

计算机视觉—运动估计

申抒含

中国科学院自动化研究所
模式识别国家重点实验室

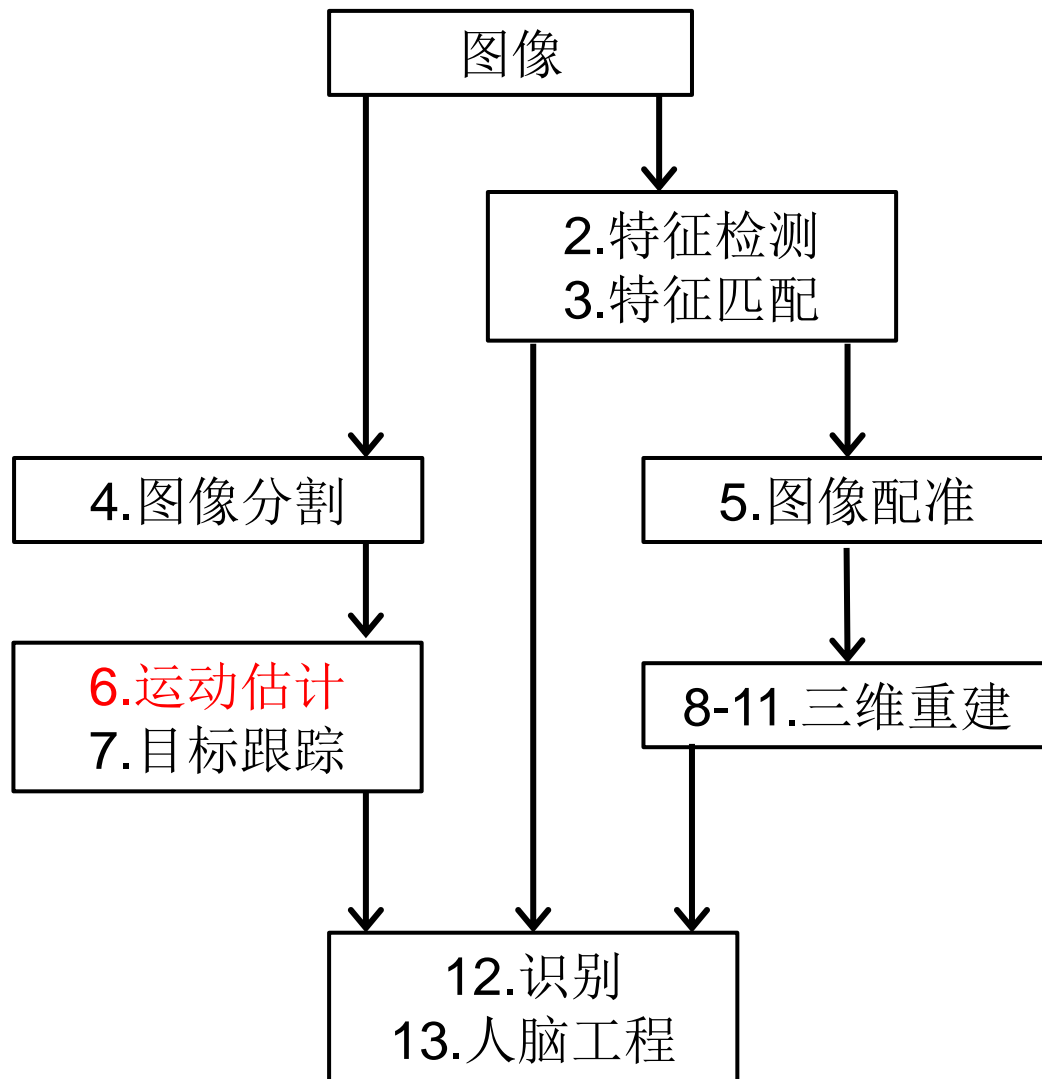


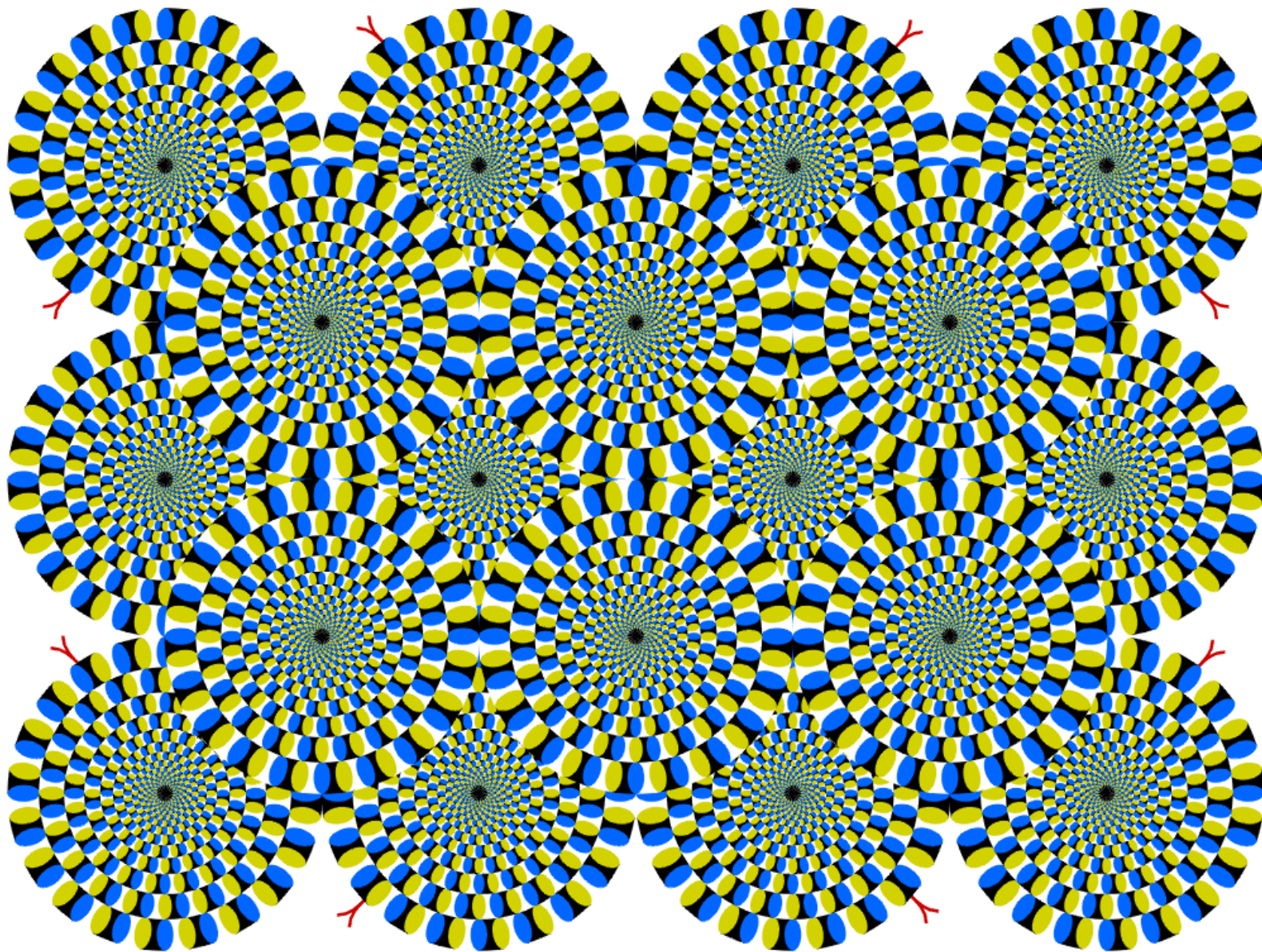
Robot Vision Group

National Laboratory of Pattern Recognition

Institute of Automation, Chinese Academy of Sciences

计算机视觉课程结构图





Akiyoshi's illusion

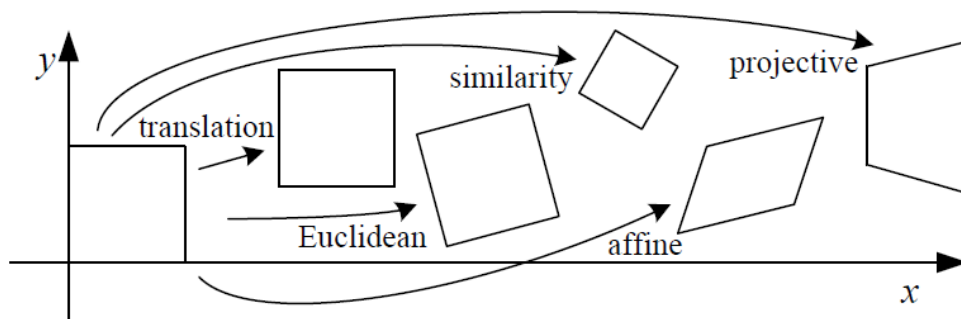
什么是运动估计

输入：图像序列

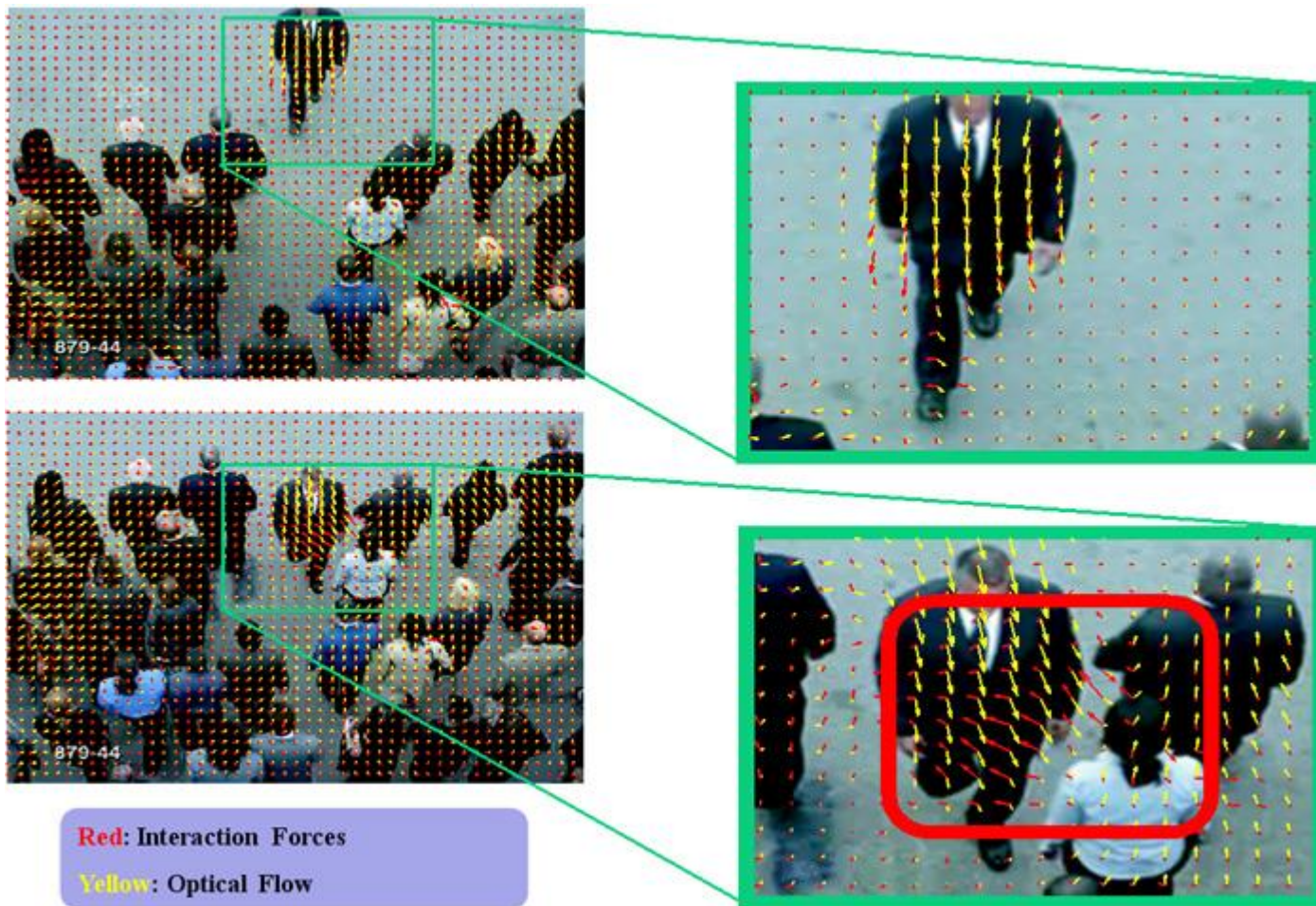
输出：相邻帧间像素点的位置偏移（运动）



运动模型：



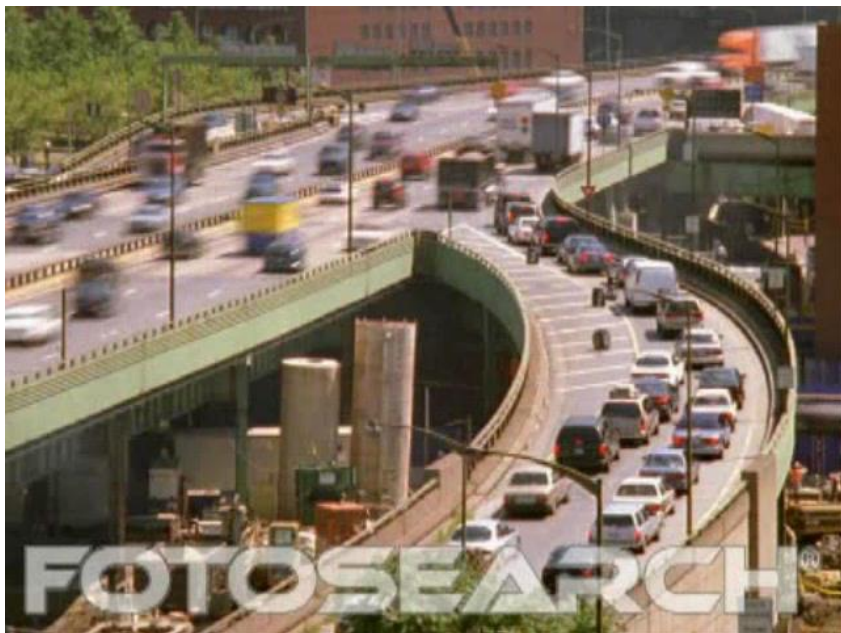
运动估计的应用



人群异常检测

Mehran, Oyama, and Shah, 2009

运动估计的应用



交通流监控

运动估计的应用



无人车

运动估计的基本思想

运动估计的基本任务：

- 给定图像 I 和 T ，寻找 I 中每一个像素点在 T 中的对应点。

运动估计的基本假设：

- 亮度一致：图像 I 和 T 中的对应像素点亮度一致；
- 微小运动：对应像素点间的运动（位移）很小。

运动估计的基本任务是寻找点对应，因此也可以看作一种图像配准，属于基于像素的短基线配准。

运动估计的基本思想

回忆一下上节课的内容（基于特征点与基于像素的配准）

	基于特征点	基于像素
优点	<ul style="list-style-type: none">• 变换模型计算只依赖特征点位置，对图像外观变化有一定鲁棒性；• 能够计算宽基线时的变换；	<ul style="list-style-type: none">• 使用了所有图像数据；• 求解精度通常更高；• 不需要特征检测和匹配；
缺点	<ul style="list-style-type: none">• 没有使用所有图像数据；• 需要进行特征检测和匹配。	<ul style="list-style-type: none">• 容易受到图像外观变化影响；• 一般只适用于窄基线时的变换。

运动估计

平移运动估计

最简单的运动估计是两幅图像 I 和 T 间仅存在2D平移，平移量 \mathbf{u} 可通过最小化误差平方和（Sum of Squared Differences, SSD）求取：

$$E_{SSD}(\mathbf{u}) = \sum_i [I(\mathbf{x}_i + \mathbf{u}) - T(\mathbf{x}_i)]^2 = \sum_i e_i^2$$

其中 $\mathbf{u}=(u,v)$ ， $e_i=I(\mathbf{x}_i+\mathbf{u})-T(\mathbf{x}_i)$ 为残留误差。SSD的基本假设是两幅图像光照一致。

为提高误差函数的鲁棒性，可以使用一些鲁棒误差函数，如Huber函数：

$$E_{SRD}(\mathbf{u}) = \sum_i h(e_i) \quad h(e) = \begin{cases} |e|^2, & \text{if } |e| < \sigma \\ 2\sigma |e| - \sigma^2, & \text{if } |e| \geq \sigma \end{cases}$$

