Visual Tracking I:
Monocular 2D point tracking

Kim Steenstrup Pedersen

Plan for today

- The monocular visual 2D tracking problem
  - Definition of the problem
  - Examples
  - Applications

- The Kanade-Lucas-Tomasi (KLT) feature tracker
  - The basic algorithm
  - Extension: Good features to track

Monocular visual 2D tracking

- Monocular = single camera

- Visual 2D tracking:
  - To follow objects apparent 2 dimensional motion in a video sequence.
  - At every frame, locate objects and associate with objects currently being tracked.
  - One object is “easy” – for multiple objects we need to keep track of the identity of objects.

- Similar to optical flow (dense), but we are interested in motion of regions of the image (sparse).

Example: 2D tracking of object

Results from Hauberg-Lauze-Pedersen, JMIV 2012

Challenges in visual tracking
(Inspired by David J. Fleet)

- Choices to make:
  - What to model / estimate: Shape (2D/3D), appearance, object dynamics.
  - What to measure: Feature points, optical flow, color histograms, edges, etc.

- Some of the main challenges are:
  - Objects with many degrees of freedom, affecting shape, appearance, and motion.
  - Occlusion and large scale changes.
  - Multiple objects and background clutter.

Examples of applications of tracking

- Surveillance (parking lot, airports, shops, etc.)
- Target tracking (military and civil)
- Automated car driver assistance (avoid other cars and pedestrians, stay on the road)
- Device-less human computer interaction (e.g. simple 2D gesture recognition)
- And many more …

Approaches to 2D visual tracking

- Tracking by visual feature matching:
  - Represent an object by a collection of image features.
  - Tracking is performed by matching features between frames.
- Region / blob tracking:
  - Represent and track the appearance of the interior of the object
- Contour tracking:
  - Represent and track the outline contour of the object

Example: 2D tracking of object (region based)

Results from Hauberg-Lauze-Pedersen, JMIV 2012

Examples of a contour tracker

Isard & Blake., “CONDENSATION” IJCV ’98
http://www.robots.ox.ac.uk/~misard/condensation.html

Common steps in tracking

- Target representation and localization:
  - Choose an appropriate representation for the target (e.g. position and scale, 2D/3D graphics model, …)
  - Localization: Estimate the parameters in the target representation.
- Filtering:
  - Noise and numerical instabilities introduce drift in the tracker.
  - This can be handled introducing temporal filtering, such as Bayesian stochastic filtering.
- Data association:
  - How do we make sure that we are tracking the same object in all frames.
  - Especially important for multi-target tracking.

The filtering problem

Results from Hauberg-Lauze-Pedersen, JMIV 2012

Data association is a hard problem: Keeping track of the player IDs

Okuma et al., “Boosted Particle Filter.” ECCV ‘04

Tracking by matching

Kanade-Lucas-Tomasi (KLT) tracker

- Basic Idea:
  - Track features between frames essentially using the Lucas-Kanade optic flow algorithm.
  - Only select good features to track and drop features when they become unstable.
- Video sequence: $I(x, y, t)$
- Local image model: $J(\mathbf{x}+\mathbf{d}) = I(\mathbf{x}) + n(\mathbf{x})$
  where $J(\mathbf{x}+\mathbf{d}) = I(x+\xi, y+\eta, t+\tau)$ is the image at $t+\tau$,
  $I(\mathbf{x}) = I(x, y, t)$ is the image at time t,
  the displacement $\mathbf{d} = (\xi, \eta)^T$, and $n(\mathbf{x})$ represent noise.
- We want to find displacement **d** that minimize the residue (matching error) on a patch window *W*:

$$\varepsilon = \int_W \left(J(\mathbf{x}+\mathbf{d}) - I(\mathbf{x})\right)^2 w(\mathbf{x}) d\mathbf{x}$$

