Lucas-Kanade Method

Shree K. Nayar

Columbia University

Topic: Motion and Optical Flow, Module: Reconstruction II

First Principles of Computer Vision

Lucas-Kanade Solution

Assumption: For each pixel, assume Motion Field, and hence Optical Flow (u, v), is constant within a small neighborhood W.

[Lucas 1981]

Lucas-Kanade Solution

Assumption: For each pixel, assume Motion Field, and hence Optical Flow (u, v), is constant within a small neighborhood W.

[Lucas 1981]

Lucas-Kanade Solution

Assumption: For each pixel, assume Motion Field, and hence Optical Flow (u, v), is constant within a small neighborhood W.

[Lucas 1981]

Lucas-Kanade Solution

Assumption: For each pixel, assume Motion Field, and hence Optical Flow (u, v), is constant within a small neighborhood W.

That is for all points $(k, l) \in W$:

$$I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$$

[Lucas 1981]

Lucas-Kanade Solution

Assumption: For each pixel, assume Motion Field, and hence Optical Flow (u, v), is constant within a small neighborhood W.

That is for all points $(k, l) \in W$:

$$I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$$

[Lucas 1981]

Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$

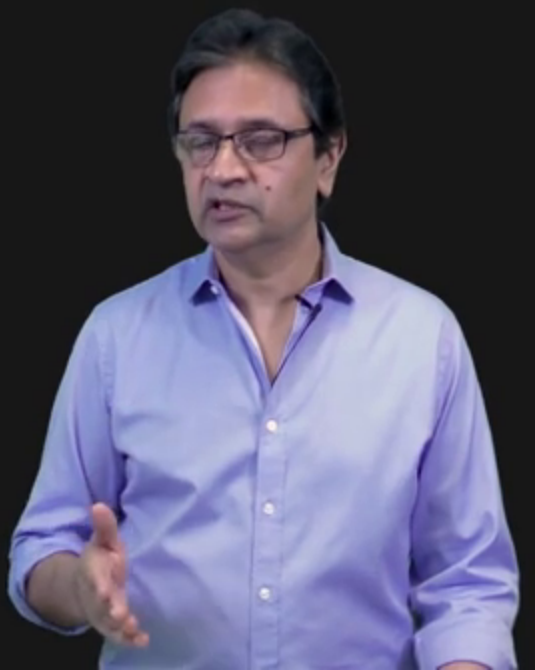
Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$



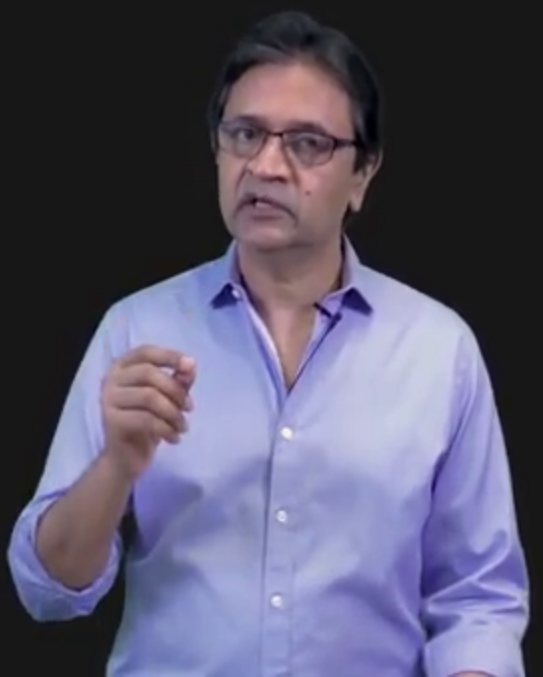
Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$



Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$

Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$

A (Known) $n^2 \times 2$ | $\mathbf{u}$ (Unknown) 2×1 | B (Known) $n^2 \times 1$

Lucas-Kanade Solution

For all points $(k,l) \in W$: $I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$

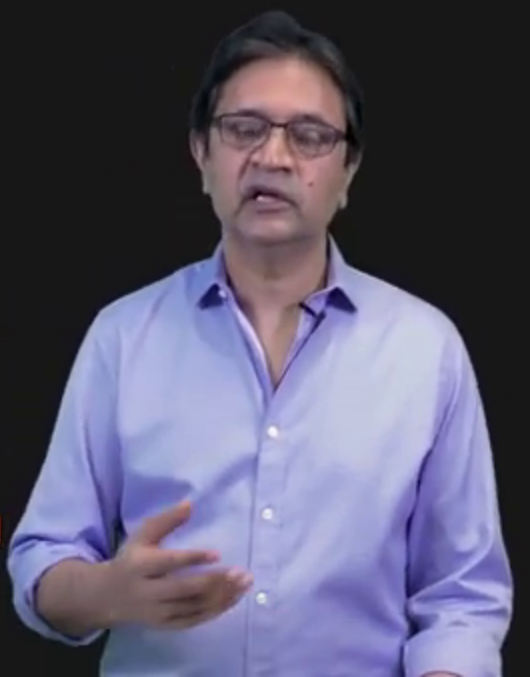
Let the size of window W be $n \times n$

