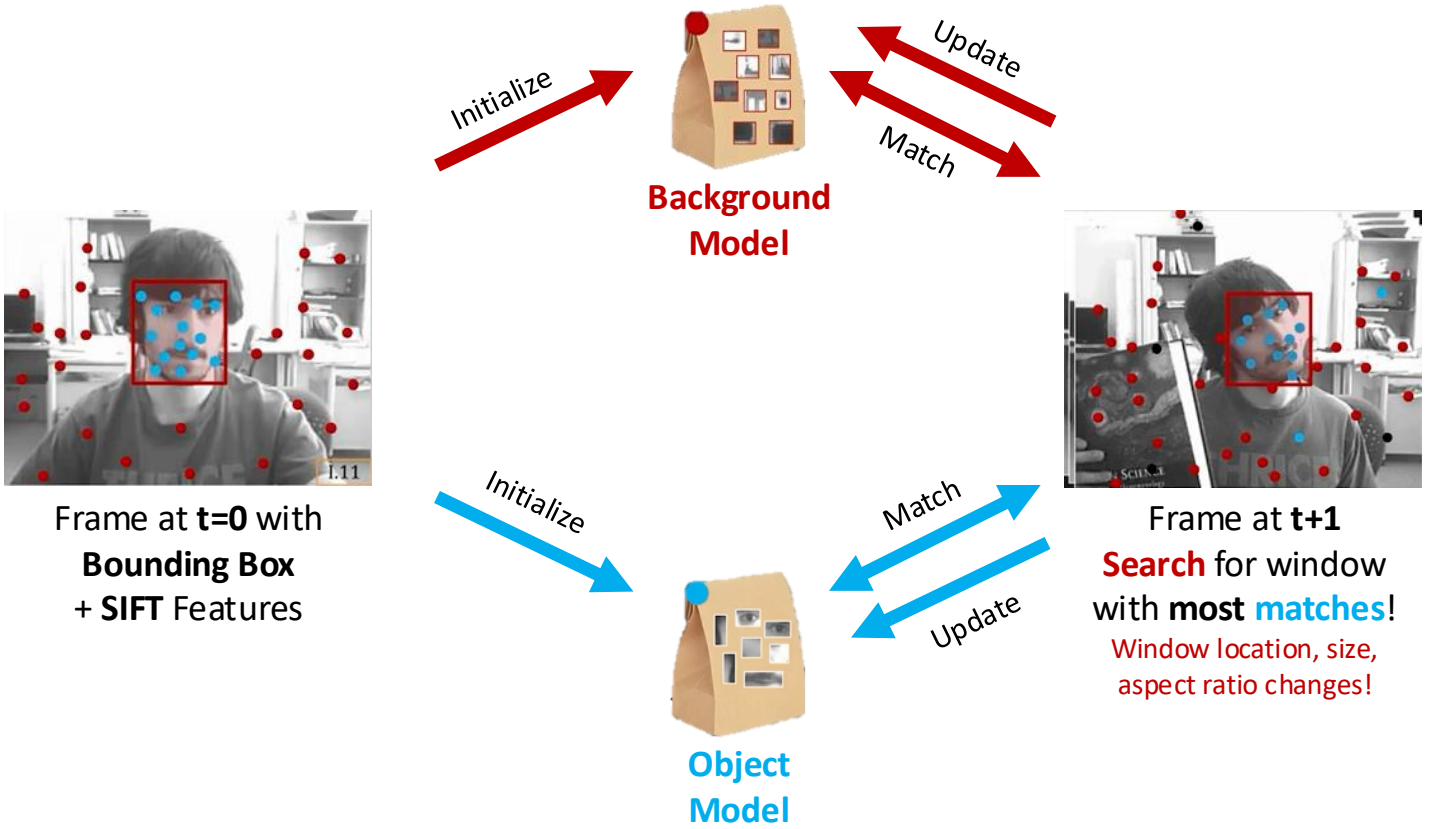
Tracking by Detection – SIFT



Slide by S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT



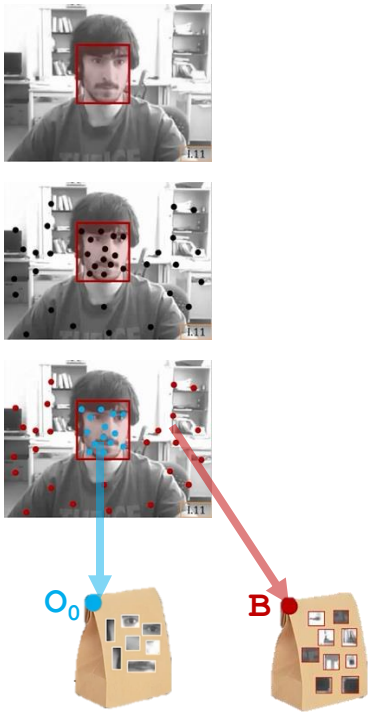
Slide by S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT

Frame at **t=0**

- User selects a Bounding Box $\mathbf{w}_1$ as object/target
- Compute SIFT features (or similar) for the frame
- Classify features in Box as Object & assign to set O_0
- Classify other features as BackGround & assign to **B**



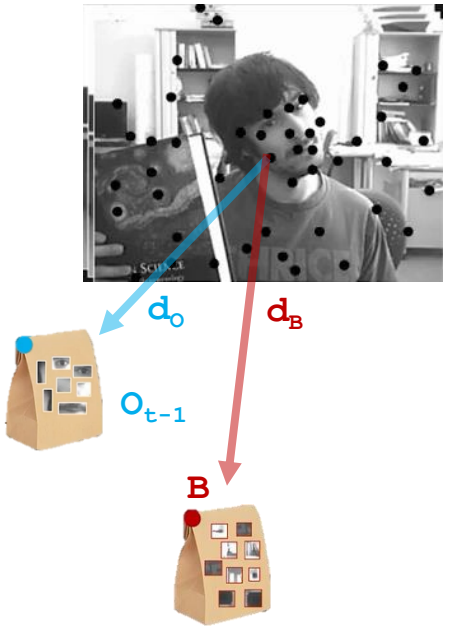
Slide by
S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT

Frame at $t>0$

- Compute SIFT features & descriptors $\{v_1, \ldots, v_k\}$
- For each feature & corresponding descriptor v_i
  - Compute distance d_O between v_i and the best match in object set O_{t-1}
  - Compute distance d_B between v_i and the best match in object set B
  - $C(v_i) = \begin{cases} +1 & \text{if } d_O/d_B < 0.5 \ (v_i \text{ belongs to object}) \\ -1 & \text{otherwise } (v_i \text{ does not belong to object}) \end{cases}$



