The Lucas & Kanade Algorithm

Instructor - Simon Lucey

16-423 - Designing Computer Vision Apps
• Review - Warp Functions.
• Linearizing Registration.
• Lucas & Kanade Algorithm.
3 Biggest Problems in Computer Vision?
3 Biggest Problems in Computer Vision?

REGISTRATION
REGISTRATION
REGISTRATION
REGISTRATION

(Prof. Takeo Kanade - Robotics Institute - CMU.)
What is Registration?

“Source Image”

“Template”
What is Registration?

“Source Image”

“Template”
What is Registration?

“Source”

“Template”
Our goal is to find the warp parameter vector $p$!

$x = \text{coordinate in template } [x, y]^T$

$x' = \text{corresponding coordinate in source } [x', y']^T$

$\mathcal{W}(x; p) = \text{warping function such that } x' = \mathcal{W}(x; p)$

$p = \text{parameter vector describing warp}$
Different Warp Functions

- translation
- rotation
- aspect
- affine
- perspective
- cylindrical

(Szeliski and Fleet)
Defining Warp Functions

\[ W(x; p) = x + p \]

\[ p = [p_1 \ p_2]^T \]

\[ W(x; p) = M \begin{bmatrix} x \\ 1 \end{bmatrix} \]

\[ M = \begin{bmatrix} 1 - p_1 & p_2 & p_3 \\ p_4 & 1 - p_5 & p_6 \end{bmatrix} \]

\[ p = [p_1 \ldots p_6]^T \]
Naive Approach to Registration

• If you were a person coming straight from machine learning you might suggest,

“Images of Object at various warps”
Naive Approach to Registration

• If you were a person coming straight from machine learning you might suggest,

“Images of Object at various warps”

“Vectors of pixel values at each warp position”
Naive Approach to Registration

- If you were a person coming straight from machine learning you might suggest,

  ```
  [255,134,45,.......34,12,124,67]
  [123,244,12,.......134,122,24,02]
  [67,13,245,.......112,51,92,181]
  [65,09,67,.......78,66,76,215]
  ```

  “Vectors of pixel values at each warp position”

  \[ f(\text{object}) \geq Th \]

  \[ f(\text{background}) \prec Th \]
Naive Approach to Registration

• Problem,
  • Do we sample every warp parameter value?

  • Plausible if we are only doing translation?

  • If the image is high resolution, do we need to sample every pixel?

  • What happens if warps are more complicated (e.g., scale)?

  • Becomes prohibitively expensive computationally as warp dimensionality expands.
Measures of Image Similarity

• Although not always perfect, a common measure of image similarity is:
  • Sum of squared differences (SSD)

\[
SSD(p) = \sum_{i=1}^{N} \| I\{ \mathcal{W}(x_i; p) \} - T(x_i) \|_2^2
\]
Measures of Image Similarity

• Although not always perfect, a common measure of image similarity is:
  • Sum of squared differences (SSD)

\[
SSD(p) = \sum_{i=1}^{N} \left\| \mathcal{I}\{\mathcal{W}(x_i; p)\} - \mathcal{T}(x_i) \right\|^2_2
\]
Measures of Image Similarity

- Although not always perfect, a common measure of image similarity is:
  - Sum of squared differences (SSD)

\[
\text{SSD}(\mathbf{p}) = \left\| \mathcal{I}(\mathbf{p}) - \mathcal{T}(0) \right\|^2_2
\]

\[
\begin{bmatrix}
\mathcal{W}(x_1; p) \\
\vdots \\
\mathcal{W}(x_N; p)
\end{bmatrix}
\quad \text{“Source Image”}
\]

\[
\begin{bmatrix}
\mathcal{W}(x_1; 0) \\
\vdots \\
\mathcal{W}(x_N; 0)
\end{bmatrix}
\quad \text{“Template”}
\]
Fractional Coordinates

