Final: KLT 角点跟踪算法的实现和加速

一、 算法背景

KLT 角点跟踪算法全称 Kanade-Lucas-Tomasi Tracking，又称 LK 跟踪算法。是经典的角点跟踪算法。算法假设目标在视频流中，只产生一致性的小位移，并且目标的灰度变化不大。那么算法必须是在以下 3 个假设成立的前提下发挥良好的效果：

（1）亮度恒定。

（2）时间连续或者运动位移小。

（3）空间一致性，邻近点有相似运动，保持相邻。

满足假设 1 保证目标不受亮度的影响；满足假设 2 保证在目标领域内能够对应的特征点；满足假设 3 保证在同一窗口内，所有点的位移相同。

在局部窗口 w 上，所有(x, y)都往一个方向移动了(dx, dy)，从而得到(x', y')，即 t 时刻的(x, y)点在 t+ τ 时刻为(x+dx, y+dy)，所以寻求匹配的问题可化为对以下的式子寻求最小值，或叫做最小化以下式子：

$$\epsilon(\mathbf{d}) = \epsilon(d_x, d_y) = \sum_{x=u_x-\omega_x}^{u_x+\omega_x} \sum_{y=u_y-\omega_y}^{u_y+\omega_y} \left(I(x,y) - J(x+d_x, y+d_y)\right)^2 .$$

用积分来表示上述式子，以上式子可等效为：

$$\epsilon = \int\int_W [J(\mathbf{x}+\frac{\mathbf{d}}{2}) - I(\mathbf{x}-\frac{\mathbf{d}}{2})]^2 w(\mathbf{x})\, d\mathbf{x},$$

这个式子的含义，即找到两副图像中，在 W 窗口中，I、J 的差异，其中 I 以 x-d/2 为中心，J 以 x+d/2 为中心，w/2 为半径的一个矩形窗口间的差异，函数 ε (d)要取得最小值，这个极值点的导数一定为 0，即

$$\frac{\partial \epsilon}{\partial \mathbf{d}} = 2\int\int_W [J(\mathbf{x}+\frac{\mathbf{d}}{2}) - I(\mathbf{x}-\frac{\mathbf{d}}{2})][\frac{\partial J(\mathbf{x}+\frac{\mathbf{d}}{2})}{\partial \mathbf{d}} - \frac{\partial I(\mathbf{x}-\frac{\mathbf{d}}{2})}{\partial \mathbf{d}}] w(\mathbf{x})\, d\mathbf{x}$$

的值为 0，由泰勒展开的性质：

$$I(\mathbf{x}-\frac{\mathbf{d}}{2}) \approx I(\mathbf{x}) - \frac{d_x}{2}\frac{\partial I}{\partial x}(\mathbf{x}) - \frac{d_y}{2}\frac{\partial I}{\partial y}(\mathbf{x})$$

$$J(\mathbf{x}+\frac{\mathbf{d}}{2}) \approx J(\mathbf{x}) + \frac{d_x}{2}\frac{\partial J}{\partial x}(\mathbf{x}) + \frac{d_y}{2}\frac{\partial J}{\partial y}(\mathbf{x}).$$

于是，问题转化为：

$$\frac{\partial \epsilon}{\partial \mathbf{d}} = 2\int\int_W [J(\mathbf{x}+\frac{\mathbf{d}}{2}) - I(\mathbf{x}-\frac{\mathbf{d}}{2})][\frac{\partial J(\mathbf{x}+\frac{\mathbf{d}}{2})}{\partial \mathbf{d}} - \frac{\partial I(\mathbf{x}-\frac{\mathbf{d}}{2})}{\partial \mathbf{d}}]w(\mathbf{x})\,d\mathbf{x}$$
$$\approx \int\int_W [J(\mathbf{x}) - I(\mathbf{x}) + \mathbf{g}^T\mathbf{d}]\mathbf{g}(\mathbf{x})w(\mathbf{x})\,d\mathbf{x},$$