Slide by S. Nayar

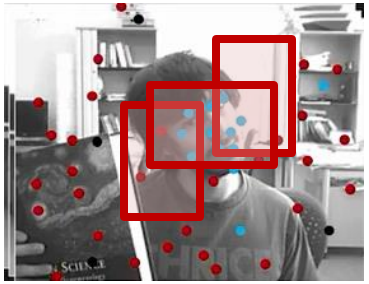
Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

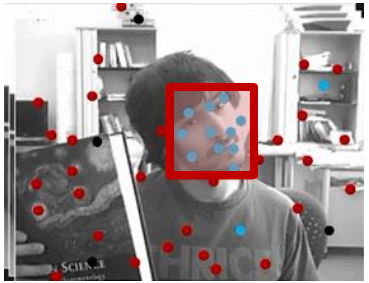
Continues at next slide

Tracking by Detection – SIFT

Continued from previous slide

- For each Search Box **W**
  - Compute $\varphi(W) = \Sigma C(v_i)$ for all features v_i in **W**
  - Compute a heuristic $\tau(W, W_{t-1})$ that **penalizes large deviations** from location, size, shape of W_{t-1}
  - Compute **Match Score**: $\mu(W) = \varphi(W) - \tau(W, W_{t-1})$
- Select Box W_t with **best Match Score** as new object location
- Update object appearance model:
  $O_t = O_{t-1} \cup \{Vi\}$ with all v_i in W_t with $C(v_i = +1)$



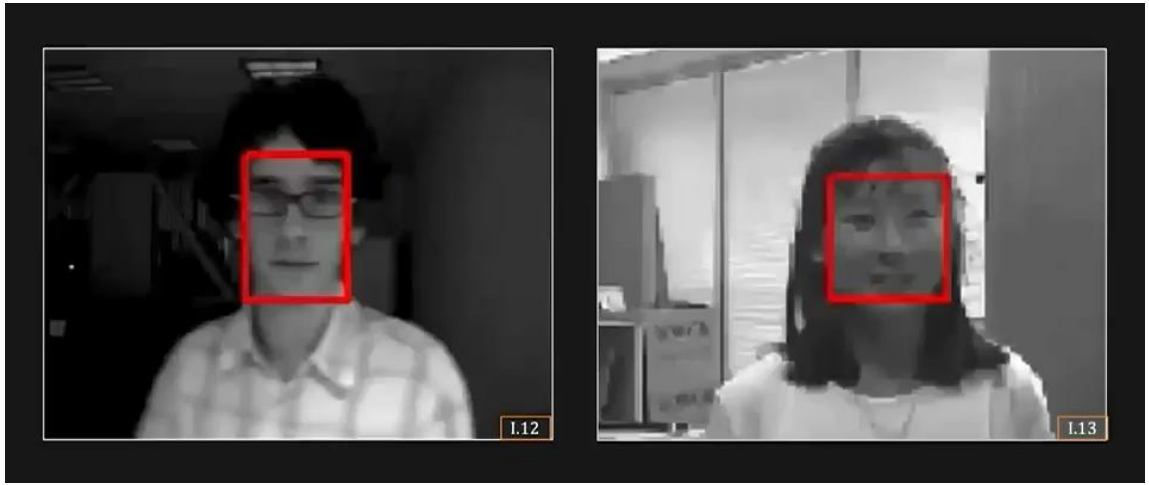


Slide by S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT

More robust to **occlusion**, **scale** & **orientation** changes than template-based tracking



Slide by S. Nayar

Gu et al., *'Efficient Visual Object Tracking with Online Nearest Neighbor Classifier'*, ACCV 2010

Tracking by Detection – SIFT

Tracking People in the Wild

Slide by
S. Nayar

Benfold and Reid, 'Stable Multi-Target Tracking in Real-Time Surveillance Video', CVPR 2011

Tracking by Detection – SIFT

Customer Behavior – In-store Analytics

Slide by
S. Nayar

https://abcnews.go.com/Business/stores-snooping-holiday-shoppers/story?id=21153815

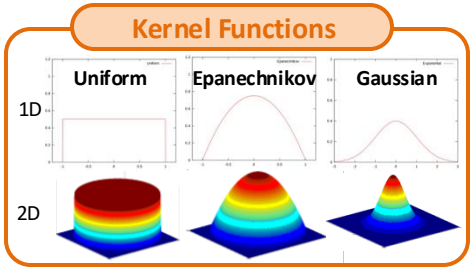
Outline

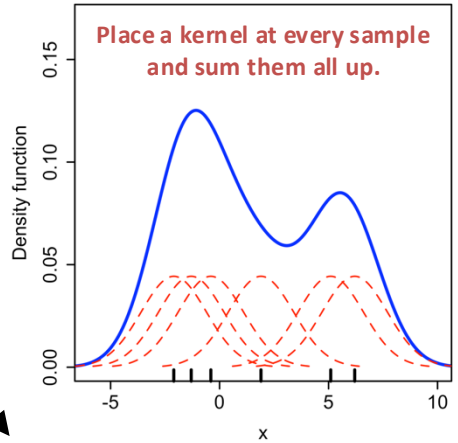
- Motion
- Optical Flow (OF)
  - Aperture Problem
  - Lukas-Kanade OF
  - Coarse-to-fine OF Estimation
  - Applications
- Tracking
  - Template Tracking
  - Tracking by Detection
  - Mean-shift Tracking

Detour: Kernel Density Estimation (KDE)

Idea: Estimate a probability distribution of a given point set.
Compose the distribution as a sum of n basis/kernel functions K.

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

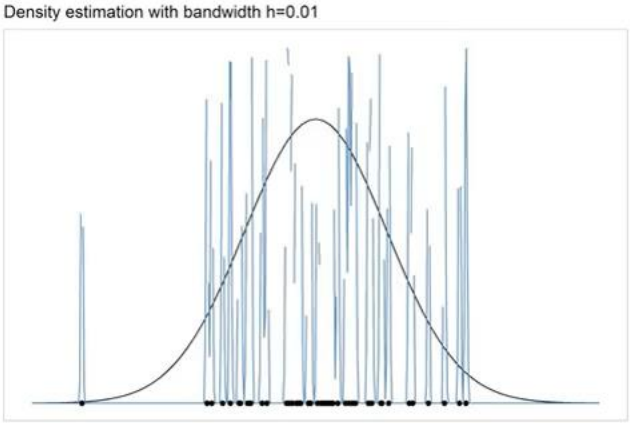


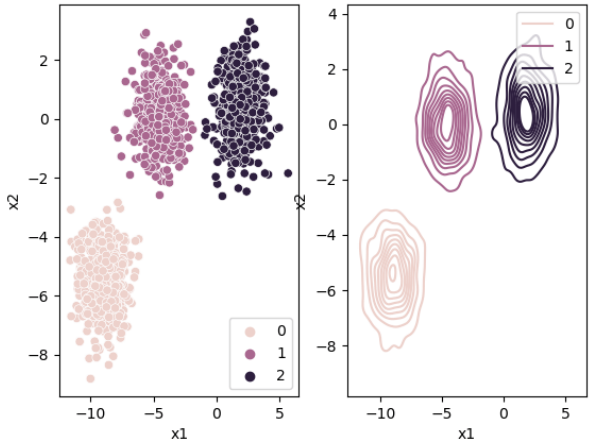


Bandwidth h:

Width of kernel
(sigma for Gaussian)

Defines level of detail
and number of modes.

