Ultrafast focus detection using multi-scale histologic features

Maksim Levental
University of Chicago

Ryan Chard
Argonne National Laboratory

Gregg A. Wildenberg
University of Chicago

ABSTRACT

We present a fast out-of-focus detection algorithm for electron microscopy images collected serially. Such images are collected for the purposes of post-processing tasks such as montaging, alignment, and image segmentation. Such an algorithm is necessitated by recent increases in collection rates owing to advances in microscopy technology. Our technique, *Multi-scale Histologic Feature Detection*, adapts classical computer vision techniques and is based on detecting various fine-grained histologic features. We further exploit the inherent parallelism in the technique by employing GPGPU primitives in order to accelerate characterization. Tests are performed that demonstrate near-real-time detection of out-of-focus conditions. <We also deploy to funcX something something>. We discuss extensions that enable scaling out to support multi-beam microscopes and integration with existing focus systems for purposes of implementing auto-focus.

1 INTRODUCTION

A fundamental goal of neuroscience is to map the anatomical relationships of the brain. Currently this is challenging because electron microscopy, an imaging method traditionally limited to small images, is the only imaging modality with sufficient resolution to directly visualize the connections, or synapses, between neurons. Recently, automated serial electron microscopy, broadly called *connectomics*, has been developed. This technique is characterized by thousands, if not tens of thousands, of individual images being automatically acquired in series and then registered to produce a volumetric dataset. Such datasets allow neuroscientists to follow the tortuous path neurons take through the brain to connect with each other (hence the name connectomics). However, many of the steps that comprise the collection of such datasets for connectomics require manual inspection causing significant slowdowns in the rate at which datasets can be acquired. Such bottlenecks significantly impact the size of the datasets that can be reasonably acquired and studied. Furthermore, advances in electron microscopes have increased the rate that datasets can be acquired (e.g. ~10 Tbs/24hr Carl Zeiss AG [3]) . This further underscores the need for automation (in order that end-to-end high throughput is achieved).

Auto-focus technology is a critical component of many imaging systems; from consumer cameras (for purposes of convenience) to industrial inspection tools to scientific instrumentation [16]. Such technology is typically either *active* or *passive*; active methods exploit some auxiliary device or mechanism to measure the distance of the optics from the scene, while passive methods analyze the definition or sharpness of an image by virtue of a proxy measure called a *criterion function.* Many electron microscopes incorporate auto-focus techniques that attempt to focus the microscope before image acquisition. Despite such functionality, out-of-focus (OOF) images still occur at high rates (between 1% and 10%), depending on the quality of the tissue sections being imaged. Such error rates prevent effective automation since a prerequisite of the downstream operations are that the images collected all have high degree-of-focus (DOF). Without properly focused images, all downstream computational steps (e.g. 2D tile montaging, 3D alignment, automatic segmentation) will fail.

Thus, we seek to further the aims of automation by ensuring that images acquired by the electron microscope have high DOF. While seemingly a small step in the process, focus detection is nevertheless an extremely critical step. Consequently, because imaging sections requires loading and unloading sets of ~100-200 sections at a time, failure to manually detect an out of focus image in real time causes significant delays. The affected sample sets need to be reloaded, desired field of view must be reconfigured, and reaquired images need to be reinserted into the image stack. All such remediation steps are time and labor intensive.

Our proposed technique, *Multi-scale Histologic Feature Detection* (MHFD), involves a second pass over the collected image, after it has been acquired, using a computer vision system to detect a failure to successfully achieve high DOF. We use feature detection [8] as a criterion function, reasoning that the quantity of features detected is positively correlated with DOF. To this end, we develop a feature detector based on scale-space representations of images (see section (2)) but optimized for latency (rather than for accuracy). Our solution achieves low latency detection of the OOF condition with high correlation (see section (4)).

Note that we explicitly aim to augment existing microscopy equipment without the need for costly and complex retrofitting. This precludes mere improvements to auto-focus systems as they are, in essence, proprietary black boxes from the perspective of the end user of a commerical electron microscope.

This rest of this article is organized as follows: section (2) quickly reviews backround on scale-space feature detectors, section (3) describes our focus detection method in the abstract and particular optimizations made in order to achieve near-real-time performance, section (4) reports results of evaluating our method on sequences of images collected at varying focus depths, section (5) discuss those results, and section (6) discusses related work and how our work is distinct therefrom.

2 SCALE-SPACE REPRESENTATIONS

We base our multi-scale histologic feature detection technique on classical scale-space representations of signals and images. We give a brief overview (see [8] for a more comprehensive review).