• The warp function gives nearly always fractional output.
• But we can only deal in integer pixels.
• What happens if we need to subsample an image?
• Simply take the Taylor series approximation.

\[ I(x_0 + \Delta x) \approx I(x_0) + \frac{\partial I(x_0)}{\partial x}^T \Delta x \]
• Simply take the Taylor series approximation.

\[ I(x_0 + \Delta x) \approx I(x_0) + \left( \frac{\partial I(x_0)}{\partial x} \right)^T \Delta x \]
Exhaustive Search

“Images at various warps”
Exhaustive Search

“Images at various warps”

[255,134,45,…….,34,12,124,67]
[123,244,12,…….,134,122,24,02]
[67,13,245,…….,112,51,92,181]
[65,09,67,…….,78,66,76,215]

“Vectors of pixel values at each warp position”
Exhaustive Search

$p = \{p_1, p_2\}$

“Possible Source Warps”

“Possible Template Warps”
Exhaustive Search

- One can see that as the dimensionality $D$ of $p$ increases, and, assuming the same number of discrete samples $n$ per dimension, the number of searches becomes,

$$\text{number of searches} \rightarrow O(n^D)$$

![Graph showing the relationship between dimensionality and number of searches](image-url)
Exhaustive Search

- One can see that as the dimensionality $D$ of $p$ increases, and, assuming the same number of discrete samples $n$ per dimension, the number of searches becomes,

$$\text{number of searches} \rightarrow O(n^D)$$

"Impractical for $D \geq 3"
Today

• Review - Warp Functions.
• Linearizing Registration.
• Lucas & Kanade Algorithm.
Slightly Less Naive Approach

- Instead of sampling through all possible warp positions to find the best match, let us instead learn a regression,

![Graph showing the relationship between Warp Displacement and Appearance Displacement](image)
Slightly Less Naive Approach

- Instead of sampling through all possible warp positions to find the best match, let us instead learn a regression,

\[
\begin{align*}
\text{Warp Displacement} & \quad \text{Appearance Displacement} \\
\text{No longer have to make discrete assumptions about warps.} & \\
\text{Makes warp estimation computationally feasible.}
\end{align*}
\]
Problems?

• For us to learn this regression effectively we need to make a couple of assumptions.
  • What is the distribution $\Delta p$ of warp displacements?
  • Is there a relationship between appearance displacement and warp displacement?
  • When does this relationship occur, when does it fail?

\[
  \mathcal{T}(0) \quad \mathcal{T}(\Delta p)
\]
Problems?

• For us to learn this regression effectively we need to make a couple of assumptions.
  • What is the distribution $\Delta p$ of warp displacements?
  • Is there a relationship between appearance displacement and warp displacement?
  • When does this relationship occur, when does it fail?

$\mathcal{T}(0)$  $\mathcal{T}(\Delta p)$  

$T(0)$  $T(\Delta p)$
$I(x, y)$

$I(x + 1, y + 1)$

Simoncelli & Olshausen 2001
$I(x, y)$

$I(x + 8, y + 8)$

Simoncelli & Olshausen 2001
$I(x, y)$

$I(x + 16, y + 16)$

Simoncelli & Olshausen 2001
\[ I(x + 50, y + 50) \]

Simoncelli & Olshausen 2001
$I(x, y)$

$I(x + 50, y + 50)$
What if I want to know $\Delta x$ given that I have only the appearance at $I(x_0)$ and $I(x_0 + \Delta x)$?
Again, simply take the Taylor series approximation?

\[ \mathcal{I}(x_0 + \Delta x) \approx \mathcal{I}(x_0) + \frac{\partial \mathcal{I}(x_0)}{\partial x^T} \Delta x \]
Again, simply take the Taylor series approximation?

\[ I(x_0 + \Delta x) \approx I(x_0) + \frac{\partial I(x_0)}{\partial x^T} \Delta x \]
Linearizing Registration