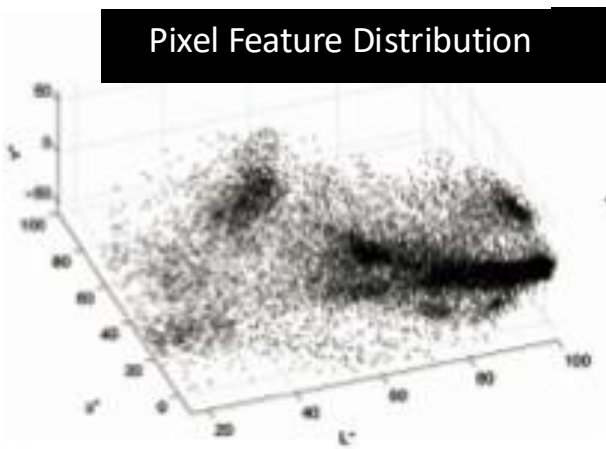




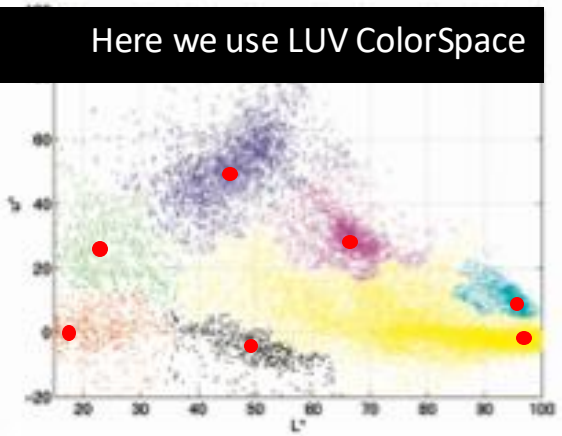
Mean Shift – Segmentation / Clustering

Input image

Pixel Feature Distribution



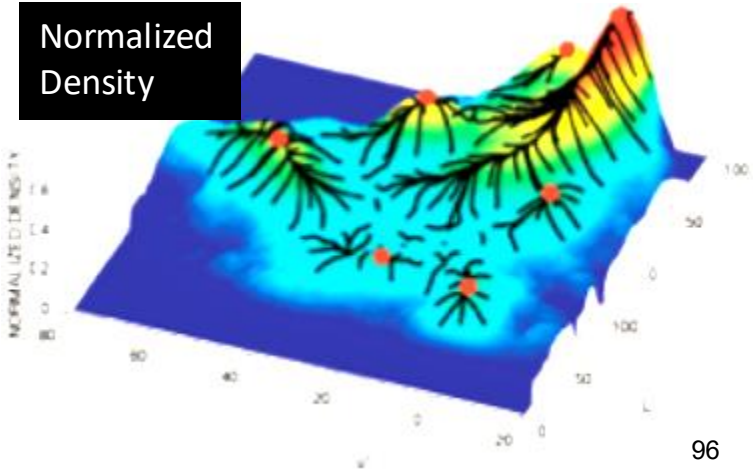
Here we use LUV ColorSpace



Idea: Density-based clustering

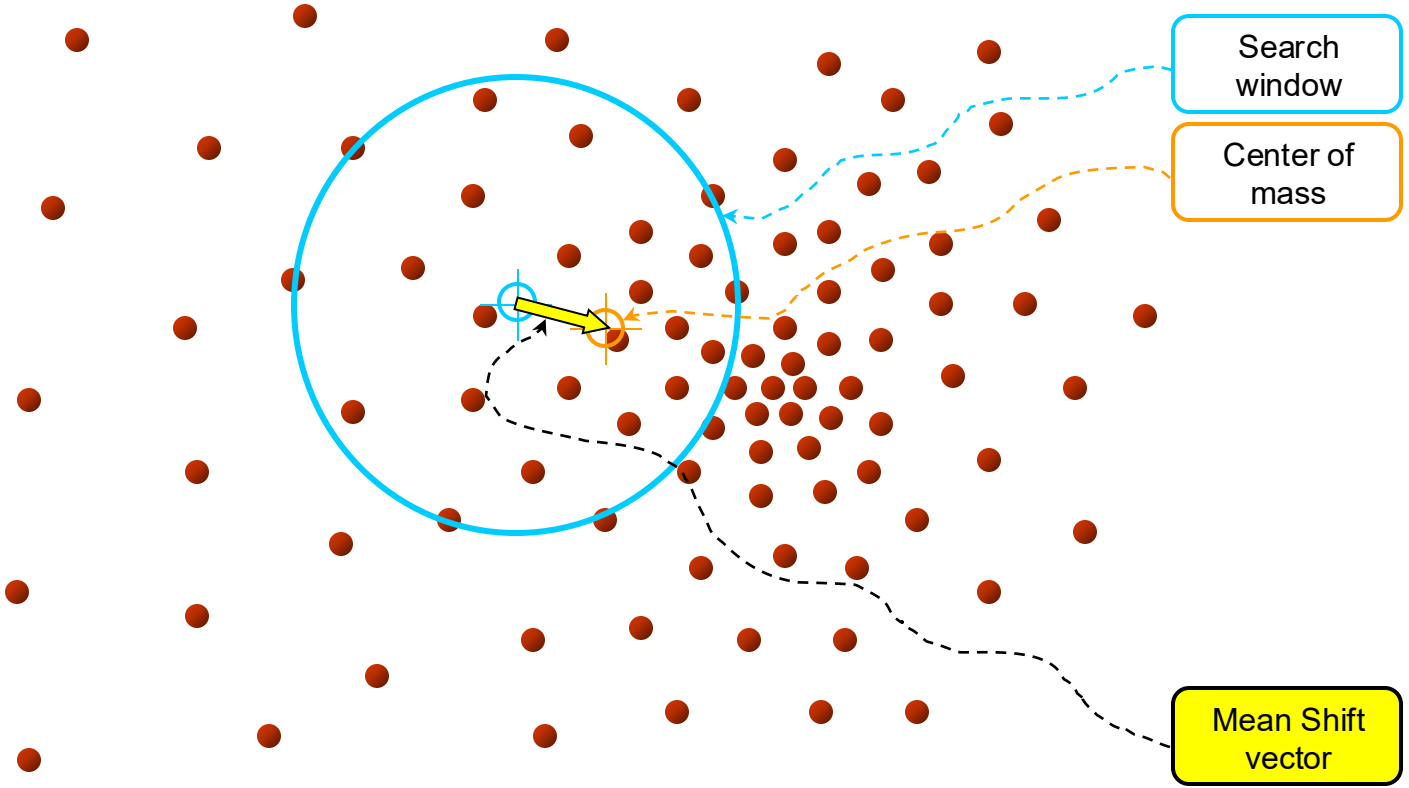
Goal: Find all modes of a given probability distribution