平移运动估计

在对计算速度要求较高的场合（如视频编码），可以使用绝对值误差和（Sum of Absolute Differences, SAD）作为误差函数：

$$E_{SAD}(\mathbf{u}) = \sum_i |I(\mathbf{x}_i + \mathbf{u}) - T(\mathbf{x}_i)| = \sum_i |e_i|$$

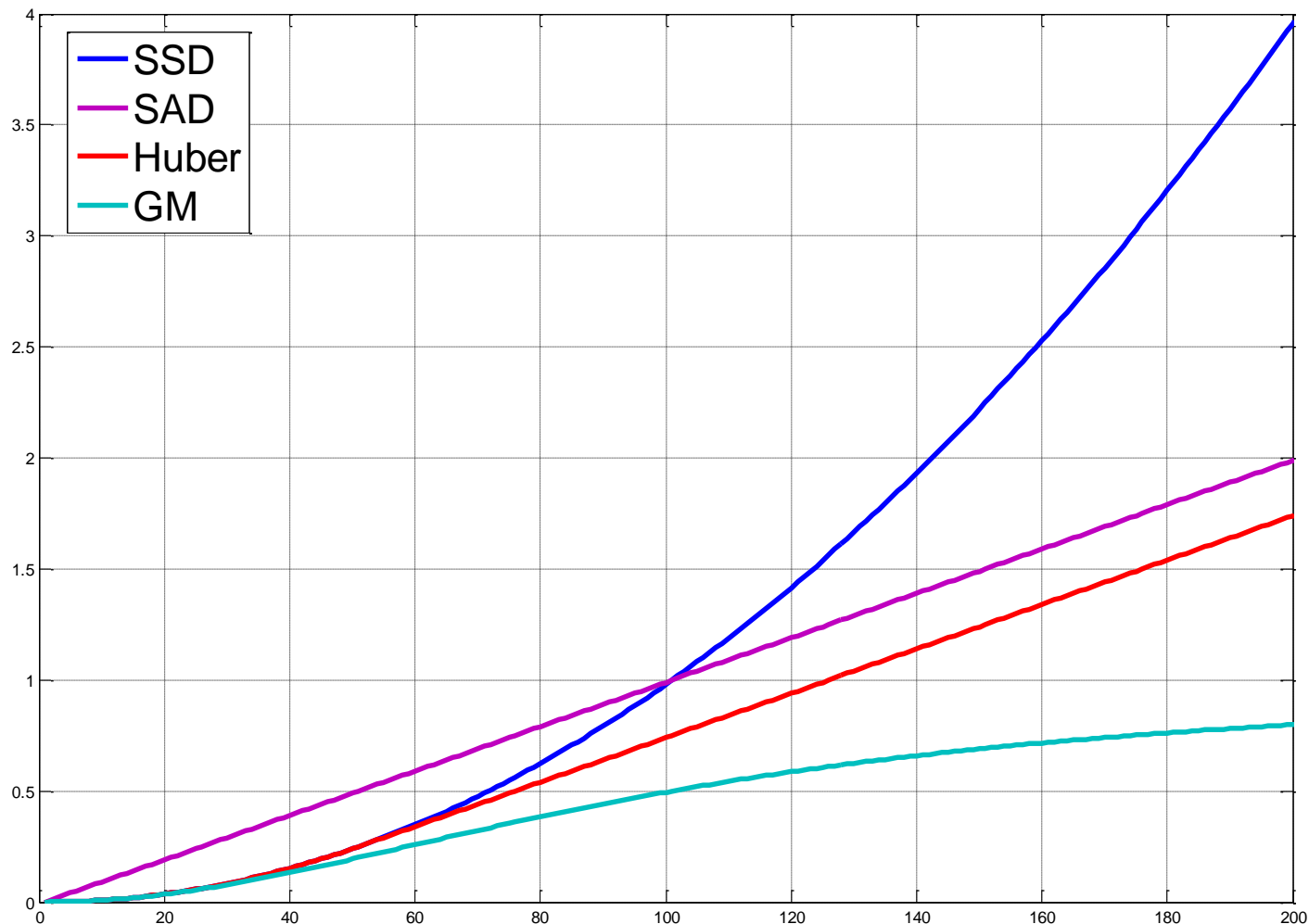
E_{SAD} 不可导，因此无法使用基于梯度下降的优化方法。

另一种在运动估计中常用的是German-McClure误差函数：

$$E_{GM}(\mathbf{u}) = \sum_i \rho_{GM}(e_i) \quad \rho_{GM}(e) = \frac{e^2}{1 + e^2 / a^2}$$

平移运动估计

几种误差函数的比较:



平移运动估计

如果图像 I 和 T 不满足光照恒定条件（如视频中光照突变），则可以将增益（Gain）和偏移（Bias）加入自变量：

$$E_{BG}(\mathbf{u}, \alpha, \beta) = \sum_i [I(\mathbf{x}_i + \mathbf{u}) - (1 + \alpha)T(\mathbf{x}_i) - \beta]^2$$

其中 α 是增益系数， β 是偏移系数。

另一种处理光照变换问题的方法是使用归一化互相关（Normalized Cross Correlation, NCC）：

$$E_{NCC}(\mathbf{u}) = \frac{\sum_i [I(\mathbf{x}_i + \mathbf{u}) - \bar{I}][T(\mathbf{x}_i) - \bar{T}]}{\sqrt{\sum_i [I(\mathbf{x}_i + \mathbf{u}) - \bar{I}]^2} \sqrt{\sum_i [T(\mathbf{x}_i) - \bar{T}]^2}}$$

其中 \bar{I} 和 \bar{T} 分别为图像 I 和 T 的亮度均值。

平移运动估计

除了直接在图像空间定义误差，还可以在将图像转换到频域空间进行误差衡量。

$$F\{I(\mathbf{x} + \mathbf{u})\} = F\{I(\mathbf{x})\}e^{-j\mathbf{u}\omega}$$

图像 $I(\mathbf{x})$ 平移后， $I(\mathbf{x} + \mathbf{u})$ 与 $I(\mathbf{x})$ 的傅里叶变换幅值相同，仅相位发生改变。因此可以使用图像互相关的傅里叶变换来定义误差函数：

$$\begin{aligned} F\{E_{CC}(\mathbf{u})\} &= F\left\{\sum_i I(\mathbf{x}_i + \mathbf{u})T(\mathbf{x}_i)\right\} \\ &= F\{I(\mathbf{x} + \mathbf{u}) * T(\mathbf{x})\} \\ &= F\{I(\mathbf{x} + \mathbf{u})\}F\{I(\mathbf{x})\} \end{aligned}$$