In matrix form:

$$\begin{bmatrix} I_x(1,1) & I_y(1,1) \\ I_x(k,l) & I_y(k,l) \\ \vdots & \vdots \\ I_x(n,n) & I_y(n,n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(1,1) \\ I_t(k,l) \\ \vdots \\ I_t(n,n) \end{bmatrix}$$

A (Known) $n^2 \times 2$ | $\mathbf{u}$ (Unknown) 2×1 | B (Known) $n^2 \times 1$

n^2 Equations, 2 Unknowns: Find Least Squares Solution

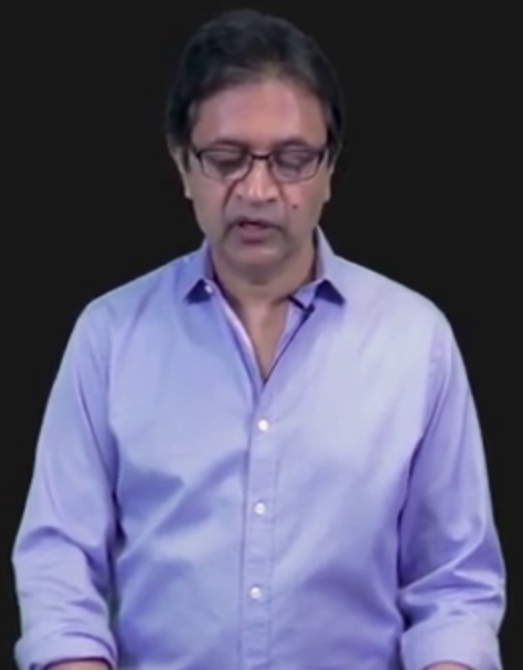


Least Squares Solution

Solve linear system: $A\mathbf{u} = B$

$$A^T A\mathbf{u} = A^T B$$

(Least-Squares using Pseudo-Inverse)



Least Squares Solution

Solve linear system: $A\mathbf{u} = B$

$$A^T A\mathbf{u} = A^T B$$ (Least-Squares using Pseudo-Inverse)

In matrix form:

$$\begin{bmatrix} \sum_w I_x I_x & \sum_w I_x I_y \\ \sum_w I_x I_y & \sum_w I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\sum_w I_x I_t \\ -\sum_w I_y I_t \end{bmatrix}$$

$A^T A$	$\mathbf{u}$	$A^T B$
(Known)	(Unknown)	(Known)
2×2	2×1	2×1

Indices (k, l) not written for simplicity

Least Squares Solution

Solve linear system: $A\mathbf{u} = B$

$$A^T A\mathbf{u} = A^T B$$ (Least-Squares using Pseudo-Inverse)

In matrix form:

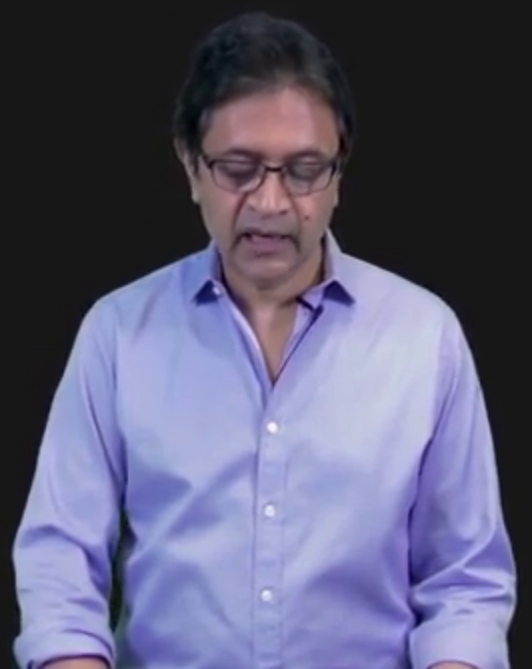
$$\begin{bmatrix} \sum_w I_x I_x & \sum_w I_x I_y \\ \sum_w I_x I_y & \sum_w I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\sum_w I_x I_t \\ -\sum_w I_y I_t \end{bmatrix}$$

Indices (k, l) not written for simplicity

$A^T A$	$\mathbf{u}$	$A^T B$
(Known)	(Unknown)	(Known)
2×2	2×1	2×1

$$\mathbf{u} = (A^T A)^{-1} A^T B$$

Fast and Easy to Solve



Least Squares Solution

Solve linear system: $A\mathbf{u} = B$

$$A^T A\mathbf{u} = A^T B$$ (Least-Squares using Pseudo-Inverse)