The fundamental principle of scale-space feature detection is that natural images possess structureful features at multiple scales and that features at a particular scale isolated from features at other scales. Thus any image $I(x, y)$ can transformed into a scale-space representation $L(x, y, t)$, where $L(x', y', t')$ represents the pixel intensity[1] at pixel coordinates (x', y') and *scale* t'. How to produce the representation of the image at each scale is discussed in the

[1] Hence, scale-space since we consider the image along dimensions of scale and space.

forth coming. More importantly, such a representation lends itself readily to scale sensitive feature detection owing to the fact that features at a particular scale are decoupled from features at other scales, thereby eliminating confounding detections. Examples of structureful features that can be detected and characterized using scale-space representations include edges, corners, ridges, and so called blobs (roughly circular regions of uniform intensity).

A scale-space representation at a particular scale is constructed by convolution of the image with a filter that satisfies the constraints of non-enhancement of local extrema, scale invariance and rotational invariance (along with some others [5]). One such filter [6] is the symmetric, mean zero, two dimensional, Gaussian filter

$$G(x, y, \sigma) := \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Thus, define the scale-space representation $L(x, y, t)$ of an image $I(x, y)$ to be the convolution of that image with a mean zero Gaussian filter:

$$L(x, y, t) := G(x, y, t) * I(x, y)$$

where t determines the scale. $L(x, y, t)$ has the interpretation that image structures of scale smaller than $\sqrt{t^2} = t$ have been removed due to blurring. This is due to the fact that the variance of the Gaussian filter is t^2 and features of this scale are therefore “beneath the noise floor” of the filter or, in effect, suppressed by filtering procedure. A corollary is that features with approximate length scale t will have maximal response upon being filtered by $G(x, y, t)$. That is to say, for a t scale feature at pixel coordinates (x, y) and for scales $t' < t < t''$ we have

$$L\left(x, y, t'\right) < L(x, y, t) < L\left(x, y, t''\right)$$

This is due to the fact that for scales $t' < t$, small scale features will dominate the response and for $t < t''$, as already mentioned, the feature will have been suppressed.

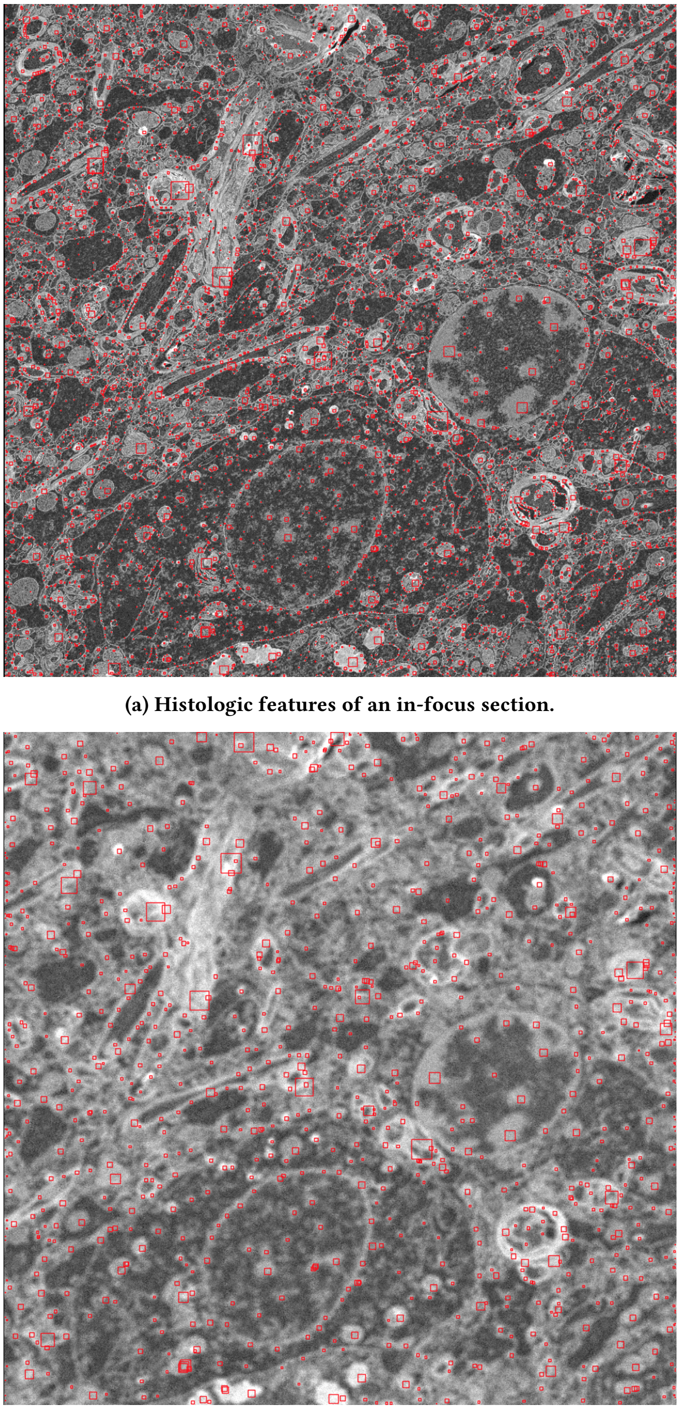
Note that the aforementioned presumes having identified the pixel coordinates (x, y) as the locus of the feature. Thus, in order to detect features across scales and space, maximal responses in spatial dimensions (x, y) need to also be characterized. For such characterization one generally employs standard calculus in order to identify critical points of second order derivatives. Hence, we can construct scale-sensitive feature detectors by considering critical points of linear and non-linear combinations of spatial derivatives ∂_x, ∂_y and derivatives in scale ∂_t. For example the scale derivative of the Laplacian

