A Scalable Distributed Framework for Multimodal GigaVoxel Image Registration

Anonymous authors
Paper under double-blind review

Abstract

In this work, we propose **FFDP**, a set of IO-aware non-GEMM fused kernels supplemented with a distributed framework for image registration at unprecedented scales. Image registration is an inverse problem fundamental to biomedical and life sciences, but algorithms have not scaled in tandem with image acquisition capabilities. Our framework complements existing model parallelism techniques proposed for large-scale transformer training by optimizing non-GEMM bottlenecks and enabling convolution-aware tensor sharding. We demonstrate unprecedented capabilities by performing multimodal registration of a $100\mu m$ *ex-vivo* human brain MRI volume at native resolution – an inverse problem more than $570\times$ larger than a standard clinical datum in about a minute using only 8 A6000 GPUs. FFDP accelerates existing state-of-the-art optimization and deep learning registration pipelines by upto $6-7\times$ while reducing peak memory consumption by $20-59\%$. Comparative analysis on a $250\mu m$ dataset shows that FFDP can fit upto $64\times$ larger problems than existing SOTA on a single GPU, and highlights both the performance and efficiency gains of FFDP compared to SOTA image registration methods.

1 Introduction

Image registration (also called 'image alignment' or 'image matching') is a non-linear inverse problem ubiquitous in biomedical and life sciences. Given d-dimensional images $F : \Omega \to \mathbb{R}^d$ and $M : \Omega \to \mathbb{R}^d$ defined on domain Ω (usually a compact subset of $\mathbb{R}^d$), image registration seeks to find a coordinate transform $\varphi : \Omega \to \Omega$ that deforms the moving image M to look similar to the fixed image F. Mathematically, we minimize the following objective (Fig. 1):

$$\varphi^* = \arg\min_{\varphi \in G} L(\varphi) \doteq C(F, M \circ \varphi) + R(\varphi) \tag{1}$$

where C is a cost or dissimilarity function, and $\circ$ is the interpolation operator, i.e. $(I \circ g)(x) = I(g(x))$ for all $x \in \Omega$. Popular choices of φ are affine and deformable transforms, i.e. $\varphi(x) = Ax + t$, and $\varphi(x) = x + u(x)$. Modern registration pipelines (Hoffmann et al., 2021; Jena et al., 2024a) consider an affine matching followed by a deformable matching step, resulting in a composite transform $\varphi(x) = Ax + t + u(x)$. u is called the displacement field, modeled as a grid of per-voxel vectors $u(x) \in \mathbb{R}^d$. For an image of size N, the displacement field is a tensor of size dN. We use $[\mathbf{x}]_\Omega$, $A[\mathbf{x}]_\Omega + t$, and $[\mathbf{u}]_\Omega$ to denote the identity grid, grid of affine transformed coordinates, and deformation grid defined on Ω respectively. Common choices of C are mean squared error, Localized Normalized Cross Correlation (Avants et al., 2008a), and Mattes Mutual Information (Mattes et al., 2001). Common choices of R include Sobolev norm of the gradient or warp fields (Beg et al., 2005; Mang et al., 2019; Avants et al., 2008b), total variation, and inverse-consistency (Christensen & Johnson, 2001). To optimize Eq. (1), iterative methods optimize φ^* directly using gradient descent, and deep learning methods learn a deep neural network $\varphi = f_\theta(F, M)$. Image registration establishes a common coordinate system, aligning scans across individuals and atlases (Hering et al., 2022; Marcus et al., 2007; Murphy et al., 2011). This alignment is a prerequisite for multimodal data fusion, cross-subject comparison, morphometric analysis (Das et al., 2009), and construction of large-scale atlases (Wang et al., 2020b). Establishing such voxelwise correspondence is fundamental for studying anatomical variability, detecting pathological signatures (Ravikumar et al., 2021), and advancing precision medicine (Börner et al., 2022; Jonsson et al., 2022). The saliency and centrality of the task across various biomedical and life science applications has spurred numerous methodological

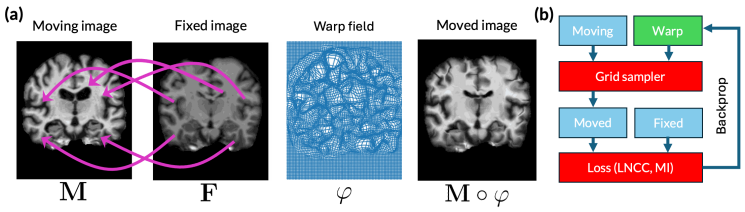


Figure 1: Image Registration Problem. (a): The task is to find a coordinate transform that warps the moving image M to the fixed image F. Individual corresponding points are shown as violet arrows; the per-pixel coordinate transform is shown as a warp field φ, and the transformed image $M \circ \varphi$. **(b)**: A typical registration pipeline - the grid sampler warps the moving image, that is then compared to the fixed image using a loss function. **Green** denotes the optimizable warp, **red** denotes the primary bottlenecks that we optimize in this paper.

advances in the field, spanning more than three decades of research (Gee et al., 1993; Tian et al., 2024).

Over the past decade, advances in MRI, CT, PET, STPT, and microscopy have enabled ultra-high-resolution imaging, often more than three orders of magnitude larger than macroscopic biomedical domains (Balchandani & Naidich, 2015; Esquivel et al., 2022; Badawi et al., 2019; Gambarotto et al., 2019; Wassie et al., 2019; Kleven et al., 2023; Wang et al., 2020b; Mansour et al., 2025; Kleinfeld et al., 2011). While a typical clinical registration problem involves $\sim$20M parameters, high-resolution ex-vivo human brain scans can require solving up to 11B parameters, far beyond the $\sim$50M-parameter scale at which current registration methods remain reliable. As a result, state-of-the-art deformable image alignment struggles to scale to the resolutions demanded in modern neuroimaging, computational pathology, developmental biology, and connectomics, creating a substantial performance gap. In parallel, innovations in large-scale transformer training such as IO-aware fused operations (Dao et al., 2022; Dao, 2023; Spector et al., 2025) and 5D parallelism for distributing larger-than-memory workloads (Shoeybi et al., 2019; Li et al., 2023; Jacobs et al., 2024; Li et al., 2024; Zhao et al., 2023; Ansel et al., 2024) optimize GEMM-like workflows. However, the fundamental concepts utilized by these methods (IO-awareness, recomputing and aggregating intermediates on shared memory to minimize high bandwidth memory (HBM) storage, identifying partial aggregates across hosts to minimize communication overheads for distributed optimization) are broadly applicable to a wide class of problems of the non-GEMM nature.

In this paper, we apply these concepts to scale image registration algorithms to match parity with the developments in both increasing resolution of image acquisition *and* compute capabilities. To that end, our contributions are twofold. First, we identify key compute and memory bottlenecks in image registration algorithms, and propose novel components that fit problems upto $64\times$ larger than existing algorithms on a single GPU. Second, we propose **F**lash **F**used **D**istributed **P**rimitives (FFDP), a distributed framework to scale registration to an arbitrary number of GPUs, thereby scaling to ultra high-resolution problems. We present a first-of-its-kind demonstration: aligning a $250\mu m$ in-vivo MRI (Lüsebrink et al., 2017) to a $100\mu m$ *ex-vivo* human brain FLASH volume (Edlow et al., 2019) – a multimodal registration problem more than **570**$\times$ larger than a standard clinical datum (Marcus et al., 2007), with over 11.8B transform parameters – completed in *one minute* using only 8 A6000 GPUs. FFDP accelerates existing traditional registration pipelines by upto $7.48\times$ while reducing memory consumption by upto 59%, and deep learning pipelines by upto $6.14\times$ while consuming upto 24% less memory. We highlight the necessity of performing high-resolution registration by comparing our method with various SOTA optimization and deep learning baselines on a $250\mu m$ T1-weighted MRI dataset, showing unprecedented performance and gains in efficiency.

2 Related Work

2.1 Memory Efficient and Large Scale Optimization

Recent years have also witnessed tremendous innovations in large-scale transformer model training. IO-aware implementations typically include individual fused kernels (Dao et al., 2022; Dao, 2023) and domain-specific languages (Spector et al., 2025; PyTorch, 2025) to minimize launch latency and large memory overheads. To distribute larger-than-memory model training workloads across multiple GPUs, 5D parallelism techniques (Shoeybi et al., 2019; Li et al., 2023; Jacobs et al.,

2024; Li et al., 2024; Zhao et al., 2023; Ansel et al., 2024) have been proposed. Many of these techniques leverage a divide-and-conquer approach to break down a larger GEMM-like operation like matrix multiplication or attention into smaller sub-problems that can be executed on multiple GPUs and synchronized to compute the final result. To our knowledge, most of these techniques are tailored to transformer-specific architectures and GEMM-like operations (self attention, FeedForward, LayerNorm, etc.) only, and a Model Parallel variant for convolution-aware tensor sharding and synchronization is not available.

2.2 Large Scale Registration in Life Sciences and Biomedical Imaging

***Ex-vivo* neuroimaging.** A large body of neuroanatomical studies are performed in conjunction with high-resolution ex-vivo MRI, blockfacing imaging, and histology to create detailed, microscopic anatomical references by integrating structural, molecular, and cytoarchitectural information across imaging modalities (Casamitjana et al., 2025; Ravikumar et al., 2024). In-vivo MRI is typically limited by resolution due to constraints on scan time. Consequently, high-resolution ex-vivo scans and blockface imaging are used as a bridge between in-vivo and histology, with the latter used as a gold standard for ground-truth microscopic tissue characterization and pathology. Numerous studies on neurodegenerative diseases including Alzheimer's disease, Parkinson's, and Multiple Sclerosis use high resolution ex-vivo MRI and histology to study disease progression and treatment effects (Madsen et al., 2021; Echávarri et al., 2011; Welton et al., 2023). Most studies only quantify local effects due to the significant computational cost of registering the entire brain at high resolution. Recently, multiple large scale consortia including Seattle Alzheimer's Disease Brain Cell Atlas (SEA-AD) and the Human Mouse Brain Atlas (HMBA) consortium are aimed at creating detailed, multimodal brain atlases linking cellular, molecular, and anatomical organization across species and disease states - combining individual efforts from multiple institutions together into a unified resource. Moreover, submillimeter whole-brain datasets (Edlow et al., 2019; Lüsebrink et al., 2017; Mahler et al., 2024) have been acquired with the goal to facilitate the development and validation of new algorithms for high-res data, provide detailed studies of the brain anatomy, and act as a high-resolution template for integration with other modalities or individual brain studies. However, existing tools cannot register these datasets at native resolution due to excessive memory requirements; we show that our method can register these datasets at native resolution in a minute using only 8 A6000 GPUs (see Section 5.2). High-resolution imaging *and* registration are essential in these contexts because they enable accurate cross-modal alignment and preservation of fine anatomical detail that would otherwise be lost through downsampling.

Large-scale registration in model organisms. Over the past decade, imaging across the life sciences and biomedical domains has progressed from mesoscale surveys to organ- and organism-wide acquisitions at cellular or even subcellular resolution. These span transparent organisms and small animal models (e.g., *C. elegans*, zebrafish, adult *Drosophila*) (Varol et al., 2020; Venkatachalam et al., 2016; Marquart et al., 2017; Gupta et al., 2018; Peng et al., 2011; Brezovec et al., 2024), adult mouse and rat brains imaged at sub-micron resolutions (Gong et al., 2016; Wang et al., 2020a; Kleven et al., 2023) using Light Sheet Fluorescence Microscopy (LSFM) and Serial Two-Photon Microscopy (STPT) imaging. Such modalities routinely generate gigavoxel to teravoxel volumes (Kutten et al., 2016; Nazib et al., 2018). Their scientific utility, however, hinges on the ability to perform registration at the native resolution of acquisition, i.e. aligning specimens (or modalities) in a common coordinate system without sacrificing the fine-scale morphologies including cell bodies, layers, axon bundles, synaptic neighborhoods, etc. that motivate high-resolution acquisition in the first place (Nazib et al., 2018; Goubran et al., 2013).

Across these diverse domains, the unifying requirement demands access to scalable multimodal registration algorithms - a challenge we address in this work. We provide an extended discussion of more related work and the necessity of our approach in Section A.

3 Fused Kernels for Memory Efficient Registration on a Single GPU

Bottlenecks of a deformable image registration pipeline Our primary objective is to identify compute and memory bottlenecks in *large-scale* image matching tasks. In identifying these bottlenecks, training-free optimization methods are better suited than deep networks since the latter has a much larger activation memory footprint, which forms the primary memory bottleneck (Tazi et al., 2024).