KLT: Solving for the image displacement

- Linearization of image by truncated Taylor series

$$J(\mathbf{x}+\mathbf{d}) \approx J(\mathbf{x}) + \mathbf{g}^T\mathbf{d}$$

where $J(\mathbf{x}) = J(x, y, t+\tau)$, $\mathbf{g} = \left(\frac{\partial J}{\partial x}, \frac{\partial J}{\partial y}\right)^T$, $\mathbf{d} = (\xi, \eta)^T$

- Plug into residue expression

$$\varepsilon = \int_W \left(J(\mathbf{x}) - I(\mathbf{x}) + \mathbf{g}^T\mathbf{d}\right)^2 w(\mathbf{x})d\mathbf{x}$$

- A closed form solution exist by differentiation wrt. $\mathbf{d} = (\xi, \eta)^T$

$$\frac{\partial \varepsilon}{\partial \mathbf{d}} = 2\int_W \left(J(\mathbf{x}) - I(\mathbf{x}) + \mathbf{g}^T\mathbf{d}\right)\mathbf{g}w(\mathbf{x})d\mathbf{x} = 0$$

Use chain rule and

$$\frac{d\mathbf{g}^T\mathbf{d}}{d\mathbf{d}} = \mathbf{g}$$

(continues on next slide)

KLT: Solving for the image displacement

$$\frac{d\varepsilon}{d\mathbf{d}} = 2\int_W \left(J(\mathbf{x}) - I(\mathbf{x}) + \mathbf{g}^T\mathbf{d}\right)\mathbf{g}w(\mathbf{x})d\mathbf{x} = 0 \;\Rightarrow$$

$$\left(\int_W \left(\mathbf{g}\mathbf{g}^T\right)w(\mathbf{x})d\mathbf{x}\right)\mathbf{d} = -\int_W \left(J(\mathbf{x}) - I(\mathbf{x})\right)\mathbf{g}w(\mathbf{x})d\mathbf{x}$$

- Use that

$$\left(\mathbf{g}^T\mathbf{d}\right)\mathbf{g} = \left(\mathbf{g}\mathbf{g}^T\right)\mathbf{d}$$

- This is just a system of linear equations $\mathbf{G}\mathbf{d} = \mathbf{e}$ with solution $\mathbf{d} = \mathbf{G}^{-1}\mathbf{e}$
- 2×2 "Harris" matrix aka structure tensor

$$\mathbf{G} = \int_W \left(\mathbf{g}\mathbf{g}^T\right)w(\mathbf{x})d\mathbf{x} \approx \sum_{\mathbf{x}\in W} \begin{bmatrix} g_x^2(\mathbf{x}) & g_x(\mathbf{x})g_y(\mathbf{x}) \\ g_x(\mathbf{x})g_y(\mathbf{x}) & g_y^2(\mathbf{x}) \end{bmatrix} w(\mathbf{x})$$

- Residue projected on gradient

$$\mathbf{e} = \int_W \left(I(\mathbf{x}) - J(\mathbf{x})\right)\mathbf{g}w(\mathbf{x})d\mathbf{x} \approx \sum_{\mathbf{x}\in W} \begin{bmatrix} \left(I(\mathbf{x}) - J(\mathbf{x})\right)g_x(\mathbf{x})w(\mathbf{x}) \\ \left(I(\mathbf{x}) - J(\mathbf{x})\right)g_y(\mathbf{x})w(\mathbf{x}) \end{bmatrix}$$

