Lecture 16: Tracking

Iterative KLT tracker

Juan Carlos Niebles and Jiajun Wu

CS131 Computer Vision: Foundations and Applications

What will we learn today?

- Iterative KLT tracker

Reading: [Szeliski] Chapters: 8.4, 8.5

[Fleet & Weiss, 2005]
http://www.cs.toronto.edu/pub/jepson/teaching/vision/2503/opticalFlow.pdf

Problem setting

- Given a video sequence, find all the features and track them across the video.
- First, use Harris corner detection to find features and their location $\boldsymbol{x}$.
- For each feature at location $\boldsymbol{x} = [x \;\; y]^T$:
  - Create an initial template for that feature: $T(\boldsymbol{x})$.
  - $T(\boldsymbol{x})$ is usually an image patch around $\boldsymbol{x}$.
- Goal: find new location of feature $\boldsymbol{x}$ at the next frame.
- We will assume $\boldsymbol{x}$ undergoes a transformation (translation, affine, …) parametrized by $\boldsymbol{p}$ to reach its new location $W(\boldsymbol{x};\, \boldsymbol{p})$.

KLT objective

- Our aim is to find the $\boldsymbol{p}$ that minimizes the difference between the template $T(\boldsymbol{x})$ and the image region around the new location of $\boldsymbol{x}$ after undergoing the transformation.

$$\sum_{x}[I(W(\boldsymbol{x};\boldsymbol{p})) - T(x)]^2$$

- $W(\boldsymbol{x};\boldsymbol{p})$ is the new location of feature $\boldsymbol{x}$.
- $I(W(\boldsymbol{x};\boldsymbol{p}))$ is image intensity at the new location.
- Recall that $\boldsymbol{p}$ is our vector of parameters that define the transformation that took $\boldsymbol{x}$ to its new location $W(\boldsymbol{x};\boldsymbol{p})$.
- Sum is over an image patch around $\boldsymbol{x}$.

KLT objective

- Since $\boldsymbol{p}$ may be large, minimizing this function may be difficult:

$$\sum_{x}[I(W(\boldsymbol{x};\boldsymbol{p})) - T(x)]^2$$

- We will instead break down $\boldsymbol{p} = \boldsymbol{p_0} + \Delta\boldsymbol{p}$
  - Large + small/residual motion
  - Where $\boldsymbol{p_0}$ is going to be fixed and we will solve for $\Delta\boldsymbol{p}$, which is a small value.
  - We can initialize $\boldsymbol{p_0}$ with our best guess of what the motion is; we can then calculate $\Delta\boldsymbol{p}$.
- We can substitute $\boldsymbol{p}$ to get:

$$\sum_{x}[I(W(\boldsymbol{x};\boldsymbol{p_0} + \Delta\boldsymbol{p})) - T(x)]^2$$

A little bit of math: Taylor series

- Taylor series is defined as:

$$f(x+\ \Delta x) = f(x) + \Delta x\frac{\partial f}{\partial x} + \Delta x^2\frac{\partial^2 f}{\partial x^2} + \ldots$$

- Assuming that Δx is small.
- We can apply this expansion to the KLT tracker and only use the first two terms:

Expanded KLT objective

$$\sum_x [I(W(\boldsymbol{x};\boldsymbol{p_0}+\Delta\boldsymbol{p}))-T(x)]^2$$

$$\approx \sum_x \left[I(W(\boldsymbol{x};\boldsymbol{p_0}))+\nabla I\frac{\partial W}{\partial \boldsymbol{p}}\Delta\boldsymbol{p}-T(x)\right]^2$$

It's a good thing we have already calculated what $\frac{\partial W}{\partial \boldsymbol{p}}$ would look like for affine, translations and other transformations!

Expanded KLT objective

- So our aim is to find the $\Delta \boldsymbol{p}$ that minimizes the following:

$$\underset{\Delta \boldsymbol{p}}{\operatorname{argmin}} \sum_{x} \left[I(W(\boldsymbol{x}; \boldsymbol{p_0})) + \nabla I \frac{\partial W}{\partial \boldsymbol{p}} \Delta \boldsymbol{p} - T(x) \right]^2$$

- Where $\nabla I = \begin{bmatrix} I_x & I_y \end{bmatrix}$
- Differentiate wrt $\Delta \boldsymbol{p}$ and setting it to zero:

$$\sum_{x} \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}} \right]^T \left[I(W(\boldsymbol{x}; \boldsymbol{p_0})) + \nabla I \frac{\partial W}{\partial \boldsymbol{p}} \Delta \boldsymbol{p} - T(x) \right] = 0$$