$$\partial_t \nabla^2 L := \partial_t \left(\partial_x^2 + \partial_y^2\right) L \tag{1}$$

effectively detects regions of uniform pixel intensity (i.e. blobs).

3 MULTI-SCALE HISTOLOGIC FEATURE DETECTION

We propose to use histologic feature detection at multiple scales as a criterion function, reasoning that the absolute quantity of features detected at multiple scales is positively correlated with DOF (see figure (1)). For our particular use-case this is tantamount to detecting histologic structures ranging from cell walls to whole organelles. The key insight is that the ability to resolve structure



(a) Histologic features of an in-focus section.

(b) Histologic features of an out-of-focus section.

Figure 1: Comparison of sections with histologic feature recognition as a function of focal depth.

across the range of feature scales is highly correlated with a high-definition image. To this end, we develop a feature detector based on eqn. (1) but optimized for latency (rather than for accuracy).

Firstly, in order to verify our hypothesis that detecting features across a range of scales is correlated with DOF, we compare the number of histologic features detected as a function of absolute deviation from in-focus ($|f - f'|$ where f' is the correct focal depth) for a series of sections with known focal depth (see figure (2a)). We observe a very strong log-linear relationship (see figure (2b)). Fitting a log-linear relationship produces a line with $r = -0.9754$, confirming our hypothesis that quantity of histologic features detected is a good proxy measure for DOF. Note that the log-linear relationship corresponds to a roughly quadratic decrease in the number of histologic features detected. This is to be expected since, intuitively, a twice improved DOF of a two dimensional image yields improved detection along both spatial dimensions and thus a four times increased quantity of histologic features detected.

We now discuss the design and implementation[2] of our multi-scale histologic feature detector, with particular attention paid to optimizations in consideration of inference latency. Eqn. (1) permits a discretization[3] called *Difference of Gaussians* (DoG) (see [10])

$$t^2\nabla^2 L \approx t \times (L(x, y, t + \delta t) - L(x, y, t))$$

Therefore, define

- n, which determines the granularity of the scales detected
- $\min_t$, the minimum scale detected
- $\max_t$, the maximum scale detected
- $\delta t := (\max_t - \min_t)/n$
- $t_i := \min_t + (i-1) \times \delta t$, the discrete scales detected