KLT: Select features for tracking

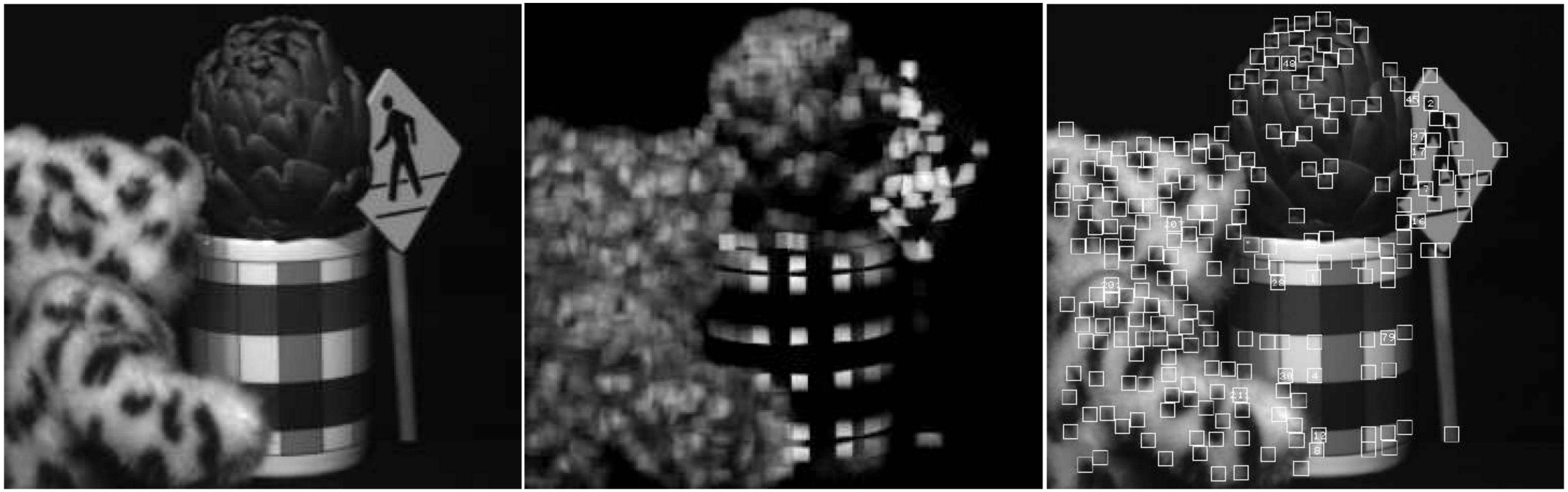
- Choose a fixed patch size
- We want to only keep patches which provide good displacement estimates – check eigenvalues of **G**:
  - If both eigenvalue are small – nearly flat patch (not good)
  - If one eigenvalue large and one small – edge (not good)
  - Both eigenvalues sufficiently large – corner or texture (good)
- We keep a patch / feature, if

  $$\min\left(\lambda_1, \lambda_2\right) > \lambda$$

  where λ_1, λ_2 are the eigenvalues of **G** and λ a threshold.
- Select non-overlapping patches in image passing the eigenvalue test.

KLT: Example of feature candidates

Patch size = 15 x 15 pixel, eigenvalues threshold = 10



Tomasi & Kanade, Tech. report CMU ‘91

The KLT algorithm

1. Find new non-overlapping feature patch candidates
   - Check feature stability, keep if $\min(\lambda_1, \lambda_2) > \lambda$
   - Store intensity patch from detection frame and current position of feature patch.
2. For each feature patch
   - (Check feature stability, keep if $\min(\lambda_1, \lambda_2) > \lambda$)
   - Estimate displacement by $\mathbf{d}_{new} = \mathbf{G}^{-1}\mathbf{e}$
   - Update feature position: $\mathbf{x}_{new} = \mathbf{x}_{old} + \mathbf{d}_{new}$
3. Loop back to 1. and process next frame.

The KLT algorithm

1. Find new non-overlapping feature patch candidates
   – Check feature stability, keep if $\min(\lambda_1, \lambda_2) > \lambda$
   – Store intensity patch from detection frame and current position of feature patch.
2. For each feature patch
   – (Check feature stability, keep if $\min(\lambda_1, \lambda_2) > \lambda$)
   – Iterate until convergence in residue $\varepsilon = \int_W \left(J(\mathbf{x}+\mathbf{d}) - I(\mathbf{x})\right)^2 w(\mathbf{x}) d\mathbf{x}$
     • Estimate displacement by $\mathbf{d}_{new} = \mathbf{G}^{-1}\mathbf{e}$
     • Resample patch $J(\mathbf{x}+\mathbf{d}_{new})$ with sub-pixel precision by bilinear interpolation.
   – If estimate do not converge – drop feature
   – Update feature position: $\mathbf{x}_{new} = \mathbf{x}_{old} + \mathbf{d}_{new}$
3. Loop back to 1. and process next frame.

