Lecture 6: Optical Flow

Mark Hasegawa-Johnson

ECE 417: Multimedia Signal Processing, Fall 2021
1. Image Gradient
2. Optical Flow
3. The Lucas-Kanade Algorithm
4. Pseudo-Inverse
5. Summary
Outline

1. Image Gradient
2. Optical Flow
3. The Lucas-Kanade Algorithm
4. Pseudo-Inverse
5. Summary
The image gradient is a way of characterizing the distribution of light and dark pixels in an image. Suppose the image intensity is $f(t, r, c)$. The image gradient is:

$$\nabla f = \begin{bmatrix} \frac{\partial f(t, r, c)}{\partial c} \\ \frac{\partial f(t, r, c)}{\partial r} \end{bmatrix}$$
Image gradient
Image gradient

Public domain image, Njw00, 2010
How do you calculate the image gradient?

Basically, use one of the standard numerical estimates of a derivative. For example, the central-difference operator:

\[
\nabla f = \begin{bmatrix} \frac{\partial f(t,r,c)}{\partial c} \\ \frac{\partial f(t,r,c)}{\partial r} \end{bmatrix} = \begin{bmatrix} \frac{f[t,r,c+1] - f[t,r,c-1]}{2} \\ \frac{f[t,r+1,c] - f[t,r-1,c]}{2} \end{bmatrix}
\]

Wikipedia has a good listing of other methods you can use.
Outline

1. Image Gradient
2. Optical Flow
3. The Lucas-Kanade Algorithm
4. Pseudo-Inverse
5. Summary
Definition: **optical flow** is the vector field $\vec{v}(t, r, c)$ specifying the current apparent velocity of the pixel at position $(r, c)$. It depends on motion of (1) the object observed, and (2) the observer.

Definition: **optical flow** is the vector field $\vec{v}(t, r, c)$ specifying the current apparent velocity of the pixel at position $(r, c)$. It depends on motion of (1) the object observed, and (2) the observer.

Optical Flow

For example, you can use it to track a user-specified rectangle in the ultrasound video of a tendon.

How to calculate optical flow

General idea:

- Treat the image as a function of continuous time and space, $f(t, r, c)$.
- If the image intensity is changing, as a function of time, then try to explain it by moving pixels around.
Calculating optical flow

More formally, let’s treat local variation of $f(t, r, c)$ using a first-order Taylor series:

$$f(t + \Delta t, r + \Delta r, c + \Delta c) \approx f(t, r, c) + \Delta t \frac{\partial f}{\partial t} + \Delta r \frac{\partial f}{\partial r} + \Delta c \frac{\partial f}{\partial c}$$

Hypothesize that all intensity variations are caused by pixels moving around. Then

$$f(t + \Delta t, r + \Delta r, c + \Delta c) - f(t, r, c) = 0$$
Calculating optical flow

\[ 0 = f(t + \Delta t, r + \Delta r, c + \Delta c) - f(t, r, c) \approx \Delta t \frac{\partial f}{\partial t} + \Delta r \frac{\partial f}{\partial r} + \Delta c \frac{\partial f}{\partial c} \]
Calculating optical flow

\[ 0 \approx \Delta t \frac{\partial f}{\partial t} + \Delta r \frac{\partial f}{\partial r} + \Delta c \frac{\partial f}{\partial c} \]

Dividing through by \( \Delta t \), and taking the limit as \( \Delta t \to 0 \), we get

\[ 0 \approx \frac{\partial f}{\partial t} + \left( \frac{\partial r}{\partial t} \right) \frac{\partial f}{\partial r} + \left( \frac{\partial c}{\partial t} \right) \frac{\partial f}{\partial c} \]
Calculating optical flow

Re-arranging gives us the optical flow equation:

\[- \frac{\partial f}{\partial t} \approx \left( \frac{\partial r}{\partial t} \right) \frac{\partial f}{\partial r} + \left( \frac{\partial c}{\partial t} \right) \frac{\partial f}{\partial c} \]
How to calculate optical flow