and finally the discretized DoG

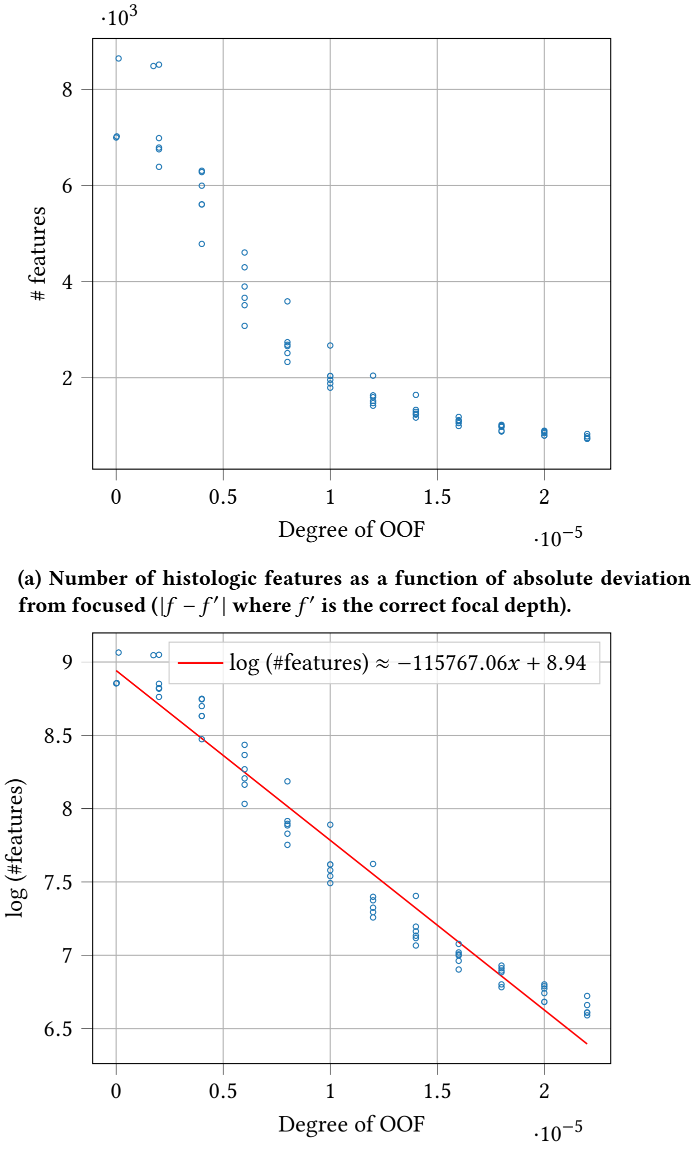
$$\text{DoG}(x, y, i) := t_i \times (L(x, y, t_{i+1}) - L(x, y, t_i)) \tag{2}$$

This produces a sequence $\{\text{DoG}(x, y, i) \mid i = 1, \ldots, n\}$ of filtered and scaled images (called a Gaussian pyramid [4]). Note that there are alternative convetions for how each difference in the definition of $\text{DoG}(x, y, i)$ should be scaled (including partitioning into so called *octaves* [2]) but we empirically determine that linear scaling is sufficient for our needs.

Computing maxima of $\text{DoG}(x, y, i)$ in the scale dimension (equivalently critical points of eqn. (1)) necessarily entails computing local[4] maxima at every scale. We make the heuristic assumption that, in each pixel neighborhood that corresponds to a feature, there is a single unique and maximal response at some scale t. This response corresponds to the scale at which the variance of the Gaussian filter G most closely corresponds to the scale of the feature (see section (2)). We therefore search for local maxima in spatial dimensions x, y but global maxima in the scale dimension

$$\left\{\left(\hat{x}_j, \hat{y}_j, \hat{i}_j\right)\right\} := \underset{x,y}{\text{argmaxlocal}}\,\underset{i}{\text{argmax}}\,\text{DoG}(x, y, i) \tag{3}$$

where the subscript j indexes over the features detected. Once all such maxima are identified it suffices to compute and report the cardinality of $\left\{\left(\hat{x}_j, \hat{y}_j, \hat{i}_j\right)\right\}$ as criterion function value.



(a) Number of histologic features as a function of absolute deviation from focused ($|f - f'|$ where f' is the correct focal depth).

(b) Log plot and line fit with $r = -0.9754$.

Figure 2: Comparison of histologic feature recognition as a function of focal depth.

We now discuss practical (i.e. implementation) optimizations. It is readily apparent that our histologic feature detector is parallelizable; for each scale t_i we can compute $L(x, y, t_i)$ independently of all other $L(x, y, t_j)$ (for $j \neq i$). A further parallelization is possible for the argmax operation, since the maxima are computed independently across distinct neighborhoods of pixels. In order to maximally exploit this we first perform the inner argmax in eqn. (3) on block of columns of $\{\text{DoG}(x, y, i)\}$ in parallel, thereby effectively reducing the Gaussian pyramid to a single image. Note that when GPU memory is sufficient we can compute the argmax across all columns simulataneously (and other wise within a constant number

[2] https://github.com/maksleventhal/cuda_blob/
[3] By virtue of G being the Green's function of the heat equation $t\nabla^2 G = \partial_t G$.
[4] In a small pixel neighborhood in both space and scale dimensions.