Aside: Resampling the patch with sub-pixel precision

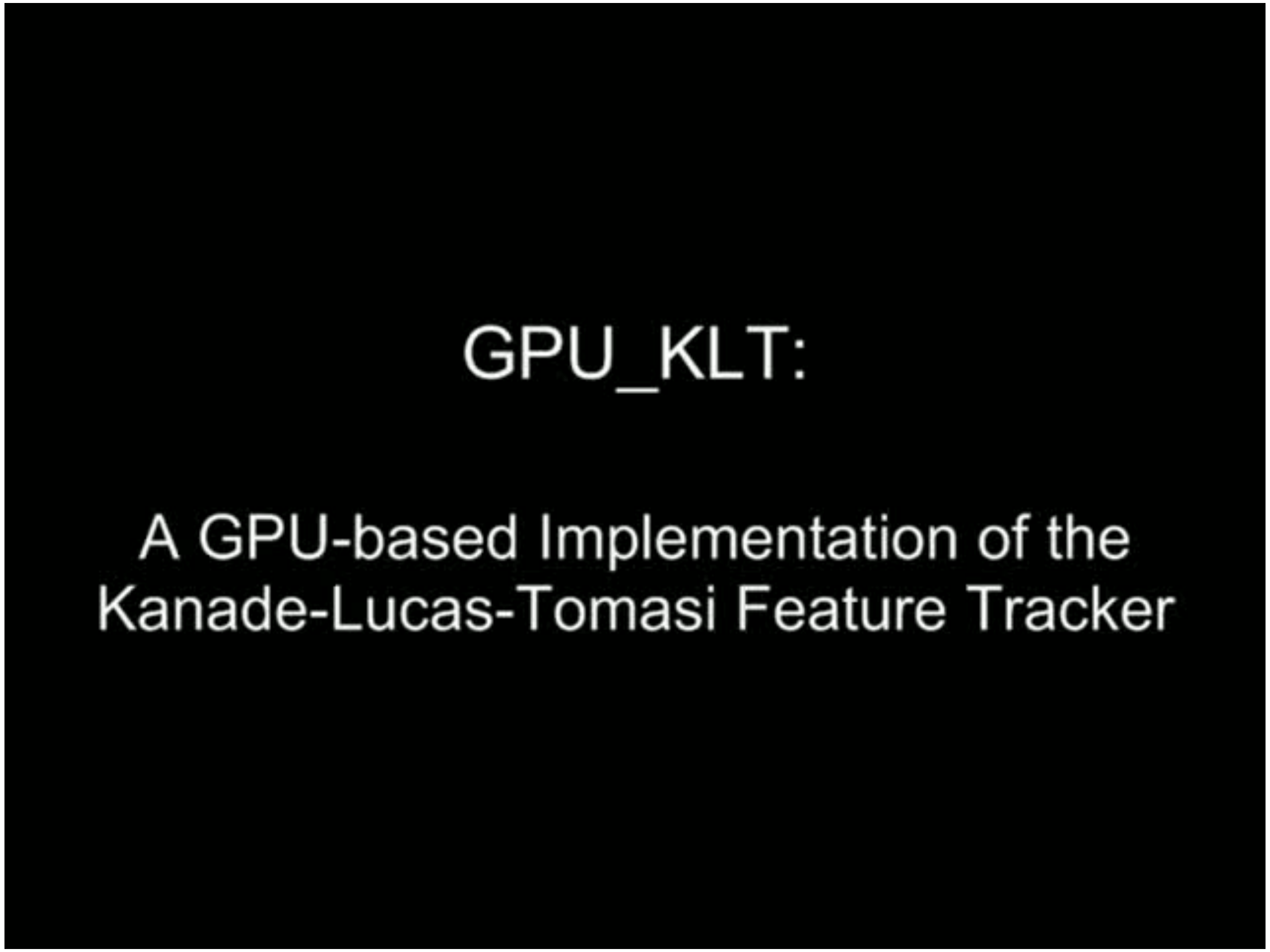
- Bilinear interpolation:
  See e.g. http://en.wikipedia.org/wiki/Bilinear_interpolation