Solving for $\Delta\boldsymbol{p}$

- Solving for $\Delta\boldsymbol{p}$ in:

$$\sum_x \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}}\right]^T \left[I(W(\boldsymbol{x};\boldsymbol{p}_0)) + \nabla I \frac{\partial W}{\partial \boldsymbol{p}} \Delta\boldsymbol{p} - T(x)\right] = 0$$

- we get:

$$\boxed{\Delta\boldsymbol{p} = H^{-1} \sum_x \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}}\right]^T \left[T(x) - I(W(\boldsymbol{x};\boldsymbol{p}_0))\right]}$$

where $H = \sum_x \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}}\right]^T \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}}\right]$

H must be invertible!

Interpreting the H matrix for translation transformations

$$H = \sum_{x} \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}} \right]^T \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}} \right]$$

Recall that

1. $\nabla I = \begin{bmatrix} I_x & I_y \end{bmatrix}$ and
2. for translation motion, $\frac{\partial W}{\partial \boldsymbol{p}}(\boldsymbol{x};\boldsymbol{p}) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Therefore,

$$H = \sum_{x} \left[\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right]^T \left[\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right]$$

$$= \boxed{\sum_{x} \begin{bmatrix} {I_x}^2 & I_x I_y \\ I_x I_y & {I_y}^2 \end{bmatrix}}$$

That's the matrix from the Harris corner detector we learnt in class!

Interpreting the H matrix for affine transformations

$$H = \sum_{\mathbf{x}} \begin{bmatrix} I_x^2 & I_x I_y & x I_x^2 & y I_x I_y & x I_x I_y & y I_x I_y \\ I_x I_y & I_y^2 & x I_x I_y & y I_y^2 & x I_y^2 & y I_y^2 \\ x I_x^2 & y I_x I_y & x^2 I_x^2 & y^2 I_x I_y & xy I_x I_y & y^2 I_x I_y \\ y I_x I_y & y I_y^2 & xy I_x I_y & y^2 I_y^2 & xy I_y^2 & y^2 I_y^2 \\ x I_x I_y & x I_y^2 & x^2 I_x I_y & xy I_y^2 & x^2 I_y^2 & xy I_y^2 \\ y I_x I_y & y I_y^2 & xy I_x I_y & y^2 I_y^2 & xy I_y^2 & y^2 I_y^2 \end{bmatrix}$$

Overall KLT tracker algorithm

$$\Delta \boldsymbol{p} = H^{-1} \sum_{x} \left[\nabla I \frac{\partial W}{\partial \boldsymbol{p}} \right]^T [T(x) - I(W(\boldsymbol{x}; \boldsymbol{p_0}))]$$

Given the features from Harris detector:

1. Initialize $\boldsymbol{p_0}$.
2. Compute the initial templates $T(x)$ for each feature.
3. Transform the features in the image I with $W(\boldsymbol{x}; \boldsymbol{p_0})$.
4. Measure the error: $I(W(\boldsymbol{x}; \boldsymbol{p_0})) - T(x)$.
5. Compute the image gradients $\nabla I = [I_x \quad I_y]$.
6. Evaluate the Jacobian $\frac{\partial W}{\partial \boldsymbol{p}}$.
7. Compute steepest descent $\nabla I \frac{\partial W}{\partial \boldsymbol{p}}$.
8. Compute Inverse Hessian H^{-1}
9. Calculate the change in parameters $\Delta \boldsymbol{p}$
10. Update parameters $\boldsymbol{p_0} = \boldsymbol{p_0} + \Delta \boldsymbol{p}$
11. Repeat 3 to 10 until $\Delta \boldsymbol{p}$ is small.

KLT over multiple frames

- Once you find a transformation for two frames, you will repeat this process for every couple of frames.
- Run Harris detector every 15-20 frames to find new features.

Challenges to consider

Implementation issues

- Window size (size of neighborhood/template around x)
  – Small window more sensitive to noise and may miss larger motions (without pyramid)
  – Large window more likely to cross an occlusion boundary (and it's slower)
  – 15x15 to 31x31 seems typical
- Weighting the window
  – Common to apply weights so that center matters more (e.g., with Gaussian)

Summary

- Iterative KLT tracker