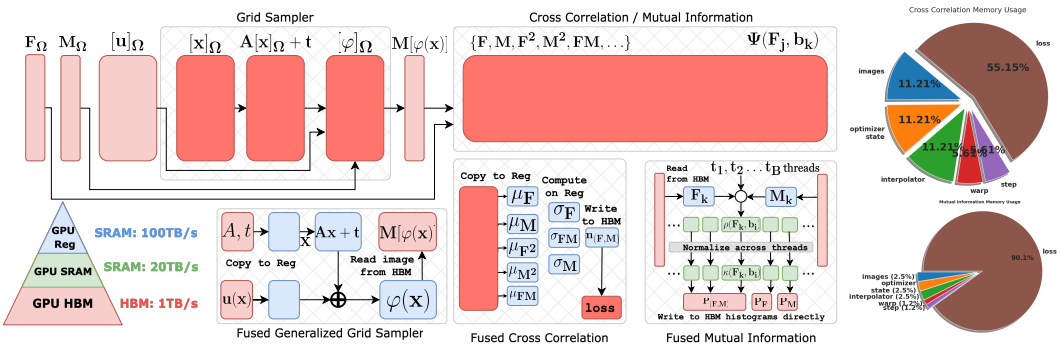


Figure 2: Left: FFDP uses fused kernels to eliminate intermediate HBM memory usage (in **dark red**) for memory-bound workhorse operations (`grid_sampler`, LNCC, MI) for large-scale image registration. For `grid_sampler` and LNCC, additional intermediate per-pixel variables (warp coordinates, patchwise statistics) are computed per-pixel in registers (blue). For MI, the Parzen Windowing and histogram aggregation is performed using shared memory (green), avoiding large HBM overheads. **Right**: Pie charts show the breakdown of memory overheads for storing the image, grid, optimizer state, and intermediate variables for MI and LNCC losses.

For instance, for a $250\mu m$ image pair, a standard deep learning method (Hoffmann et al., 2021) generates an activation map of size 27GB only after the first layer. Extrapolating memory usage for clinical data, existing deep networks will require upto 1.2TB of GPU memory at inference to process these image volumes at native resolution. In contrast, a training-free optimizer can fit this problem in less than 45GB of GPU memory. We use FireANTs (Jena et al., 2024a) as our base framework to identify compute and memory bottlenecks in a typical image registration problem. We analyze the flamegraph of a typical clinical MRI registration task from the *OASIS* brain dataset (Marcus et al., 2007) in Fig. 20. We identify three key memory bottlenecks in image matching pipelines (1) deformable interpolation and warp composition (2) cross-correlation loss, and (3) mutual information loss (see Fig. 2(right)). [1] We first propose efficient designs to fit larger problems on a single GPU, and then extend the framework to distributed registration.

3.1 Composite Implicit Grid Sampler

A fundamental operation used in image registration is the *grid sampler*. This operator allows us to warp an image M using a deformation field $\varphi : \Omega \to \Omega$ and computes the image $M' : M'(x) = M(\varphi(x))$. Virtually every image registration pipeline uses this operation to warp the moving image using an affine, deformable, or composite transform. For affine and composite transforms, the operator initializes a regular grid $[\mathbf{x}]_\Omega$, a grid of size $3N$. The affine grid $A[\mathbf{x}]_\Omega + t$ is another grid of size $3N$. If a deformable grid $[\mathbf{u}]_\Omega$ is optimized, then a third grid $A[\mathbf{x}]_\Omega + t + [\mathbf{u}]_\Omega$ is materialized, costing a total of $9N$ overhead for an image of size N. To consolidate these memory overheads, we propose a composite implicit grid sampler. This is a fused CUDA kernel that performs the following operation:

$$\texttt{fused\_grid\_sampler}(I; A, t, [\mathbf{u}], S, x_{\text{bounds}})(x) = I(Ax + t + Su(x))$$

where $A, S \in GL(d, \mathbb{R})$ are affine matrices, t is a translation vector, $[\mathbf{u}]$ is the deformation grid, and x_{bounds} are the bounds of the (implicit) identity grid $[\mathbf{x}]_\Omega$. There are three benefits of this approach. First, the kernel avoids materializing any additional grids in HBM, reducing the memory overhead of the kernel from $O(n)$ to $O(1)$ with no loss in runtime or accuracy. Second, when the warp $[\mathbf{u}]_\Omega$ is sharded across hosts in a distributed setting, the identity grid $[\mathbf{x}]_\Omega$ needs to be sharded correctly too. Since the identity grid is implicitly defined by its bounds $x_{\text{bounds}} = (x_{\min}, x_{\max}) \in \mathbb{R}^{2d}$, our implementation can be easily used in a distributed optimization setting without instantiating partial shards $[\mathbf{x}]_{\Omega_h}$. Finally, the matrix S is used to rescale the deformation field to sample from the coordinates of the sharded images I_h which lie on the grid Ω_h instead of Ω (see Section I.2) without initializing additional memory. The backward pass is very similar to the existing PyTorch implementation, with the exception of the gradient of the affine matrix. We discuss the derivation and pseudocode of the forward and backward pass in the Section H.

[1] A GPU's memory hierarchy spans multiple tiers: registers (per-thread, single-cycle), shared memory/L1 cache (on-chip, tens of KB, low latency within a block), L2 cache (MBs, shared across SMs, moderate latency), and global memory (**HBM**). Our work focuses on reducing HBM usage for key non-GEMM operations used in image registration, by maximizing register and shared memory usage while minimizing global memory traffic.

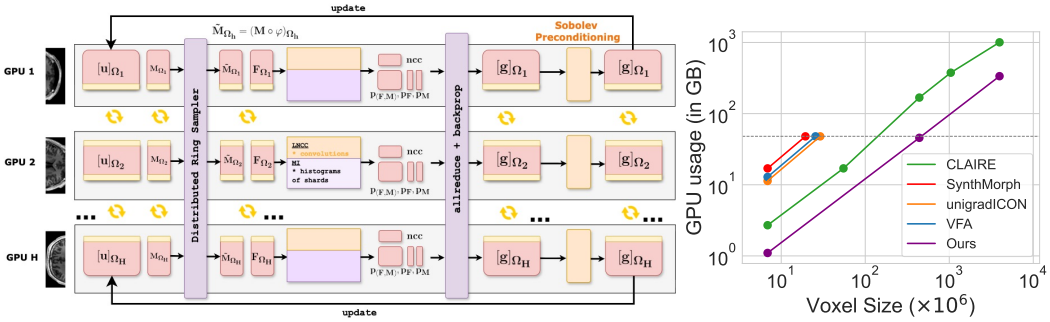


Figure 3: Left: Overview of our distributed framework. GridParallel (GP) shards the fixed and moving images (F, M) and the warp field $[\mathbf{u}]$ across multiple GPUs. Yellow blocks and arrows denote synchronized halo boundaries between GPUs, enabling smoothing on images and warp fields without an allgather. The ring sampler (violet) computes interpolated image shards on the fly, avoiding materialization of the full moving image. We then compute losses (MSE, LNCC, MI), compute gradients w.r.t. each warp shard, apply Sobolev regularization with GP, and update shards by gradient descent. **Right**: Scaling efficiency compared to deep methods and CLAIRE (Mang et al., 2019), a distributed registration method. Most SOTA deep learning baselines require orders-of-magnitude more memory for the same problem size and scalability is limited to a single GPU (dotted line). Our framework scales to arbitrarily large problem sizes while using about $5\times$ less memory than CLAIRE.

3.2 Implicit Parzen Windowing for Mutual Information

Mattes Mutual Information (MI) is one of the most commonly used loss functions for *multimodal* image matching (Chen et al., 2022; Avants et al., 2009; Mattes et al., 2001). For random variables X and Y, MI is the KL divergence between the joint distribution $P(X, Y)$ and product of marginal distributions $P(X)P(Y)$ of the intensities of the two images. For image matching, X and Y are the pixel intensities for the images I, J. The distributions are estimated using a kernel density estimator:

$$P_I(v) = \frac{1}{N}\sum_k \kappa(v - I_k), \quad P_{(I,J)}(v, w) = \frac{1}{N}\sum_k \kappa(v - I_k)\kappa(w - J_k) \tag{2}$$

where κ is a kernel function of choice. Common choices of κ are the Gaussian (Guo, 2019) and 3rd order B-Spline kernels (Thévenaz & Unser, 2000). To empirically compute the KL divergence, the distributions Eq. (2) are discretized over B equally spaced bins on the domain of $u \in I, v \in J$. However, to compute the joint histogram of size B^2, this method requires materializing the entire Parzen Block $\Psi_I(j, k) = \kappa(b_j - I_k)$ of size $2k_P BN$, where k_P is a kernel-dependent constant. Since $N >> B$ (B is typically chosen to be 32), this operation becomes a significant memory bottleneck for large N. For instance, a typical clinical image volume ($N \approx 30$MB) with 32 bins will consume **7.5GB** of HBM - a significantly huge cost that grows much faster for larger problems.

Our efficient implementation leverages the fact that B is small to avoid materializing the tensors $\Psi_I, \Psi_J \in \mathbb{R}^{B\times N}$ altogether and use high-throughput shared memory to compute and accumulate the histogram entries and partial gradients for each image pixel. We provide the detailed derivation in Section G. This leads to an efficient implementation that consumes $O(1)$ additional HBM instead of $O(N)$ (holding B constant). This leads to upto **98%** lesser HBM usage for images considered in our experiments, and an asymptotic 100% reduction in HBM usage for large images (Fig. 7(top-right)).

3.3 Efficient Implicit Fused Cross-Correlation

Local Normalized Cross-Correlation (LNCC) is used ubiquitously in signal and image processing as a similarity metric. In deformable image registration, it is used as a robust similarity function to compare anatomical similarities (Chen et al., 2022; Hoffmann et al., 2021; Avants et al., 2008b; Wu et al., 2024). Most LNCC implementations are memory-bound due to the large number of intermediate variables. Our analysis in Section F shows that the computational graph adds $16\times$ HBM overhead, and upto another $16\times$ HBM overhead for computing gradients with respect to all intermediates.

To avoid these huge memory overheads, we fuse all the intermediate computation in a fused kernel. Our fused forward pass requires only $5\times$ memory for storing all intermediates (I, J, I^2, J^2, IJ convolved with matrix w). In Section F we analytically derive the gradient and show that the input gradients can be computed by modifying the saved intermediates *in-place*. This leads upto a **76.5% reduction** in memory (see Table 3) and outperforms even `torch.compile` implementations.

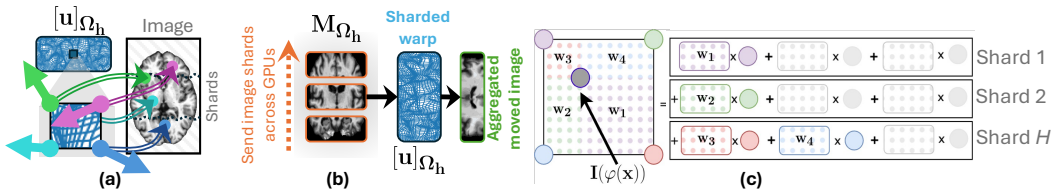


Figure 4: **(a)** Neighboring coordinates in the warp field may refer to pixel locations on arbitrary image shards due to the deformable nature of the warp field, making distributed interpolation non-trivial. **(b)** Ring Sampler interleaves fetching of image shards and aggregating the partial sums of interpolated values, avoiding a memory-expensive allgather. **(c)** Bilinear Interpolation is decomposed into partial sums over image shards, which are accumulated with a ring topology communication, similar to Liu et al. (2024b).

4 Extending Image Registration to Multiple GPUs

Our composite implicit grid sampler and improved loss functions allows optimizing problems with image sizes that are upto two magnitudes larger than other baselines on a single A6000 GPU (Fig. 5a). However, many applications using mesoscopic and microscopic data require registration of images that do not fit on a single GPU. Inspired by distributed frameworks for LLM training (Shoeybi et al., 2019; Rajbhandari et al., 2020) and initial work on distributed image registration (Mang et al., 2019), we propose a distributed framework that allows sharding large images across multiple GPUs to efficiently scale to arbitrarily large problem sizes with any similarity loss function.

Distributed Setting. For distributed registration with H hosts or GPUs, we partition the domain $P(\Omega) = \{\Omega_1, \Omega_2, \dots \Omega_H\}$ such that $|\Omega_i| = N/H$, $\Omega_i \cap \Omega_j = \phi \quad \forall i \neq j$ and $\cup_i \Omega_i = \Omega$. We use $[\mathbf{x}]_{\Omega_h}$, $A[\mathbf{x}]_{\Omega_h} + t$, and $[\mathbf{u}]_{\Omega_h}$ to denote the sharded tensors defined on domain Ω_h.

4.1 Grid Parallel for Boundary-Synchronized Image Sharding

Techniques like Tensor/Sequence/Expert/Context Parallel have been tremendously successful in distributed optimization by sharding large models and sequences across multiple GPUs (Shoeybi et al., 2019; Li et al., 2023; Liu et al., 2024b;a). However, these techniques work for transformer-like architectures and input sequences where the model parameters and activations do not require boundary synchronization. In contrast, image registration contains operations that require boundary synchronization between image and grid shards to perform mathematically correct convolutions. Examples of such operations include convolutions for calculating LNCC, total variation loss, Sobolev norm of the gradient and warp fields (Mang et al., 2019; Avants et al., 2008b; Beg et al., 2005).