其中：

$$\mathbf{g} = \left[\frac{\partial}{\partial x}\left(\frac{I+J}{2}\right) \quad \frac{\partial}{\partial y}\left(\frac{I+J}{2}\right)\right]^T$$

从而，问题即为：

$$\frac{\partial \epsilon}{\partial \mathbf{d}} = \int\int_W [J(\mathbf{x}) - I(\mathbf{x}) + \mathbf{g}^T(\mathbf{x})\mathbf{d}]\mathbf{g}(\mathbf{x})w(\mathbf{x})\,d\mathbf{x} = 0$$

$$\int\int_W [J(\mathbf{x}) - I(\mathbf{x})]\mathbf{g}(\mathbf{x})w(\mathbf{x})\,d\mathbf{x} = -\int\int_W \mathbf{g}^T(\mathbf{x})\mathbf{d}\mathbf{g}(\mathbf{x})w(\mathbf{x})\,d\mathbf{x},$$
$$= -\left[\int\int_W \mathbf{g}(\mathbf{x})\mathbf{g}^T(\mathbf{x})w(\mathbf{x})\,d\mathbf{x}\right]\mathbf{d}$$

即其等式可看作为：

$$Z\mathbf{d} = \mathbf{e}$$

其中，Z 为一个 2*2 的矩阵，e 为一个 2*1 的向量，

$$Z = \int\int_W \mathbf{g}(\mathbf{x})\mathbf{g}^T(\mathbf{x})w(\mathbf{x})\,d\mathbf{x}$$

$$\mathbf{e} = \int\int_W [I(\mathbf{x}) - J(\mathbf{x})]\mathbf{g}(\mathbf{x})w(\mathbf{x})\,d\mathbf{x}$$

求解方程，不一定能得到精确解，可以利用牛顿迭代法求解，当残差小于一定阈值时，就认为得到了近似解。

KLT 对于图像中的目标，并不是目标框和跟踪框内的所有的点都求取偏移量，而是选择一些特征不变的角点（corners），可以不同的特征不变的角点作为跟踪点，比如 SIFT、SURF、FAST、SUSAN、HARRIS 等。

在这里 shi-tomasi 提出了一种 Good Features 的角点。他们认为，式中的对称矩阵 Z 包含了噪声和良好条件，当矩阵 Z 的两个特征值较小时，意味着以该点为中心 W 为窗口内是平坦的；一大一小的两个特征值，对应无方向的纹理特征；两个较大的特征值代表了当前点是角点或者椒盐纹理。所以，当两个特征值大于一定阈值时，选择这个点作为角点。