Normalized Density



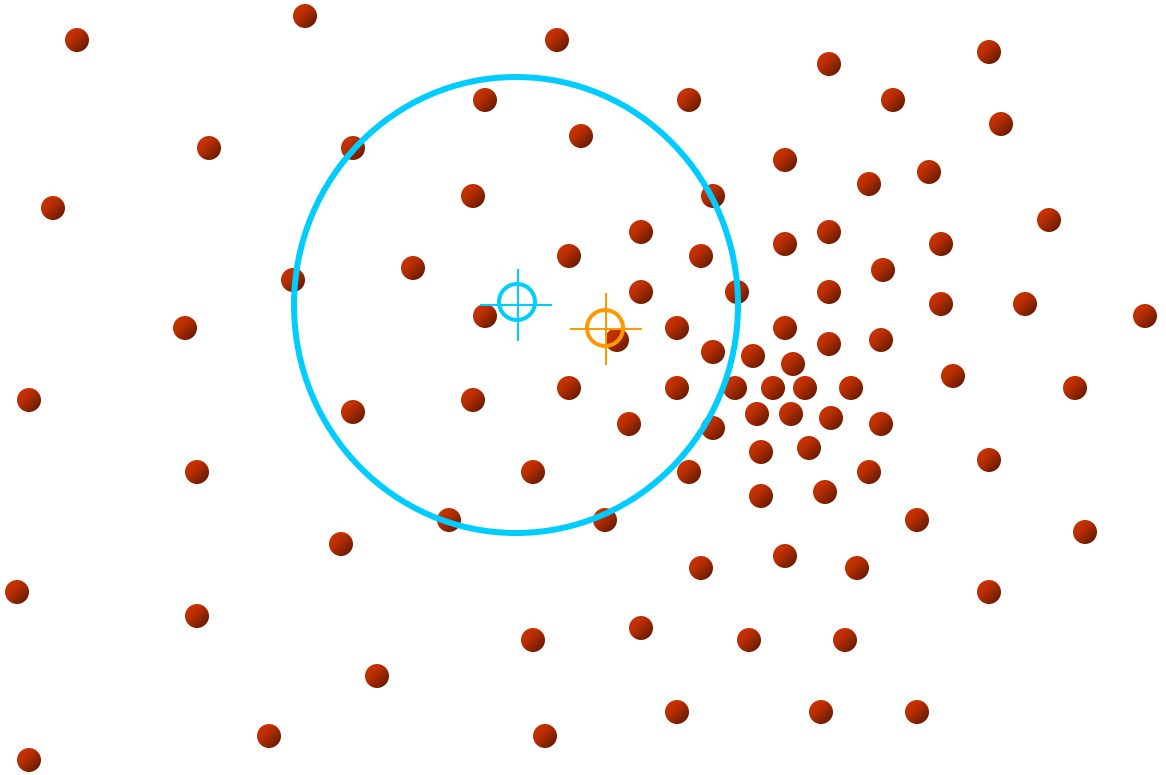
[Comaniciu, Meer. Mean shift: A robust approach toward feature space analysis, TPAMI, 2002]

Mean Shift Algorithm – 2D Toy Example



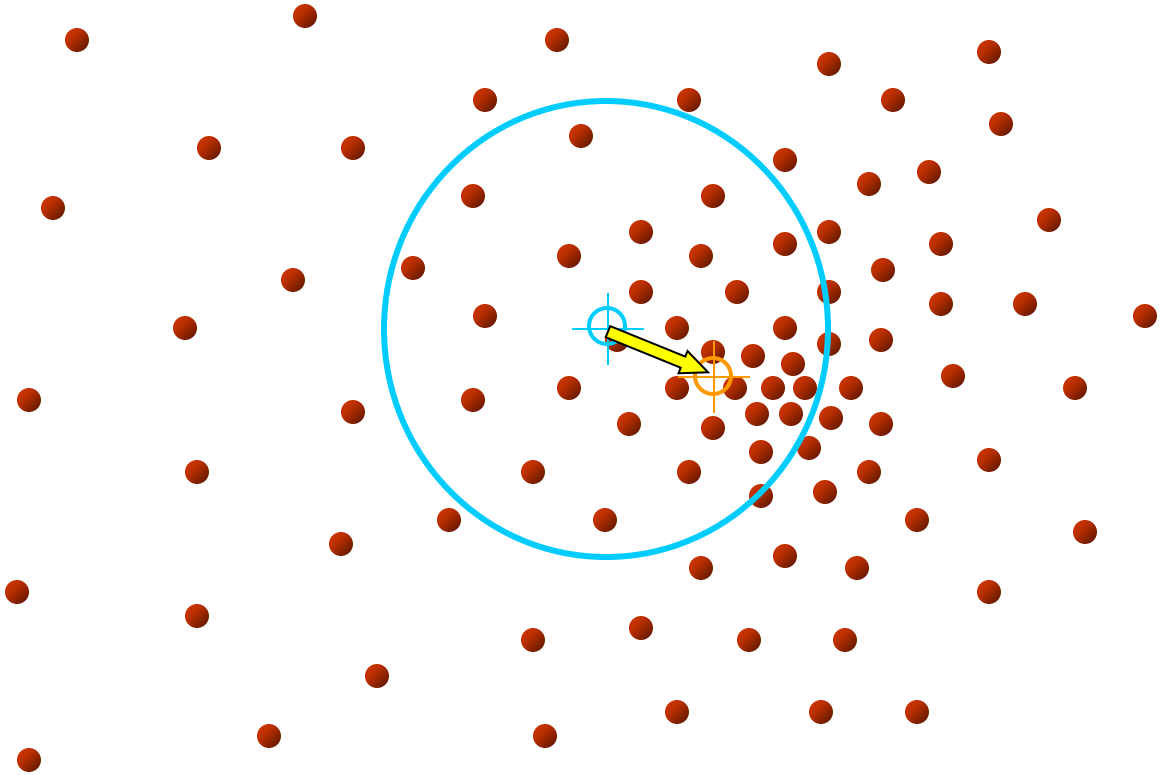
[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm – 2D Toy Example