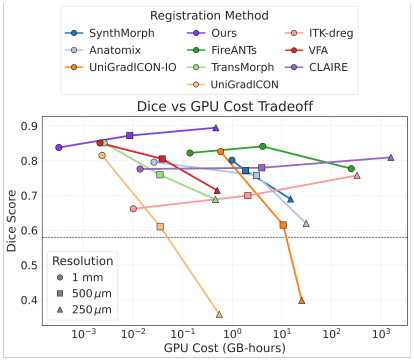
To enable these functionalities and complement existing parallelism techniques, we propose '*Grid Parallel*' (GP) as an abstraction on a tensor. GP shards a tensor across hosts, stores the sharded dimension and bounds as metadata, and provides synchronization operations to augment the tensor with sufficient boundary padding from neighboring shards prior to performing a convolution operation. GP allows us to partition the fixed images, $[\mathbf{u}]$, and the optimizer state $[\mathbf{m_1}], [\mathbf{m_2}]$ – essentially sharding the entire problem across H hosts while allowing the user to apply convolutional operations seamlessly. We compare the performance of GP with naive DTensor sharding in Section D.

4.2 Distributed Ring Sampler

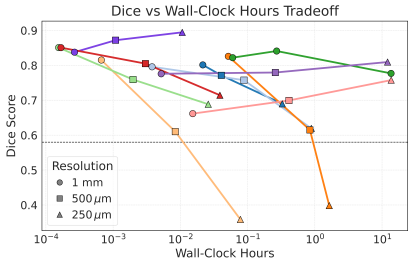
Despite the sharding in GP, the moving image M cannot be sharded across GPUs due to the random-access nature of the `grid_sample` operation applied on M. In general, the warp vector $\varphi(x)$ residing on GPU i can point to coordinates that reside on the sharded image on GPU j for any $j \neq i$. Even for neighboring coordinates $x_s, x_u \in [\mathbf{x}]_i$, the coordinates $\varphi(x_s)$ and $\varphi(x_u)$ can point to different shards $j_1 \neq j_2 \neq i$. This is illustrated in Fig. 4(a). Keeping the entire moving image in memory limits the maximum problem size to $N \leq V$, where V is the memory per GPU, regardless of the number of hosts H. However, we want the maximum problem size to scale with H. Therefore, we propose a distributed `grid_sampler` that allows us to *correctly* interpolate the moving image with sharded images scattered across multiple hosts without performing an `allgather` operation on the moving image.

(a) Performance comparison across methods and resolutions.

Resolution	Method	AvgDice Score ↑	InvDice Score ↑	AvgHD90^{cum} (mm) ↓
1 mm	Baseline	0.579 ± 0.055	0.141 ± 0.142	1.587 ± 0.908
	Anatomix	0.796 ± 0.035	0.386 ± 0.138	0.468 ± 0.137
	CLAIRE	0.776 ± 0.044	0.344 ± 0.120	0.554 ± 0.150
	FireANTs	0.822 ± 0.032	0.435 ± 0.147	0.393 ± 0.126
	ITK-dreg	0.662 ± 0.055	0.199 ± 0.125	1.002 ± 0.277
	SynthMorph	0.801 ± 0.022	0.378 ± 0.133	0.455 ± 0.098
	TransMorph	0.851 ± 0.016	0.468 ± 0.161	0.310 ± 0.064
	UniGradICON (IO)	0.826 ± 0.022	0.391 ± 0.155	0.384 ± 0.095
	UniGradICON	0.815 ± 0.026	0.393 ± 0.156	0.419 ± 0.113
	VFA	0.851 ± 0.023	0.494 ± 0.169	0.323 ± 0.096
	Ours	0.838 ± 0.028	0.436 ± 0.148	0.341 ± 0.109
500 µm	Baseline	0.580 ± 0.055	0.138 ± 0.143	1.357 ± 0.326
	Anatomix†	0.758 ± 0.040	0.325 ± 0.159	0.619 ± 0.169
	CLAIRE	0.779 ± 0.051	0.275 ± 0.210	0.570 ± 0.211
	FireANTs	0.841 ± 0.033	0.489 ± 0.163	0.340 ± 0.127
	ITK-dreg	0.699 ± 0.056	0.240 ± 0.130	0.834 ± 0.254
	SynthMorph†	0.771 ± 0.035	0.337 ± 0.133	0.557 ± 0.144
	TransMorph†	0.759 ± 0.028	0.300 ± 0.175	0.624 ± 0.127
	UniGradICON†	0.610 ± 0.044	0.133 ± 0.122	1.231 ± 0.262
	UniGradICON (IO)†	0.615 ± 0.047	0.149 ± 0.136	1.527 ± 1.495
	VFA†	0.805 ± 0.044	0.419 ± 0.181	0.462 ± 0.163
	Ours	0.872 ± 0.028	0.528 ± 0.180	0.258 ± 0.099
250 µm	Baseline	0.580 ± 0.055	0.136 ± 0.141	1.409 ± 0.322
	Anatomix†	0.620 ± 0.031	0.161 ± 0.115	1.179 ± 0.190
	CLAIRE	0.809 ± 0.054	0.378 ± 0.133	0.570 ± 0.211
	FireANTs†	0.777 ± 0.064	0.341 ± 0.199	0.629 ± 0.295
	ITK-dreg	0.758 ± 0.048	0.299 ± 0.125	0.613 ± 0.191
	SynthMorph†	0.690 ± 0.052	0.243 ± 0.164	0.882 ± 0.239
	TransMorph†	0.689 ± 0.044	0.191 ± 0.132	0.973 ± 0.245
	UniGradICON (IO)†	0.398 ± 0.062	0.063 ± 0.071	3.491 ± 3.198
	UniGradICON†	0.359 ± 0.044	0.045 ± 0.056	2.992 ± 0.670
	VFA†	0.714 ± 0.066	0.281 ± 0.216	0.821 ± 0.300
	Ours	0.895 ± 0.029	0.597 ± 0.204	0.216 ± 0.098



(b) Accuracy vs. GPU Compute Cost.



(c) Accuracy vs. Wall-clock Time.

Figure 5: Registration performance on Faux-OASIS dataset at 1 mm, 500 µm, and 250 µm (native 250 µm); mean ± std over pairs. ↑ higher is better; ↓ lower is better. HD90 values are reported using our cumulative definition (see Sec. K.2). (Green)/ (Yellow) = best/second; †= patch-based

Our approach leverages the key observation that (bi/tri)linear interpolation can be decomposed as an aggregate of partial sums of interpolated values on individual image shards. Fig. 4(b) illustrates this example. These individual image shards are sent across hosts in a ring topology, similar to Liu et al. (2024b), and the partial sum is aggregated inplace. This operation only incurs an additional N/H HBM overhead for fetching the sharded image from other hosts, scaling efficiently to arbitrary large problem sizes for sufficiently large H. The detailed derivation and correctness of this operation is shown in Section I.

4.3 Distributed Loss Functions

Since the moved image and fixed image are sharded cross H hosts, the loss function must take this into account to compute the loss function correctly.

Mean Squared Error (MSE). Since MSE is a per-pixel loss, we compute the individual MSE on host h and perform an `allreduce` operation.

Localized Normalized Cross Correlation (LNCC). The LNCC computes per-pixel patch similarities for each pixel, using a convolution over its neighbors. For sharded images, the patch statistics at the boundary requires a boundary synchronization with its neighboring shards which is provided by our GP implementation. After computing the LNCC for all pixels in each shard, we perform another `allreduce` to compute the LNCC over the entire image.

Mutual Information (MI). The MI loss computes the joint histograms $p_{(I,J)}(x, y)$ and marginals $p_I(x), p_J(y)$. However, these distributions are partial aggregates from the sharded images on each GPU. Eq. (2) can be rewritten as $p_I(v) = \sum_h \frac{N_h}{N} \left(\frac{1}{N_h} \sum_{k \in \Omega_h} \kappa(v - I_k) \right), p_{IJ}(v, w) = \sum_h \frac{N_h}{N} \left(\frac{1}{N_h} \sum_{k \in \Omega_h} \kappa(v - I_k) \kappa(w - J_k) \right)$, where the red terms correspond to the per-host histogram computation. Performing an `allreduce` to compute the weighted average of these histograms (with weights N_h/N) results in a valid and correct joint and marginal distributions over all hosts.

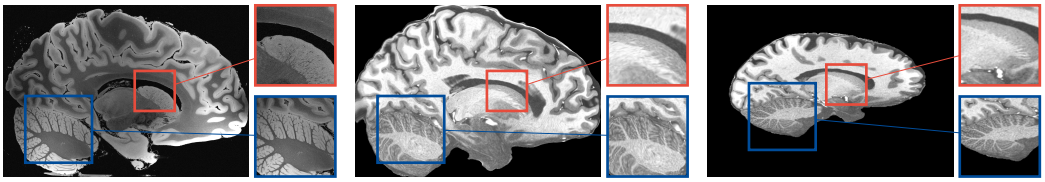


Figure 6: Qualitative comparison on registration of 100μm ex-vivo brain MRI (T1 $\rightarrow$ FLASH) image. Fine details like cerebellar white matter are not visible at macroscopic scales, but are aligned at 100μm. Fixed image is of size $1760 \times 1760 \times 1278$. Best viewed zoomed in. More results in Fig. 11.

This also leads to only a $B^2 + 2B$ communication overhead regardless of N, making a distributed implementation highly practical.

5 Experiments

Our primary goals are to (a) accelerate both optimization and neural network based registration workflows, and (b) solve significantly larger image registration problems. We show the efficacy of our method by accelerating existing registration workflows on standard clinical data. This is followed by optimizing a multimodal registration task with more than 11.8B optimizable parameters, an unprecedented result in large-scale registration. We compare the performance and computational efficiency of our method with various state-of-the-art baselines on a simulated 250μm ex-vivo brain MRI dataset, followed by ablations on various components of our framework.

Baselines. To accelerate existing registration workflows, we compare against TransMorph (Chen et al., 2022) and FireANTs (Jena et al., 2024a), which are state-of-the-art deep learning and optimization based registration frameworks respectively. In addition, we perform comparative evaluation with two methods explicitly designed for large-scale registration: ITK-DReg (itk) (CPU-based) and CLAIRE (Mang et al., 2019) (multi-GPU), and several SOTA learning-based approaches for clinical data - SynthMorph (Hoffmann et al., 2021), Vector-Field Attention (Liu et al., 2024c), unigradICON (Tian et al., 2024) (with/without instance optimization), anatomix+ConvexAdam (Dey et al., 2025).

5.1 Accelerating existing registration workflows and ablations

For deep networks, we train `TransMorph-large` under three loss configurations: (a) LNCC+Dice, (b) MI+Dice, and (c) LNCC+scaling-and-squaring (Ashburner, 2007) +Dice. For each configuration shown in Table 1, we either use the vanilla PyTorch implementation (Baseline) or our kernels (Ours). For classical optimization, we benchmark runtime and memory against multiple LNCC backends (FireANTs, VoxelMorph/TransMorph, Fast LNCC, `torch.compile`, and Ours) and MI backends (PyTorch and Ours with and without `torch.compile`). Tables 1 and 4 and Fig. 12 show that during network training our kernels converge 6.1$\times$ faster with LNCC while using 16.5% less memory, and reduce MI memory usage by 24.7%. Despite being designed for very large images, the runtime and memory benefits are significant for clinical-scale data (i.e., 30MB for OASIS). Optimization frameworks see larger gains: FireANTs achieves up to 95.2% memory savings and 2.6$\times$ speedup with MI, and a 7.5$\times$ speedup over FastLNCC (Jia et al., 2025) (and 2.9$\times$ over FireANTs' LNCC backend which applies separable convolutions on FastLNCC), with 44-59% lower memory usage overall.

5.2 Registration to a 100 micron ex-vivo brain MRI volume

To showcase the efficacy of our method on real large scale images, we register a 250μm in-vivo MRI image (Lüsebrink et al., 2017) to a 100 μm *ex-vivo* FLASH human brain volume (Edlow et al., 2019). This represents an inverse problem with more than 11.2B optimizable parameters (compared to $\sim$20M for clinical datasets), or 44.8GB of GPU memory. The entire problem does not fit on most GPUs, necessitating distributed multimodal registration. We optimize a composite transform - affine followed by a diffeomorphic mapping; details can be found in Section E.1. Multimodal deformable registration took $\sim$58 seconds on 8 NVIDIA A6000 GPUs, which is unprecedented at this resolution. Fig. 6 shows qualitative results, highlighting the ability to register highly detailed

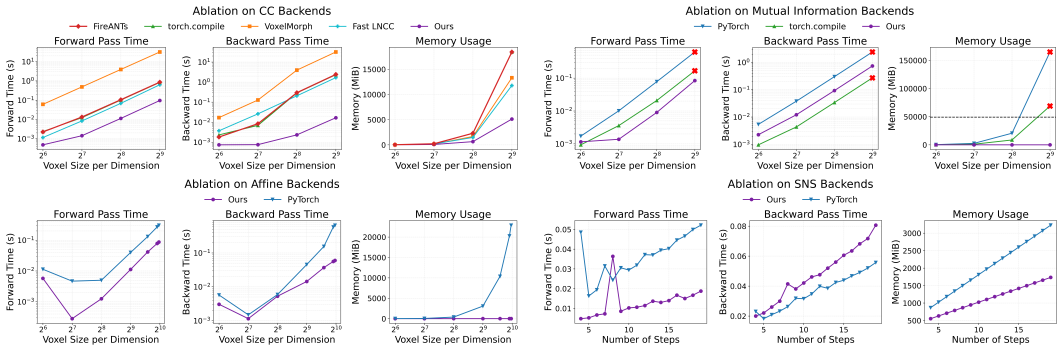


Figure 7: Ablations on key workhorse operations: LNCC, MI, `grid_sampler`, and scaling-and-squaring operations. Our fused kernels consume significantly less HBM and runtime.

structures such as cerebellar white matter; these structures are not visible at macroscopic scales. The resultant advantages of performing registration at this scale can allow researchers to characterize the neuroanatomy at microscopic resolutions and allow morphometric analysis of cortical layers and subcortical nuclei among other structures.