$$min(\lambda_1, \lambda_2) > \lambda$$

其中λ是设置的阈值。

综上所述，当我们采用角点进行跟踪时，如果通过方程解得的残差 e 足够小（设定的阈值），认为是跟踪到的一个角点，并求出了角点的偏移$\mathbf{d}$。

二、 算法设计

1.CPU 朴素

依照公式计算，其中图像长*宽*窗口大小*窗口大小的四重循环为主要优化目标。

2.CPU 优化

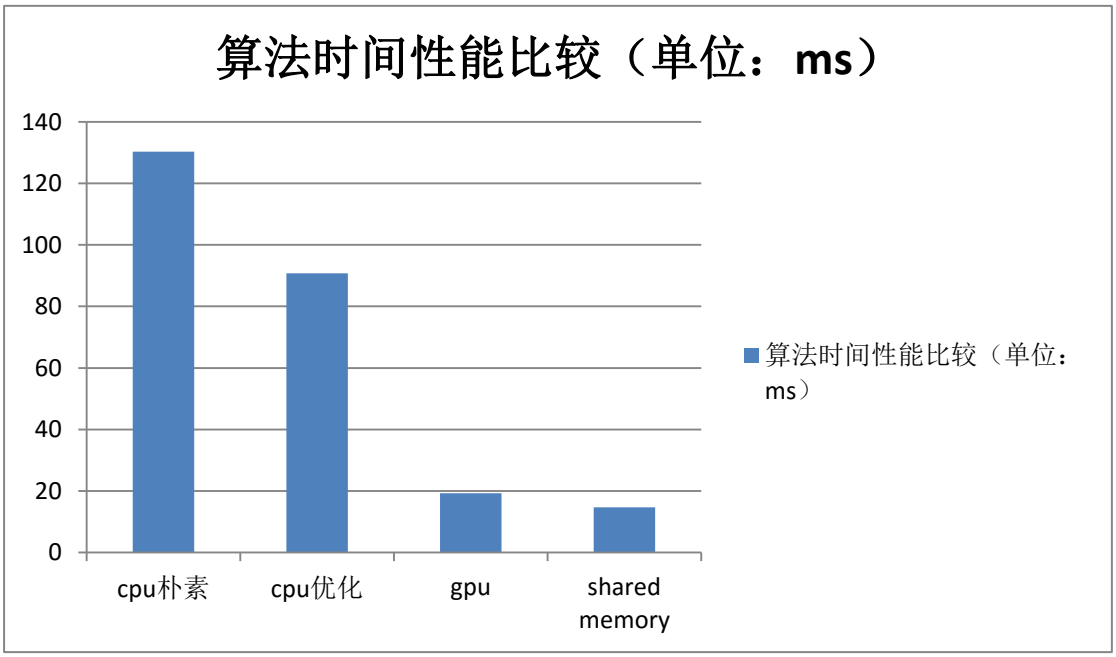
利用相邻子矩阵里大部分元素相同的特性，每个相邻矩阵相差为两行/列，在计算元素之和时可以减去一行/列并加上一行/列的形式计算，起到加速效果。