- Resample patch from *J* at $J(\mathbf{x}+\mathbf{d}_{new})$ by bilinear interpolation:

  Make a drawing on whiteboard

- Use the resampled patch to compute $\mathbf{G}$, $\mathbf{e}$ and residual ε

Kanade-Lucas-Tomasi (KLT) tracker



Sinha et al. “GPU_KLT: A GPU-based Implementation of the KLT Feature Tracker” 2006
http://cs.unc.edu/~ssinha/Research/GPU_KLT/

KLT: Assumptions

- Linear displacement:
  - Patches must be planar or near planar.
  - High curvature will introduce deformation of the patch violating the linear assumption.
  - As tracking time increases, patches deform in a non-linear fashion or start to straddle occlusion boundaries.
- Patch size:
  - We assume that one patch size fits all – not really correct.
  - Too small patches leads to noisy displacement estimates.
  - Too large patches leads to problems with occlusion boundaries.

Extension: Good features to track

- In order to detect occlusion boundaries, introduce an affine deformation model and a dissimilarity measure.
- Dissimilarity between first patch and current patch:

$$\varepsilon = \int_W \left(J((\mathbf{I}+\mathbf{D})\mathbf{x}+\mathbf{d}) - I(\mathbf{x}) \right)^2 w(\mathbf{x}) d\mathbf{x}$$