In matrix form:

$$\begin{bmatrix} \sum_w I_x I_x & \sum_w I_x I_y \\ \sum_w I_x I_y & \sum_w I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\sum_w I_x I_t \\ -\sum_w I_y I_t \end{bmatrix}$$

Indices (k, l) not written for simplicity

$A^T A$	$\mathbf{u}$	$A^T B$
(Known)	(Unknown)	(Known)
2×2	2×1	2×1

$$\mathbf{u} = (A^T A)^{-1} A^T B$$

Fast and Easy to Solve

Least Squares Solution

Solve linear system: $A\mathbf{u} = B$

$$A^T A\mathbf{u} = A^T B$$ (Least-Squares using Pseudo-Inverse)

In matrix form:

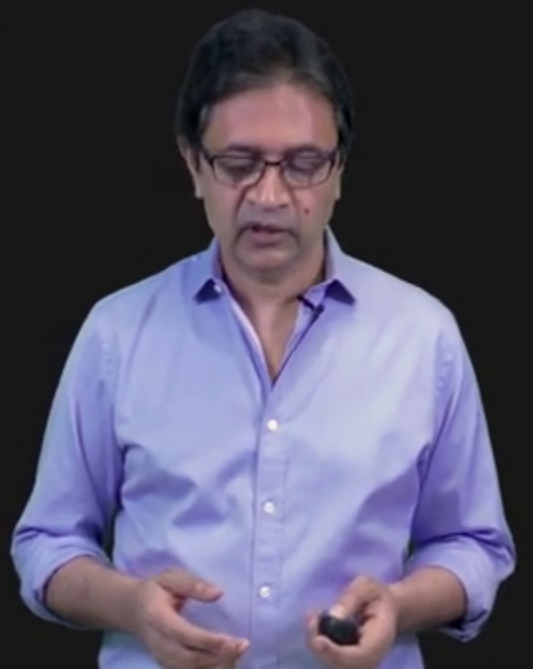
$$\begin{bmatrix} \sum_w I_x I_x & \sum_w I_x I_y \\ \sum_w I_x I_y & \sum_w I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\sum_w I_x I_t \\ -\sum_w I_y I_t \end{bmatrix}$$

Indices (k, l) not written for simplicity

$A^T A$	$\mathbf{u}$	$A^T B$
(Known)	(Unknown)	(Known)
2×2	2×1	2×1

$$\mathbf{u} = (A^T A)^{-1} A^T B$$

Fast and Easy to Solve



When Does Optical Flow Estimation Work?

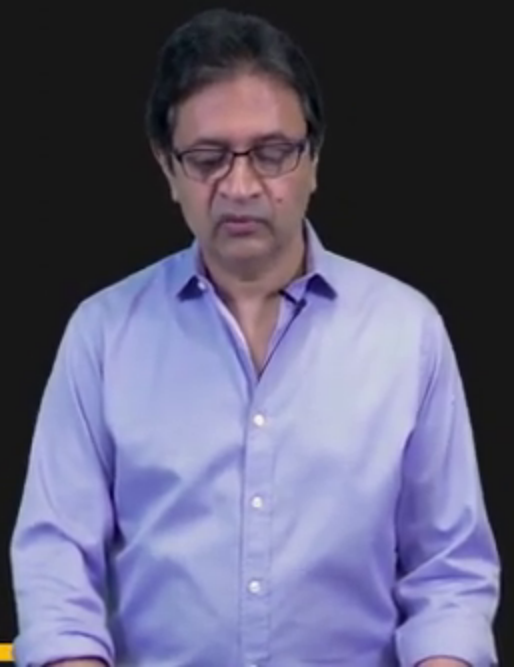
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



When Does Optical Flow Estimation Work?

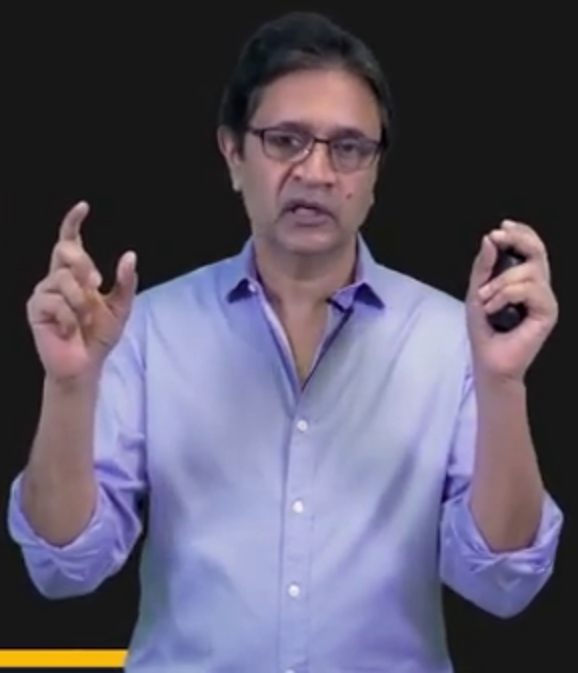
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



When Does Optical Flow Estimation Work?