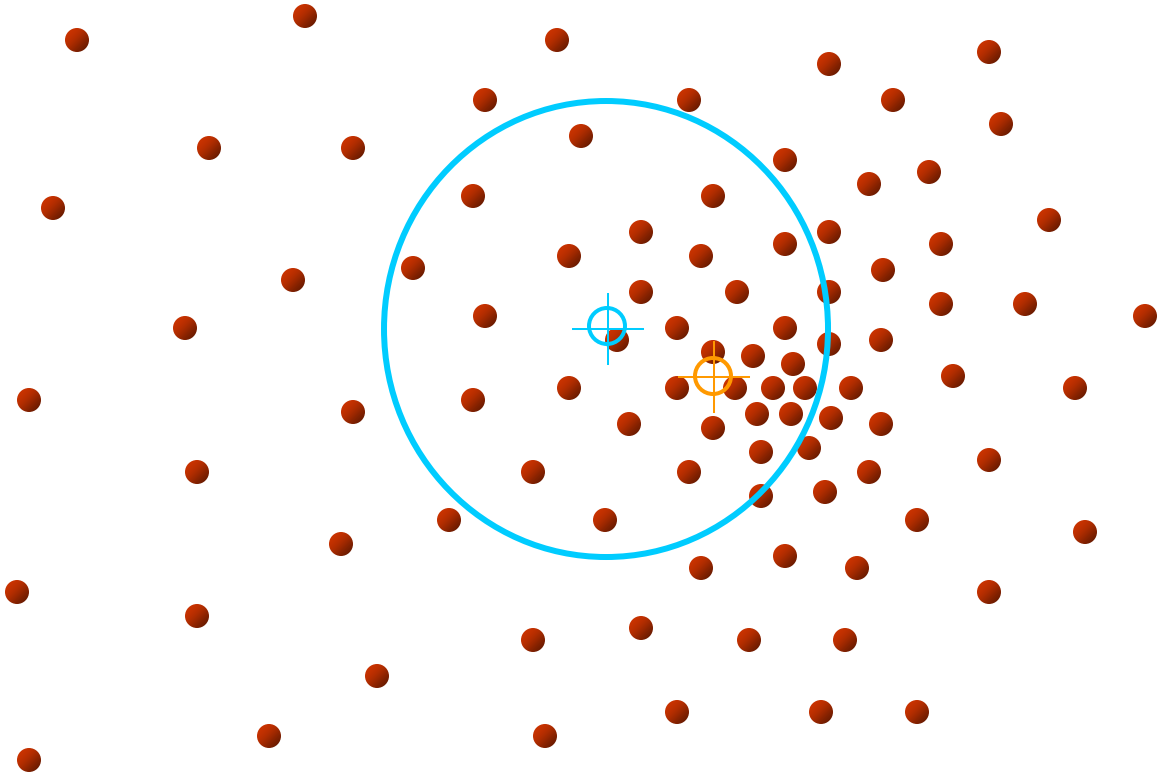
[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm – 2D Toy Example



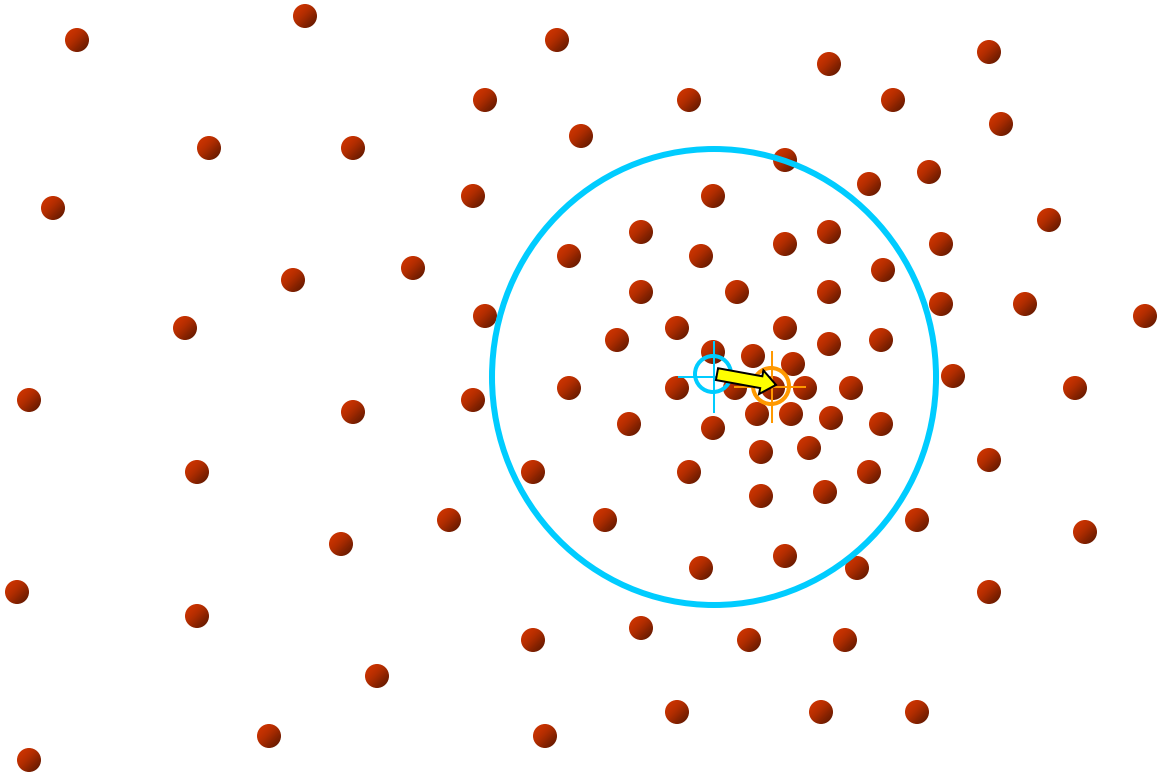
[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm – 2D Toy Example