时域卷积等价于频域乘积。

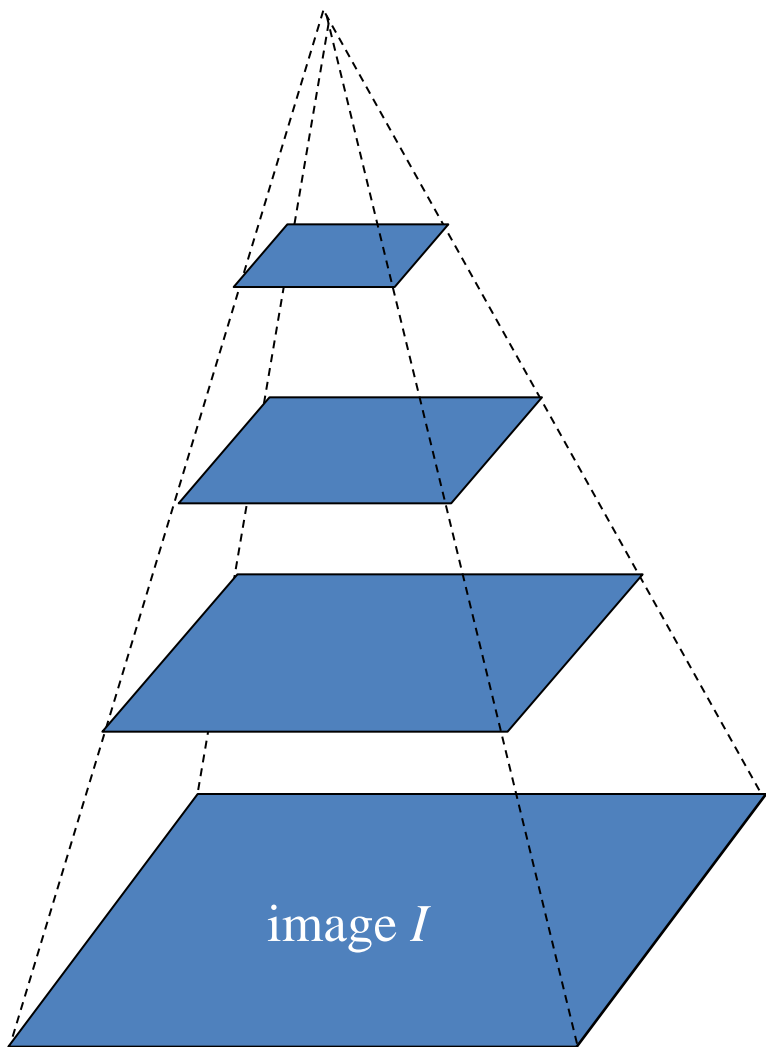
平移运动估计

给定误差函数后，可以通过误差函数最小化获得平移向量最优值。最简单的优化方法是穷举法，及遍历所有可能的运动向量找最优解。

在基于运动补偿的视频压缩中，对图像块使用水平和竖直方向（ ± 16 像素）进行遍历，每个图像块需测试 $32 \times 32 = 1024$ 次。

为加快遍历法的速度，可以使用由粗到细图像金字塔进行遍历求解。

平移运动估计



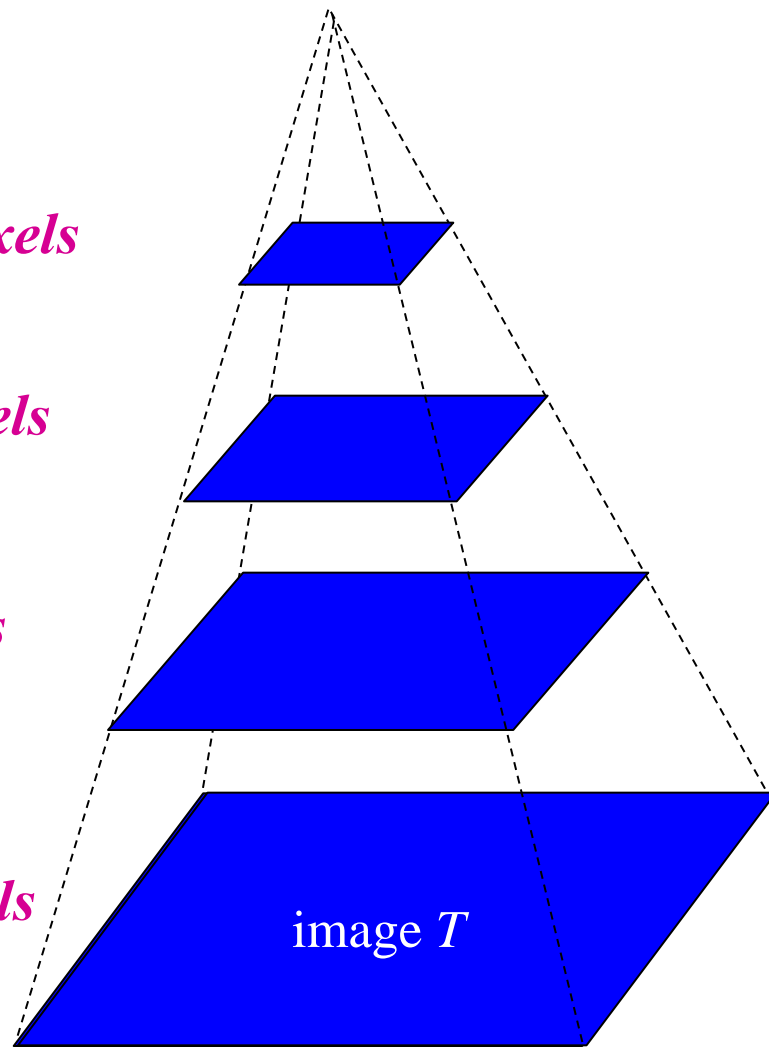
Pyramid of image I

$u=1.25$ pixels

$u=2.5$ pixels

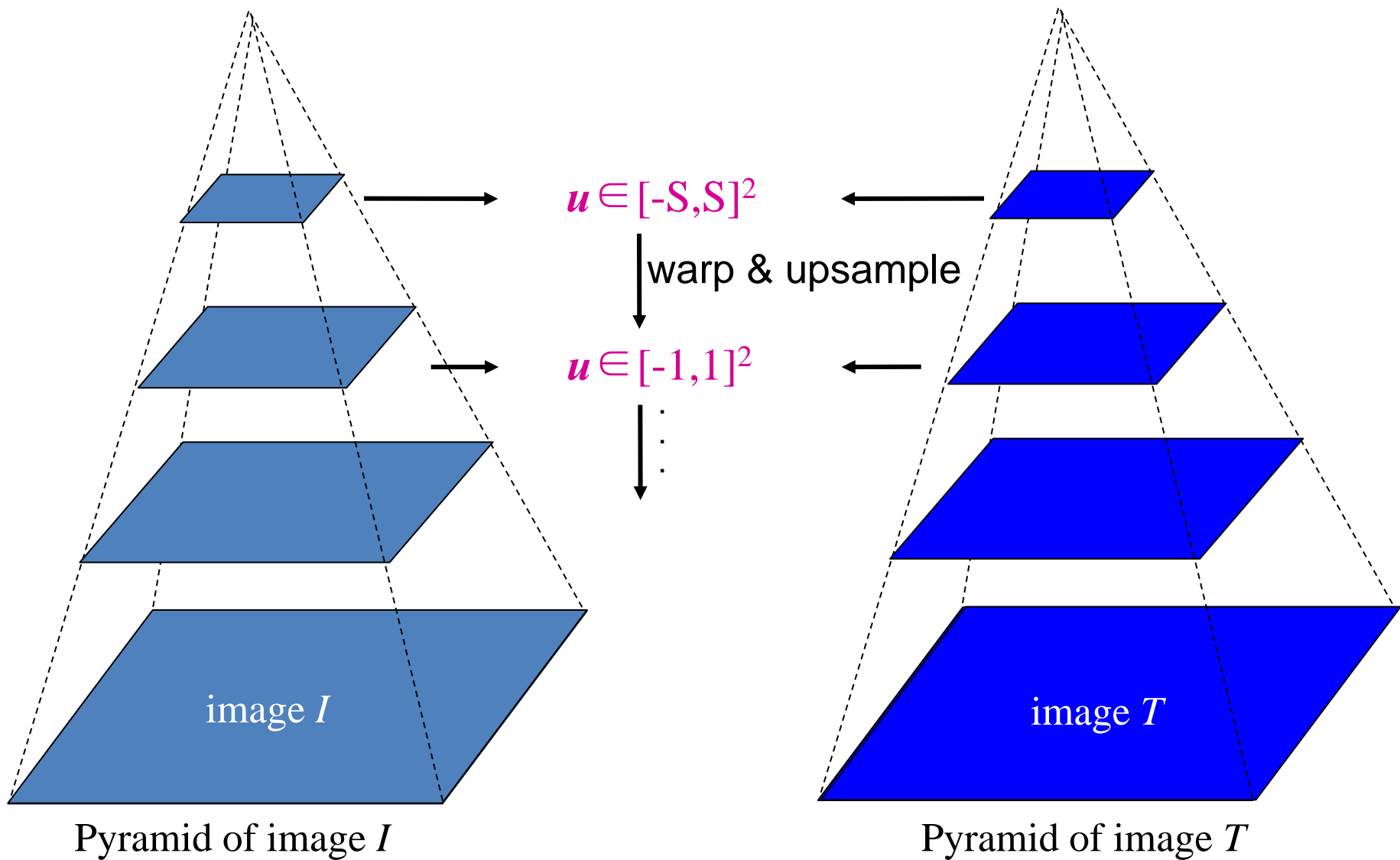
$u=5$ pixels

$u=10$ pixels



Pyramid of image T

平移运动估计



平移运动估计

上述方法能够获得整数像素的运动估计，但在一些对估计精度要求较高的场合（如视频去抖动）需要精度更高（亚像素）的估计结果。

$$E_{SSD}(\mathbf{u}) = \sum_i [I(\mathbf{x}_i + \mathbf{u}) - T(\mathbf{x}_i)]^2$$

能否把误差函数 $E_{SSD}(\mathbf{u})$ 写成线性最小二乘问题？

$$E_{SSD}(\mathbf{u}) = \sum_i (a_i \mathbf{u} - b_i)^2$$

不能， $I(\mathbf{x}_i + \mathbf{u})$ 不是 \mathbf{u} 的线性函数， $I(\mathbf{x}_i + \mathbf{u})$ 是一个查表函数。

平移运动估计

能否把误差函数 $E_{SSD}(\mathbf{u})$ 写成非线性最小二乘问题？

可以，查表函数 $I(\mathbf{x}_i + \mathbf{u})$ 可以看作 \mathbf{u} 的非线性函数。最常用的亚像素精度运动估计方法是Lucas和Kanade于1981年提出的基于高斯牛顿法的最小二乘求解方法。

回顾一下高斯牛顿法求解步骤：

最小化：
$$E_{SSD}(\mathbf{u}) = \sum_i [I(\mathbf{x}_i + \mathbf{u}) - T(\mathbf{x}_i)]^2$$

转化为迭代最小化：
$$E_{LK-SSD}(\Delta\mathbf{u}) = \sum_i [I(\mathbf{x}_i + \mathbf{u} + \Delta\mathbf{u}) - T(\mathbf{x}_i)]^2$$