3. GPU

将图像按像素分块，每个像素点作为一个线程计算。

4. GPU shared memory

将同一个线程块内的公共元素提前从 global memory 加载到 shared memory，起到加速效果。

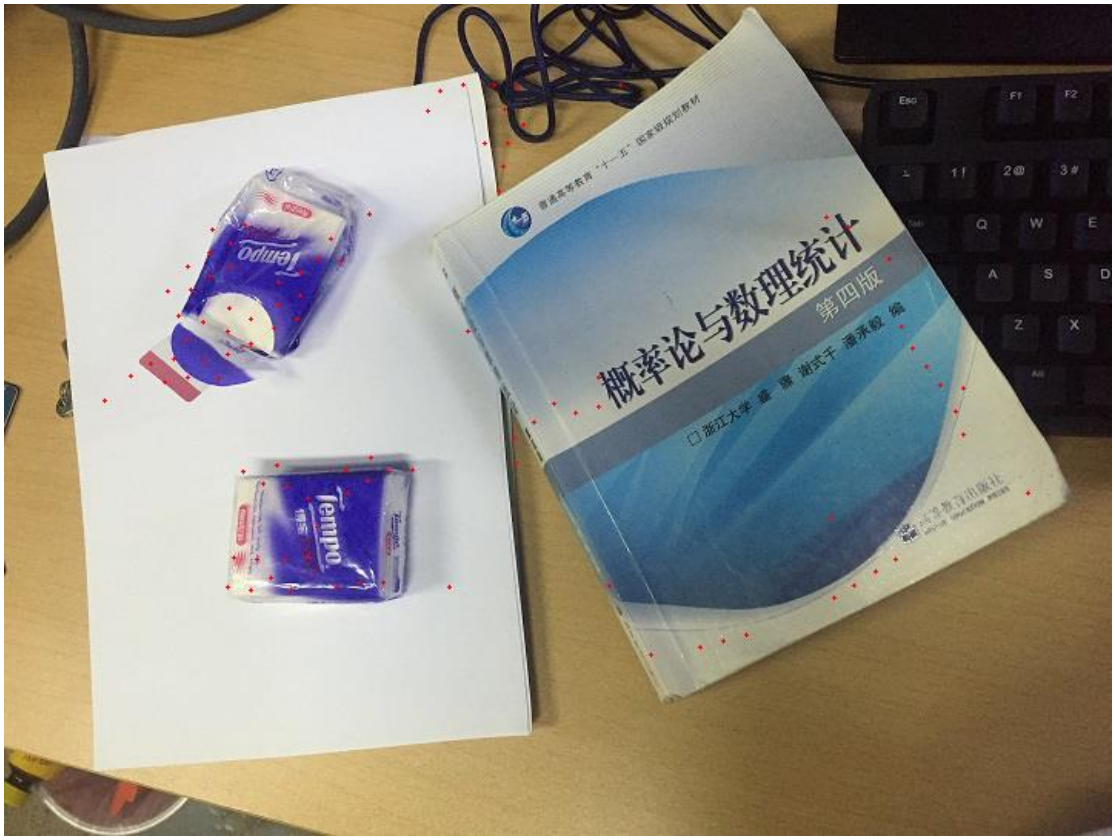


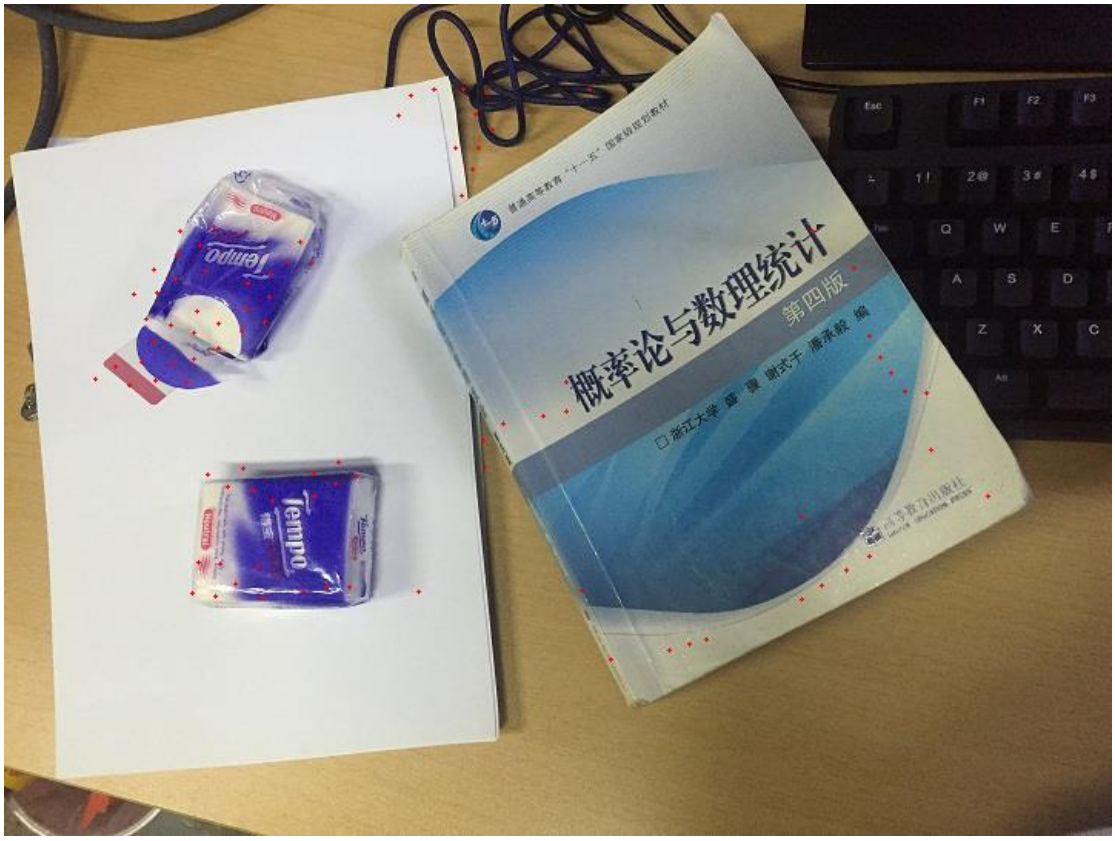
角点跟踪率：91%
加速比：约是 cpu 朴素算法的 10 倍，约是 cpu 优化算法的 7 倍。