Registration accuracy in these studies is measured using privately annotated fiducial markers, hindering reproducibility and comparability of methodological advances. Due to lack of scalable frameworks, most high-resolution studies simply run ANTs at a significantly downsampled resolution (Kleven et al., 2023; Mansour et al., 2025; Wang et al., 2020b; Kronman et al., 2024; Bogovic et al., 2020; Edlow et al., 2019) and upsample the warp field to the native resolution.

5.3 Comparative Analysis on a Simulated ex-vivo Brain MRI Dataset

The faux-OASIS dataset To compare registration performance at high resolutions and leverage existing methods as baselines, we synthesize the *faux-OASIS* dataset, which mimics the anatomical distribution of an MRI dataset at $250\mu m$ isotropic resolution (more details in Section K). At $250\mu m$, the deformation field has 1.32B degrees of freedom per image pair, compared to $\sim$20M for OASIS.

Table 1: Accelerating TransMorph (**Top**) and FireANTs (**Bottom**) training with various computation backends.

Variant	Loss	Diffeomorphic	Training Time (h)	GPU Mem (GB)	Val DSC
Baseline	LNCC	✗	171.20	20.01	86.74
Ours	LNCC	✗	27.84	16.95	87.23
Baseline	LNCC	✓	171.42	21.28	86.55
Ours	LNCC	✓	27.93	17.34	87.09
Baseline	MI	✗	26.09	22.34	86.74
Ours	MI	✗	24.94	16.80	86.80

Loss	Backend	Dice Score ↑	Runtime (s) ↓	Memory (MB) ↓
LNCC	FireANTs	78.81 ± 3.87	1.44 ± 0.08	1044.5 ± 0.0
LNCC	FastLNCC	76.96 ± 3.60	3.76 ± 0.16	1026.3 ± 0.0
LNCC	VXM/TM	76.96 ± 3.60	57.08 ± 2.45	1418.5 ± 0.0
LNCC	`torch.compile`	69.35 ± 4.09	0.82 ± 0.04	860.7 ± 0.0
LNCC	Ours	78.67 ± 3.04	0.50 ± 0.01	577.5 ± 0.0
MI	PyTorch	75.88 ± 3.45	7.51 ± 0.37	12206.3 ± 0.0
MI	`torch.compile`	75.88 ± 3.45	1.05 ± 0.05	3865.5 ± 0.0
MI	Ours	75.88 ± 3.44	2.90 ± 0.16	577.5 ± 0.0
MI	`torch.compile`+Ours	75.93 ± 3.47	2.95 ± 0.16	657.3 ± 0.0

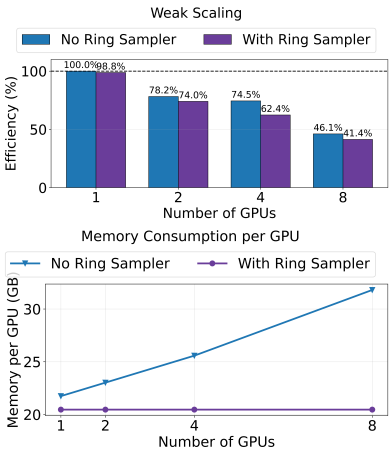
Baselines and evaluation. All methods (including CLAIRE and FireANTs without FFDP) run out of memory at $250\mu m$ resolution. We proposed two modifications to deep learning based methods to enable them to work on this dataset: (a) inspired by several high-resolution studies (Wang et al., 2020b; Mansour et al., 2025; Edlow et al., 2019), we register the images at a downsampled resolution, and then upsample the deformation field (b) inspired by several histology registration methods (Wodzinski et al., 2024; Lotz et al., 2015; Liang et al., 2021), we perform patchwise registration and mosaicing of the final deformation. We compare the methods at three resolutions: 1mm, $500\mu m$, and $250\mu m$. At 1mm, the full image fits within a patch, providing a baseline reference comparable to reported OASIS performance. At higher resolutions, patches are defined by each method's default input size with stride equal to 50% of the patch size. FireANTs augmented with FFDP is denoted as *Ours*. We report Dice, inverse-weighted Dice (InvDice; Mang et al. (2019)), and average Haussdorf distance capped at 90 percentile (AvgHD90). To compare efficiency, we measure both wall-clock time and GPU-hours.

Results. Fig. 5a summarizes performance metrics. At 1mm, most methods achieve performance consistent with their reported performance on OASIS, including VFA and TransMorph which were trained on the OASIS dataset with label supervision. At higher resolutions, nearly all methods degrade, especially for InvDice and HD90, which emphasize alignment of fine structures. In contrast,

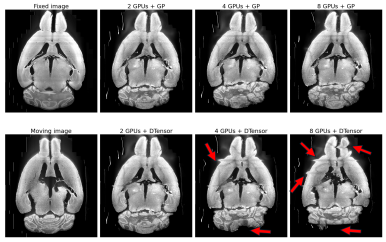
our method improves in accuracy: at $250\mu m$, we improve Dice by 18.1 points, InvDice by 31.6 points, and reduce AvgHD90 by 62.1%. The correlation between resolution and performance is also observed in (Mang et al., 2019; Mang & Ruthotto, 2017; Nazib et al., 2018); in addition we verify that patch-based methods *degrade* in performance at higher resolutions.

This degradation among patchwise methods is expected; histology-style pipelines typically register consecutive slides with small deformations after affine alignment. At high resolution, patching reduces anatomical context and the patches become progressively more out-of-distribution (see Fig. 19). Patchwise or downsampling strategies are therefore insufficient for ultra-high resolution large-scale registration, and existing deep methods cannot be repurposed to work at higher resolutions efficiently. Accuracy-efficiency tradeoffs in Figs. 5b and 5c show that our method is Pareto-efficient compared to all other methods (CPU, deep learning, and distributed GPU methods), requiring up to $500\times$ fewer GPU-hours compared to alternatives at $250\mu m$.

5.4 Ablation Studies



(a) Weak scaling and Per-GPU memory consumption of FFDP.



(b) Qualitative ablation of GP synchronization in FFDP on the fMOST mouse brain dataset (Tustison et al., 2024). Red arrows highlight regions affected by incorrect boundary effects due to no GP. See Fig. 10 for more examples.

Figure 8: Scaling and GP ablations.

We ablate on the efficiency of various workhorse operations used in image registration in Fig. 7 and Table 3. We compare our implementations to community-standard PyTorch implementation (Jia et al., 2025; Chen et al., 2022) and `torch.compile` versions. For grid sampler and MI kernels, our kernels have $O(1)$ extra HBM overhead instead of $O(N)$ in the PyTorch implementation. For LNCC, our implementation achieves an average speedup in the forward pass by $5.22\times$ and $56.98\times$ in the backward pass. Our `grid_sampler` also leads to an efficient scaling-and-squaring operation, commonly used in deep learning registration pipelines (Chen et al., 2022), with a memory reduction of 50% compared to the baseline implementation.

Scalability Analysis. We test the weak scaling of our distributed framework by registering synthetic images with increasing voxel sizes. For H GPUs, we instantiate an image pair of size $700 \times 700 \times 700H$ and shard the images, warp, and optimizer state across H GPUs. Fig. 8a shows weak scaling of FFDP with and without ring sampler. Without the ring sampler, the `grid_sample` operation requires storing the moving image of size $700 \times 700 \times 700H$ on each GPU, leading to peak HBM memory increasing linearly with H. This implies the framework would not scale to arbitrarily large problem sizes, regardless of cluster size H. Peak Memory consumption is independent of H with the Ring Sampler, and scaling efficiency is only minimally affected.

Ablation on GP. We ablate the effect of GP by replacing it with DTensor sharding (no boundary sync). Figs. 8b, 9 and 10 show that incorrect boundary synchronization leads to undesirable artifacts in the moved images, and reduces labelmap overlap.

6 Conclusion

We propose a novel distributed framework for arbitrarily large image registration problems. Our work identifies and proposes IO-aware and distributed-friendly implementations of workhorse operations in image registration algorithms, enabling registration of images at arbitrarily large resolutions on a single GPU. Our fused primitives demonstrate compelling results in both improving existing registration pipelines and scaling to arbitrarily large, multimodal problems pertinent in modern life science applications, that were previously infeasible without approximations. FFDP shows unprecedented registration capabilities that will enable researchers to leverage and effectively work with large-scale image volumes and unearth new insights leveraging the large resolution images.

REFERENCES

Allen brain atlas. URL https://atlas.brain-map.org/.

Itk-dreg: A framework for distributed, large-scale image registration. URL https://itk-dreg.readthedocs.io/en/latest/.

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

Jesper LR Andersson, Mark Jenkinson, Stephen Smith, et al. Non-linear registration, aka spatial normalisation fmrib technical report tr07ja2. *FMRIB Analysis Group of the University of Oxford*, 2 (1):1–22, 2007.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pp. 929–947, 2024.

ANTsX. Antsx: Advanced normalization tools (ants). URL https://github.com/ANTsX/ANTs. GitHub repository.

John Ashburner. A fast diffeomorphic image registration algorithm. *Neuroimage*, 38(1):95–113, 2007.

John Ashburner. Spm: a history. *Neuroimage*, 62(2):791–800, 2012.

B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee. Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain. *Medical Image Analysis*, 12(1):26–41, February 2008a. ISSN 1361-8423. doi: 10.1016/j.media.2007.06.004.

B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee. Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Medical Image Analysis*, 12(1):26–41, February 2008b. ISSN 1361-8415. doi: 10.1016/j.media.2007.06.004. URL https://www.sciencedirect.com/science/article/pii/S1361841507000606.

Brian B. Avants, P. Thomas Schoenemann, and James C. Gee. Lagrangian frame diffeomorphic image registration: Morphometric comparison of human and chimpanzee cortex. *Medical Image Analysis*, 10(3):397–412, June 2006. ISSN 13618415. doi: 10.1016/j.media.2005.03.005. URL https://linkinghub.elsevier.com/retrieve/pii/S1361841505000411.

Brian B Avants, Nick Tustison, Gang Song, et al. Advanced normalization tools (ants). *Insight j*, 2 (365):1–35, 2009.

Ramsey D Badawi, Hongcheng Shi, Pengcheng Hu, Shuguang Chen, Tianyi Xu, Patricia M Price, Yu Ding, Benjamin A Spencer, Lorenzo Nardo, Weiping Liu, et al. First human imaging studies with the explorer total-body pet scanner. *Journal of Nuclear Medicine*, 60(3):299–303, 2019.

Guha Balakrishnan, Amy Zhao, Mert R. Sabuncu, John Guttag, and Adrian V. Dalca. VoxelMorph: A Learning Framework for Deformable Medical Image Registration. *IEEE Transactions on Medical Imaging*, 38(8):1788–1800, August 2019. ISSN 0278-0062, 1558-254X. doi: 10.1109/TMI.2019.2897538. URL http://arxiv.org/abs/1809.05231. arXiv:1809.05231 [cs].

P Balchandani and TP Naidich. Ultra-high-field mr neuroimaging. *American Journal of Neuroradiology*, 36(7):1204–1215, 2015.

Erin S Beck, Pascal Sati, Varun Sethi, Tobias Kober, Blake Dewey, Pavan Bhargava, Govind Nair, Irene C Cortese, and Daniel Salo Reich. Improved visualization of cortical lesions in multiple sclerosis using 7t mp2rage. *American Journal of Neuroradiology*, 39(3):459–466, 2018.

M Faisal Beg, Michael I Miller, Alain Trouvé, and Laurent Younes. Computing large deformation metric mappings via geodesic flows of diffeomorphisms. *International journal of computer vision*, 61:139–157, 2005.

Ganesh Bikshandi and Jay Shah. A case study in cuda kernel fusion: Implementing flashattention-2 on nvidia hopper architecture using the cutlass library. *arXiv preprint arXiv:2312.11918*, 2023.

Benjamin Billot, Douglas N Greve, Oula Puonti, Axel Thielscher, Koen Van Leemput, Bruce Fischl, Adrian V Dalca, Juan Eugenio Iglesias, et al. Synthseg: Segmentation of brain mri scans of any contrast and resolution without retraining. *Medical image analysis*, 86:102789, 2023.

John A Bogovic, Hideo Otsuna, Larissa Heinrich, Masayoshi Ito, Jennifer Jeter, Geoffrey Meissner, Aljoscha Nern, Jennifer Colonell, Oz Malkesman, Kei Ito, et al. An unbiased template of the drosophila brain and ventral nerve cord. *Plos one*, 15(12):e0236495, 2020.

Katy Börner, Andreas Bueckle, Bruce W Herr, Leonard E Cross, Ellen M Quardokus, Elizabeth G Record, Yingnan Ju, Jonathan C Silverstein, Kristen M Browne, Sanjay Jain, et al. Tissue registration and exploration user interfaces in support of a human reference atlas. *Communications Biology*, 5(1):1369, 2022.