参数迭代更新：
$$\mathbf{u} \leftarrow \mathbf{u} + \alpha\Delta\mathbf{u}$$

平移运动估计

将 $I(\mathbf{x}_i + \mathbf{u} + \Delta \mathbf{u})$ 在 $\mathbf{x}_i + \mathbf{u}$ 处一阶Taylor展开，将关于 \mathbf{u} 的非线性查表函数转化为关于 $\Delta \mathbf{u}$ 的线性函数：

$$\begin{aligned} E_{LK-SSD}(\Delta \mathbf{u}) &= \sum_i [I(\mathbf{x}_i + \mathbf{u} + \Delta \mathbf{u}) - T(\mathbf{x}_i)]^2 \\ &\approx \sum_i [I(\mathbf{x}_i + \mathbf{u}) + \nabla I(\mathbf{x}_i + \mathbf{u}) \Delta \mathbf{u} - T(\mathbf{x}_i)]^2 \\ &= \sum_i [J_I(\mathbf{x}_i + \mathbf{u}) \Delta \mathbf{u} + e_i]^2 \end{aligned}$$

其中：

$$J_I = \nabla I = \left(\frac{\partial I}{\partial u}, \frac{\partial I}{\partial v} \right) \qquad e_i = I(\mathbf{x}_i + \mathbf{u}) - T(\mathbf{x}_i)$$

$I(\mathbf{x} + \mathbf{u})$ 的雅克比矩阵

运动误差

平移运动估计

将 $E_{SSD}(\Delta \mathbf{u})$ 进一步展开:

$$\begin{aligned} E_{LK-SSD}(\Delta \mathbf{u}) &\approx \sum_i [J_I(\mathbf{x}_i + \mathbf{u})\Delta \mathbf{u} + e_i]^2 \\ &= \sum_i [\Delta \mathbf{u}^T J_I^T(\mathbf{x}_i + \mathbf{u}) J_I(\mathbf{x}_i + \mathbf{u}) \Delta \mathbf{u} + 2J_I^T(\mathbf{x}_i + \mathbf{u}) e_i \Delta \mathbf{u} + e_i^2] \end{aligned}$$

使 $E_{LK-SSD}(\Delta \mathbf{u})$ 最小化的 $\Delta \mathbf{u}$ 满足:

$$\nabla E_{LK-SSD}(\Delta \mathbf{u}) = \sum_i [2J_I^T(\mathbf{x}_i + \mathbf{u}) J_I(\mathbf{x}_i + \mathbf{u}) \Delta \mathbf{u} + 2J_I^T(\mathbf{x}_i + \mathbf{u}) e_i] = 0$$

写成紧凑形式: $\mathbf{A}\Delta \mathbf{u} = \mathbf{b}$

$$\mathbf{A} = \sum_i J_I^T(\mathbf{x}_i + \mathbf{u}) J_I(\mathbf{x}_i + \mathbf{u}) \quad \mathbf{b} = -\sum_i J_I^T(\mathbf{x}_i + \mathbf{u}) e_i$$

平移运动估计

$$\mathbf{A}\Delta\mathbf{u} = \mathbf{b}$$

$$\text{其中: } \mathbf{A} = \sum_i J_I^T(\mathbf{x}_i + \mathbf{u}) J_I(\mathbf{x}_i + \mathbf{u}) \quad \mathbf{b} = -\sum_i J_I^T(\mathbf{x}_i + \mathbf{u}) e_i$$

$$J_I = \nabla I = \left(\frac{\partial I}{\partial u}, \frac{\partial I}{\partial v} \right) \quad e_i = I(\mathbf{x}_i + \mathbf{u}) - T(\mathbf{x}_i)$$

Gauss-Newton方向为: $\Delta\mathbf{u} = -\mathbf{A}^{-1}\mathbf{b}$

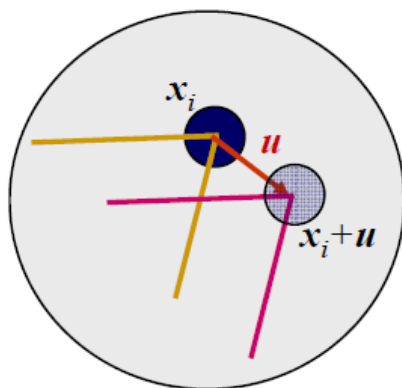
回忆一下Newton法方向: $\Delta\mathbf{u} = -\mathbf{H}^{-1}\mathbf{g}$

因此, $\Delta\mathbf{u} = -\mathbf{A}^{-1}\mathbf{b}$ 中的 \mathbf{A} 是对Hessian矩阵的高斯牛顿近似, \mathbf{b} 是梯度权值残差向量。

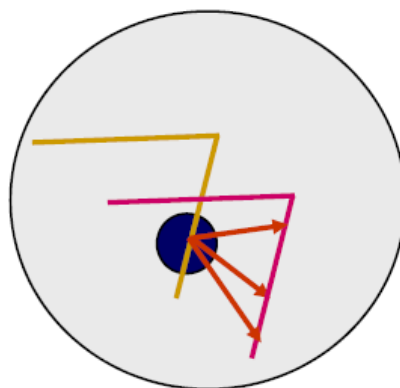
平移运动估计

使用高斯牛顿法最小化 $E_{LK-SSD}(\mathbf{u})$ 的迭代终止条件是 $\|\Delta\mathbf{u}\|$ 小于一定阈值（如1/10像素）。

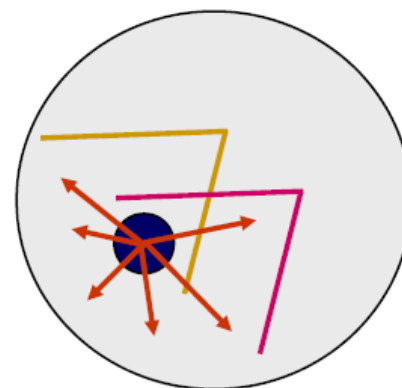
需要注意的是，如果在图像中分块估计 $\Delta\mathbf{u}$ ，那么在图像中某些区域（如边缘、弱纹理区域等），可能出现 $\mathbf{A}\Delta\mathbf{u}=\mathbf{b}$ 中的矩阵 \mathbf{A} 欠定的情况（ \mathbf{A} 的某些特征值为零或趋近于零），此时 $\Delta\mathbf{u}$ 非唯一解，这种情况称为孔径问题（Aperture problem）。



(a)

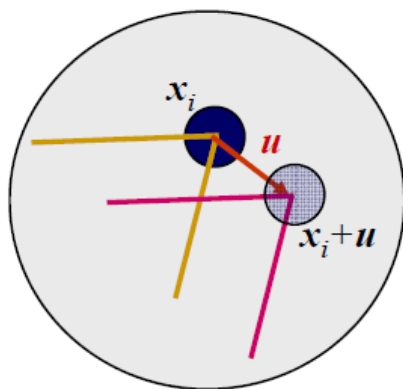


(b)



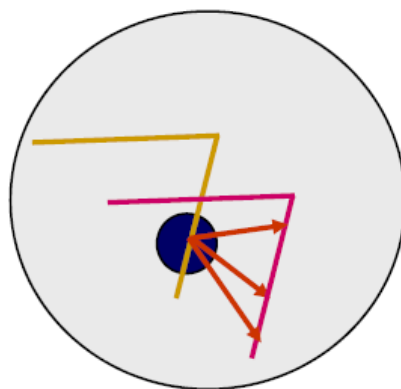
(c)

平移运动估计



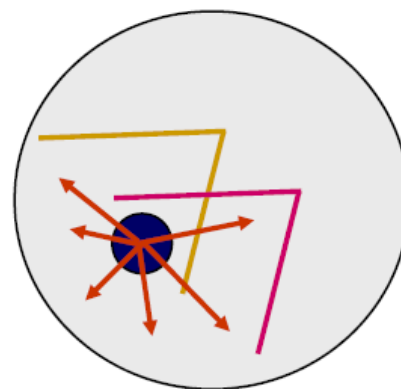
(a)

角点



(b)

边缘点



(c)

弱纹理点

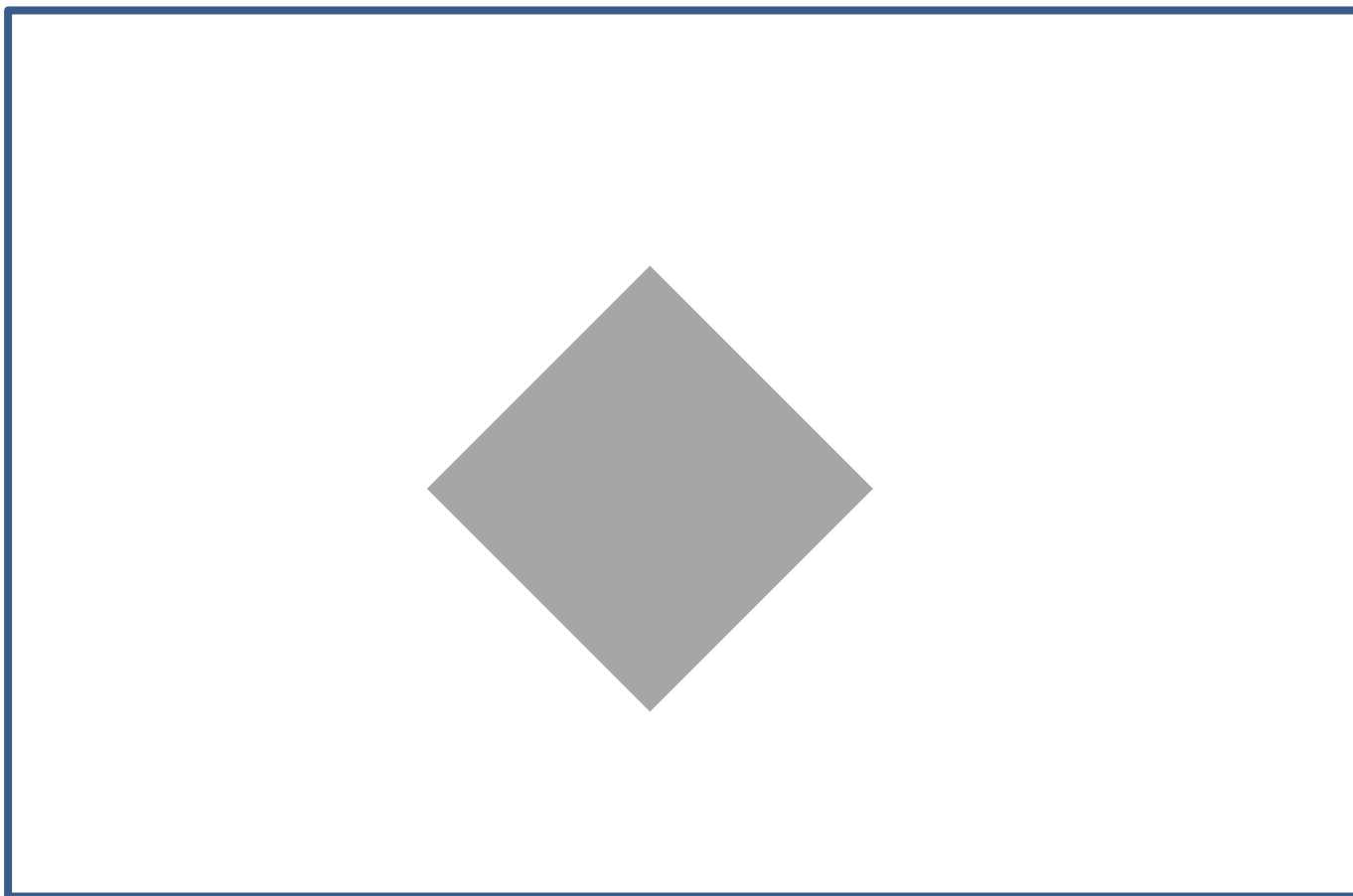
是不是似曾相识？

A就是Harris角点检测子使用的自相关矩阵。

$$\mathbf{A}(x) = \sum J_I^T J_I = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