三、 实际意义

作为计算机视觉中的一个基本问题，目标跟踪需要尽可能地达到实时效果，人的舒适感官需要大约每秒 60 帧左右，即需要在 20ms 内完成一帧图像的检测和跟踪。而 CPU 实现难以达到该效果，经测试基本在 100ms 左右，因此需要用 GPU 加速以更好的应对相关实际问题。经 GPU 加速后，基本已经达到需要的效果。

四、 实验结果





五、 参考文献

[1]. Lucas B D, Kanade T. An iterative image registration technique with an application to stereo vision[J]. 1981.

[2]. Marroquim R. Kanade-Lucas-Tomasi Tracker[J].

[3]. Shi J. Good features to track[C]//Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on. IEEE, 1994: 593-600.

[4]. Birchfield S. Derivation of kanade-lucas-tomasi tracking equation[J]. unpublished notes, 1997.

[5].Tomasi C, Kanade T. Detection and tracking of point features[J]. 1991.

六、 附录

CPU: i5-4590

GPU: NVIDIA GeForce GTX 750

Device_Query

```
PS C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v8.0\extras\demo_suite> .\deviceQuery.exe
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v8.0\extras\demo_suite\deviceQuery.exe Starting...

 CUDA Device Query (Runtime API) version (CUDART static linking)

Detected 1 CUDA Capable device(s)

Device 0: "GeForce GTX 750"
  CUDA Driver Version / Runtime Version          8.0 / 8.0
  CUDA Capability Major/Minor version number:    5.0
  Total amount of global memory:                 2048 MBytes (2147483648 bytes)
  ( 4) Multiprocessors, (128) CUDA Cores/MP:     512 CUDA Cores
  GPU Max Clock rate:                            1085 MHz (1.08 GHz)
  Memory Clock rate:                             2505 Mhz
  Memory Bus Width:                              128-bit
  L2 Cache Size:                                 2097152 bytes
  Maximum Texture Dimension Size (x,y,z)         1D=(65536), 2D=(65536, 65536), 3D=(4096, 4096, 4096)
  Maximum Layered 1D Texture Size, (num) layers  1D=(16384), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers  2D=(16384, 16384), 2048 layers
  Total amount of constant memory:               65536 bytes
  Total amount of shared memory per block:       49152 bytes
  Total number of registers available per block: 65536
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  2048
  Maximum number of threads per block:           1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size    (x,y,z): (2147483647, 65535, 65535)
  Maximum memory pitch:                          2147483647 bytes
  Texture alignment:                             512 bytes
  Concurrent copy and kernel execution:          Yes with 1 copy engine(s)
  Run time limit on kernels:                     Yes
  Integrated GPU sharing Host Memory:            No
  Support host page-locked memory mapping:       Yes
  Alignment requirement for Surfaces:            Yes
  Device has ECC support:                        Disabled
  CUDA Device Driver Mode (TCC or WDDM):         WDDM (Windows Display Driver Model)
  Device supports Unified Addressing (UVA):      Yes
  Device PCI Domain ID / Bus ID / location ID:   0 / 1 / 0
  Compute Mode:
     < Default (multiple host threads can use ::cudaSetDevice() with device simultaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 8.0, CUDA Runtime Version = 8.0, NumDevs = 1, Device0 = GeForce
 GTX 750
Result = PASS
```