Define the optical flow vector, $\vec{v}(t, r, c)$, and image gradient, $\nabla f(t, r, c)$:

$$\vec{v} = \begin{bmatrix} \frac{\partial c}{\partial t} \\ \frac{\partial c}{\partial r} \\ \frac{\partial f}{\partial t} \end{bmatrix}, \quad \nabla f = \begin{bmatrix} \frac{\partial f}{\partial c} \\ \frac{\partial f}{\partial r} \end{bmatrix}$$

Then the optical flow equation is:

$$-\frac{\partial f}{\partial t} = (\nabla f)^T \vec{v}$$
Outline

1. Image Gradient
2. Optical Flow
3. The Lucas-Kanade Algorithm
4. Pseudo-Inverse
5. Summary
How to calculate optical flow

So we have this optical flow equation:

$$- \frac{\partial f}{\partial t} = (\nabla f)^T \vec{v}$$

Assume that we can calculate $\partial f / \partial t$ and $\nabla f$, using standard image gradient methods. Now we just need to find $\vec{v}$. But $\vec{v} = [v_r, v_c]^T$ is a vector of two unknowns, so the equation above is one equation in two unknowns!
How to calculate optical flow

The solution is to assume that a small block of pixels all move together:

CC-SA 4.0 by German iris, 2017
The Lucas-Kanade Algorithm

The Lucas-Kanade Algorithm replaces this equation

\[- \frac{\partial f}{\partial t} = (\nabla f)^T \vec{v}\]

with this equation:

\[- \begin{bmatrix}
\frac{\partial f[t,r-W,c-W]}{\partial t} \\
\vdots \\
\frac{\partial f[t,r+W,c+W]}{\partial t}
\end{bmatrix}
\begin{bmatrix}
\frac{\partial f[t,r-W,c-W]}{\partial c} \\
\vdots \\
\frac{\partial f[t,r+W,c+W]}{\partial c}
\end{bmatrix}
= \begin{bmatrix}
\frac{\partial f[t,r-W,c-W]}{\partial r} \\
\vdots \\
\frac{\partial f[t,r+W,c+W]}{\partial r}
\end{bmatrix}
\begin{bmatrix}
V_c \\
V_r
\end{bmatrix}\]

so that we are averaging over a block of size \((2W + 1) \times (2W + 1)\) pixels.
The Lucas-Kanade algorithm solves the equation

\[ \vec{b} = A \vec{v} \]

where

\[ \vec{b} = - \begin{bmatrix} \frac{\partial f[t, r-W, c-W]}{\partial t} \\ \vdots \\ \frac{\partial f[t, r+W, c+W]}{\partial t} \end{bmatrix}, \quad A = \begin{bmatrix} \frac{\partial f[t, r-W, c-W]}{\partial c} & \frac{\partial f[t, r-W, c-W]}{\partial r} \\ \vdots & \vdots \\ \frac{\partial f[t, r+W, c+W]}{\partial c} & \frac{\partial f[t, r+W, c+W]}{\partial r} \end{bmatrix} \]

\[ \vec{v} = \begin{bmatrix} v_c[t, r, c] \\ v_r[t, r, c] \end{bmatrix} \]
The Lucas-Kanade Algorithm

The Lucas-Kanade algorithm solves the equation

\[ \vec{b} = A\vec{v} \]

... but now \( A \) is a matrix of size \((2W + 1) \times 2\), so it's still not invertible! How do we solve that?
Outline

1. Image Gradient
2. Optical Flow
3. The Lucas-Kanade Algorithm
4. Pseudo-Inverse
5. Summary
Pseudo-Inverse

The pseudo-inverse, $A^\dagger$, of any matrix $A$, is a matrix that acts like $A^{-1}$ in many ways, but it doesn’t require $A$ to be square. Here are some of its properties:

$$AA^\dagger A = A$$

$$A^\dagger AA^\dagger = A$$
Of particular interest to us, the vector $\vec{v} = A^\dagger \vec{b}$ “pseudo-solves” the equation $\vec{b} = A\vec{v}$. By pseudo-solve, we mean that

- If $A$ is a short fat matrix, then there are an infinite number of different vectors $\vec{v}$ that solve $\vec{b} = A\vec{v}$. $\vec{v} = A^\dagger \vec{b}$ is one of those; specifically, it’s the one that minimizes $\|\vec{v}\|^2$.

