GPU acceleration of initial search method for regional image matching

Student's ID: 15B15829 Name: Moran Lee Supervisor: Yamasita Yukihiko

1 Introduction

Image matching is applied to object detection and pattern recognition, which is one of the most important methods in image processing. Therefore, many image matching methods have been proposed. They can be roughly divided into two types, one is a feature point matching method as extracting feature points from the image and evaluating the consistency of the feature points, and the other is a region matching method as evaluating the consistency of a image region.

Furthermore, depending on the spatial relationship between the template image and the input image, matching can be divided into whole-to-part and whole-to-whole matching. Yamashita and Wakahara proposed a fast initial search method for whole-to-part matching. GPU can be used to further accelerate the method for real time computation. In this paper I propose the implementation method and compare it with CPU calculation.

2 Fast calculation of Gaussian convolution integral

Gaussian smoothing is a process of blurring an image using a Gaussian function. It can be applied to noise removal, preprocessing of edge extraction, etc.

Gaussian smoothing is a convolutional sum given by the following expression $O(n) \simeq \sum_{k=-K}^{K} G(k)I(n-k)$.

Since Gaussian function is an even function, the differentiated Gaussian function is an odd function. Therefore, Gaussian function can be approximated by the cosine function ($G(k) \simeq \sum_{p=0}^{P} a_p \cos(\beta pk)$) and differential Gaussian function can be approximated by the sine function($G_d(k) \simeq \sum_{p=1}^{P} b_p \sin(\beta pk)$).

We define the convolutional sum of finite interval trigonometric functions as the following equations, $c_p[n] = \sum_{k=-K}^{K} I[n-k]\cos(\beta pk)$, $s_p[n] = \sum_{k=-K}^{K} I[n-k]\sin(\beta pk)$. Then, we can get a $O(PN)$ algorithm to calculate Gaussian convolutional sum as follows, $[v[n] = e^{i\beta p}v[n-1] + I[n]$, $c_p[n] + is_p[n] = (-1)^p(v[n+K] - v[n-K] + I[n-K])$.

3 Initial search method for regional image matching

The edge direction histogram $D(\theta)$ is weighted sum of the strength of edges in each edge direction in a certain region. It can be described by a Q-th order Fourier series as $D(\theta) = \sum_{q=-Q}^{Q} d_n e^{iq\theta}$. The coefficients can be calculated as $d_q \simeq \frac{I_X(x,y) - iI_Y(x,y)^q}{2\pi(\|I_1(x,y)\|^2+\epsilon)^{\frac{q-1}{2}}}$. In this case, when rotating the reference direction by θ_0, the coefficients d'_q of the rotated histogram can be obtained as $d\prime_q = d_q e^{-iq\theta_0}$.

Using fast calculation of Gaussian convolutional sum and Q-th order Fourier series of edge direction histogram, the feature value calculation of the initial search method for regional image matching can be described as follow.

Algorithm 1 Calculate Feature Values of Region

Require: Input Image I, a set of scales S_a, a set of center points for feature values T_S, The regularized coordinates of points where histograms are used (x_m, y_m) $(m = 1, 2, ..., M)$

for $S \in S_a$ **do**
  Calculate Gaussian differentiation of I with scale $S/8$.
  Calculate the edge direction histogram d_q.
  Calculate Gaussian smoothing of d_q in scales S and $S/4$.
  for $(x, y) \in T_S$ **do**
    Calculate the orientation(s) of the region with the smoothed coefficient in scale S.
    for $\theta_0 \in$(the set of orientions) **do**
      Calculate cordinates of points where histograms are used.

$$x'_m = S(x_m \cos\theta_0 - y_m \sin\theta_0) + x$$

$$y'_m = S(x_m \sin\theta_0 + y_m \cos\theta_0) + y$$

      Rotated edge direction histogram in scale $S/4$ and get $M(2Q+1)$-dimensional vector.
      Normalise the obtained vector as its norm is 1 and store it as feature values at the region.
    end for
  end for
end for

The feature values of the target region in the template image and of regions around sample points are calculated. Then, the inner product between feature values in the template image and input image are calculated. The region in input image that maximizes the inner products is given as the matched region.

4 Calculation by GPU

GPU calculation is a kind of SIMD (single instruction multi data) calculation, which means the same task is applied to different cores to process multiple

data. GPUs of NVIDIA use warp schedulers to control jobs in the core. Therefore, we need to decrease the branches (if statement) of calculation to increase efficiency of calculation.

GPUs also use cache memory to accelerate calculation. Therefore, the efficiency of calculation can be increased by increasing the cache hit rate. And the best access pattern is given by Figure 1, where the cache hit rate is 100%.

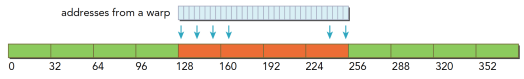

Figure 1: Best memory access pattern

The worst memory access pattern is given by Figure 2, where the cache hit rate is 12.5%.

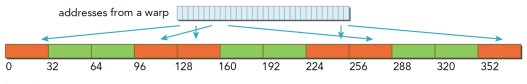

Figure 2: Worst memory access pattern

5 Optimization of initial search method by GPU

In the initial search method for region-based image matching, at first, we calculate Gaussian smoothing of d_q in scale S and $S/4$ at all points. Then we calculate the feature values for sampling region, and sort to find the point which matches the region in the template image best.

Because of the property of the warp scheduler, GPU is not suitable for sorting. Therefore we use CPU to do this job. Because of the same reason, we use CPU to find the point which matches the region best.

Almost all jobs in calculation of all Gaussian smoothing of d_q in scale S and $S/4$ can be calculated in parallel, because the jobs for a point don't have dependency. But the calculation of Gaussian convolution integral at a point is dependent of the result at the former point. Therefore, we divided tasks by rows and tasks for a row is assigned to a core.

But if we assign jobs for a row into a core, the memory access pattern becomes the worst. Therefore I optimize the calculation to increase the cache hit rate. For the purpose, I store results for a row into a column memory structure. Then transpose the data for the next step.

6 Experiment

I use a standard image Lenna (512px×512px) to compare the effect between CPU calculation and optimized GPU calculation of the initial search method for regional image matching introduced before.

At first, I calculate the compute time of CPU calculation (CPU). Secondly, I calculate the normal compute time of GPU calculation (GPU 1). Thirdly, we calculate the compute time of GPU calculation in which memory access is optimized (GPU 2). I calculate compute time of optimized GPU calculation with three scales which can show its efficiency in real world application (GPU 3).

Because the hardware spec affects the calculation efficiency, we use two types of computers to collect experiment data.

Table 1: Comparison of Compute Time(ms)

	940MX(6500U)	1060(7700K)
CPU	742	509
GPU 1	160	66
GPU 2	156	31
GPU 3	136	22

As we can see, the computation time of CPU is at least 5 times more than computation time of GPU. Furthermore optimized calculation has a better efficiency than normal one. Also, it can be calculated within 100 ms for the calculation with three scales, and it will enable 10fps in the initial search. Therefore, the propesd method it can be applied to real-time applications.

7 Conclusion

This paper proposed to accelerate initial search method for regional image matching and realize real-time computing by GPU. We confirm the acceleration and its practicality for real-time processing by experiments. Future works are to increase the hit rate of cache memory in read access, which may further accelerates the method.

References

[1] John Cheng,Grossman,McKercher,(2015)『CUDA C professional programming』,(translated by Sinya Morino) Impress Company, 2015.

[2] Yukihiko Yamasita and Toru Wakahara,"An initial search method for region-based image matching",Technical Report of IEICE, vol.118, no. 219, PP.119-124, 2018.