Bilinear Upsampling On CUDA

Bilgin Aksoy

Project Phase-I

Bilgin Aksoy
MMI
2252286
13 November 2017

1 Abstract

Object segmentation can be thought as a deep learning classification problem but without a final layer which classifies the inputs. Using lower layers activations deep learning can easily segment the object from a given input. This idea is generally called Fully Convolutional Network (FCN) and first proposed by Long et al. [1] But the activations of lower layers are smaller than the input due to the max-pooling operation. I will implement a kernel which takes three inputs (activations), applies bilinear interpolation upsampling, and averaging the output.

2 Introduction-Literature Survey

2.1 Bilinear Interpolation

Bilinear Interpolation: Bilinear interpolation is used to know values at random position from the weighted average of the four closest pixels to the specified input coordinates, and assigns that value to the output coordinates.[2] Equation-(1)

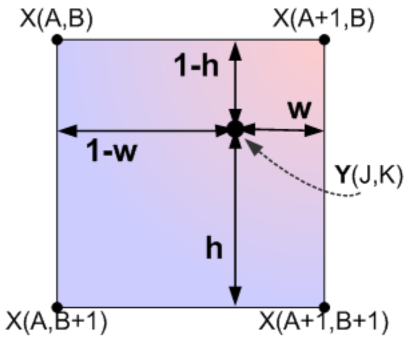

Figure 1: Image from `https://www.giassa.net/?page_id=240`[3]

$$Y(J,K) = (1-W)(1-H)X(A,B)+(W)(1-H)X(A+1,B)+(1-W)(H)X(A,B+1)+(W)(H)X(A+1,B+1) \quad (1)$$

2.2 Literature

Qingshuang et al.[4] proposed ”GPGPU-based parallel algorithm can take full advantage of the GPU’s parallel computing capabilities, and can achieve about 40 times speedup. (Table-1)”

Table 1: The Time-Consuming Of Serial And Parallel Bilinear Spatial Interpolation Of Different Grid Size [4]

Grid Size	CPU (sec)	GPU (sec)
40x40	1.70	0.11
100x100	10.34	0.37
200x200	41.43	1.21
300x300	93.13	2.36
500x500	240.02	5.70
800x800	630.92	14.40
1200x1200	1432.33	32.41

Akgn and Gevrekci have been proposed ”a massively multi-threaded implementation of super-resolution image formation on the NVIDIA CUDA architecture.” [5] ”Super-resolution reconstruction begins with an initial estimate image which is typically chosen as a bilinearly upscaled version of the original low resolution frame." Their kernel's profile can be seen in Table-2.

Table 2: Bilinear filtering code analysis [5]

Thread numbers (x,y)	(32,32)
Shared memory bank conflict	N/A
Global memory BW efficiency	100
Register usage	17
Occupancy	0,877
Total execution time per frame	161 microsec

2.3 Literature-Real Life Implementations

There are multiple implementations using software /hardware acceleration on CPU/GPU. One of most used framework OpenCV has both CPU and GPU implementation of upsampling using bilinear interpolation.

Listing 1: OpenCV Prototype of Bilinear Upsampling Function on CPU

```
void cv::resize (InputArray src, OutputArray dst, Size dsize, double fx=0,
    double fy=0, int interpolation=INTER_LINEAR           )
```

Listing 2: OpenCV Prototype of Bilinear Upsampling Function on GPU

```
void cv::cuda::resize   (InputArray src, OutputArray dst, Size dsize, double fx
     = 0, double fy = 0, int interpolation = INTER_LINEAR, Stream &stream=Stream
    ::Null())
```

Intel offers developers high-quality, production-ready, low-level building blocks for image processing, signal processing, and data processing applications by using a library called Intel Integrated Performance Primitives (Intel IPP). This library optimized for wide range of Intel architecture (Intel Quark, Intel Atom, Intel Core, Intel Xeon, and Intel Xeon Phi processors). Their bencmarks for image resizing is in Figure-2.

Intel IPP library unfortunately is not free. It's price starts from 249$ for academic licence. OpenCV 3.x has native support for IPP-ICV library which is a special subset of Intel IPP functions for image processing and computer vision.

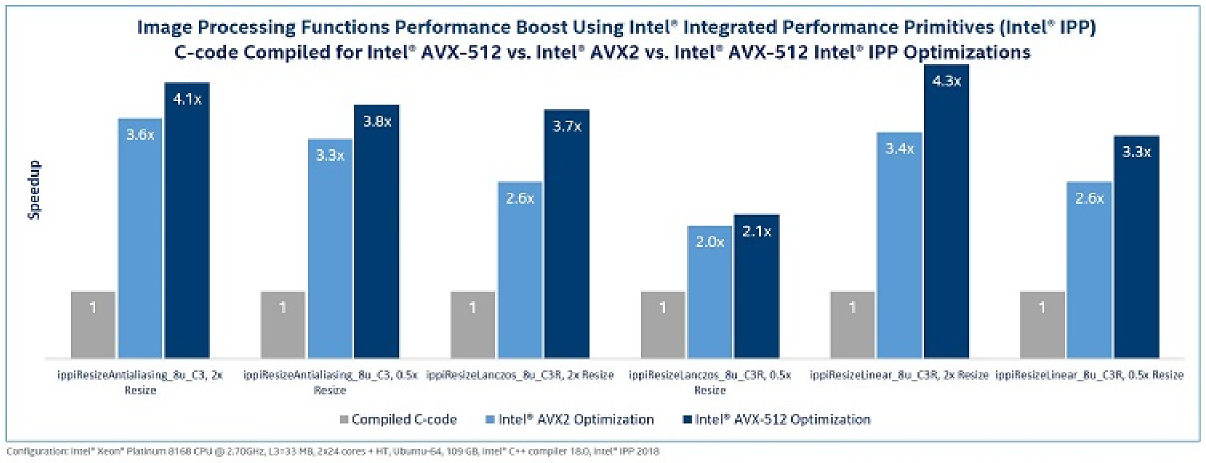

Figure 2: Image from `https://software.intel.com`

Listing 3: Intel IPP Prototype of Bilinear Upsampling Function

```
IppStatus ippiResizeLinear_<mod>(const Ipp<datatype>* pSrc, Ipp32s srcStep, Ipp
    <datatype>* pDst, Ipp32s dstStep, IppiPoint dstOffset, IppiSize dstSize,
    IppiBorderType border, const Ipp<datatype>* pBorderValue, const
    IppiResizeSpec_32f* pSpec, Ipp8u* pBuffer)
```

ImageMagick is an another commonly used software, which can be instantiated over command line, has support for many media formats, and can be used for many operation over media files including image resizing.

Listing 4: Sample code for image resizing.

```
$ convert input.jpg -scale 200% -interpolate bilinear output.jpg
```

References

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[2] S. Fadnavis, "Image interpolation techniques in digital image processing: An overview," *International Journal of Engineering Research and Applications*, vol. 4, no. 10, pp. 70–73, 2014.

[3] "Bilinear interpolation." `https://www.giassa.net/?page_id=240`.

[4] W. QINGSHUANG, W. QIANG, and C. XIANGFU, "Parallel bilinear spatial interpolation algorithm based on gpgpu.," *Journal of Theoretical & Applied Information Technology*, vol. 48, no. 3, 2013.

[5] T. Akgün and M. Gevrekci, "Accelerating super-resolution reconstruction using gpu by cuda," in *Advances in Computational Science, Engineering and Information Technology*, pp. 47–58, Springer, 2013.