• Again, simply take the Taylor series approximation,

\[ I(x_0 + \Delta x) \approx I(x_0) + \frac{\partial I(x_0)}{\partial x^T} \Delta x \]

• Therefore,

\[ \Delta x \approx \left[ \frac{\partial I(x_0)}{\partial x} \frac{\partial I(x_0)}{\partial x^T} \right]^{-1} \frac{\partial I(x_0)}{\partial x} [I(x_0 + \Delta x) - I(x_0)] \]

• We refer to \( \frac{\partial I(x_0)}{\partial x} \) as the gradient of the image at \( x_0 \).
Gradients through Filters

• Traditional method for calculating gradients in vision is through the use of edge filters. (e.g., Sobel, Prewitt).

\[
\begin{align*}
\nabla I_x & \approx \mathbf{H} \star I \\
\nabla I_y & \approx \mathbf{V} \star I
\end{align*}
\]

where, \[
\frac{\partial I(x)}{\partial x} = [\nabla I_x(x), \nabla I_y(x)]^T
\]
Gradients through Filters

- Traditional method for calculating gradients in vision is through the use of edge filters. (e.g., Sobel, Prewitt).

\[ \nabla I_x, \nabla I_y \]

- Often have to apply a smoothing filter as well.

\[ \frac{\partial I(x)}{\partial x} = [\nabla I_x(x), \nabla I_y(x)]^T \]
• Another strategy is to learn a least-squares regression for every pixel in the source image such that,

\[ I(x + \Delta x) \approx I(x) + w_x^T \Delta x + b \]
Another strategy is to learn a least-squares regression for every pixel in the source image such that,

\[ I(x + \Delta x) \approx I(x) + w_x^T \Delta x + b \]

\( w_x \in \mathbb{R}^N \)
Gradients through Regression

Another strategy is to learn a least-squares regression for every pixel in the source image such that,

\[ I(x + \Delta x) \approx I(x) + w_x^T \Delta x + b \]

Why does \( b \) have to equal zero?
Gradients through Regression

• Can be written formally as,

\[
\arg \min_{\mathbf{w}_x} \sum_{\Delta \mathbf{x} \in \mathcal{N}} \| I(\mathbf{x} + \Delta \mathbf{x}) - I(\mathbf{x}) - \mathbf{w}_x^T \Delta \mathbf{x} \|^2
\]

• Solution is given as,

\[
\mathbf{w}_x = \left( \sum_{\Delta \mathbf{x} \in \mathcal{N}} \Delta \mathbf{x} \Delta \mathbf{x}^T \right)^{-1} \left( \sum_{\Delta \mathbf{x} \in \mathcal{N}} \Delta \mathbf{x} [I(\mathbf{x} + \Delta \mathbf{x}) - I(\mathbf{x})] \right)
\]
Gradients through Regression

• Can solve for gradients using least-squares regression.
• Has nice properties: expand neighborhood, no heuristics.

Where,

\[
\frac{\partial I(x)}{\partial x} = [\nabla I_x(x), \nabla I_y(x)]^T = [w_x, w_y]^T
\]
Possible Solution?

• Could we now just solve for individual pixel translation, and then estimate a complex warp from these motions?

\[
\begin{bmatrix}
\Delta x_1 \\
\vdots \\
\Delta x_N \\
\end{bmatrix} \rightarrow \Delta p
\]

“Motion Field for Rotation”
Spatial Coherence

• Problem is pixel noise. Individual pixels are too noisy to be reliable estimators of pixel movement (optical flow).