Algorithm 1 Multi-scale Histologic Feature Detection

```
Input: $I(x, y), n, \min_t, \max_t, M$
$I'(x, y) :=$ HistorgramStretch($I(x, y)$)
Broadcast($I'(x, y), M$)
parfor $m := 1, \ldots, M$ do
    parfor $i \in I_m$ do
        $L(x, y, t_i) := \mathcal{F}^{-1}\{\mathcal{F}\{G(x, y, t_i)\} \cdot \mathcal{F}\{I'(x, y)\}\}$
    end
end
Gather($L(x, y, t_i), M$)
parfor $i := 1, \ldots, n + 1$ do
    $\text{DoG}(x, y, i) := t_i \times (L(x, y, t_{i+1}) - L(x, y, t_i))$
end
$\{(\hat{x}_j, \hat{y}_j, \hat{t}_j)\} := \text{argmaxlocal}_{x,y} \text{argmax}_i \text{DoG}(x, y, i)$
Output: $\text{DOF} := |\{(\hat{x}_j, \hat{y}_j, \hat{t}_j)\}|$
```

of steps). We then perform the outer argmaxlocal$_{x,y}$ on disjoint pixel neighborhoods of the flattened image in parallel as well.

Note that the implementation of the inner argmax is "free", since the argmax primitive is implemented in exactly this way in most GPGPU libraries [11] and thus our substitution of argmax$_i$ for argmaxlocal$_i$ yields a small but not insignificant latency decrease. The outer argmaxlocal is implemented using a comparison against MaxPool2D(n, n) (with $n = 3$) (see [7] for details on this technique). Employing MaxPool2D in this way has the added benefit of effectively performing non-maximum suppression [12], since it effectively rejects spurrious candidate maxima within a 3×3 neighborhood of a true maximum.

Typically one would compute $L(x, y, t_i)$ in the conventional way (by linearly convolving G and I) but prior work has shown [7] that performing the convolution in the Fourier domain is much more efficient; namely

$$L(x, y, t_i) = \mathcal{F}^{-1}\{\mathcal{F}\{G(x, y, t_i)\} \cdot \mathcal{F}\{I(x, y)\}\}$$

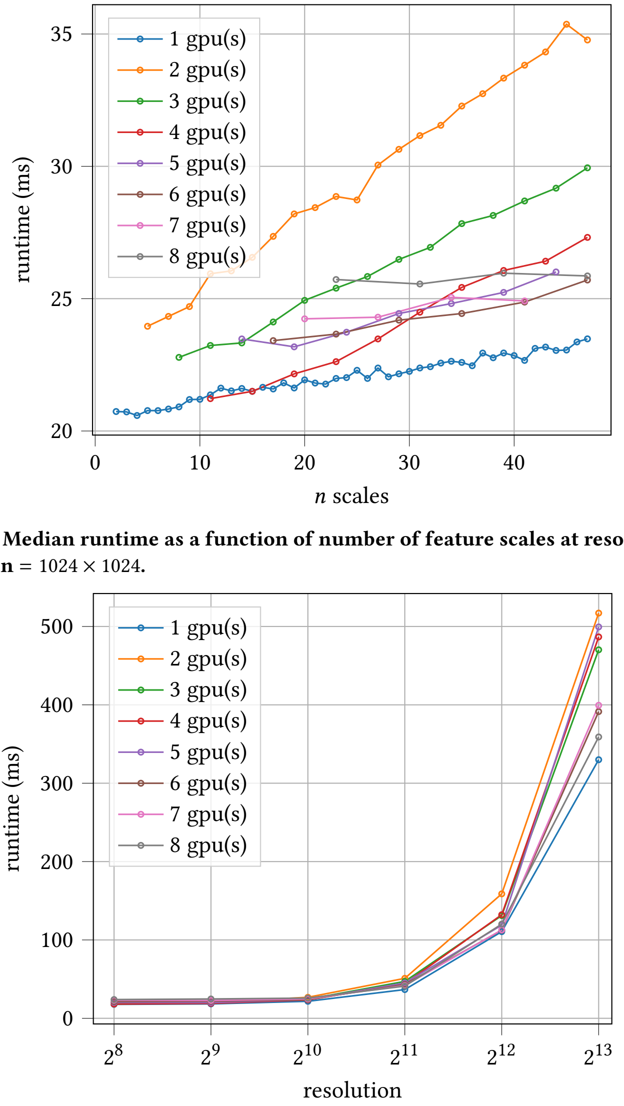
where $\mathcal{F}\{\cdot\}, \mathcal{F}^{-1}\{\cdot\}$ are the Fourier transform and inverse Fourier transform, respectively. This approach has the additional advantage that we can make use of highly optimized Fast Fourier Transform (FFT) routines made available by GPGPU libraries.