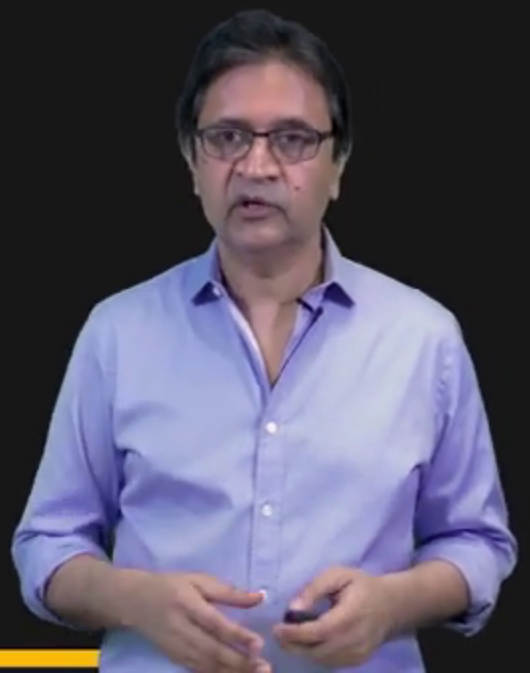
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



When Does Optical Flow Estimation Work?

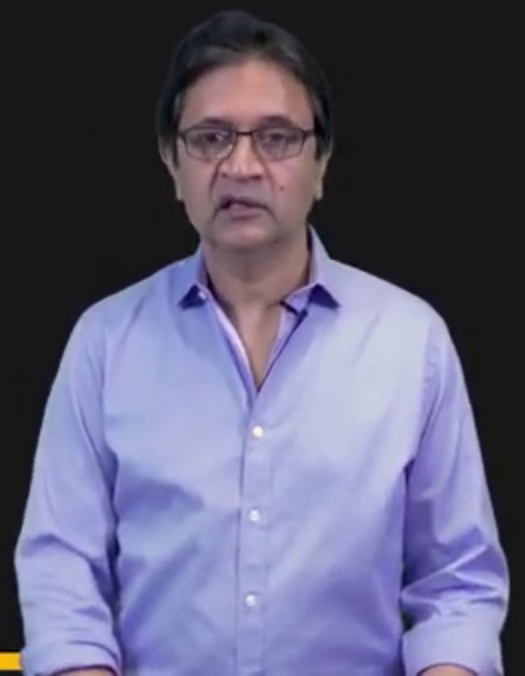
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



When Does Optical Flow Estimation Work?

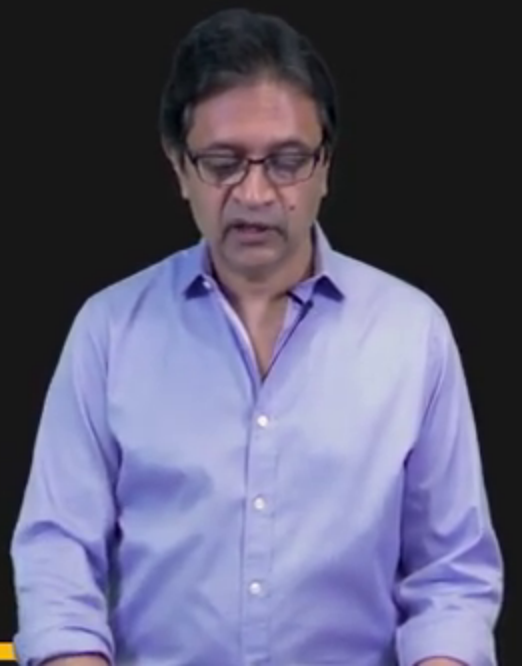
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



When Does Optical Flow Estimation Work?