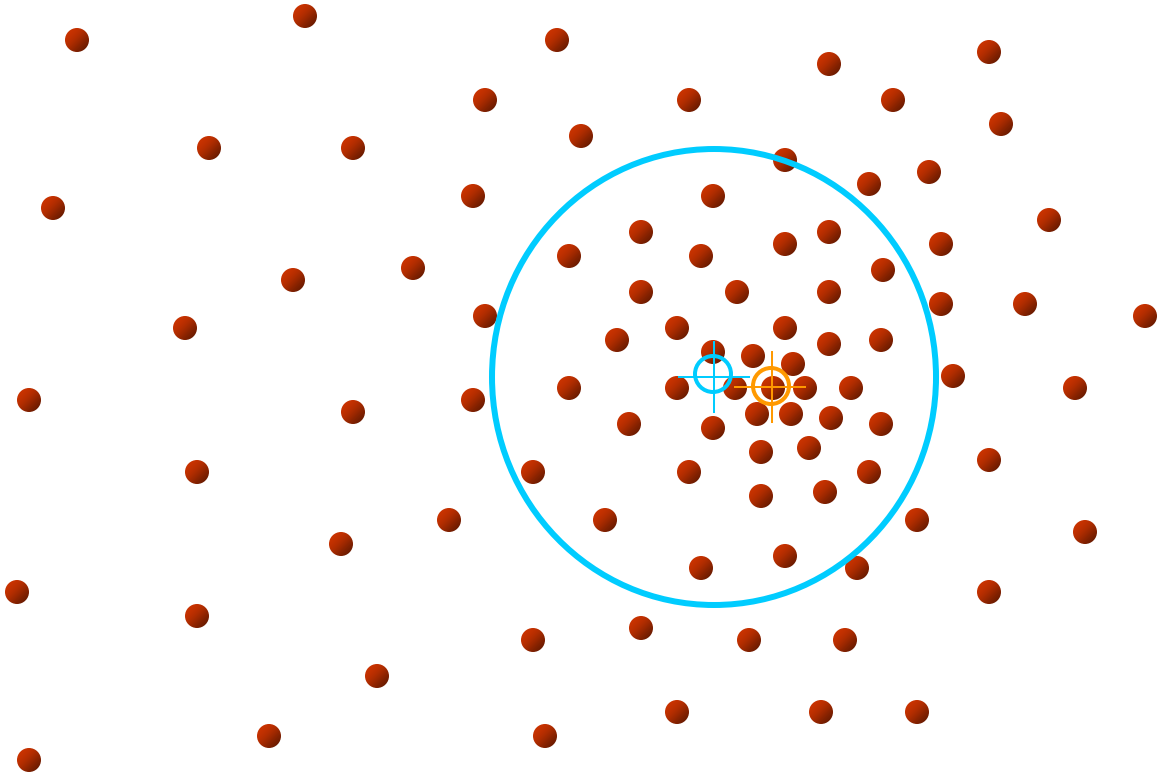
[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm – 2D Toy Example



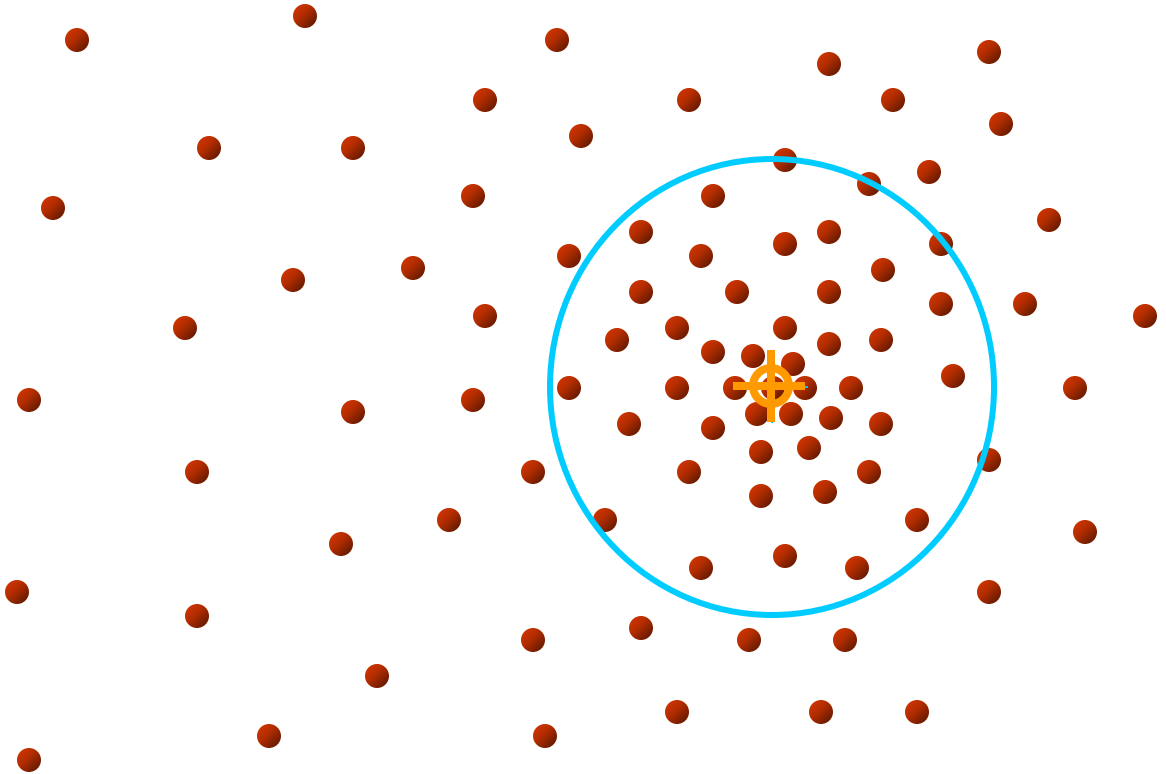
[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm – 2D Toy Example