One remaining detail is histogram stretching of the images. Due to the dynamic range (i.e. variable bit depth) of the microscope we need to normalize the histogram of pixel values; we do this by saturating .175% of the darkest pixels, saturating .175% of the lightest pixels, and mapping the entire range to [0, 1]. We find this gives us consistently robust results with respect to noise and anomalous features. This histogram normalization is also parallelized using GPU primitives.

In summary our technique is presented in alg. (1).

4 EVALUATION

Brains were prepared in the same manner and as previously described]. Briefly, an anesthetized animal was first transcardially perfused with 10ml 0.1 M Sodium Cacodylate (cacodylate) buffer, pH 7.4 (Electron microscopy sciences (EMS) followed by 20 ml of fixative containing 2% paraformaldehyde (EMS), 2.5% glutaraldehyde (EMS) in 0.1 M Sodium Cacodylate (cacodylate) buffer, pH 7.4 (EMS).



(a) Median runtime as a function of number of feature scales at resolution = 1024×1024.

(b) Median runtime as a function of section resolution with 16 feature scales.

Figure 3: Scaling experiments for runtime with respect to number of GPUs, resolution, and number of feature scales.

The brain was removed and placed in fixative for at least 24 hours at 4C. A series of 300 um vibratome sections were prepared and put into fixative for 24 hours at 4C. The primary visual cortex (V1) was identified using areal landmarks and reference atlases. A small piece (2 x 2 mm) containing V1 was cut out and prepared for EM by staining sequentially with 2% osmium tetroxide (EMS) in cacodylate buffer, 2.5% potassium ferrocyanide (Sigma-Aldrich), thiocarbohydrazide, unbuffered 2% osmium tetroxide, 1% uranyl acetate, and 0.66% Aspartic acid buffered Lead (II) Nitrate with extensive rinses between each step with the exception of potassium ferrocyanide. The tissue was then dehydrated in ethanol and propylene oxide and infiltrated with 812 Epon resin (EMS, Mixture: 49% Embed 812, 28%

Table 1: Test platform

CPU	Dual AMD Rome 7742 @ 2.25GHz
GPU	8x NVIDIA A100-40GB
HD	4x 3.84 U.2 NVMe SSD
RAM	1TB
Software	CuPy-8.3.0, CUDA-11.0, NVIDIA-450.51.05

DDSA, 21% NMA, and 2.0% DMP 30). The resin-infiltrated tissue was cured at 60oC for 3 days. Using a commercial ultramicrotome (Powertome, RMC), the cured block was trimmed to a 1.0mm x 1.5 mm rectangle and 2,000, 40nm thick sections were collected on polyimide tape (Kapton) using an automated tape collecting device (ATUM, RMC) and assembled on silicon wafers as previously described (ref??). Images at different focal distances were acquired using backscattered electron detection with a Gemini 300 scanning electron microscope (Carl Zeiss), equipped with ATLAS software for automated imaging. Dwell times for all datasets were 1.0 microsecond.

We perform runtime experiments across a range of parameters of interest (section resolution, number of feature scales). Our test platform is a NVIDIA DGX A100 (see table 1). Experiments consist of computing the DOF of a sample section for a given configuration. All experiments are repeated k times (with $k = 21$) and all metrics reported are in fact median statistics[5].

For a section resolution of 1024×1024 pixels we achieve approximately a 50Hz runtime in the single GPU configuration; this is near-real-time. We observe that, as expected, runtime grows linearly with the number of feature scales and quadratically with the resolution of the section; naturally, this is owing to the parallel architecture of the GPU. The principle defect of our technique is that it is highly dependent on the available RAM of the GPU it is deployed to. In practice, most GPUs available at the edge, i.e. proximal to microscopy instruments, will have insufficient ram to accommodate large section resolutions and wide feature scale ranges. In fact, even the 40GB of the DGX's A100 is exhausted at resolutions above 4096×4096 for more than approximately 20 feature scales.