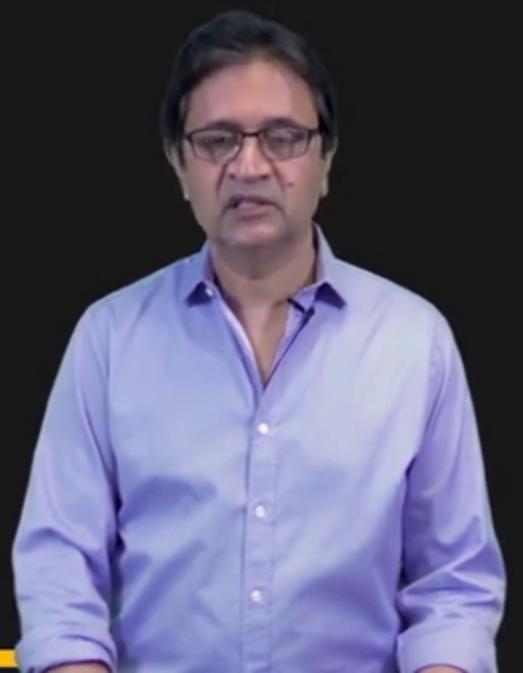
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

  If λ_1 and λ_2 are eigen values of $A^T A$, then

  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



When Does Optical Flow Estimation Work?

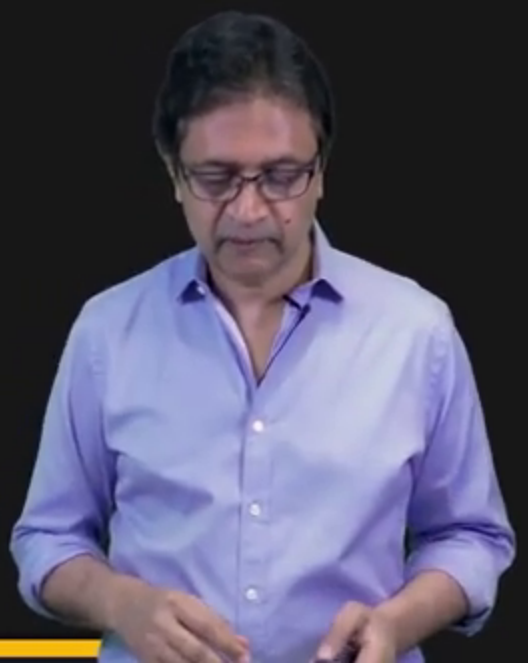
$$A\mathbf{u} = B \qquad A^T A\mathbf{u} = A^T B$$

- $A^T A$ must be invertible. That is $\det(A^T A) \neq 0$
- $A^T A$ must be well-conditioned.

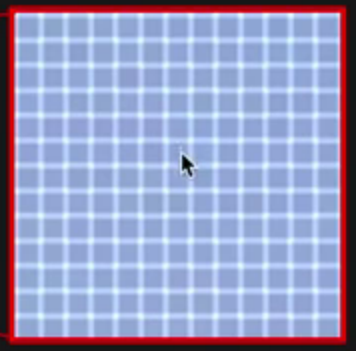
  If λ_1 and λ_2 are eigen values of $A^T A$, then

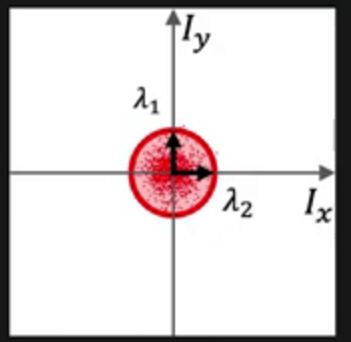
  $\lambda_1 > \epsilon$ and $\lambda_2 > \epsilon$

  $\lambda_1 \geq \lambda_2$ but not $\lambda_1 \ggg \lambda_2$



Smooth Regions (Bad)

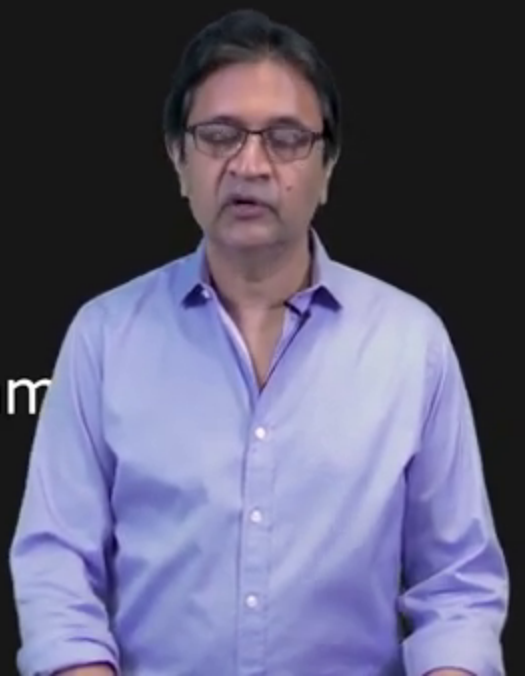




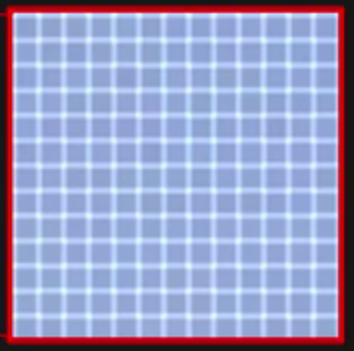
$\lambda_1 \sim \lambda_2$
Both are Small