[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm – 2D Toy Example

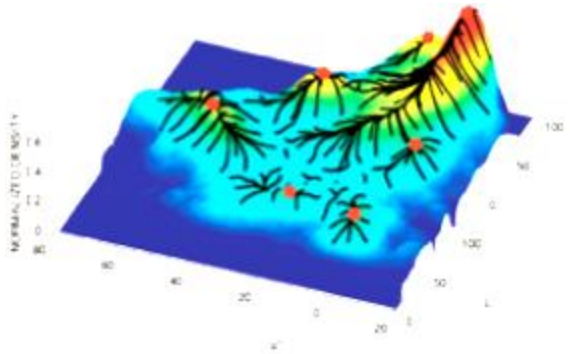


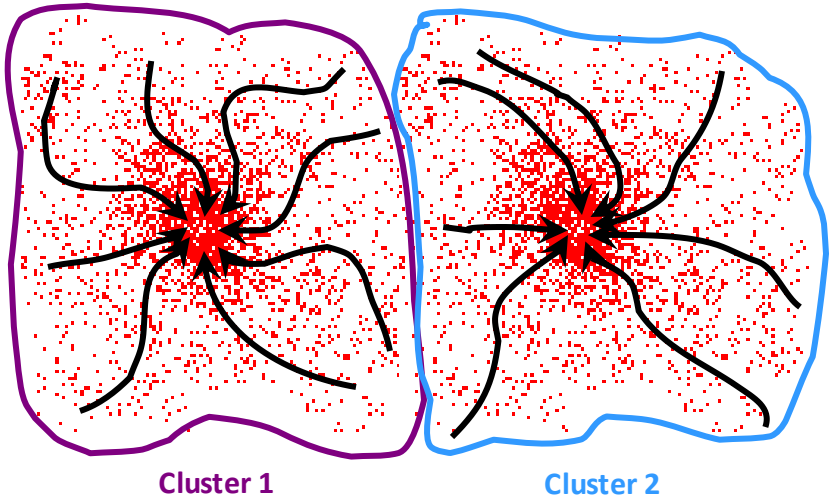
[Slide credit: Y. Ukrainitz & B. Sarel]

Mean Shift Algorithm

The mean shift algorithm seeks the **modes** of the given point set

1. Choose kernel and bandwidth
2. For each point:
   a) Center a window with a kernel on that point
   b) Compute the mean of the data in the search window
   c) Center the search window at the new mean location
   d) Repeat (b,c) until convergence
3. Assign points that lead to nearby modes to the same cluster



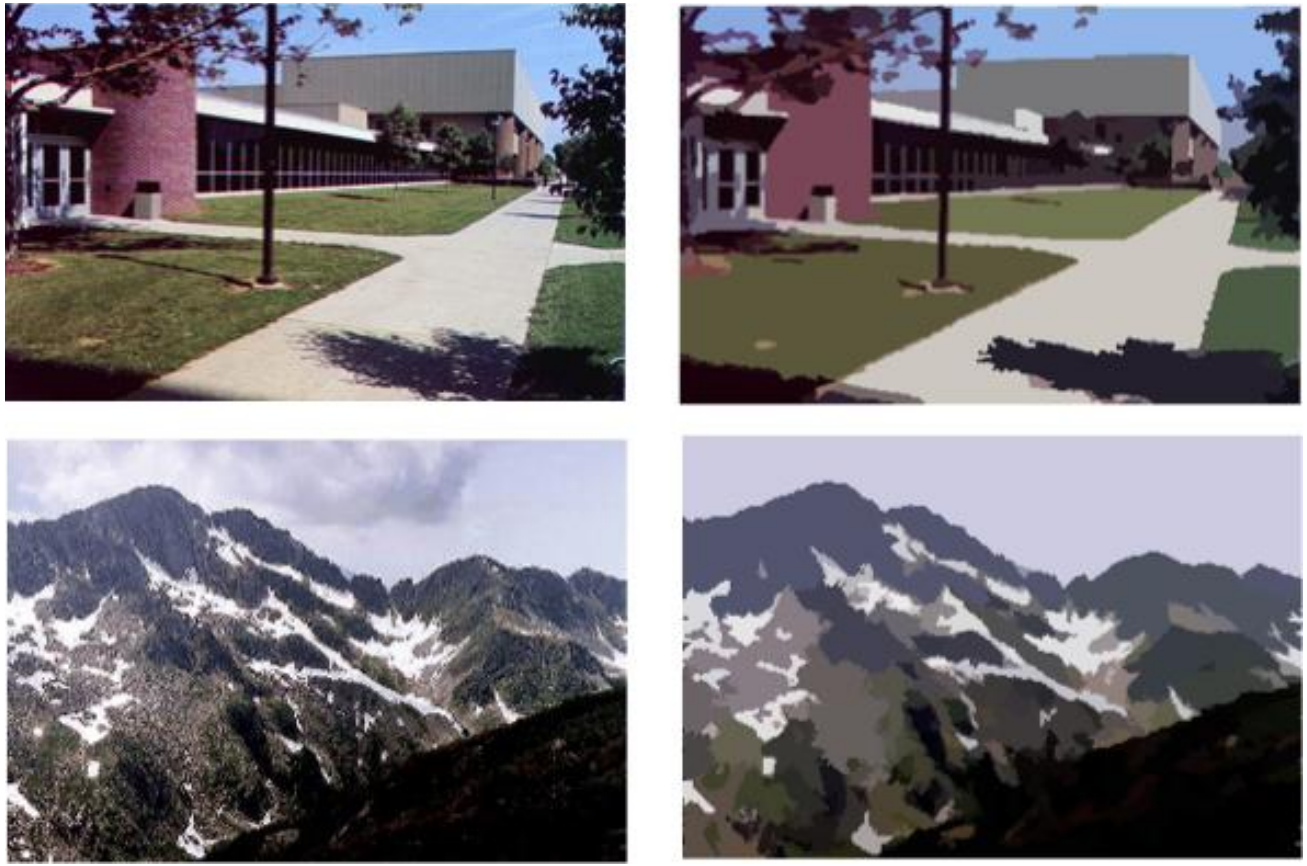


[Figure from: Y. Ukrainitz & B. Sarel]

- **Attraction basin:** Region for which all trajectories lead to the same mode
- **Cluster:** All points in the attraction basin of a mode
- **Note:** Bandwith and kernel type controls the size of attraction basins and the number of modes; and thus eventually the number of found clusters.