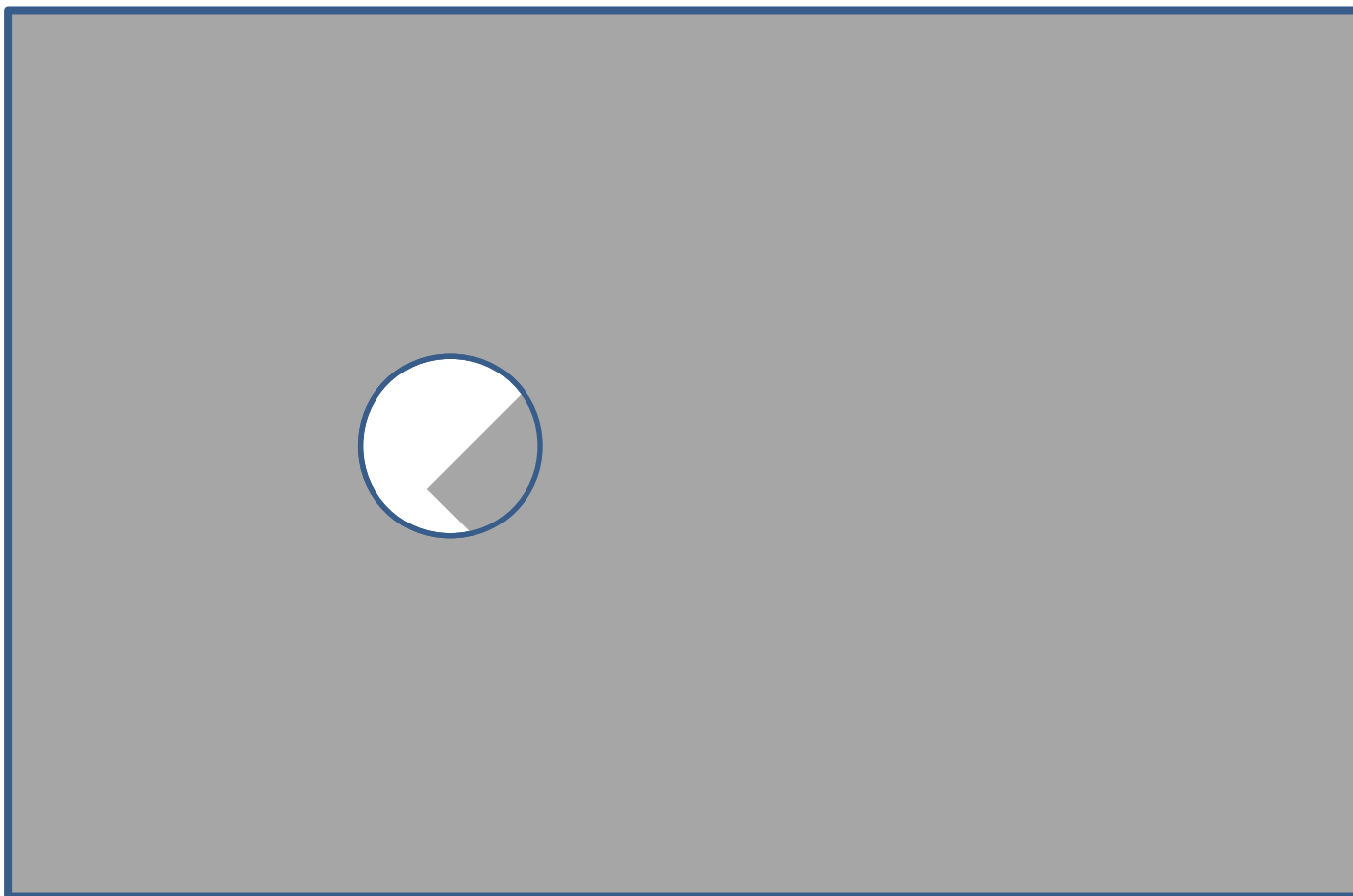
平移运动估计

孔径问题（Aperture problem）



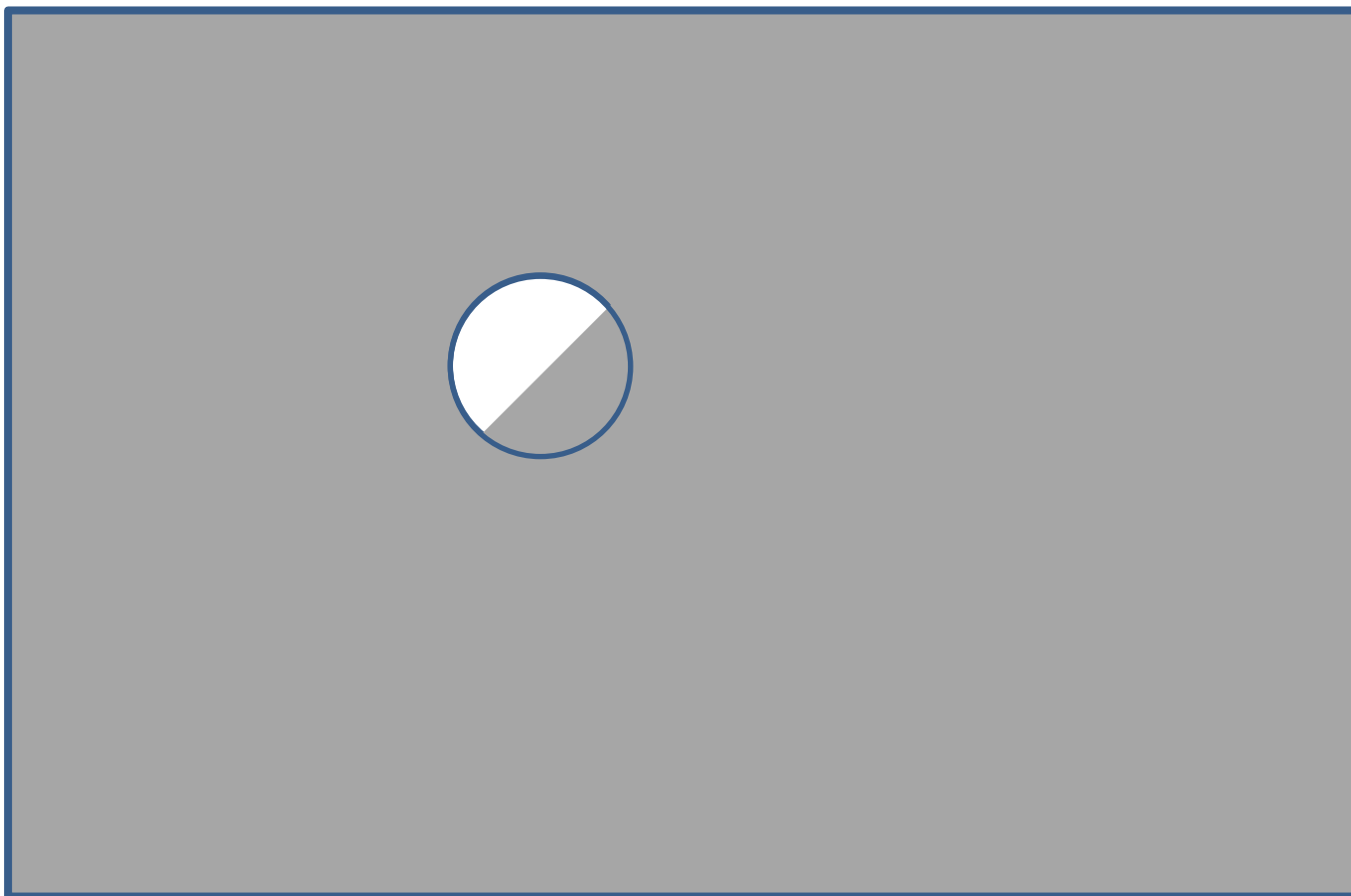
平移运动估计

孔径问题（Aperture problem）



平移运动估计

孔径问题（Aperture problem）



平移运动估计

应用：视频编码中基于块的运动补偿



参数化运动估计

当运动估计所使用的运动模型比平以运动更复杂时，如仿射变换，参数空间纬度远高于平移的二维，因此不适合使用穷举法遍历整个参数空间。此时一般使用Lucas-Kanade算法进行非线性优化。

令 $\mathbf{W}(\mathbf{x};\mathbf{p})$ 表示图像点 \mathbf{x} 经过参数 \mathbf{p} 构成的运动模型变换后的图像点，则参数化运动估计的最小化目标函数为：

$$E_{LK-PM}(\mathbf{p}) = \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) - T(\mathbf{x}_i)]^2$$

则每一步L-K迭代可以写成如下最小化问题：

$$E_{LK-PM}(\Delta\mathbf{p}) = \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p} + \Delta\mathbf{p})) - T(\mathbf{x}_i)]^2$$
$$\mathbf{p} \leftarrow \mathbf{p} + \alpha\Delta\mathbf{p}$$

参数化运动估计

$$\begin{aligned} E_{LK-PM}(\Delta \mathbf{p}) &= \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p} + \Delta \mathbf{p})) - T(\mathbf{x}_i)]^2 \\ &\approx \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) + J_I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))\Delta \mathbf{p} - T(\mathbf{x}_i)]^2 \\ &= \sum_i [J_I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))\Delta \mathbf{p} + e_i]^2 \end{aligned}$$

$$e_i = I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) - T(\mathbf{x}_i) \quad \text{图像误差}$$

$$\begin{aligned} J_I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) &= \frac{\partial I}{\partial \mathbf{p}} = \nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) \frac{\partial \mathbf{W}}{\partial \mathbf{p}}(\mathbf{x}_i) \\ &= \nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{W}}(\mathbf{x}_i) \end{aligned}$$

上式表示 $I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))$ 的雅可比矩阵 J_I 等于 $I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))$ 的梯度图像乘以运动模型 $\mathbf{W}(\cdot)$ 的雅可比矩阵。

参数化运动估计

常见运动模型 $\mathbf{W}(\cdot)$ 的雅可比矩阵:

Transform	Matrix	Parameters p	Jacobian J
translation	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$	(t_x, t_y)	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Euclidean	$\begin{bmatrix} c_\theta & -s_\theta & t_x \\ s_\theta & c_\theta & t_y \end{bmatrix}$	(t_x, t_y, θ)	$\begin{bmatrix} 1 & 0 & -s_\theta x - c_\theta y \\ 0 & 1 & c_\theta x - s_\theta y \end{bmatrix}$
similarity	$\begin{bmatrix} 1+a & -b & t_x \\ b & 1+a & t_y \end{bmatrix}$	(t_x, t_y, a, b)	$\begin{bmatrix} 1 & 0 & x & -y \\ 0 & 1 & y & x \end{bmatrix}$
affine	$\begin{bmatrix} 1+a_{00} & a_{01} & t_x \\ a_{10} & 1+a_{11} & t_y \end{bmatrix}$	$(t_x, t_y, a_{00}, a_{01}, a_{10}, a_{11})$	$\begin{bmatrix} 1 & 0 & x & y & 0 & 0 \\ 0 & 1 & 0 & 0 & x & y \end{bmatrix}$
projective	$\begin{bmatrix} 1+h_{00} & h_{01} & h_{02} \\ h_{10} & 1+h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix}$	$(h_{00}, h_{01}, \dots, h_{21})$	$\frac{1}{D} \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y \end{bmatrix}$

参数化运动估计

最小化目标函数： $E_{LK-PM}(\Delta \mathbf{p}) \approx \sum_i [J_I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))\Delta \mathbf{p} + e_i]^2$

$$J_I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) = \nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))J_{\mathbf{w}}(\mathbf{x}_i)$$