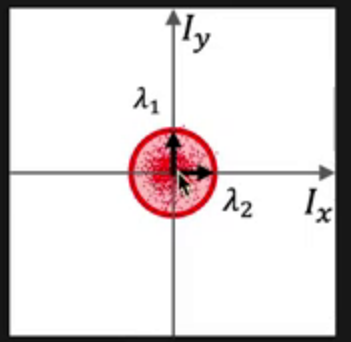
Equations for all pixels in window are more or less the sam

Cannot reliably compute flow!



Smooth Regions (Bad)



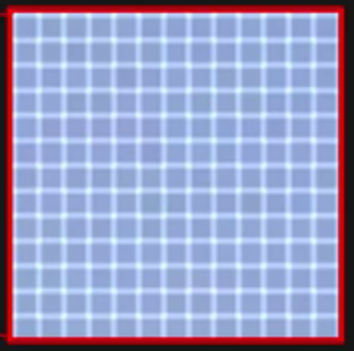


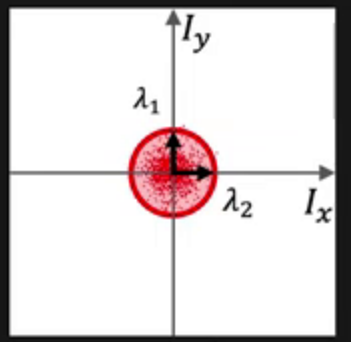
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the same

Cannot reliably compute flow!

Smooth Regions (Bad)



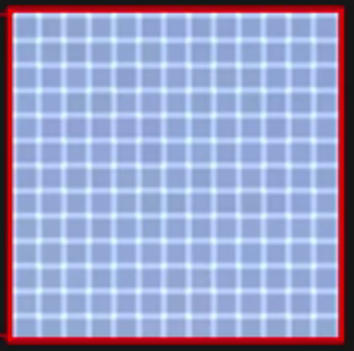


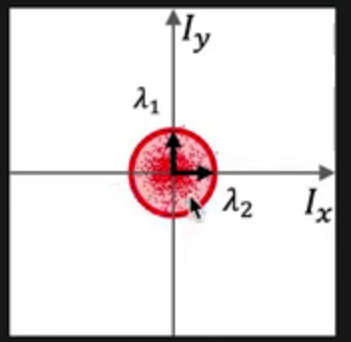
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the sam

Cannot reliably compute flow!

Smooth Regions (Bad)



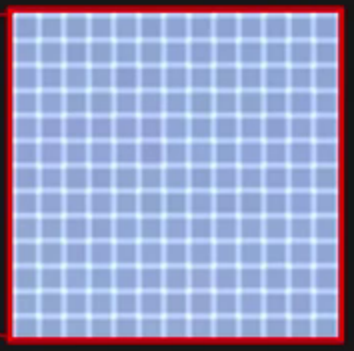


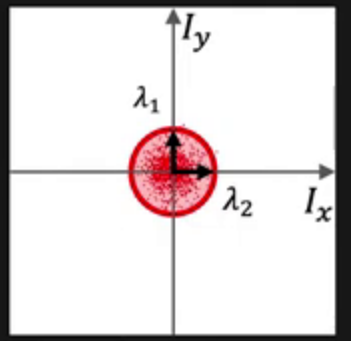
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the same

Cannot reliably compute flow!

Smooth Regions (Bad)



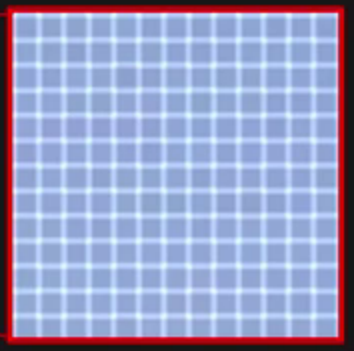


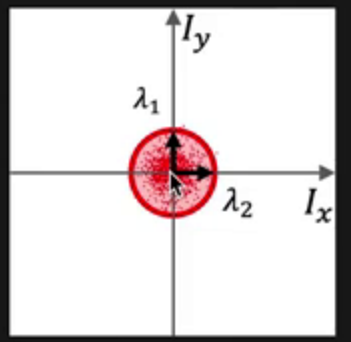
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the sam

Cannot reliably compute flow!