Therefore, we further investigate parallelizing MHFD across multiple GPUs. Our implementation parallelizes MHFD in a straightforward fashion: we partition the set of filters across the GPUs, perform the "lighter" FFT-IFFT pair on each constituent GPU, and then gather the results to the root GPU (arbitrarily chosen). That is to say we actually carry out

$$\{L(x, y, t_i) \mid i \in I_m\} = \left\{\mathcal{F}^{-1}\left\{\mathcal{F}\left\{G(x, y, t_i)\right\} \cdot \mathcal{F}\left\{I(x, y)\right\}\right\} \mid i \in I_m\right\}$$

where for $m = 1, \ldots, M$ the set I_m indexes the scales allocated to a node m. By partitioning the set of Gaussian filters $\{G(x, y, t_i)\}$ across M nodes, we effectively perform distributed filtering. We use CUDA-aware OpenMPI to implement the distribution. Note that for such multi-GPU configurations the range of feature scales was chosen to be a multiple of the number of GPUs (hence the proportionally increasing sparsity of data in figure (3a)). We observe that, as one would expect, runtime is inversely proportional to number of GPUs (see figure (3b)) but that for instances where a single GPU configuration is sufficient it is also optimal. More precise timing reveals that parallelization across multiple GPUs incurs high network copy costs during the gather phase (see figure (4)). Note that this latency persists even after taking advantage of CUDA IPC [13]. In effect, this is a fairly obvious demonstration of Amdahl's law. Therefore, parallelization across multiple GPUs should be considered in instances where full resolution section images are necessary[6].

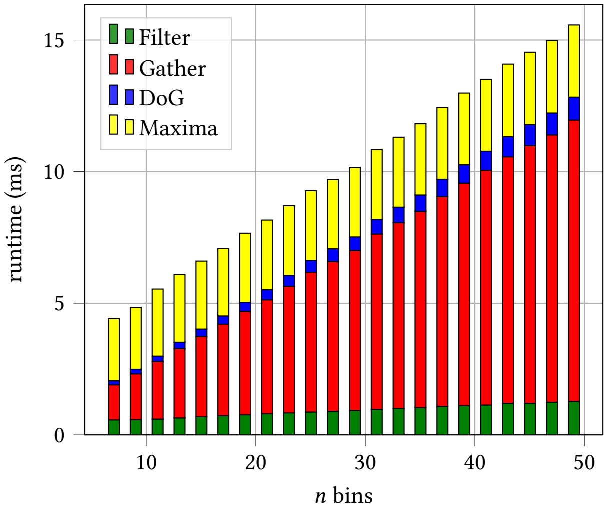


Figure 4: Breakdown of runtime into the four major phases for two GPUs across feature scales at resolution = 1024×1024.

5 DISCUSSION

As stated in the introduction our intent here was to study an OOF detection technique in order to augment existing microscopy instrumentation. Rather than focusing the microscope, as auto-focusing algorithms would, our algorithm operates downstream of image acquisition and reports out-of-focus events to the user. This enables the user to intervene and initiate reacquisition protocols (on the microscope) before unknowingly proceeding with collecting the next series of images or proceeding with downstream image processing and analysis. Our technique is effective and operates at near-real-time latencies. Thus, this human-in-the-loop remediation protocol already saves the user much wasted collection time and tedium in triaging defective collection runs. Note that MHFD could in fact be employed in an iterative mode in order that the DOF reported were used to adjust the focus of the microscope. This would require close integration with the existing software and actuation hardware of the microscope.

Though our technique is efficient for use with a single GPU for small to moderately sized image sections we emphasize that distribution across multiple nodes will inevitably be necessary for use with microscopy instrumentation in the near future. The current state of the art ZEISS MultiSEM 505/506 employs 91 parallel electron

[5] We discard the first execution since it is an outlier due to various initializations (e.g. pinning CUDA memory).

[6] For example, when feature scale range are very wide, with detection at the lower end of the scale being critical. In all other cases downsampling by bilinear interpolation in order to satisfy GPU RAM constraints yields a more than reasonable tradeoff between accuracy and latency.

beams and images an entire 52 tile series in approximately 1.3s [3]; this is approximately 25ms per tile or exactly 50Hz (i.e. exactly the rate at which our technique operates). For maximal efficiency in the end-to-end automation of connectomics our solution (or refinements thereof) will need to be deployed and made available to researchers.

6 RELATED WORK

There is much work in developing and improving auto-focus algorithms and their applications to microscopy. Yeo et al. [15] was one of the first investigations of applying auto-focus to microscopy. They compare several criterion functions and conclude that the so-called Tenengrad criterion function is most accurate and most robust to noise. The crucial difference between their evaluation criteria and ours is they select for criterion functions that are suited for optical microscopy, i.e. criterion functions that are robust to staining/coloring (where as all of our samples are grayscale). Redondo et al. [14] reviews sixteen criterion functions and their computational cost in the context of automated microscopy. Bian et al. [1] address the same issues that motivate us in that they aim to support automated processes in the face of topographic variance in the samples (which lead to compare OOF rates). Their solutions distinguish themselves in that they employ active devices (such as low-coherence interferometry). Interestingly, seemingly contemporaneously with our project Luo et al. Luo et al. [9] proposed a deep learning architecture that auto-focuses in a "single-shot" manner. Such a solution is quite appealing given the affinity with our own application of GPGPU to the problem and we intend to experiment with applying it to our data.