$\Delta \mathbf{p}$ 最优解满足： $\mathbf{A}\Delta \mathbf{p} = \mathbf{b}$

上式中的（高斯牛顿近似）Hessian矩阵 \mathbf{A} 和梯度权值残差向量 \mathbf{b} 分别为：

$$\mathbf{A} = \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))J_{\mathbf{w}}(\mathbf{x}_i)]^T [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))J_{\mathbf{w}}(\mathbf{x}_i)]$$

$$\mathbf{b} = -\sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))J_{\mathbf{w}}(\mathbf{x}_i)]^T e_i$$

Gauss-Newton方向： $\Delta \mathbf{p} = -\mathbf{A}^{-1}\mathbf{b}$

参数化运动估计

参数化运动估计中的 \mathbf{A} 和 \mathbf{b} :

$$\mathbf{A} = \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]^T [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]$$

$$\mathbf{b} = -\sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]^T e_i$$

对比一下平移运动 ($\mathbf{W}(\mathbf{x}; \mathbf{u}) = \mathbf{x} + \mathbf{u}$) 中的 \mathbf{A} 和 \mathbf{b} :

$$\mathbf{A} = \sum_i \nabla I^T(\mathbf{x}_i + \mathbf{u}) \nabla I(\mathbf{x}_i + \mathbf{u})$$

$$\mathbf{b} = -\sum_i \nabla I^T(\mathbf{x}_i + \mathbf{u}) e_i \quad \text{平移的雅可比矩阵: } J_{\mathbf{w}}^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

参数化运动估计

回顾一下使用Lucas-Kanade算法进行参数化运动估计的流程

最小化目标函数：
$$E_{LK-PM}(\mathbf{p}) = \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) - T(\mathbf{x}_i)]^2$$

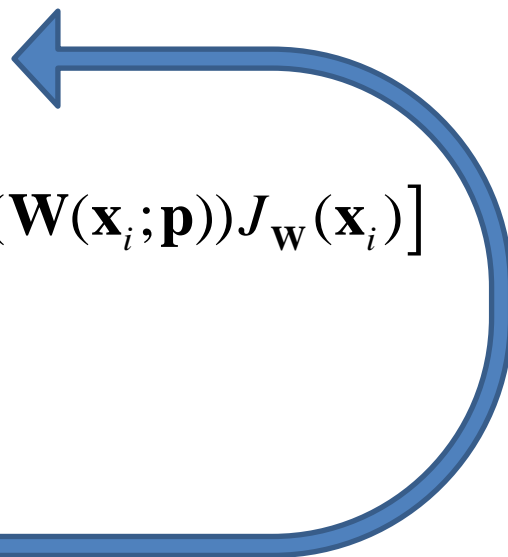
1、给定初始运动参数： \mathbf{p}

2、Gauss-Newton方向： $\Delta\mathbf{p} = -\mathbf{A}^{-1}\mathbf{b}$

$$\mathbf{A} = \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]^T [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]$$

$$\mathbf{b} = -\sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]^T e_i$$

3、运动参数更新： $\mathbf{p} \leftarrow \mathbf{p} + \alpha \Delta\mathbf{p}$



参数化运动估计

回顾一下使用Lucas-Kanade算法进行参数化运动估计的流程

最小化目标函数:
$$E_{LK-PM}(\mathbf{p}) = \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) - T(\mathbf{x}_i)]^2$$

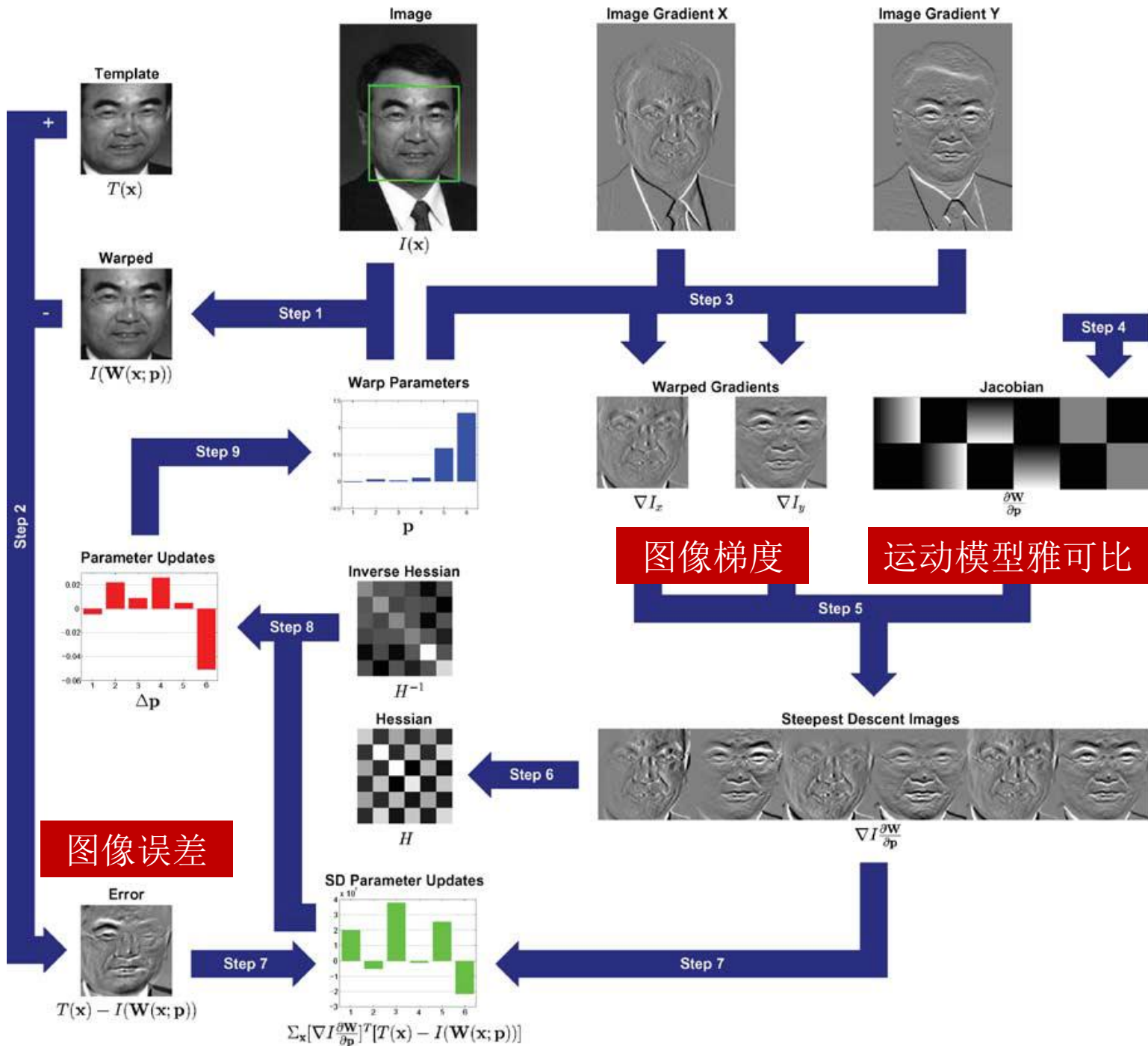
关键计算参数:

$$\mathbf{A} = \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) \underline{J_{\mathbf{w}}(\mathbf{x}_i)}]^T [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) \underline{J_{\mathbf{w}}(\mathbf{x}_i)}]$$

$$\mathbf{b} = -\sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) \underline{J_{\mathbf{w}}(\mathbf{x}_i)}]^T \underline{e_i}$$

图像梯度 $\nabla I = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$ 运动模型 \mathbf{W} 雅可比矩阵 $J_{\mathbf{w}} = \frac{\partial J}{\partial \mathbf{p}}$

图像误差 $e = I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - T(\mathbf{x})$



参数化运动估计

Lucas-Kanade算法在每一步迭代过程中使用运动模型 $\mathbf{W}(\cdot; \mathbf{p})$ 将源图像 I 向模版图像 T 变换，因此也称为正向合成算法（forward compositional algorithm）。

正向合成的Lucas-Kanade算法在每一步需要对 I 的梯度图像进行变换，并计算运动模型雅可比矩阵，计算复杂度较高。为此，Baker和Matthews与2004年提出一种逆向合成算法（inverse compositional algorithm）。

逆向合成算法的核心思想是将模版图像 T 向源图像 I 变换，从而避免反复计算图像梯度和雅可比矩阵。

参数化运动估计

正向合成Lucas-Kanade算法:

$$\begin{aligned} E_{LK-PM}(\Delta \mathbf{p}) &= \sum_i [I(\mathbf{W}(\mathbf{x}_i; \mathbf{p} + \Delta \mathbf{p})) - T(\mathbf{x}_i)]^2 \\ &\approx \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{W}}(\mathbf{x}_i) \Delta \mathbf{p} + e_i]^2 \quad \mathbf{p} \leftarrow \mathbf{p} + \Delta \mathbf{p} \end{aligned}$$

逆向合成Lucas-Kanade算法:

$$\begin{aligned} E_{LK-PM-INV}(\Delta \mathbf{p}) &= \sum_i [T(\mathbf{W}(\mathbf{x}_i; \Delta \mathbf{p})) - I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))]^2 \\ &\approx \sum_i [T(\mathbf{x}_i) + J_T(\mathbf{x}_i) \Delta \mathbf{p} - I(\mathbf{W}(\mathbf{x}_i; \mathbf{p}))]^2 \\ &= \sum_i [\nabla T(\mathbf{x}_i) J_{\mathbf{W}}(\mathbf{x}_i) \Delta \mathbf{p} - e_i]^2 \end{aligned}$$

$$\mathbf{W}(\mathbf{x}; \mathbf{p}) \leftarrow \mathbf{W}(\mathbf{x}; \mathbf{p}) \circ \mathbf{W}(\mathbf{x}; \Delta \mathbf{p})^{-1}$$

参数化运动估计

正向合成Lucas-Kanade算法:

$$E_{LK-PM}(\Delta \mathbf{p}) \approx \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i) \Delta \mathbf{p} + e_i]^2$$

$$\mathbf{A} = \sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]^T [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]$$

$$\mathbf{b} = -\sum_i [\nabla I(\mathbf{W}(\mathbf{x}_i; \mathbf{p})) J_{\mathbf{w}}(\mathbf{x}_i)]^T e_i$$