Smooth Regions (Bad)



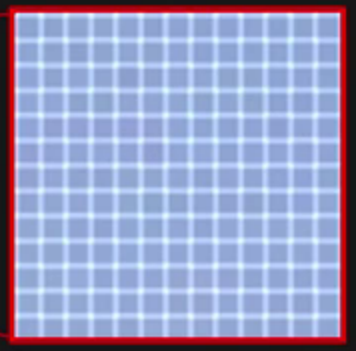


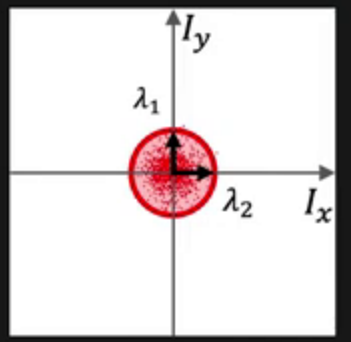
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the sam

Cannot reliably compute flow!

Smooth Regions (Bad)



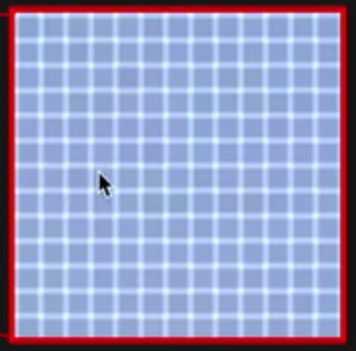


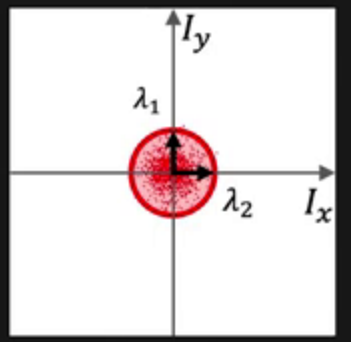
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the sam

Cannot reliably compute flow!

Smooth Regions (Bad)



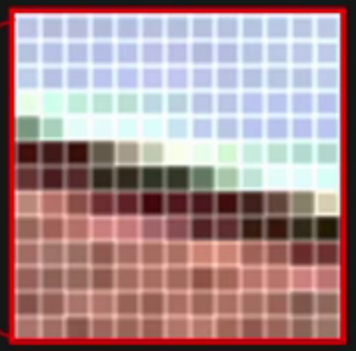


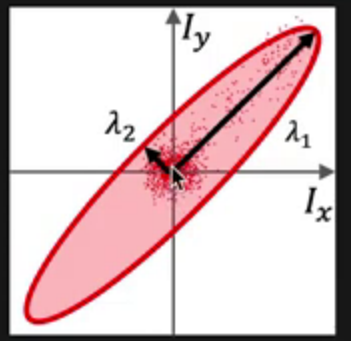
$\lambda_1 \sim \lambda_2$
Both are Small

Equations for all pixels in window are more or less the sam

Cannot reliably compute flow!

Edges (Bad)





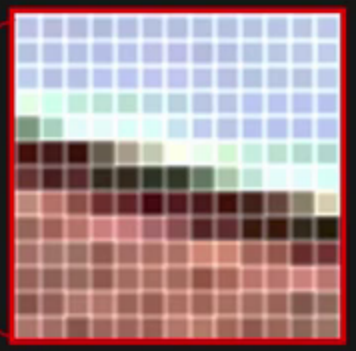
$\lambda_1 \ggg \lambda_2$

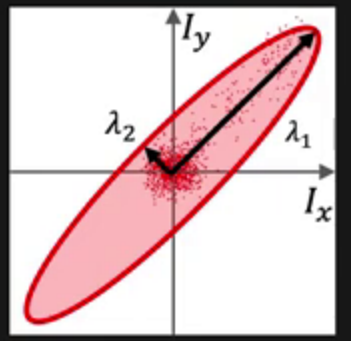
Badly conditioned. Prominent gradient in one direction.

Cannot reliably compute flow!

Same as Aperture Problem.

Edges (Bad)





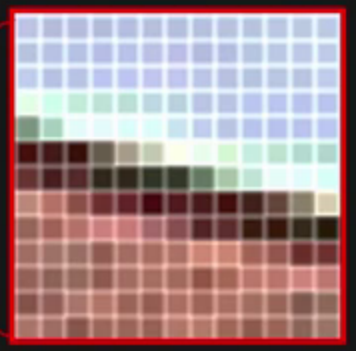
$\lambda_1 \ggg \lambda_2$

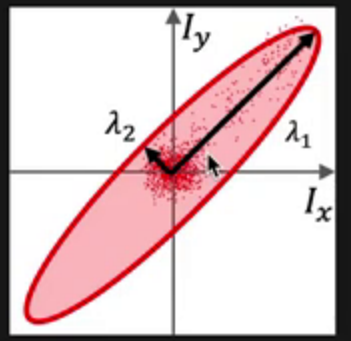
Badly conditioned. Prominent gradient in one direction

Cannot reliably compute flow!