Bella E Brezovec, Andrew B Berger, Yukun A Hao, Feng Chen, Shaul Druckmann, and Thomas R Clandinin. Mapping the neural dynamics of locomotion across the drosophila brain. *Current Biology*, 34(4):710–726, 2024.

Xiaohuan Cao, Jianhua Yang, Jun Zhang, Dong Nie, Minjeong Kim, Qian Wang, and Dinggang Shen. Deformable image registration based on similarity-steered cnn regression. In *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pp. 300–308. Springer, 2017.

Adrià Casamitjana, Matteo Mancini, Eleanor Robinson, Loïc Peter, Roberto Annunziata, Juri Althonayan, Shauna Crampsie, Emily Blackburn, Benjamin Billot, Alessia Atzeni, et al. A probabilistic histological atlas of the human brain for mri segmentation. *Nature*, pp. 1–8, 2025.

Junyu Chen, Eric C. Frey, Yufan He, William P. Segars, Ye Li, and Yong Du. TransMorph: Transformer for unsupervised medical image registration. *Medical Image Analysis*, 82:102615, November 2022. ISSN 13618415. doi: 10.1016/j.media.2022.102615. URL http://arxiv.org/abs/2111.10480. arXiv:2111.10480 [cs, eess].

Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. TVM: An automated End-to-End optimizing compiler for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pp. 578–594, Carlsbad, CA, October 2018. USENIX Association. ISBN 978-1-939133-08-3. URL https://www.usenix.org/conference/osdi18/presentation/chen.

Gary E Christensen and Hans J Johnson. Consistent image registration. *IEEE transactions on medical imaging*, 20(7):568–582, 2001.

Gilberto Corso, Gabriel MF Ferreira, and Thomas M Lewinsohn. Mutual information as a general measure of structure in interaction networks. *Entropy*, 22(5):528, 2020.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Sandhitsu R Das, Brian B Avants, Murray Grossman, and James C Gee. Registration based cortical thickness measurement. *Neuroimage*, 45(3):867–879, 2009.

Chris Davis and A Murat Maga. Image registration and template based annotation of great ape skulls. In *American Journal of Physical Anthropology*, volume 165, pp. 60–61. WILEY 111 RIVER ST, HOBOKEN 07030-5774, NJ USA, 2018.

Bob D De Vos, Floris F Berendsen, Max A Viergever, Hessam Sokooti, Marius Staring, and Ivana Išgum. A deep learning framework for unsupervised affine and deformable image registration. *Medical image analysis*, 52:128–143, 2019.

Neel Dey, Benjamin Billot, Hallee E. Wong, Clinton Wang, Mengwei Ren, Ellen Grant, Adrian V Dalca, and Polina Golland. Learning general-purpose biomedical volume representations using randomized synthesis. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xOmC5LiVuN.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex attention: A programming model for generating optimized attention kernels. *arXiv preprint arXiv:2412.05496*, 2024.

Carmen Echávarri, P Aalten, Harry BM Uylings, HIL Jacobs, Pieter Jelle Visser, EHBM Gronenschild, FRJ Verhey, and S Burgmans. Atrophy in the parahippocampal gyrus as an early biomarker of alzheimer's disease. *Brain Structure and Function*, 215(3):265–271, 2011.

Brian L Edlow, Azma Mareyam, Andreas Horn, Jonathan R Polimeni, Thomas Witzel, M Dylan Tisdall, Jean C Augustinack, Jason P Stockmann, Bram R Diamond, Allison Stevens, et al. 7 tesla mri of the ex vivo human brain at 100 micron resolution. *Scientific data*, 6(1):244, 2019.

Andrea Esquivel, Andrea Ferrero, Achille Mileto, Francis Baffour, Kelly Horst, Prabhakar Shantha Rajiah, Akitoshi Inoue, Shuai Leng, Cynthia McCollough, and Joel G Fletcher. Photon-counting detector ct: key points radiologists should know. *Korean journal of radiology*, 23(9):854, 2022.

Fenja Falta, Christoph Großbröhmer, Alessa Hering, Alexander Bigalke, and Mattias P Heinrich. Lung250m-4b: A combined 3d dataset for CT- and point cloud-based intra-patient lung registration. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=FC0dsvguFi.

Miriam Friedel, Matthijs C van Eede, Jon Pipitone, M Mallar Chakravarty, and Jason P Lerch. Pydpiper: a flexible toolkit for constructing novel registration pipelines. *Frontiers in neuroinformatics*, 8:67, 2014.

Sarah F Frisken. Surfacenets for multi-label segmentations with preservation of sharp boundaries. *The Journal of computer graphics techniques*, 11(1):34, 2022.

Brett M Frye, Suzanne Craft, Thomas C Register, Jeongchul Kim, Christopher T Whitlow, Richard A Barcus, Samuel N Lockhart, Kiran Kumar Solingapuram Sai, and Carol A Shively. Early alzheimer's disease-like reductions in gray matter and cognitive function with aging in nonhuman primates. *Alzheimer's & Dementia: Translational Research & Clinical Interventions*, 8(1):e12284, 2022.

Davide Gambarotto, Fabian U Zwettler, Maeva Le Guennec, Marketa Schmidt-Cernohorska, Denis Fortun, Susanne Borgers, Jörn Heine, Jan-Gero Schloetel, Matthias Reuss, Michael Unser, et al. Imaging cellular ultrastructures using expansion microscopy (u-exm). *Nature methods*, 16(1): 71–74, 2019.

James C Gee, Martin Reivich, and Ruzena Bajcsy. Elastically deforming a three-dimensional atlas to match anatomical brain images. 1993.

Hui Gong, Dongli Xu, Jing Yuan, Xiangning Li, Congdi Guo, Jie Peng, Yuxin Li, Lindsay A Schwarz, Anan Li, Bihe Hu, et al. High-throughput dual-colour precision imaging for brain-wide connectome with cytoarchitectonic landmarks at the cellular level. *Nature communications*, 7(1):12142, 2016.

Maged Goubran, Cathie Crukley, Sandrine De Ribaupierre, Terence M Peters, and Ali R Khan. Image registration of ex-vivo mri to sparsely sectioned histology of hippocampal and neocortical temporal lobe specimens. *Neuroimage*, 83:770–781, 2013.

Courtney K Guo. *Multi-modal image registration with unsupervised deep learning*. PhD thesis, Massachusetts Institute of Technology, 2019.

Tripti Gupta, Gregory D Marquart, Eric J Horstick, Kathryn M Tabor, Sinisa Pajevic, and Harold A Burgess. Morphometric analysis and neuroanatomical mapping of the zebrafish brain. *Methods*, 150:49–62, 2018.

Alessa Hering, Keelin Murphy, and Bram van Ginneken. Learn2reg challenge: Ct lung registration - training data, 2020. URL https://doi.org/10.5281/zenodo.3835682.

Alessa Hering, Lasse Hansen, Tony CW Mok, Albert CS Chung, Hanna Siebert, Stephanie Häger, Annkristin Lange, Sven Kuckertz, Stefan Heldmann, Wei Shao, et al. Learn2reg: comprehensive multi-task medical image registration challenge, dataset and evaluation in the era of deep learning. *IEEE Transactions on Medical Imaging*, 42(3):697–712, 2022.

Malte Hoffmann, Benjamin Billot, Douglas N Greve, Juan Eugenio Iglesias, Bruce Fischl, and Adrian V Dalca. Synthmorph: learning contrast-invariant registration without acquired images. *IEEE transactions on medical imaging*, 41(3):543–558, 2021.

Junhao Hu, Weijie Gan, Zhixin Sun, Hongyu An, and Ulugbek S. Kamilov. A Plug-and-Play Image Registration Network, March 2024. URL http://arxiv.org/abs/2310.04297. arXiv:2310.04297 [eess].

Phillip Isola, Daniel Zoran, Dilip Krishnan, and Edward H Adelson. Crisp boundary detection using pointwise mutual information. In *European conference on computer vision*, pp. 799–814. Springer, 2014.

Sam Ade Jacobs, Masahiro Tanaka, Chengming Zhang, Minjia Zhang, Reza Yazdani Aminadabi, Shuaiwen Leon Song, Samyam Rajbhandari, and Yuxiong He. System optimizations for enabling training of extreme long sequence transformer models. In *Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing*, PODC '24, pp. 121–130, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706684. doi: 10.1145/3662158.3662806. URL https://doi.org/10.1145/3662158.3662806.

Rohit Jena, Pratik Chaudhari, and James C Gee. Fireants: Adaptive riemannian optimization for multi-scale diffeomorphic registration. *arXiv preprint arXiv:2404.01249*, 2024a.

Rohit Jena, Deeksha Sethi, Pratik Chaudhari, and James C. Gee. Deep learning in medical image registration: Magic or mirage? In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 108331–108353. Curran Associates, Inc., 2024b. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c3fe2a07ec47b89c50e89706d2e23358-Paper-Conference.pdf.

Rohit Jena, Pratik Chaudhari, and James C. Gee. Deep implicit optimization enables robust learnable features for deformable image registration. *Medical Image Analysis*, 103:103577, 2025. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2025.103577. URL https://www.sciencedirect.com/science/article/pii/S1361841525001240.

Xi Jia, Joseph Bartlett, Tianyang Zhang, Wenqi Lu, Zhaowen Qiu, and Jinming Duan. U-net vs transformer: Is u-net outdated in medical image registration? *arXiv preprint arXiv:2208.04939*, 2022.

Xi Jia et al. A naive trick to accelerate training of lncc-based deep image registration models. *Preprints*, February 2025. doi: 10.20944/preprints202502.2200.v1.

Bailiang Jian, Jiazhen Pan, Morteza Ghahremani, Daniel Rueckert, Christian Wachinger, and Benedikt Wiestler. Mamba? catch the hype or rethink what really helps for image registration. *arXiv preprint arXiv:2407.19274*, 2024.

Hanna Jonsson, Simon Ekstrom, Robin Strand, Mette A Pedersen, Daniel Molin, Hakan Ahlstrom, and Joel Kullberg. An image registration method for voxel-wise analysis of whole-body oncological pet-ct. *Scientific Reports*, 12(1):18768, 2022.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Ankita Joshi and Yi Hong. Diffeomorphic Image Registration using Lipschitz Continuous Residual Networks. pp. 13.

Justin W Kenney, Patrick E Steadman, Olivia Young, Meng Ting Shi, Maris Polanco, Saba Dubaishi, Kristopher Covert, Thomas Mueller, and Paul W Frankland. A 3d adult zebrafish brain atlas (azba) for the digital age. *Elife*, 10:e69988, 2021.

Stefan Klein, Marius Staring, Keelin Murphy, Max A Viergever, and Josien PW Pluim. Elastix: a toolbox for intensity-based medical image registration. *IEEE transactions on medical imaging*, 29 (1):196–205, 2009.

David Kleinfeld, Arjun Bharioke, Pablo Blinder, Davi D Bock, Kevin L Briggman, Dmitri B Chklovskii, Winfried Denk, Moritz Helmstaedter, John P Kaufhold, Wei-Chung Allen Lee, et al. Large-scale automated histology in the pursuit of connectomes. *Journal of Neuroscience*, 31(45): 16125–16138, 2011.

Heidi Kleven, Ingvild E Bjerke, Francisco Clascá, Henk J Groenewegen, Jan G Bjaalie, and Trygve B Leergaard. Waxholm space atlas of the rat brain: a 3d atlas supporting data analysis and integration. *Nature methods*, 20(11):1822–1829, 2023.

Julian Krebs, Tommaso Mansi, Hervé Delingette, Li Zhang, Florin C Ghesu, Shun Miao, Andreas K Maier, Nicholas Ayache, Rui Liao, and Ali Kamen. Robust non-rigid registration through agent-based action learning. In *Medical Image Computing and Computer Assisted Intervention-MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pp. 344–352. Springer, 2017.

Fae N Kronman, Josephine K Liwang, Rebecca Betty, Daniel J Vanselow, Yuan-Ting Wu, Nicholas J Tustison, Ashwin Bhandiwad, Steffy B Manjila, Jennifer A Minteer, Donghui Shin, et al. Developmental mouse brain common coordinate framework. *Nature communications*, 15(1):9072, 2024.

Kwame S. Kutten, Joshua T. Vogelstein, Nicolas Charon, Li Ye, Karl Deisseroth M.D., and Michael I. Miller. Deformably registering and annotating whole CLARITY brains to an atlas via masked LDDMM. In Peter Schelkens, Touradj Ebrahimi, Gabriel Cristóbal, Frédéric Truchetet, and Pasi Saarikko (eds.), *Optics, Photonics and Digital Technologies for Imaging Applications IV*, volume 9896, pp. 989616. International Society for Optics and Photonics, SPIE, 2016. doi: 10.1117/12.2227444. URL https://doi.org/10.1117/12.2227444.

Joel Lamy-Poirier. Breadth-first pipeline parallelism. *Proceedings of Machine Learning and Systems*, 5:48–67, 2023.