Mean Shift Segmentation

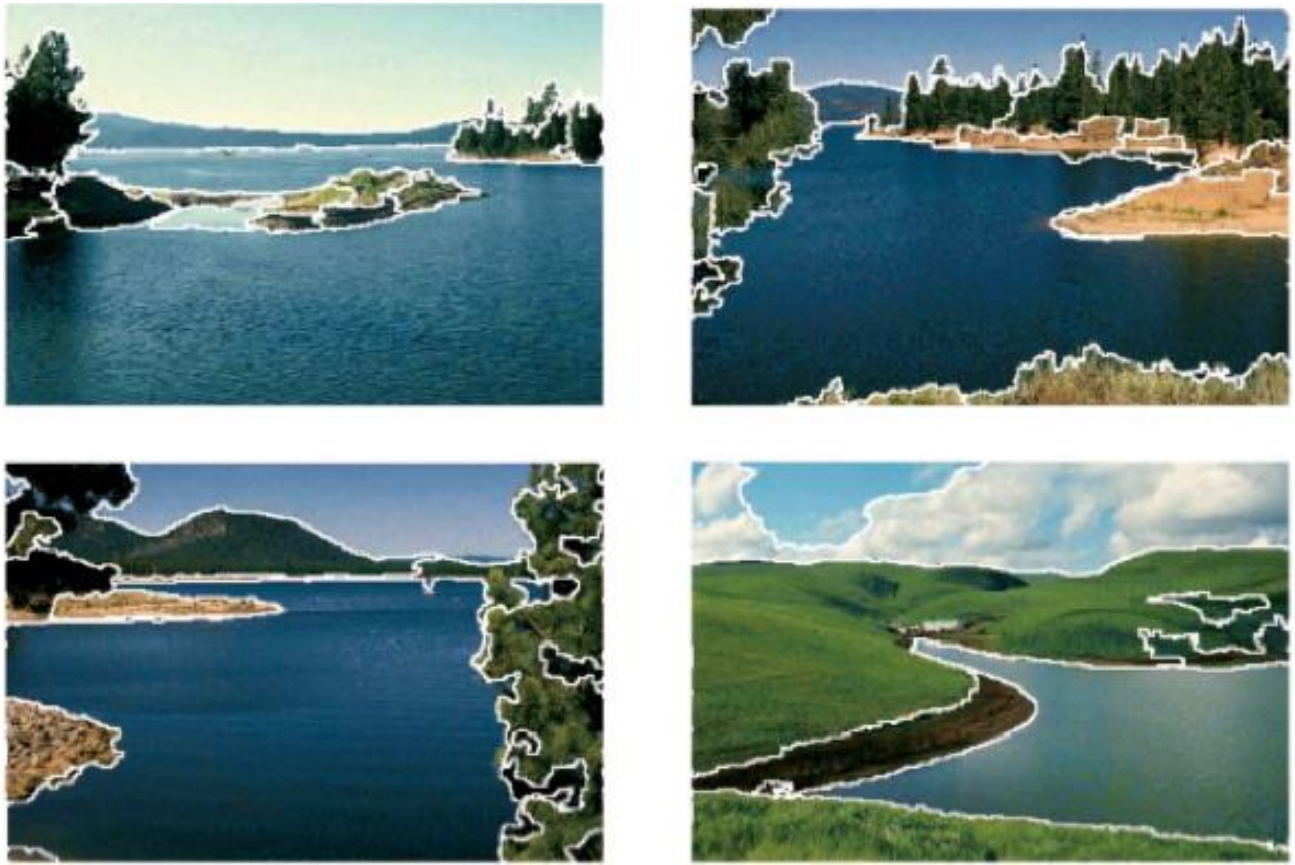
Results of clustering in LUV color space.



[Comaniciu, Meer. Mean shift: A robust approach toward feature space analysis, TPAMI, 2002]

Mean Shift Segmentation

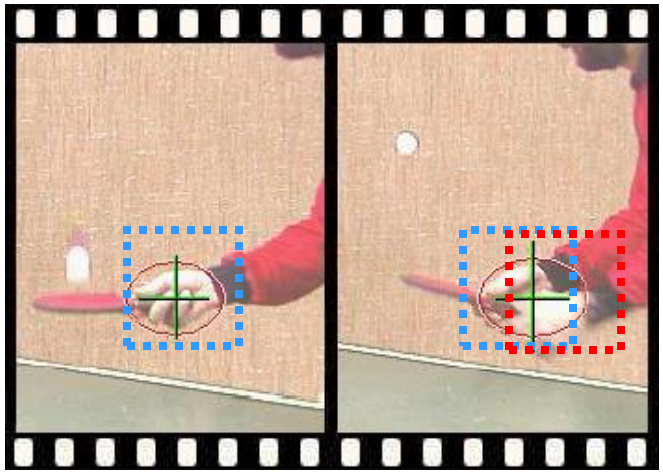
Results of clustering in LUV color space.



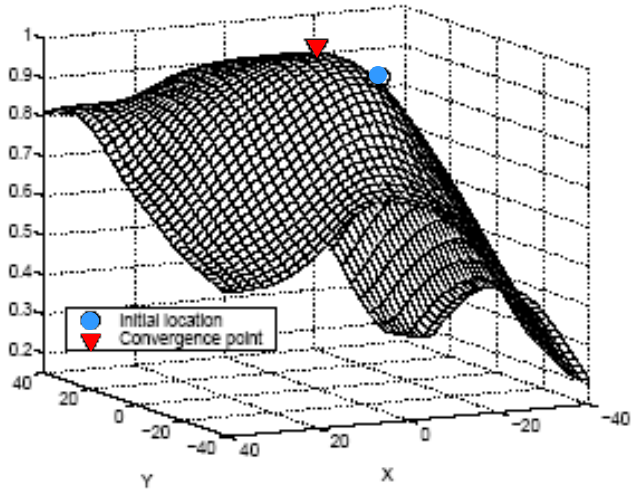
[Comaniciu, Meer. Mean shift: A robust approach toward feature space analysis, TPAMI, 2002]

Mean Shift Tracking

- Define a **target region**: $\vec{q} = (q_1, \dots, q_m)$ with **color distribution** from m pixels
- Next frame: Search vicinity with **candidate region**: $\vec{p}(y) = (p_1(y), \dots, p_m(y))$ at position y
- With a **similarity function** $f[\vec{p}(y), \vec{q}]$ for **comparing histograms** (e.g. cosine similarity), **mean shift** finds the next best position of the target candidate region



target region: $\vec{q}$ candidate region: $\vec{p}(y)$



similarity function $f[\vec{p}(y), \vec{q}]$

In every time step:

- Use mean shift update
- to find **best location** y
- by locally maximizing the region **similarity function** starting from the previous frame position.

Mean Shift Tracking Results

Closing Remarks

Action Points

- This week:
  - Finish Lab-2 assignment
  - Start Lab-3 assignment – Release tomorrow

    Disclaimer: might include next-Tuesday's material

Disclaimer

Many of the slides used here are obtained from online resources (including many open lecture materials) without appropriate acknowledgement. They are used here for the sole purpose of classroom teaching. All the credit and all the copyrights belong to the original authors. You should not copy it, redistribute it, put it online, or use it for any other purposes than for this course.