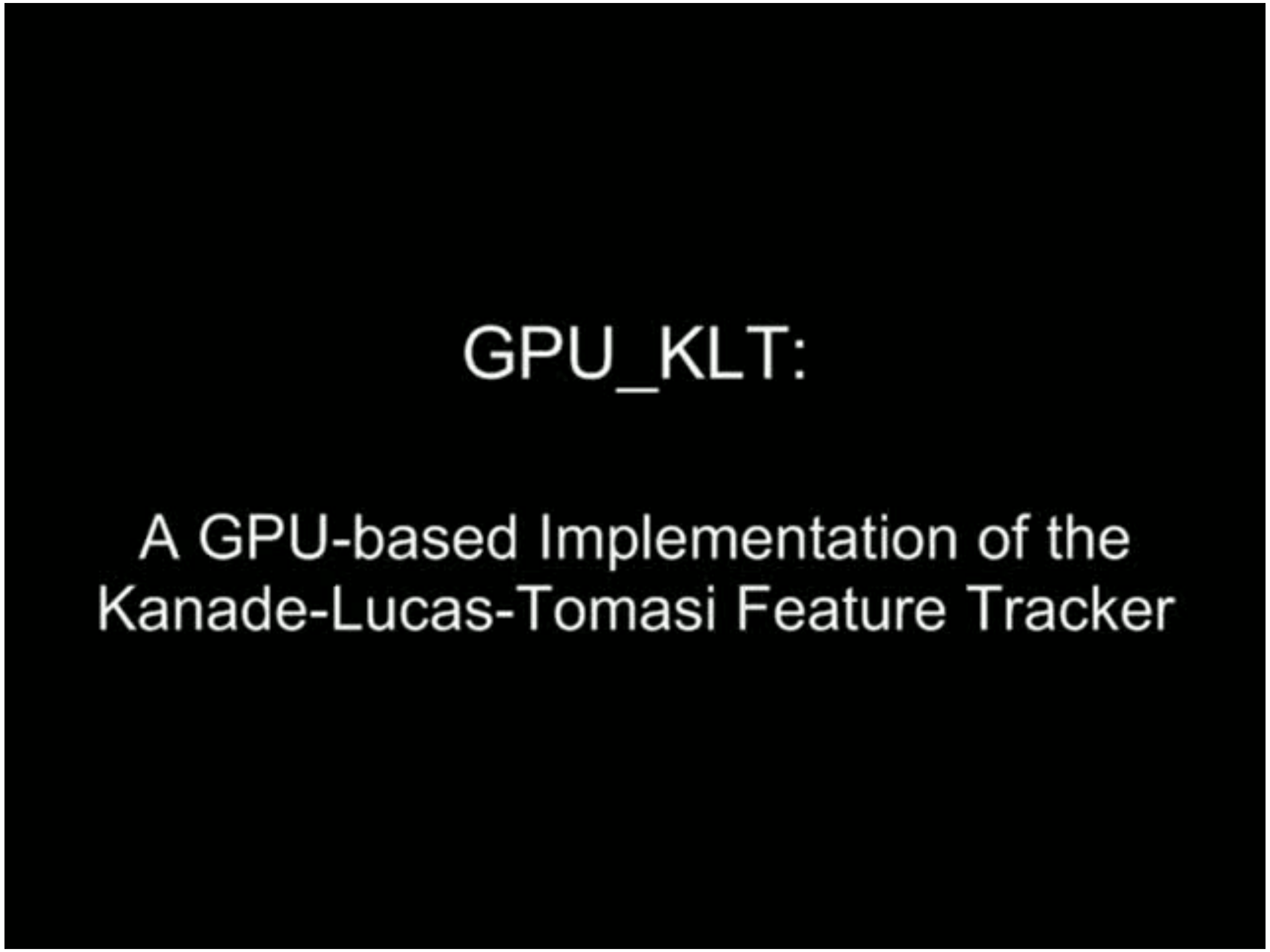
- First estimate **D** and **d** by solving

$$\mathbf{Tz} = \mathbf{a} \Rightarrow \mathbf{z} = \mathrm{pinv}(\mathbf{T})\mathbf{a} = \left(\mathbf{T}^T\mathbf{T}\right)^{-1}\mathbf{T}^T\mathbf{a}$$

where $\mathbf{z} = \left(D_{xx}, D_{xy}, D_{yx}, D_{yy}, d_x, d_y\right)^T$ and see Shi-Tomasi for definition of **T** and **a**.

- Detect occlusions and instability by thresholding the dissimilarity measure. Remove features that fail this test.

Kanade-Lucas-Tomasi (KLT) tracker



Sinha et al. “GPU_KLT: A GPU-based Implementation of the KLT Feature Tracker” 2006
http://cs.unc.edu/~ssinha/Research/GPU_KLT/

Assignment 4: The last assignment

- Try to implement the KLT tracker.
- Experiment with your tracker on different video sequences.

What’s missing?

- I have not covered the filtering and data association problems.
  - We cover the filtering problem on the *Advanced topics in data modeling* course.

- Word of advice: Many tracking algorithms exist, but non of them work in all situation – therefore consider your problem and pick the approach that best meets the challenges in your problem.

Summary

- The monocular visual 2D tracking problem
  - Definition
  - Examples
  - Applications
- The Kanade-Lucas-Tomasi (KLT) feature tracker
  - The basic algorithm
  - Extension from “Good features to track”

Literature

Reading material:

- Tomasi & Kanade: Detection and Tracking of Point Features. CMU Technical report, CMU-CS-91-132, 1991.
- Shi & Tomasi: Good Features to Track. IEEE Proceedings of CVPR’ 94, 593 – 600, 1994.

Additional material:

- Isard & Blake: CONDENSATION – Conditional Density Propagation for Visual Tracking. International Journal of Computer Vision, 29(1): 5 – 28, 1998.