Leo Lebrat, Rodrigo Santa Cruz, Frederic de Gournay, Darren Fu, Pierrick Bourgeat, Jurgen Fripp, Clinton Fookes, and Olivier Salvado. CorticalFlow: A Diffeomorphic Mesh Transformer Network for Cortical Surface Reconstruction. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29491–29505. Curran Associates, Inc., 2021. URL https://papers.nips.cc/paper/2021/hash/f6b5f8c32c65fee991049a55dc97d1ce-Abstract.html.

Dacheng Li, Rulin Shao, Anze Xie, Eric P. Xing, Xuezhe Ma, Ion Stoica, Joseph E. Gonzalez, and Hao Zhang. DISTFLASHATTN: Distributed memory-efficient attention for long-context LLMs training. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=pUEDkZyPDl.

Shenggui Li, Fuzhao Xue, Chaitanya Baranwal, Yongbin Li, and Yang You. Sequence parallelism: Long sequence training from system perspective. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.134. URL https://aclanthology.org/2023.acl-long.134/.

Cher-Wei Liang, Ruey-Feng Chang, Pei-Wei Fang, and Chiao-Min Chen. Improving algorithm for the alignment of consecutive, whole-slide, immunohistochemical section images. *Journal of Pathology Informatics*, 12(1):29, 2021.

Jiayong Liang, Xiaoping Liu, Kangning Huang, Xia Li, Dagang Wang, and Xianwei Wang. Automatic registration of multisensor images using an integrated spatial and mutual information (smi) metric. *IEEE transactions on geoscience and remote sensing*, 52(1):603–615, 2013.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Hao Liu, Matei Zaharia, and Pieter Abbeel. Ringattention with blockwise transformers for near-infinite context. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=WsRHpHH4s0.

Yihao Liu, Junyu Chen, Lianrui Zuo, Aaron Carass, and Jerry L Prince. Vector field attention for deformable image registration. *Journal of Medical Imaging*, 11(6):064001–064001, 2024c.

Josephine K Liwang, Hannah C Bennett, Hyun-Jae Pi, and Yongsoo Kim. Protocol for using serial two-photon tomography to map cell types and cerebrovasculature at single-cell resolution in the whole adult mouse brain. *STAR protocols*, 4(1):102048, 2023.

Josephine K Liwang, Fae N Kronman, Hyun-Jae Pi, Yuan-Ting Wu, Daniel J Vanselow, Steffy B Manjila, Deniz Parmaksiz, Donghui Shin, Yoav Ben-Simon, Michael Taormina, et al. epdevatlas: mapping gabaergic cells and microglia in the early postnatal mouse brain. *Nature Communications*, 16(1):9538, 2025.

William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pp. 347–353. 1998.

Johannes Lotz, Janine Olesch, Benedikt Müller, Thomas Polzin, P Galuschka, JM Lotz, Stefan Heldmann, Hendrik Laue, Margarita González-Vallinas, Arne Warth, et al. Patch-based nonlinear image registration for gigapixel whole slide images. *IEEE Transactions on Biomedical Engineering*, 63(9):1812–1819, 2015.

Xin Luo, Zhigang Liu, Mingsheng Shang, Jungang Lou, and MengChu Zhou. Highly-accurate community detection via pointwise mutual information-incorporated symmetric non-negative matrix factorization. *IEEE Transactions on Network Science and Engineering*, 8(1):463–476, 2021. doi: 10.1109/TNSE.2020.3040407.

Falk Lüsebrink, Alessandro Sciarra, Hendrik Mattern, Renat Yakupov, and Oliver Speck. T1-weighted in vivo human whole brain mri dataset with an ultrahigh isotropic resolution of 250 μm. *Scientific data*, 4(1):1–12, 2017.

Mads AJ Madsen, Vanessa Wiggermann, Stephan Bramow, Jeppe Romme Christensen, Finn Sellebjerg, and Hartwig R Siebner. Imaging cortical multiple sclerosis lesions with ultra-high field mri. *NeuroImage: Clinical*, 32:102847, 2021.

Lucas Mahler, Julius Steiglechner, Benjamin Bender, Tobias Lindig, Dana Ramadan, Jonas Bause, Florian Birk, Rahel Heule, Edyta Charyasz, Michael Erb, et al. Ultracortex: Submillimeter ultra-high field 9.4 t brain mr image collection and manual cortical segmentations. *arXiv preprint arXiv:2406.18571*, 2024.

Andreas Mang and Lars Ruthotto. A lagrangian gauss–newton–krylov solver for mass-and intensity-preserving diffeomorphic image registration. *SIAM Journal on Scientific Computing*, 39(5): B860–B885, 2017.

Andreas Mang, Amir Gholami, Christos Davatzikos, and George Biros. CLAIRE: A distributed-memory solver for constrained large deformation diffeomorphic image registration. *SIAM Journal on Scientific Computing*, 41(5):C548–C584, January 2019. ISSN 1064-8275, 1095-7197. doi: 10.1137/18M1207818. URL http://arxiv.org/abs/1808.04487. arXiv:1808.04487 [cs, math].

Harrison Mansour, Ryan Azrak, James J Cook, Kathryn J Hornburg, Yi Qi, Yuqi Tian, Robert W Williams, Fang-Cheng Yeh, Leonard E White, and G Allan Johnson. The duke mouse brain atlas: Mri and light sheet microscopy stereotaxic atlas of the mouse brain. *Science Advances*, 11(18): eadq8089, 2025.

Daniel S Marcus, Tracy H Wang, Jamie Parker, John G Csernansky, John C Morris, and Randy L Buckner. Open access series of imaging studies (oasis): cross-sectional mri data in young, middle aged, nondemented, and demented older adults. *Journal of cognitive neuroscience*, 19(9): 1498–1507, 2007.

Gregory D Marquart, Kathryn M Tabor, Eric J Horstick, Mary Brown, Alexandra K Geoca, Nicholas F Polys, Damian Dalle Nogare, and Harold A Burgess. High-precision registration between zebrafish brain atlases using symmetric diffeomorphic normalization. *GigaScience*, 6(8):gix056, 2017.

David Mattes, David R Haynor, Hubert Vesselle, Thomas K Lewellyn, and William Eubank. Nonrigid multimodality image registration. In *Medical imaging 2001: image processing*, volume 4322, pp. 1609–1620. Spie, 2001.

Michael P Milham, Lei Ai, Bonhwang Koo, Ting Xu, Céline Amiez, Fabien Balezeau, Mark G Baxter, Erwin LA Blezer, Thomas Brochier, Aihua Chen, et al. An open resource for non-human primate imaging. *Neuron*, 100(1):61–74, 2018.

Tony C. W. Mok and Albert C. S. Chung. Large Deformation Diffeomorphic Image Registration with Laplacian Pyramid Networks, June 2020. URL http://arxiv.org/abs/2006.16148. arXiv:2006.16148 [cs, eess].

Tony CW Mok and Albert Chung. Affine medical image registration with coarse-to-fine vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20835–20844, 2022.

Keelin Murphy, Bram Van Ginneken, Joseph M Reinhardt, Sven Kabus, Kai Ding, Xiang Deng, Kunlin Cao, Kaifang Du, Gary E Christensen, Vincent Garcia, et al. Evaluation of registration methods on thoracic ct: the empire10 challenge. *IEEE transactions on medical imaging*, 30(11): 1901–1920, 2011.

Abdullah Nazib, James Galloway, Clinton Fookes, and Dimitri Perrin. Performance of registration tools on high-resolution 3d brain images. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 566–569, 2018. doi: 10.1109/EMBC.2018.8512403.

OpenAI. Triton. https://openai.com/index/triton/, 2021.

Hanchuan Peng, Phuong Chung, Fuhui Long, Lei Qu, Arnim Jenett, Andrew M Seeds, Eugene W Myers, and Julie H Simpson. Brainaligner: 3d registration atlases of drosophila brains. *Nature methods*, 8(6):493–498, 2011.

Zhen Peng, Minnan Luo, Wenbing Huang, Jundong Li, Qinghua Zheng, Fuchun Sun, and Junzhou Huang. Learning representations by graphical mutual information estimation and maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):722–737, 2023. doi: 10.1109/TPAMI.2022.3147886.

PyTorch. Fusing convolution and batch norm using custom function. https://docs.pytorch.org/tutorials/intermediate/custom_function_conv_bn_tutorial.html, 2023. Created July 22, 2021; Last updated April 18, 2023; Last verified November 5, 2024.

PyTorch. Helion. https://pytorch.org/blog/helion/, 2025.

Penghui Qi, Xinyi Wan, Guangxing Huang, and Min Lin. Zero bubble pipeline parallelism. *arXiv preprint arXiv:2401.10241*, 2023.

Huaqi Qiu, Chen Qin, Andreas Schuh, Kerstin Hammernik, and Daniel Rueckert. Learning diffeomorphic and modality-invariant registration using b-splines. 2021.

Quan Quan, Qingsong Yao, Heqin Zhu, and S Kevin Zhou. Igu-aug: Information-guided unsupervised augmentation and pixel-wise contrastive learning for medical image analysis. *IEEE Transactions on Medical Imaging*, 2024.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16, 2020. doi: 10.1109/SC41405.2020.00024.

Sadhana Ravikumar, Laura E. M. Wisse, Sydney Lim, Ranjit Ittyerah, Long Xie, Madigan L. Bedard, Sandhitsu R. Das, Edward B. Lee, M. Dylan Tisdall, Karthik Prabhakaran, Jacqueline Lane, John A. Detre, Gabor Mizsei, John Q. Trojanowski, John L. Robinson, Theresa Schuck, Murray Grossman, Emilio Artacho-Pérula, Maria Mercedes Iñiguez de Onzoño Martin, María del Mar Arroyo Jiménez, Monica Muñoz, Francisco Javier Molina Romero, Maria del Pilar Marcos Rabal, Sandra Cebada Sánchez, José Carlos Delgado González, Carlos de la Rosa Prieto, Marta Córcoles Parada, David J. Irwin, David A. Wolk, Ricardo Insausti, and Paul A. Yushkevich. Ex vivo mri atlas of the human medial temporal lobe: characterizing neurodegeneration due to tau pathology. *Acta Neuropathologica Communications*, 9(1):173, 2021. ISSN 2051-5960. doi: 10.1186/s40478-021-01275-7. URL https://doi.org/10.1186/s40478-021-01275-7.

Sadhana Ravikumar, Amanda E Denning, Sydney Lim, Eunice Chung, Niyousha Sadeghpour, Ranjit Ittyerah, Laura EM Wisse, Sandhitsu R Das, Long Xie, John L Robinson, et al. Postmortem imaging reveals patterns of medial temporal lobe vulnerability to tau pathology in alzheimer's disease. *Nature Communications*, 15(1):4803, 2024.

Marc-Michel Rohé, Manasi Datar, Tobias Heimann, Maxime Sermesant, and Xavier Pennec. Svf-net: learning deformable image registration using shape matching. In *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pp. 266–274. Springer, 2017.

Suman Sarkar and Biswajit Pandey. A study on the statistical significance of mutual information between morphology of a galaxy and its large-scale environment. *Monthly Notices of the Royal Astronomical Society*, 497(4):4077–4090, 2020.

Will Schroeder and Spiros Tsalikis. Really fast isocontouring. https://www.kitware.com/really-fast-isocontouring/, June 13 2023. Kitware blog / announcement.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *Advances in Neural Information Processing Systems*, 37:68658–68685, 2024.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Henrik Skibbe, Muhammad Febrian Rachmadi, Ken Nakae, Carlos Enrique Gutierrez, Junichi Hata, Hiromichi Tsukada, Charissa Poon, Matthias Schlachter, Kenji Doya, Piotr Majka, et al. The brain/minds marmoset connectivity resource: An open-access platform for cellular-level tracing and tractography in the primate brain. *PLoS biology*, 21(6):e3002158, 2023.

Hessam Sokooti, Bob De Vos, Floris Berendsen, Boudewijn PF Lelieveldt, Ivana Išgum, and Marius Staring. Nonrigid image registration using multi-scale 3d convolutional neural networks. In *Medical Image Computing and Computer Assisted Intervention- MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pp. 232–239. Springer, 2017.

Benjamin Frederick Spector, Simran Arora, Aaryan Singhal, Arjun Parthasarathy, Daniel Y Fu, and Christopher Re. Thunderkittens: Simple, fast, and $\textit{Adorable}$ kernels. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=0fJfVOSUra.

Bane Sullivan and Alexander Kaszynski. PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). *Journal of Open Source Software*, 4(37):1450, May 2019. doi: 10.21105/joss.01450. URL https://doi.org/10.21105/joss.01450.

Kathryn M Tabor, Gregory D Marquart, Christopher Hurt, Trevor S Smith, Alexandra K Geoca, Ashwin A Bhandiwad, Abhignya Subedi, Jennifer L Sinclair, Hannah M Rose, Nicholas F Polys, et al. Brain-wide cellular resolution imaging of cre transgenic zebrafish lines for functional circuit-mapping. *Elife*, 8:e42687, 2019.

Nouamane Tazi, Ferdinand Mom, Haojun Zhao, Phuc Nguyen, Mohamed Mekkouri, Leandro Werra, and Thomas Wolf. The ultra-scale playbook: Training LLMs on GPU clusters, 2024. URL https://huggingface.co/blog/the-ultra-scale-playbook. HuggingFace Blog.