\mathbf{A} 和 \mathbf{b} 随着 \mathbf{p} 的变化而变化

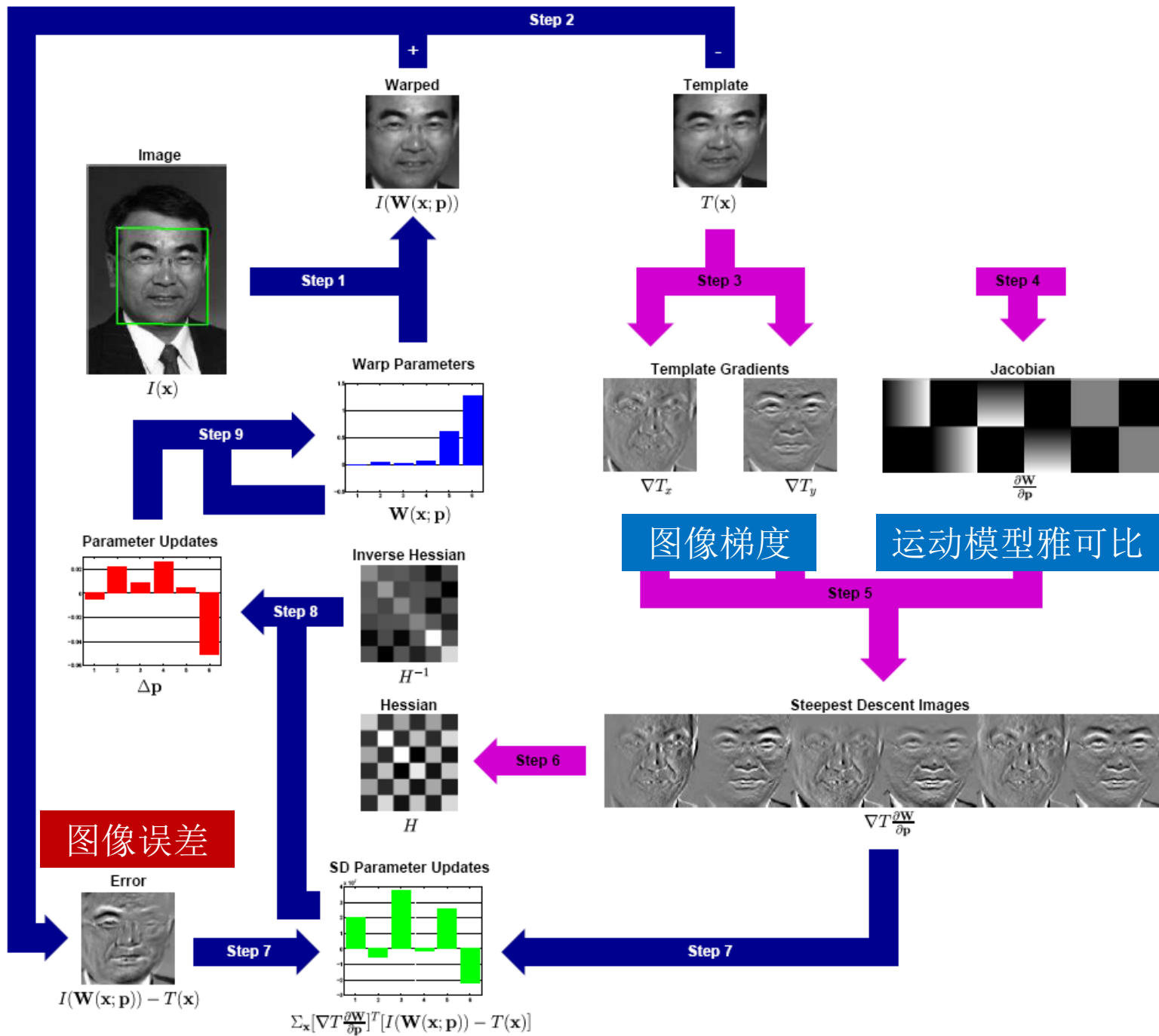
逆向合成Lucas-Kanade算法:

$$E_{LK-PM-INV}(\Delta \mathbf{p}) \approx \sum_i [\nabla T(\mathbf{x}_i) J_{\mathbf{w}}(\mathbf{x}_i) \Delta \mathbf{p} - e_i]^2$$

$$\mathbf{A} = \sum_i [\nabla T(\mathbf{x}_i) J_{\mathbf{w}}(\mathbf{x}_i)]^T [\nabla T(\mathbf{x}_i) J_{\mathbf{w}}(\mathbf{x}_i)]$$

$$\mathbf{b} = -\sum_i [\nabla T(\mathbf{x}_i) J_{\mathbf{w}}(\mathbf{x}_i)]^T e_i$$

只有 e_i 随着 \mathbf{p} 的变化而变化



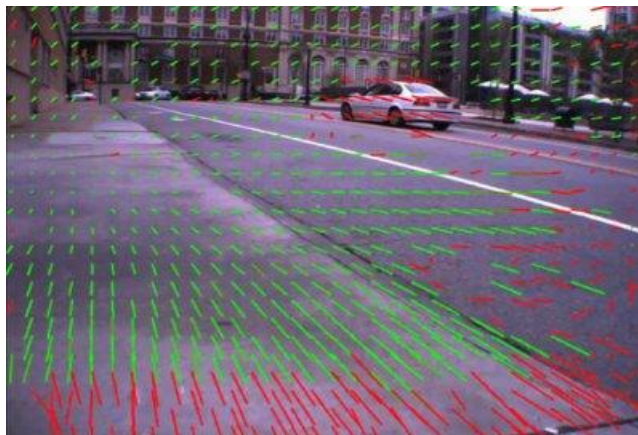
参数化运动估计

应用：视频去抖动



光流运动估计

最具一般性的运动估计方式是估计每一个像素的独立运动，称为光流（Optical Flow）估计。



当人的眼睛观察运动物体时，物体的景象在人眼的视网膜上形成一系列连续变化的图像，这一系列连续变化的信息不断“流过”视网膜（即图像平面），好像一种光的“流”，故称之为光流。

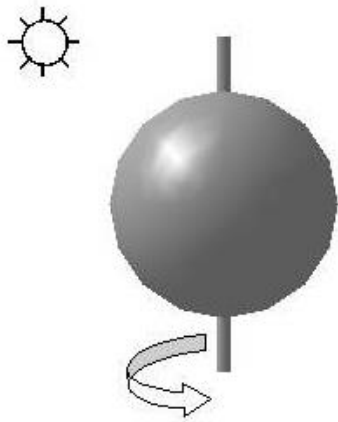
光流运动估计

点的光流和点的实际运动的区别和联系：

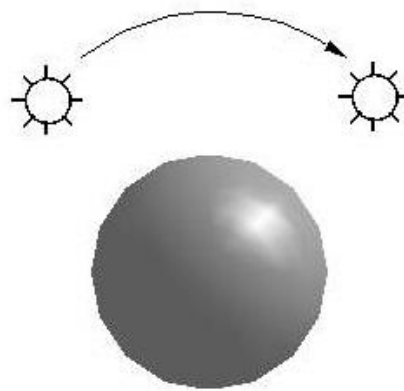
- 运动场（Motion Field）：一个运动物体在空间产生一个三维的速度场，运动前后空间对应点在图象上的投影形成一个二维运动场。
- 光流场（Optical Flow Field）：是指图像亮度模式的表面运动，是一个二维矢量场。
- 理想情况下，光流场与运动场是互相吻合的，但实际上并不一定如此。研究光流场的目的就是为从序列图像中近似计算不能直接得到的运动场。

光流运动估计

点的光流和点的实际运动的区别和联系：



光流等于零，但运动场
不等于零。



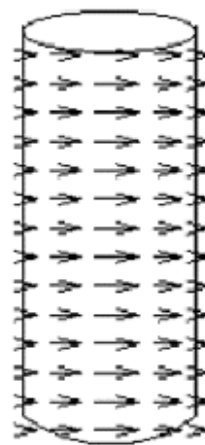
光流不等于零，但运动场
为零。

光流运动估计

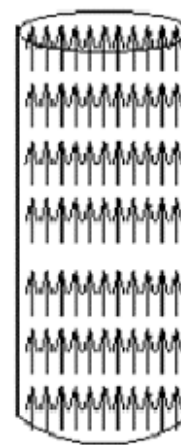
点的光流和点的实际运动的区别和联系：



Barber's pole



Motion field



Optical flow

光流运动估计

光流运动估计使用的目标函数为：

$$E_{SSD-OF}(\{\mathbf{u}_i\}) = \sum_i [I(\mathbf{x}_i + \mathbf{u}_i) - T(\mathbf{x}_i)]^2$$

上式求和符号内每一项都是独立的，因此最小化 E_{SSD-OF} 等价于最小化如下 N 个目标函数（ N 为像素点数量）：

$$E_{SSD-OF}^i(\mathbf{u}_i) = [I(\mathbf{x}_i + \mathbf{u}_i) - T(\mathbf{x}_i)]^2, \quad i = 1, \dots, N$$

如果使用L-K法求解这个非线性最小二乘问题，则每一步L-K迭代可以写成如下最小化问题：

$$\begin{aligned} E_{SSD-OF}^i(\{\Delta\mathbf{u}_i\}) &= [I(\mathbf{x}_i + \mathbf{u}_i + \Delta\mathbf{u}_i) - T(\mathbf{x}_i)]^2 \\ &\approx [I(\mathbf{x}_i + \mathbf{u}_i) + \nabla I(\mathbf{x}_i + \mathbf{u}_i)\Delta\mathbf{u}_i - T(\mathbf{x}_i)]^2 \\ &= [J_I(\mathbf{x}_i + \mathbf{u}_i)\Delta\mathbf{u}_i + e_i]^2 \end{aligned}$$

光流运动估计

最小化目标函数： $E_{SSD-OF}^i(\Delta \mathbf{u}_i) \approx [J_I(\mathbf{x}_i + \mathbf{u}_i)\Delta \mathbf{u}_i + e_i]^2$

其中： $J_I = \nabla I = (\frac{\partial I}{\partial u}, \frac{\partial I}{\partial v})$ $e_i = I(\mathbf{x}_i + \mathbf{u}_i) - T(\mathbf{x}_i)$

则（高斯牛顿近似）Hessian矩阵 \mathbf{A} 和梯度权值残差向量 \mathbf{b} 为：

$$\mathbf{A}_i = J_I^T(\mathbf{x}_i + \mathbf{u}_i)J_I(\mathbf{x}_i + \mathbf{u}_i) \quad \mathbf{b}_i = -J_I^T(\mathbf{x}_i + \mathbf{u}_i)e_i$$

$\Delta \mathbf{u}_i = -\mathbf{A}_i^{-1}\mathbf{b}_i$ 是否有解？

无解，因为 2×2 矩阵 \mathbf{A}_i 秩为1，不可逆。

比如：

$$\begin{bmatrix} 1 \\ 3 \end{bmatrix} [1 \quad 3] = \begin{bmatrix} 1 & 3 \\ 3 & 9 \end{bmatrix}$$

线性相关

光流运动估计

光流运动估计目标函数：

$$E_{SSD-OF}(\{\mathbf{u}_i\}) = \sum_i [I(\mathbf{x}_i + \mathbf{u}_i) - T(\mathbf{x}_i)]^2$$

直接求解 $\{\mathbf{u}_i\}$ 是一个欠定问题，需要添加其他约束。

根据约束类型的不同可以把光流运动估计方法分为：局部优化方法（每个像素光流单独求解）和全局优化方法（所有像素光流一起求解）。

光流运动估计

可以添加局部窗口约束，即在局部小区域内的像素点具有相同的 (u,v) 。

以每个像素为中心给定一个 $m \times m$ 的局部窗口，假设在窗口内的像素点具有同样的光流：

$$E_{SSD-OF}^i(\mathbf{u}_i) = \sum_{j \in P_i} [I(\mathbf{x}_j + \mathbf{u}_i) - T(\mathbf{x}_j)]^2$$

P_i 表示以像素 \mathbf{x}_i 为中心的一个 $m \times m$ 窗口。

问题转化为一个 $m \times m$ 局部图像的平移参数估计问题。对每个像素依次构造上述平移估计问题，可以求解整个图像的光流运动场。

光流运动估计

再来看一看光流运动估计的目标函数，如果 \mathbf{u}_i 很小，可将 $I(\mathbf{x}_i + \mathbf{u}_i)$ 在 \mathbf{x}_i 处一阶Taylor展开：

$$\begin{aligned} E_{SSD-OF}(\{\mathbf{u}_i\}) &= \sum_i [I(\mathbf{x}_i + \mathbf{u}_i) - T(\mathbf{x}_i)]^2 \\ &\approx \sum_i [I(\mathbf{x}_i) + \nabla I(\mathbf{x}_i) \mathbf{u}_i - T(\mathbf{x}_i)]^2 \end{aligned}$$