ACKNOWLEDGMENTS

This work was supported by the U.S. Department of Energy, Office of Science, under contract DE-AC02-06CH11357.

REFERENCES

[1] Zichao Bian, Chengfei Guo, Shaowei Jiang, Jiakai Zhu, Ruihai Wang, Pengming Song, Zibang Zhang, Kazunori Hoshino, and Guoan Zheng. 2020. Autofocusing technologies for whole slide imaging and automated microscopy. *Journal of Biophotonics* 13, 12 (2020), e202000227. https://doi.org/10.1002/jbio.202000227 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/jbio.202000227

[2] P. Burt and E. Adelson. 1983. The Laplacian Pyramid as a Compact Image Code. *IEEE Transactions on Communications* 31, 4 (1983), 532–540. https://doi.org/10.1109/TCOM.1983.1095851

[3] Carl Zeiss AG 2018. *ZEISS MultiSEM Research Partner Program*. Carl Zeiss AG. The World's Fastest Scanning Electron Microscopes.

[4] Konstantinos G Derpanis. 2005. The gaussian pyramid. (2005).

[5] Remco Duits, Luc Florack, Jan De Graaf, and Bart ter Haar Romeny. 2004. On the axioms of scale space theory. *Journal of Mathematical Imaging and Vision* 20, 3 (2004), 267–298.

[6] Jan J Koenderink. 1984. The structure of images. *Biological cybernetics* 50, 5 (1984), 363–370.

[7] M. Levental, R. Chard, J. A. Libera, K. Chard, A. Koripelly, J. R. Elias, M. Schwarting, B. Blaiszik, M. Stan, S. Chaudhuri, and I. Foster. 2020. Towards Online Steering of Flame Spray Pyrolysis Nanoparticle Synthesis. In *2020 IEEE/ACM 2nd Annual Workshop on Extreme-scale Experiment-in-the-Loop Computing (XLOOP)*. IEEE Computer Society, Los Alamitos, CA, USA, 35–40. https://doi.org/10.1109/XLOOP51963.2020.00011

[8] T. Lindeberg. 2004. Feature Detection with Automatic Scale Selection. *International Journal of Computer Vision* 30 (2004), 79–116.

[9] Yilin Luo, Luzhe Huang, Yair Rivenson, and Aydogan Ozcan. 2021. Single-Shot Autofocusing of Microscopy Images Using Deep Learning. *ACS Photonics* 8, 2 (2021), 625–638. https://doi.org/10.1021/acsphotonics.0c01774 arXiv:https://doi.org/10.1021/acsphotonics.0c01774

[10] David Marr and Ellen Hildreth. 1980. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207, 1167 (1980), 187–217.

[11] Duane Merrill. CUDA Unbound (CUB). https://github.com/NVIDIA/cub.

[12] A. Neubeck and L. Van Gool. 2006. Efficient Non-Maximum Suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, Vol. 3. 850–855. https://doi.org/10.1109/ICPR.2006.479

[13] S. Potluri, H. Wang, D. Bureddy, A. K. Singh, C. Rosales, and D. K. Panda. 2012. Optimizing MPI Communication on Multi-GPU Systems Using CUDA Inter-Process Communication. In *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops PhD Forum*. 1848–1857. https://doi.org/10.1109/IPDPSW.2012.228

[14] Rafael Redondo, Gabriel Cristóbal, Gloria Bueno Garcia, Oscar Deniz, Jesus Salido, Maria del Milagro Fernandez, Juan Vidal, Juan Carlos Valdiviezo, Rodrigo Nava, Boris Escalante-Ramírez, and Marcial Garcia-Rojo. 2012. Autofocus evaluation for brightfield microscopy pathology. *Journal of Biomedical Optics* 17, 3 (2012), 1 – 9. https://doi.org/10.1117/1.JBO.17.3.036008

[15] TTE Yeo, SH Ong, Jayasooriah, and R Sinniah. 1993. Autofocusing for tissue microscopy. *Image and Vision Computing* 11, 10 (1993), 629–639. https://doi.org/10.1016/0262-8856(93)90059-P

[16] Yu Sun, S. Duthaler, and B. J. Nelson. 2005. Autofocusing algorithm selection in computer microscopy. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 70–76. https://doi.org/10.1109/IROS.2005.1545017