• Fortunately, neighboring points in the scene typically belong to the same surface and hence typically have similar motions $\mathcal{W}(x; p)$. 
Reminder: Warp Functions

- To perform alignment we need a formal way of describing how the template relates to the source image.
- To do this we can employ what is known as a warp function:

\[ \mathcal{W}(\mathbf{x}; p) \]

where,

\[ x_i = i\text{-th 2D coordinate} \]
\[ p = \text{parametric form of warp} \]
Today

- Review - Warp Functions.
- Linearizing Registration.
- Lucas & Kanade Algorithm.
Lucas & Kanade Algorithm

- Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of \textit{gradients} and \textit{spatial coherence}.
- Technique applies Taylor series approximation to any spatially coherent area governed by the warp \( \mathcal{W}(x; p) \).

\[ \mathcal{I}(p + \Delta p) \approx \mathcal{I}(p) + \frac{\partial \mathcal{I}(p)}{\partial p^T} \Delta p \]
Lucas & Kanade Algorithm

• Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of gradients and spatial coherence.

• Technique applies Taylor series approximation to any spatially coherent area governed by the warp $\mathcal{W}(x; p)$.

$$\mathcal{I}(p + \Delta p) \approx \mathcal{I}(p) + \frac{\partial \mathcal{I}(p)}{\partial p^T} \Delta p$$
Lucas & Kanade Algorithm

- Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of gradients and spatial coherence.
- Technique applies Taylor series approximation to any spatially coherent area governed by the warp $\mathcal{W}(x; p)$.

$$\mathcal{I}(p + \Delta p) \approx \mathcal{I}(p) + \frac{\partial \mathcal{I}(p)}{\partial p^T} \Delta p$$

“We consider this image to always be static....”
Lucas & Kanade Algorithm

- Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of **gradients** and **spatial coherence**.
- Technique applies Taylor series approximation to any spatially coherent area governed by the warp $\mathcal{W}(x; p)$.

\[
\mathcal{I}(p + \Delta p) \approx \mathcal{I}(p) + \frac{\partial \mathcal{I}(p)}{\partial p^T} \Delta p
\]
Lucas & Kanade Algorithm

- Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of gradients and spatial coherence.
- Technique applies Taylor series approximation to any spatially coherent area governed by the warp $\mathcal{W}(x; p)$.

\[
\mathcal{I}(p + \Delta p) \approx \mathcal{I}(p) + \frac{\partial \mathcal{I}(p)}{\partial p^T} \Delta p
\]

\[
\frac{\partial \mathcal{I}(p)}{\partial p^T} = \begin{bmatrix}
\frac{\partial \mathcal{I}^T(x'_{1})}{\partial x'_{1}^T} & \cdots & 0^T \\
\vdots & \ddots & \vdots \\
0^T & \cdots & \frac{\partial \mathcal{I}^T(x'_{N})}{\partial x'_{N}^T}
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \mathcal{W}(x_{1}; p)}{\partial p^T} \\
\vdots \\
\frac{\partial \mathcal{W}(x_{N}; p)}{\partial p^T}
\end{bmatrix}
\]

\[
x' = \mathcal{W}(x; p)
\]
Lucas & Kanade Algorithm

\[
\frac{\partial \mathcal{I}(\mathbf{p})}{\partial \mathbf{p}^T} = \begin{bmatrix}
\frac{\partial \mathcal{I}(\mathbf{x}'_1)}{\partial \mathbf{x}'_1^T} & \ldots & 0^T \\
\vdots & \ddots & \vdots \\
0^T & \ldots & \frac{\partial \mathcal{I}(\mathbf{x}'_N)}{\partial \mathbf{x}'_N^T}
\end{bmatrix}
= \begin{bmatrix}
\frac{\partial \mathcal{W}(\mathbf{x}_1; \mathbf{p})}{\partial \mathbf{p}^T} \\
\vdots \\
\frac{\partial \mathcal{W}(\mathbf{x}_N; \mathbf{p})}{\partial \mathbf{p}^T}
\end{bmatrix}
\]

\[\nabla_x \mathcal{I} \]