\mathbf{u}_i 很小

注意和L-K法的区别：

$$\begin{aligned} E_{SSD-OF}(\{\Delta \mathbf{u}_i\}) &= \sum_i [I(\mathbf{x}_i + \mathbf{u}_i + \Delta \mathbf{u}_i) - T(\mathbf{x}_i)]^2 \\ &\approx \sum_i [I(\mathbf{x}_i + \mathbf{u}_i) + \nabla I(\mathbf{x}_i + \mathbf{u}_i) \Delta \mathbf{u}_i - T(\mathbf{x}_i)]^2 \end{aligned}$$

$\Delta \mathbf{u}_i$ 很小

光流运动估计

再来看一看光流运动估计的目标函数，如果 \mathbf{u}_i 很小，可将 $I(\mathbf{x}_i + \mathbf{u}_i)$ 在 \mathbf{x}_i 处一阶Taylor展开：

$$\begin{aligned} E_{SSD-OF}(\{\mathbf{u}_i\}) &= \sum_i [I(\mathbf{x}_i + \mathbf{u}_i) - T(\mathbf{x}_i)]^2 \\ &\approx \sum_i [I(\mathbf{x}_i) + \nabla I(\mathbf{x}_i) \mathbf{u}_i - T(\mathbf{x}_i)]^2 \end{aligned}$$

E_{SSD-OF} 最小，相当于： $I(\mathbf{x}_i) + \nabla I(\mathbf{x}_i) \mathbf{u}_i - T(\mathbf{x}_i) = 0$

可以进一步写成： $I_{x_i} u_i + I_{y_i} v_i + I_t = 0$

其中： $\mathbf{x}_i = (x_i, y_i)$ $\mathbf{u}_i = (u_i, v_i)$ I_t 为 I 和 T 两帧差分

$I_x u + I_y v + I_t = 0$ 称为光流约束方程

光流运动估计

光流约束方程是否可解？

$$I_x u + I_y v + I_t = 0$$

一个方程两个未知数，不可解。需要为每一个像素添加其他约束，比如刚才的局部窗口约束。

比如使用一个 5×5 的局部窗口，则每个像素 \mathbf{x} 可以得到25个约束方程（包括 \mathbf{x} 本身的光流方程）：

$$I_x(\mathbf{p}_i)u + I_y(\mathbf{p}_i)v + I_t(\mathbf{p}_i) = 0, \quad i = 1, \dots, 25$$

$\{\mathbf{p}_1, \dots, \mathbf{p}_{25}\}$ 为以 \mathbf{x} 为中心的 5×5 窗口内的25个像素点。

光流运动估计

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

$\mathbf{A}_{25 \times 2} \quad \mathbf{u}_{2 \times 1} \quad \mathbf{b}_{25 \times 1}$





$$\min \|\mathbf{A}\mathbf{u} - \mathbf{b}\|^2$$

通过求解上述线性最小二乘问题得到像素 \mathbf{x} 处的运动参数 \mathbf{u} 。

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

光流运动估计

对比一下两种局部窗口方法：

误差平方和+局部窗口约束	光流约束方程+局部窗口约束
$E_{SSD-OF}(\mathbf{u}_i) = [I(\mathbf{x}_j + \mathbf{u}_i) - T(\mathbf{x}_j)]^2$ <div></div> $\min \sum_{j \in P_i} [I(\mathbf{x}_j + \mathbf{u}_i) - T(\mathbf{x}_j)]^2$ <p>非线性最小二乘</p>	$I_x u + I_y v + I_t = 0$ <div></div> $\min \ \mathbf{A}\mathbf{u} - \mathbf{b}\ ^2$ <p>线性最小二乘</p>

本质原因：光流约束方程假设 $\mathbf{u}_i \rightarrow 0$ ，可将 $I(\mathbf{x}_i + \mathbf{u}_i)$ 在 \mathbf{x}_i 处一阶Taylor展开，舍去 \mathbf{u}_i 二阶以上项，从而将非线性简化为线性。

光流运动估计

除了对每个像素 \mathbf{x} 单独求解运动参数 \mathbf{u} ，更常用的方法是对所有像素的运动整体求解，最典型的算法是Horn和Schunck于1981年提出的基于光流平滑约束的算法。

光流的局部平滑约束可以用光流梯度来描述：

$$(\nabla u)^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 = 0$$

$$(\nabla v)^2 = \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 = 0$$

光流运动估计

将光流局部平滑约束加入光流方程，并整体计算所有像素光流运动，则光流估计可以表示为如下最小化问题：

$$\min E_{HS}$$

$$E_{HS} = \iint \left((I_x u + I_y v + I_t)^2 + \lambda^2 ((\nabla u)^2 + (\nabla v)^2) \right) dx dy$$

光流的最优解应满足：

$$\frac{\partial E_{HS}}{\partial u} = 0 \rightarrow I_x^2 u + I_x I_y v = -\lambda^2 \nabla^2 u - I_x I_t$$

$$\frac{\partial E_{HS}}{\partial v} = 0 \rightarrow I_y^2 v + I_x I_y u = -\lambda^2 \nabla^2 v - I_y I_t$$

光流运动估计

$$I_x^2 u + I_x I_y v = -\lambda^2 \nabla^2 u - I_x I_t$$

$$I_y^2 v + I_x I_y u = -\lambda^2 \nabla^2 v - I_y I_t$$

求解上述方程，可得：

$$u = \bar{u} - I_x [I_x \bar{u} + I_y \bar{v} + I_t] / (\lambda^2 + I_x^2 + I_y^2) \quad \nabla^2 u \approx u - \bar{u}$$

$$v = \bar{v} - I_y [I_x \bar{u} + I_y \bar{v} + I_t] / (\lambda^2 + I_x^2 + I_y^2) \quad \nabla^2 v \approx v - \bar{v}$$

由于 \bar{u} 和 \bar{v} 分别为 (x,y) 邻域内的光流平均值，因此 (u,v) 可采用迭代的方式求解：

$$u^{n+1} = \bar{u}^n - I_x [I_x \bar{u}^n + I_y \bar{v}^n + I_t] / (\lambda^2 + I_x^2 + I_y^2)$$

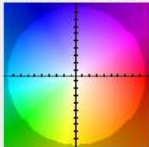
$$v^{n+1} = \bar{v}^n - I_y [I_x \bar{u}^n + I_y \bar{v}^n + I_t] / (\lambda^2 + I_x^2 + I_y^2)$$

光流运动估计

光流估计标准评测库: <http://vision.middlebury.edu/flow/>

Optical flow evaluation results										Statistics: Average SD R0.5 R1.0 R2.0 A50 A75 A95																
Show images: <input checked="" type="radio"/> below table <input type="radio"/> above table <input type="radio"/> in window										Error type: endpoint angle interpolation normalized interpolation																
Average endpoint error	avg. rank	Army (Hidden texture)			Mequon (Hidden texture)			Schefflera (Hidden texture)			Wooden (Hidden texture)			Grove (Synthetic)			Urban (Synthetic)			Yosemite (Synthetic)			Teddy (Stereo)			
		GT	im0	im1	GT	im0	im1	GT	im0	im1	GT	im0	im1	GT	im0	im1	GT	im0	im1	GT	im0	im1	GT	im0	im1	
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	
IVANN [87]	2.6	0.07	0.20	0.05	0.15	0.51	0.12	0.18	0.37	0.14	0.10	0.49	0.06	0.41	0.61	0.21	0.23	0.66	0.19	0.10	0.12	0.17	0.34	0.80	0.23	
OFLAF [77]	6.6	0.08	0.21	0.06	0.16	0.53	0.12	0.19	0.37	0.14	0.14	0.77	0.23	0.51	0.78	0.25	0.31	0.76	0.25	0.11	0.12	0.21	0.28	0.42	0.78	0.63
MDP-Flow2 [68]	7.8	0.08	0.21	0.07	0.15	0.48	0.11	0.20	0.40	0.14	0.15	0.80	0.29	0.63	0.93	0.43	0.26	0.76	0.23	0.11	0.12	0.17	0.38	0.79	0.44	
NN-field [71]	8.4	0.08	0.22	0.04	0.17	0.55	0.13	0.19	0.39	0.15	0.09	0.48	0.05	0.41	0.61	0.20	0.52	0.64	0.26	0.13	0.29	0.13	0.20	0.35	0.83	0.21
ComponentFusion [96]	9.8	0.07	0.21	0.05	0.16	0.55	0.12	0.20	0.44	0.15	0.11	0.65	0.06	0.71	0.27	0.32	0.53	0.28	0.32	0.61	0.06	0.18	0.29	0.11	0.13	0.25
WLIF-Flow [93]	14.4	0.08	0.21	0.06	0.18	0.55	0.15	0.25	0.56	0.14	0.17	0.68	0.08	0.61	0.92	0.41	0.43	0.22	0.96	0.29	0.17	0.13	0.28	0.12	0.21	
TC/T-Flow [76]	14.7	0.07	0.21	0.05	0.19	0.48	0.12	0.28	0.66	0.23	0.14	0.14	0.86	0.38	0.07	0.67	0.24	0.98	0.23	0.49	0.22	0.82	0.19	0.11	0.11	
Layers++ [37]	16.1	0.08	0.21	0.07	0.19	0.56	0.17	0.20	0.40	0.18	0.13	0.58	0.07	0.48	0.70	0.33	0.47	0.33	1.01	0.15	0.14	0.24	0.46	0.88	0.72	
LME [70]	16.6	0.08	0.22	0.06	0.15	0.49	0.11	0.30	0.26	0.64	0.18	0.31	0.67	0.66	0.20	0.96	0.53	0.28	0.37	1.18	0.31	0.28	0.12	0.18		
nLayers [57]	17.2	0.07	0.19	0.06	0.22	0.59	0.12	0.25	0.54	0.11	0.20	0.15	0.84	0.33	0.08	0.53	0.78	0.34	0.44	0.26	0.84	0.30	0.13	0.28		
IROF++ [58]	17.2	0.08	0.23	0.07	0.21	0.68	0.17	0.28	0.63	0.17	0.20	0.15	0.73	0.19	0.23	0.60	0.10	0.89	0.42	0.43	0.22	1.08	0.28			
FC-2Layers-FF [74]	19.3	0.08	0.21	0.07	0.21	0.70	0.30	0.20	0.40	0.18	0.15	0.18	0.76	0.22	0.08	0.53	0.53	0.77	0.4	0.49	0.12	0.14	0.33			
Correlation Flow [75]	19.5	0.09	0.23	0.07	0.17	0.58	0.11	0.43	0.44	0.99	0.15	0.11	0.47	0.75	0.33	1.08	0.30	0.41	0.92	0.30	0.14	0.13	0.26			
AGIF+OF [85]	20.6	0.08	0.22	0.07	0.23	0.73	0.18	0.28	0.66	0.23	0.18	0.14	0.70	0.17	0.08	0.57	0.78	0.85	0.38	0.10	0.47	0.33				

Color encoding
of flow vectors



Schefflera - Complementary-OF flow



flow error



总结一本课程内容回顾

运动估计的基本假设：

- 亮度一致：图像 I 和 T 中的对应像素点亮度一致；
- 微小运动：对应像素点间的运动很小。

平移运动估计：遍历法、LK法

参数化运动估计：正向LK法、逆向LK法

光流运动估计：局部法、全局法

总结—参考文献

- Lucas, B. D. and Kanade, T. **An iterative image registration technique with an application in stereo vision**. In Seventh International Joint Conference on Artificial Intelligence, 1981.
- Horn, B. K. P. and Schunck, B. G. **Determining optical flow**. *Artificial Intelligence*, 17:185–203, 1981.
- Baker, S. and Matthews, I. **Lucas-Kanade 20 years on: A unifying framework**. *International Journal of Computer Vision*, 56(3):221–255, 2004.