- If $A$ is a tall thin matrix, then there is usually no vector $\vec{v}$ that solves $\vec{b} = A\vec{v}$, but $\vec{v} = A^\dagger \vec{b}$ is the vector that comes closest, in the sense that

$$A^\dagger \vec{b} = \arg\min_{\vec{v}} \|\vec{b} - A\vec{v}\|^2$$
Solving for the Pseudo-Inverse

Let’s use this equation:

$$v^* = A^\dagger \vec{b} = \arg\min_v \| \vec{b} - A\vec{v} \|^2$$

to solve for the pseudo-inverse.
Solving for the Pseudo-Inverse

\[ A^\dagger \vec{b} = \arg\min_{\vec{v}} \| \vec{b} - A\vec{v} \|^2 \]

\[ = \arg\min_{\vec{v}} \left( \vec{b} - A\vec{v} \right)^T \left( \vec{b} - A\vec{v} \right) \]

\[ = \arg\min_{\vec{v}} \left( \vec{b}^T \vec{b} - 2\vec{v}^T A^T \vec{b} + \vec{v}^T A^T A\vec{v} \right) \]
Solving for the Pseudo-Inverse

\[ A^\dagger \vec{b} = \arg\min_\vec{v} \left( \vec{b}^T \vec{b} - 2\vec{v}^T A^T \vec{b} + \vec{v}^T A^T A \vec{v} \right) \]

If we differentiate the quantity in parentheses, and set the derivative to zero, we get

\[ \vec{0} = -2A^T \vec{b} + 2A^T A \vec{v} \]

Assume that the columns of \( A \) are linearly independent; then \( A^T A \) is invertible, and so the solution is

\[ \vec{v} = (A^T A)^{-1} A^T \vec{b} \]
Outline

1. Image Gradient
2. Optical Flow
3. The Lucas-Kanade Algorithm
4. Pseudo-Inverse
5. Summary
Optical flow is the vector field, $\vec{v}(t, r, c)$, as a function of pixel position and frame number. It is computed by assuming that the only changes to an image are the ones caused by motion, so that

$$f(t + \Delta t, r + \Delta r, c + \Delta c) = f(t, r, c)$$

From that assumption, we get the optical flow equation:

$$-\frac{\partial f}{\partial t} = \vec{v}^T \nabla f$$
Lucas-Kanade assumes that there is a \((2W + 1) \times (2W + 1)\) block of pixels that all move together, so that \(\vec{b} = A\vec{v}\), where

\[
\vec{b} = -\begin{bmatrix}
\frac{\partial f[t, r-W, c-W]}{\partial t} \\
\vdots \\
\frac{\partial f[t, r+W, c+W]}{\partial t}
\end{bmatrix}, \quad A = \begin{bmatrix}
\frac{\partial f[t, r-W, c-W]}{\partial c} & \frac{\partial f[t, r-W, c-W]}{\partial r} \\
\vdots & \vdots \\
\frac{\partial f[t, r+W, c+W]}{\partial c} & \frac{\partial f[t, r+W, c+W]}{\partial r}
\end{bmatrix}
\]

\[
\vec{v} = \begin{bmatrix}
v_c[t, r, c] \\
v_r[t, r, c]
\end{bmatrix}
\]
The Lucas-Kanade equation cannot be solved exactly, because it is $(2W + 1)$ equations in only two unknowns. But we can find the minimum-squared error solution, which is

$$v^*[t, r, c] = A^\dagger \vec{b} = (A^T A)^{-1} A^T \vec{b}$$