\[\nabla_y \mathcal{I} \]
\[
\frac{\partial \mathcal{I}(p)}{\partial p^T} = \begin{bmatrix}
\frac{\partial \mathcal{I}(x_1')}{\partial x_1'} & \cdots & 0^T \\
\vdots & \ddots & \vdots \\
0^T & \cdots & \frac{\partial \mathcal{I}(x_N')}{\partial x_N'}
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \mathcal{W}(x_1; p)}{\partial p^T} \\
\vdots \\
\frac{\partial \mathcal{W}(x_N; p)}{\partial p^T}
\end{bmatrix}
\]
Lucas & Kanade Algorithm

\[ \frac{\partial I(p)}{\partial p^T} = \begin{bmatrix} \frac{\partial I(x'_1)}{\partial x'^T_1} & \cdots & 0^T \\ \vdots & \ddots & \vdots \\ 0^T & \cdots & \frac{\partial I(x'_N)}{\partial x'^T_N} \end{bmatrix} \begin{bmatrix} \frac{\partial W(x_1;p)}{\partial p^T} \\ \vdots \\ \frac{\partial W(x_N;p)}{\partial p^T} \end{bmatrix} \]

\[ W(x;p) = M \begin{bmatrix} x \\ 1 \end{bmatrix} \]

\[ M = \begin{bmatrix} 1 - p_1 & p_2 & p_3 \\ p_4 & 1 - p_5 & p_6 \end{bmatrix} \]

\[ p = [p_1 \ldots p_6]^T \]

Can you determine \( \frac{\partial W(x;p)}{\partial p} \)?
Lucas & Kanade Algorithm

- Lucas & Kanade (1981) realized this and proposed a method for estimating warp displacement using the principles of *gradients* and *spatial coherence*.
- Technique applies Taylor series approximation to any spatially coherent area governed by the warp $\mathcal{W}(x; p)$.

$$\mathcal{I}(p + \Delta p) \approx \mathcal{I}(p) + \frac{\partial \mathcal{I}(p)^T}{\partial p} \Delta p$$

\[ N \times 1 \quad N \times 1 \quad N \times P \]

$N = \text{number of pixels}$

$P = \text{number of warp parameters}$
Lucas & Kanade Algorithm

• Often just refer to,

\[
J_I = \frac{\partial \mathcal{I}(p)}{\partial p^T}
\]

as the “Jacobian” matrix.

• Also refer to,

\[
H_I = J_I^T J_I
\]

as the “pseudo-Hessian”.

• Finally, we can refer to,

\[
\mathcal{T}(0) = \mathcal{I}(p + \Delta p)
\]

as the “template”.
Lucas & Kanade Algorithm

- Actual algorithm is just the application of the following steps,

**Step 1:**

\[
\Delta p = H_I^{-1} J^T_I [T(0) - I(p)]
\]

**Step 2:**

\[
p \leftarrow p + \Delta p
\]

keep applying steps until \( \Delta p \) converges.
Examples of LK Alignment
Examples of LK Alignment
Gauss-Newton Algorithm

- Gauss-Newton algorithm common strategy for optimizing non-linear least-squares problems.

\[
\arg \min_x \|y - \mathcal{F}(x)\|_2^2 \\
\text{s.t. } \mathcal{F} : \mathbb{R}^N \rightarrow \mathbb{R}^M
\]

**Step 1:**
\[
\arg \min_{\Delta x} \|y - \mathcal{F}(x) - \frac{\partial \mathcal{F}(x)}{\partial x^T} \Delta x\|_2^2
\]

**Step 2:**
\[
x \leftarrow x + \Delta x
\]

keep applying steps until \(\Delta x\) converges.
Initialization

- Initialization has to be suitably close to the ground-truth for method to work.
- Kind of like a black hole’s event horizon.
- You got to be inside it to be sucked in!
Gauss-Newton Optimization

Can we approximate the true objective with a convex one?


• Bristow & Lucey, “In Defense of Gradient-Based Alignment on Densely Sampled Sparse Features”, Springer Book on Dense Registration Methods 2015.