Same as Aperture Problem.

Edges (Bad)



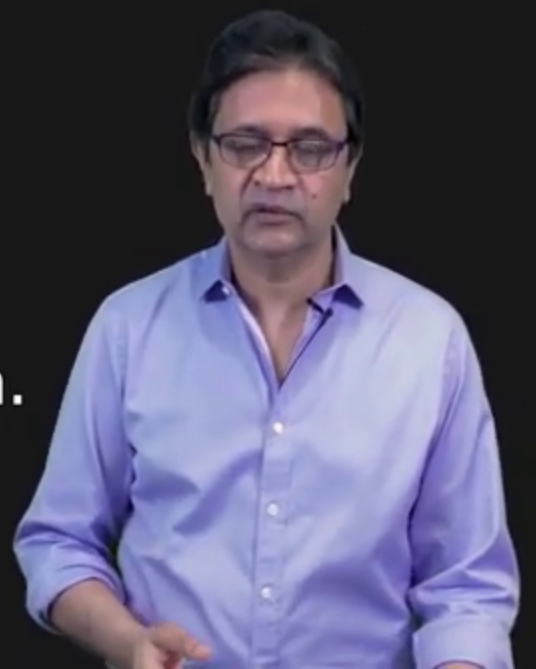


$\lambda_1 \ggg \lambda_2$

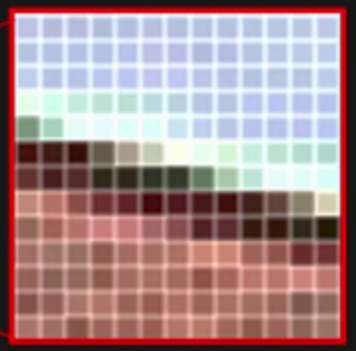
Badly conditioned. Prominent gradient in one direction.

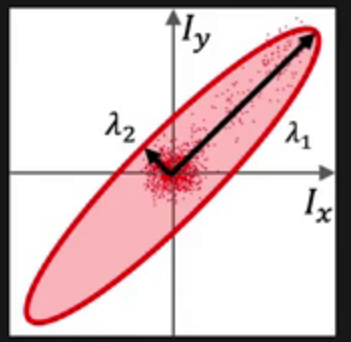
Cannot reliably compute flow!

Same as Aperture Problem.



Edges (Bad)





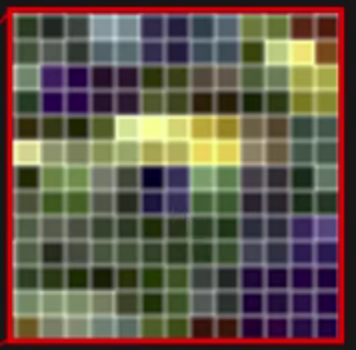
$\lambda_1 \ggg \lambda_2$

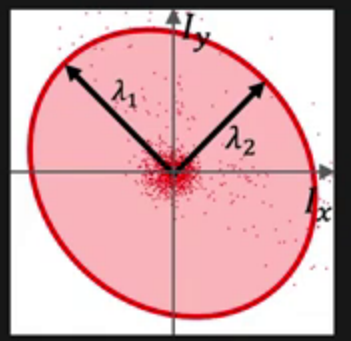
Badly conditioned. Prominent gradient in one direction.

Cannot reliably compute flow!

Same as Aperture Problem.

Textured Regions (Good)



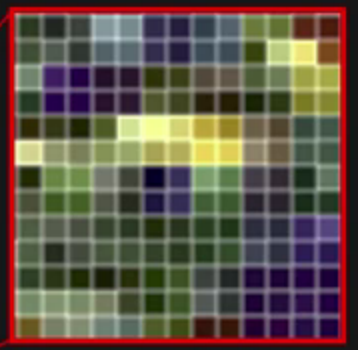


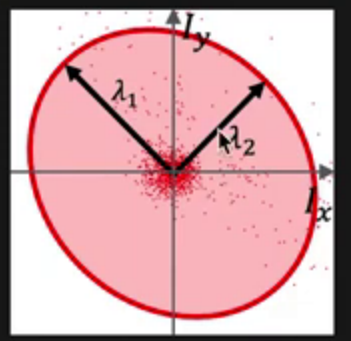
$\lambda_1 \sim \lambda_2$
Both are Large

Well conditioned. Large and diverse gradient magnitudes.

Can reliably compute optical flow.

Textured Regions (Good)





$\lambda_1 \sim \lambda_2$
Both are Large

Well conditioned. Large and diverse gradient magnitudes.

Can reliably compute optical flow.