Philippe Thévenaz and Michael Unser. Optimization of mutual information for multiresolution image registration. *IEEE transactions on image processing*, 9(12):2083–2099, 2000.

Lin Tian, Hastings Greer, Roland Kwitt, François-Xavier Vialard, Raúl San José Estépar, Sylvain Bouix, Richard Rushmore, and Marc Niethammer. unigradicon: A foundation model for medical image registration. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 749–760. Springer, 2024.

Lazaros C Triarhou. Dopamine and parkinson's disease. In *Madame curie bioscience database [internet]*. Landes Bioscience, 2013.

Nicholas J Tustison and Brian B Avants. Explicit b-spline regularization in diffeomorphic image registration. *Frontiers in neuroinformatics*, 7:39, 2013.

Nicholas J Tustison, Min Chen, Fae N Kronman, Jeffrey T Duda, Clare Gamlin, Mia G Tustison, Michael Kunst, Rachel Dalley, Staci Sorenson, Quanxin Wang, et al. The antsx ecosystem for mapping the mouse brain. *bioRxiv*, pp. 2024–05, 2024.

Erdem Varol, Amin Nejatbakhsh, Ruoxi Sun, Gonzalo Mena, Eviatar Yemini, Oliver Hobert, and Liam Paninski. Statistical atlas of c. elegans neurons. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part V 23*, pp. 119–129. Springer, 2020.

Vivek Venkatachalam, Ni Ji, Xian Wang, Christopher Clark, James Kameron Mitchell, Mason Klein, Christopher J Tabone, Jeremy Florman, Hongfei Ji, Joel Greenwood, et al. Pan-neuronal imaging in roaming caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 113(8): E1082–E1088, 2016.

Guoxia Wang, Jinle Zeng, Xiyuan Xiao, Siming Wu, Jiabin Yang, Lujing Zheng, Zeyu Chen, Jiang Bian, Dianhai Yu, and Haifeng Wang. Flashmask: Efficient and rich mask extension of flashattention. *arXiv preprint arXiv:2410.01359*, 2024.

Quanxin Wang, Song-Lin Ding, Yang Li, Josh Royall, David Feng, Phil Lesnar, Nile Graddis, Maitham Naeemi, Benjamin Facer, Anh Ho, Tim Dolbeare, Brandon Blanchard, Nick Dee, Wayne Wakeman, Karla E. Hirokawa, Aaron Szafer, Susan M. Sunkin, Seung Wook Oh, Amy Bernard, John W. Phillips, Michael Hawrylycz, Christof Koch, Hongkui Zeng, Julie A. Harris, and Lydia Ng. The Allen Mouse Brain Common Coordinate Framework: A 3D Reference Atlas. *Cell*, 181(4):936–953.e20, May 2020a. ISSN 00928674. doi: 10.1016/j.cell.2020.04.007. URL https://linkinghub.elsevier.com/retrieve/pii/S0092867420304025.

Quanxin Wang, Song-Lin Ding, Yang Li, Josh Royall, David Feng, Phil Lesnar, Nile Graddis, Maitham Naeemi, Benjamin Facer, Anh Ho, et al. The allen mouse brain common coordinate framework: a 3d reference atlas. *Cell*, 181(4):936–953, 2020b.

Asmamaw T Wassie, Yongxin Zhao, and Edward S Boyden. Expansion microscopy: principles and uses in biological research. *Nature methods*, 16(1):33–41, 2019.

Thomas Welton, Septian Hartono, Yao-Chia Shih, Stefan T Schwarz, Yue Xing, Eng-King Tan, Dorothee P Auer, Noam Harel, and Ling-Ling Chan. Ultra-high-field 7t mri in parkinson's disease: ready for clinical use?—a narrative review. *Quantitative Imaging in Medicine and Surgery*, 13(11): 7607, 2023.

Marek Wodzinski, Niccolo Marini, Manfredo Atzori, and Henning Müller. Deeperhistreg: robust whole slide images registration framework. *arXiv preprint arXiv:2404.14434*, 2024.

Yifan Wu, Tom Z. Jiahao, Jiancong Wang, Paul A. Yushkevich, M. Ani Hsieh, and James C. Gee. NODEO: A Neural Ordinary Differential Equation Based Optimization Framework for Deformable Image Registration. *arXiv:2108.03443 [cs]*, February 2022. URL http://arxiv.org/abs/2108.03443. arXiv: 2108.03443.

Yifan Wu, Mengjin Dong, Rohit Jena, Chen Qin, and James C Gee. Neural ordinary differential equation based sequential image registration for dynamic characterization. *arXiv preprint arXiv:2404.02106*, 2024.

Xiao Yang, Roland Kwitt, Martin Styner, and Marc Niethammer. Quicksilver: Fast predictive image registration–a deep learning approach. *NeuroImage*, 158:378–396, 2017.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.

Liutong Zhang, Lei Zhou, Ruiyang Li, Xianyu Wang, Boxuan Han, and Hongen Liao. Cascaded feature warping network for unsupervised medical image registration. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 913–916. IEEE, 2021.

Shengyu Zhao, Yue Dong, Eric I-Chao Chang, and Yan Xu. Recursive cascaded networks for unsupervised medical image registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019a.

Shengyu Zhao, Tingfung Lau, Ji Luo, I Eric, Chao Chang, and Yan Xu. Unsupervised 3d end-to-end medical image registration with volume tweening network. *IEEE journal of biomedical and health informatics*, 24(5):1394–1404, 2019b.

Shuai Zhao, Yang Wang, Zheng Yang, and Deng Cai. Region mutual information loss for semantic segmentation. *Advances in Neural Information Processing Systems*, 32, 2019c.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

A Related Works

A.1 Memory Efficient and Large Scale Optimization

Recent advances in large scale transformer-based model training has amassed significant attention and efforts to alleviate key bottlenecks in both memory and compute efficiency. Activation memory forms a key bottleneck in many deep learning training pipelines, and recent advances propose fused operations (Dao et al., 2022; Dao, 2023; Shah et al., 2024; PyTorch, 2023; Bikshandi & Shah, 2023; Dong et al., 2024) to significantly reduce HBM usage without approximations. Other techniques propose sub-quadratic approximations to the quadratic complexity of the attention operation and propose highly efficient and IO-aware fused kernels (Yuan et al., 2025; Dong et al., 2024; Wang et al., 2024). However, as these models and their inputs get increasingly larger in size, they do not fit on a single GPU. Various distributed techniques like Tensor Parallel (Shoeybi et al., 2019), Sequence Parallel (Li et al., 2023; Jacobs et al., 2024; Li et al., 2024), pipeline parallel (Qi et al., 2023; Lamy-Poirier, 2023; Liu et al., 2024a), fully-sharded data parallel (FSDP2) (Ansel et al., 2024; Zhao et al., 2023; Rajbhandari et al., 2020) have been proposed that distribute (shard) the model and its inputs across multiple GPUs for transformer-like models. Another research area approaches the problem of scaling large models by building compilers and intermediate representations to enable writing optimized kernels at runtime OpenAI (2021); Ansel et al. (2024); Spector et al. (2025); Chen et al. (2018); Abadi et al. (2016). To our knowledge, most of these techniques are tailored to transformer-specific architectures and GEMM-like operations (self attention, feedforward, batchnorm, etc.) only, and a Tensor/Model Parallel variant for convolution-aware sharding is not available. However, other disciplines including biomedical and clinical imaging, life sciences, climate modeling, drug discovery, genomics, geosciences, robotics leverage other key components that do not fit in the transformer-specific framework, or are GEMM-like in nature. We focus on the compute and memory bottlenecks in the image registration problem, that is a key component in a variety of biomedical and life science applications.

A.2 Large Scale Registration in Life Sciences and Biomedical Imaging

A.2.1 *Ex-vivo* neuroimaging and histology for neuroanatomical and pathological studies.

A large body of neuroanatomical studies are performed in conjunction with ex-vivo and blockface imaging and histology to create detailed, multi-scale anatomical references by integrating structural, molecular, and cytoarchitectural information across imaging modalities (Casamitjana et al., 2025; Ravikumar et al., 2024). In-vivo MRI scans are typically limited by resolution due to constraints on scan time and motion artifacts associated with longer scan times. This makes in-vivo MRI scans unsuitable for studying the microstructural changes associated with neurodegenerative disease progression. Registration of fine anatomical details like cortical layers, axonal projections, or individual nuclei are useful to understand neuropathology, and such analyses are not possible at macroscopic clinical scales. Therefore, high-resolution ex-vivo scans and blockface imaging are used as a bridge between in-vivo and histology, with the latter used as a gold standard for ground-truth microscopic tissue characterization and pathology. Many complementary stains are used to visualize neuropathological features, including protein aggregates, neuronal loss, gliosis, and myelin integrity. Accurate registration of these structures is important to improve our understanding of morphological effects of pathology. For example, Alzheimer's Disease (AD) is characterized by cortical atrophy in the medial temporal lobes, particularly hippocampus, entorhinal cortex, and parahippocampal gyrus (Ravikumar et al., 2024; Echávarri et al., 2011). Accurate atrophy quantification of these structures can only be reliably performed at $\sim$0.5 mm or better resolution MRI or ex vivo imaging, necessitating high resolution registration. Parkinson's Disease (PD) is characterized by degeneration of DA neurons in the substantia nigra (Triarhou, 2013) and subthalamic nucleus that are small ($\sim$5-10 mm), requiring $<$0.7 mm isotropic or ex vivo imaging for volumetry or susceptibility mapping for accurate delineation (Welton et al., 2023). Multiple Schelosis is characterized by cortical lesions (Madsen et al., 2021; Beck et al., 2018) that cannot be delineated at the in-vivo resolution and typically requires high resolution ex-vivo imaging and histopathology integration. Except in-vivo imaging, all other modalities are very high resolution typically ranging from 500μm up to 100μm (Ravikumar et al., 2024; Echávarri et al., 2011; Welton et al., 2023; Madsen et al., 2021) for ex vivo imaging and $\sim$ 10μm for histology sections. High-resolution imaging and registration are essential in these contexts because they enable accurate cross-modal alignment and preservation of fine anatomical

detail that would otherwise be lost through downsampling. Most of these studies, however, limit their analyses to localized effects due to the significant computational cost of registering the entire brain at high resolution. Recently, projects like Allen Brain Atlas (all) and multiple large scale consortia including Seattle Alzheimer's Disease Brain Cell Atlas (SEA-AD) consortium and the Human Mouse Brain Atlas (HMBA) consortium are aimed at creating detailed, multimodal brain atlases linking cellular, molecular, and anatomical organization across species and disease states - combining individual efforts from multiple institutions together into a unified resource. Achieving this multimodal organization at the whole brain level requires high-resolution registration tools to accurately align diverse imaging modalities while preserving fine-scale cytoarchitectural detail.

Most in-vivo to histology registration workflows require registration of an in-vivo image to its ex-vivo counterpart. The $100\mu m$ ex-vivo and $250\mu m$ in-vivo images released in Edlow et al. (2019); Lüsebrink et al. (2017) are intended to be used as high-resolution templates to enable accurate studies, but lack of computationally efficient methods restricts their broad usage in the neuroimaging community. Our paper performs a native-scale registration on a modest GPU server with 8 A6000 GPUs to showcase the distributed capabilities, democratizing the use of such high resolution data for advancing the state of neuropathology studies.

A.2.2 High resolution pulmonary imaging enables subvoxel landmark localization.

Pulmonary CT mapping is a key component of lung disease diagnosis and treatment, and accurate landmark tracking requires registration at high resolution. Lung CT images can be acquired at submillimeter resolution (Murphy et al., 2011), but deep learning methods often require downsampling to accommodate their memory requirements (Falta et al., 2023; Hering et al., 2020). In the LungCT Learn2Reg challenge, the Lung CT images have a resolution of 1.25-1.75mm and the top performing methods achieve an average landmark error of 1.83mm. However, in the EMPIRE10 challenge, the average physical resolution of the images is 0.7mm and the average landmark error of most top methods (FireANTs, DISCO) is around 0.649mm, reaching subvoxel landmark localization. This demonstrates that with an appropriately high resolution, top methods can achieve subvoxel accuracy in landmark errors without learning. Moreover, due to the large voxel sizes in the EMPIRE10 dataset (with average voxel size of $412.8 \times 317.2 \times 364.9$, about $5\times$ larger than OASIS brain MRI on average), most top performing methods are iterative methods, sometimes used in conjunction with patch based feature extractors. This retrospective analysis shows the direct impact of using higher resolution to improve landmark accuracy in pulmonary imaging, and the benefits of using native-scale registration.

A.2.3 Large scale registration in model organisms.

Over the past decade, imaging across the life sciences and biomedical domains has progressed from mesoscale surveys to organ- and organism-wide acquisitions at cellular or even subcellular resolution. These span transparent organisms and small animal models (e.g., *C. elegans*, zebrafish, adult *Drosophila*) (Varol et al., 2020; Venkatachalam et al., 2016; Marquart et al., 2017; Gupta et al., 2018; Peng et al., 2011; Brezovec et al., 2024), whole-rodent brains imaged at micron or submicron sampling (Gong et al., 2016; Wang et al., 2020a), and non-human primate (NHP) and human ex vivo MRI at hundreds of microns (Skibbe et al., 2023; Milham et al., 2018; Edlow et al., 2019; Lüsebrink et al., 2017). Such modalities routinely generate giga- to teravoxel volumes (Kutten et al., 2016; Nazib et al., 2018). Their scientific utility, however, hinges on the ability to perform registration at the native resolution of acquisition, i.e. aligning specimens (or modalities) in a common coordinate system without sacrificing the fine-scale morphologies-cell bodies, layers, axon bundles, synaptic neighborhoods– that motivate high-resolution acquisition in the first place (Nazib et al., 2018; Goubran et al., 2013).

Cellular-resolution atlases in model organisms. In *C. elegans*, statistical atlases of neuron positions require aligning whole-animal volumes to preserve the fidelity of closely apposed cells (Varol et al., 2020; Venkatachalam et al., 2016). In zebrafish, deformable registration with cellular-level precision and minimal perturbation of tissue morphology enables pooling of gene expression, single-neuron morphologies, and brain-wide activity (Marquart et al., 2017; Gupta et al., 2018). In adult *Drosophila*, whole-brain registration underpins large-scale databases and enables structure–function integration (for example, aligning two-photon functional volumes to EM-derived connectomes) (Peng et al., 2011; Brezovec et al., 2024).

Whole-brain rodent imaging Large scale efforts like NIH's Brain Research through Advancing Innovative Neurotechnologies (BRAIN) Initiative - Cell Census Network (BICCN) aims to provide researchers and the public with a comprehensive reference of the diverse cell types in human, mouse, and non-human primate brain, and researchers collect a wide range of multimodal data including MRI, sectioning tomography, microscopy, antibody stains (e.g. calbindin), and spatial transcriptomics. In rodents, fMOST pipelines yield whole-brain images at micron sampling (e.g., $0.32\mu m$ voxels generating $>$10TB datasets) for tracing long-range axons and quantifying cytoarchitecture (Gong et al., 2016). Constructing stereotaxic spaces such as the Allen CCFv3 and Waxholm rat atlas requires deformable registration that preserves layers and boundaries (Wang et al., 2020a; Kleven et al., 2023; Kronman et al., 2024). Currently, there is a huge gap between the resolution at which data is acquired and the resolution at which templates are created. For example, STPT images can be collected at less than $1\mu m$ resolution (Liwang et al., 2023), but the Allen CCFv3 template is generated at $10\mu m$ by upsampling the registrations from $25\mu m$ due to compute constraints. Extrapolating the runtime reported in the method used to generate the CCFv3 template (Wang et al., 2020a), registration will require about 19 hours for a single pair or about 7.26 CPU-years for a single iteration of template matching - and is therefore impossible to curate without access to huge HPC clusters. This is contrasted to our method that can perform registration in about a minute or two on a modest server rack with 8 GPUs (or 5.48 GPU days for a single iteration of template building) - saving a significant amount of time and resources. Certain phenomena of interest like cellular organization and brain-wide connectomes are emergent only at very high resolutions, necessitating computational tools that can scale with the data.

The lack of computational tools for large-scale registration has a trickle-down effect on follow up studies as well. For instance, ANTsX pipelines for mouse brain registration to the CCFv3 atlas is performed at $50\mu m$ instead of $10\mu m$ for compute reasons (Tustison et al., 2024). Developmental atlases also register the CCFv3 at resolutions significantly downsampled from the original $10\mu m$ template (Kronman et al., 2024; Liwang et al., 2025) citing lack of computational resources as one of the primary reasons.

Zebrafish Initially adopted as a developmental biology model because of its ease of domestication, high fecundity, and transparent early life stages, the zebrafish has gained broader prominence with advances in brain imaging, molecular genetic tools, and behavioral assays (Kenney et al., 2021). For the AZBA template (Kenney et al., 2021), the raw images are collected at $4\mu m$ but was resampled to $8\mu m$ ($8\times$ downsampling) due to system constraints. The tools used for the registration (Friedel et al., 2014) do not recommend running locally and only on a distributed cluster. Brain-wide cellular resolution imaging of transgenic zebrafish lines (Tabor et al., 2019) is performed on large clusters like Biowulf Linux cluster at the National Institutes of Health, significantly reducing accessibility of these imaging resources to researchers, signifying an unmet need for efficient and distributed multimodal registration frameworks.

Across these diverse domains, the unifying requirement demands access to scalable multimodal registration algorithms - a challenge we address in this work.

A.3 Deformable Image Registration

Given two images $F : \Omega \to \mathbb{R}^d$ and $M : \Omega \to \mathbb{R}^d$ defined on domain Ω (usually a compact subset of $\mathbb{R}^d$), Deformable Image Registration (DIR) refers to an inverse problem to find a transformation $\varphi : \Omega \to \Omega$ that warps the moving image M to the fixed image F. Prior to deep learning, the inverse problem was solved using iterative solvers (Klein et al., 2009; Tustison & Avants, 2013; Andersson et al., 2007; Ashburner, 2012; Avants et al., 2006), and has been made significantly more scalable by recent advances in GPU-based libraries (Mang et al., 2019; Mang & Ruthotto, 2017; Jena et al., 2024a). Meanwhile, earliest deep learning for image registation (DLIR) methods (Cao et al., 2017; Krebs et al., 2017; Rohé et al., 2017; Sokooti et al., 2017) used supervised learning to predict a transformation field using pseudo ground truth transformations. However, since the inverse problem is generally ill-posed, unsupervised and weakly supervised learning methods (Balakrishnan et al., 2019; Zhao et al., 2019b;a; Joshi & Hong; De Vos et al., 2019; Mok & Chung, 2020; Zhang et al., 2021; Qiu et al., 2021; Lebrat et al., 2021; Jia et al., 2022; Mok & Chung, 2022) became dominant. However, these methods perform virtually identically to iterative solvers in the unsupervised setting, and show relatively brittle performance under domain shift(Jena et al., 2024b; Jian et al., 2024; Jena

et al., 2025). Other recent work has shown increased reliability under domain shift (Liu et al., 2024c; Chen et al., 2022). Another line of work combines neural priors with iterative solvers to leverage learnable features with strong convergence properties and robustness of solvers (Wu et al., 2024; Hu et al., 2024; Wu et al., 2022; Zhao et al., 2019a;b). However, almost all deep learning-based methods typically work reliably only at a macroscopic resolution, with most methods working only at a standard resolution of 1mm or $192 \times 160 \times 224$ voxels, and running out of memory on larger images, even on other macroscopic problems like lung or full body CT unless they are significantly downsampled (Falta et al., 2023). This is a significant limitation given modern real life applications including the ultra-high-resolution image acquisition techniques used for *ex-vivo* neuroanatomical and developmental biology studies, spanning subcellular structures and connectomes in species like C.elegans (Varol et al., 2020; Venkatachalam et al., 2016), zebrafish (Marquart et al., 2017; Gupta et al., 2018), adult Drosophila (Bogovic et al., 2020; Brezovec et al., 2024; Peng et al., 2011), rodents (Wang et al., 2020b; Mansour et al., 2025; Kleven et al., 2023; Kronman et al., 2024), and non-human primates (Skibbe et al., 2023; Davis & Maga, 2018; Frye et al., 2022). The scale of these problems is often two to three orders of magnitude larger than the scale of existing deep learning methods. A simple extrapolation shows that existing deep learning methods will require ≈ 1.87 TB of GPU memory to train a model on a $250\mu m$ ex-vivo brain dataset, making them impractical for training on larger problems. (Mang et al., 2019; Mang & Ruthotto, 2017) propose a distributed framework for registering arbitrarily large images, but is limited to MSE loss function and a one-parameter subgroup of diffeomorphic transforms (stationary velocity field), which is less flexible than the entire space of diffeomorphic transforms (Mang et al., 2019; Jena et al., 2024a). Moreover, they show results on upto 256 GPUs which indicates room for improvement in terms of scaling efficiency. In our work, we propose a distributed framework that is upto an order of magnitude more efficient than (Mang et al., 2019) on large problems.

B Limitations and Future Work

One of the limitations of the proposed framework is the relatively poor weak scaling of the method in the distributed setting (41% on 8 GPUs without NVLink or Infiniband). Even so, for most life science applications feasibility is the first step towards scalable, distributed, multimodal registration, and future work will focus on improving the weak scaling of the method. Another active avenue for future work is to enable Virtual GridParallel (VGP) to use fewer GPUs by sequentially offloading and onloading consecutive shards from CPU onto a single GPU. Deformable Registration of the 100 μm volume in Appendix 5.2 took only *one minute* on 8 A6000 GPUs, but equivalently it would take around 15-20 minutes to register this pair on a single A6000 GPU with VGP, accounting for repeated CPU offloading. This is an acceptable timeframe for large-scale studies, allowing researchers to prototype and iterate on large-scale image volumes as well with a single GPU. Other avenues for future work include collecting labeled data at high-resolution for various real-world life science applications and performing comparative studies on these datasets.

C LLM Usage

We use an LLM (minimally) to polish the manuscript and improve clarity of ideas and organization. All LLM-generated text is thoroughly reviewed, proofread, and revised by the first author of the paper.

D Correctness of Grid Parallel Implementation

The GridParallel framework aims to add additional synchronization primitives for performing mathematically correct convolutions across image or grid shards. These convolutions are required to compute the LNCC loss, and applying Sobolev preconditioning of the warp field and its gradient. Without GP synchronization, the implementation is equivalent to a DTensor sharding (Ansel et al., 2024). To our knowledge, existing Model Parallel and FSDP techniques are exclusively built for model weights and activations for linear and self.-attention layers and do not support this functionality. The pseudocode for convolution with GP synchronization is provided in Algorithm 1.

To ablate the effect of GridParallel synchronization, we register images at $500\mu m$ resolution from the faux-OASIS dataset with and without the GridParallel synchronization to measure the effect on

performance. Results in Fig. 9 show only a minimal drop in performance with DTensor sharding. We posit that this is because the faux OASIS dataset does not contain real-world noise and other artifacts that can degrade performance with incorrect boundary synchronization. To study the effect of GP synchronization on a more challenging dataset, we register images at $10\mu m$ resolution from the fluorescence micro-optical sectioning tomography (fMOST) mouse brain dataset (Tustison et al., 2024) with and without the GridParallel synchronization to measure the effect on performance. This dataset contains image volumes of size $1202 \times 1078 \times 627$ voxels, or a displacement field of 9.74GB. The data contains a myriad of complex artifacts, namely stripe artifacts, boundary halo effects, and speckle noise from image stitching and reconstruction. We run FireANTs with multi-scale optimization at scales 16, 8, 4, 2, $1\times$ downsampling for 200, 200, 200, 100, 50 iterations. We use our Fused LNCC implementation with a window size of 7, and a learning rate of 0.5. Smoothing regularizations are set to $\sigma_{\text{grad}} = 1.0$ and $\sigma_{\text{warp}} = 0.5$. We ablate on 2, 4 and 8 GPUs.

Since we do not have ground truth annotations for this dataset, we only make qualitative observations. Unlike the faux-OASIS dataset, the fMOST dataset is more challenging with high levels of image heterogeneity and complex anatomical structures. Fig. 10 shows that the performance without GP synchronization is significantly affected as a function of GPUs. Specifically, the boundaries introduce undesirable artifacts due to mathematically incorrect smoothing and LNCC losses computed across shard. GP synchronization produces qualitatively better results regardless of the number of GPUs used to shard the problem.

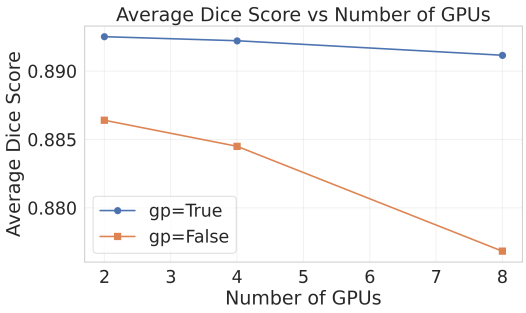


Figure 9: Quantitative ablation of GP synchronization on the faux-OASIS dataset.

Algorithm 1 Convolution with GP synchronization

Require: T (tensor), r (rank), k kernel size, W kernel filter, sharding index sh, GP size gp_size

pad $\leftarrow (k-1)/2$
bl $\leftarrow$ `None`
br $\leftarrow$ `None`
if $r > 0$ **then**
    bl $\leftarrow$ `get_boundary`$(r-1, \text{pad})$
end if
if $r <$ gp_size **then**
    br $\leftarrow$ `get_boundary`$(r+1, \text{pad})$
end if
$T_{\text{pad}} \leftarrow$ `concat`$([\text{bl}, T, \text{br}], \text{dim} = \text{sh})$
out $\leftarrow$ `conv`(T_{pad}, W)
crop_from_left $\leftarrow 0$
crop_from_right $\leftarrow 0$
if $r > 0$ **then**
    crop_from_left $\leftarrow$ pad
end if
if $r <$ gp_size **then**
    crop_from_right $\leftarrow$ pad
end if
out $\leftarrow$ `crop`$(\text{out}, (\text{crop\_from\_left}, \text{crop\_from\_right}), \text{dim} = \text{sh})$
return out

E Accelerating TransMorph

TRAINING

In this section, we plot the performance of TransMorph training with and without our fused operations. Table 1 summarizes the performance of TransMorph training with and without our fused operations

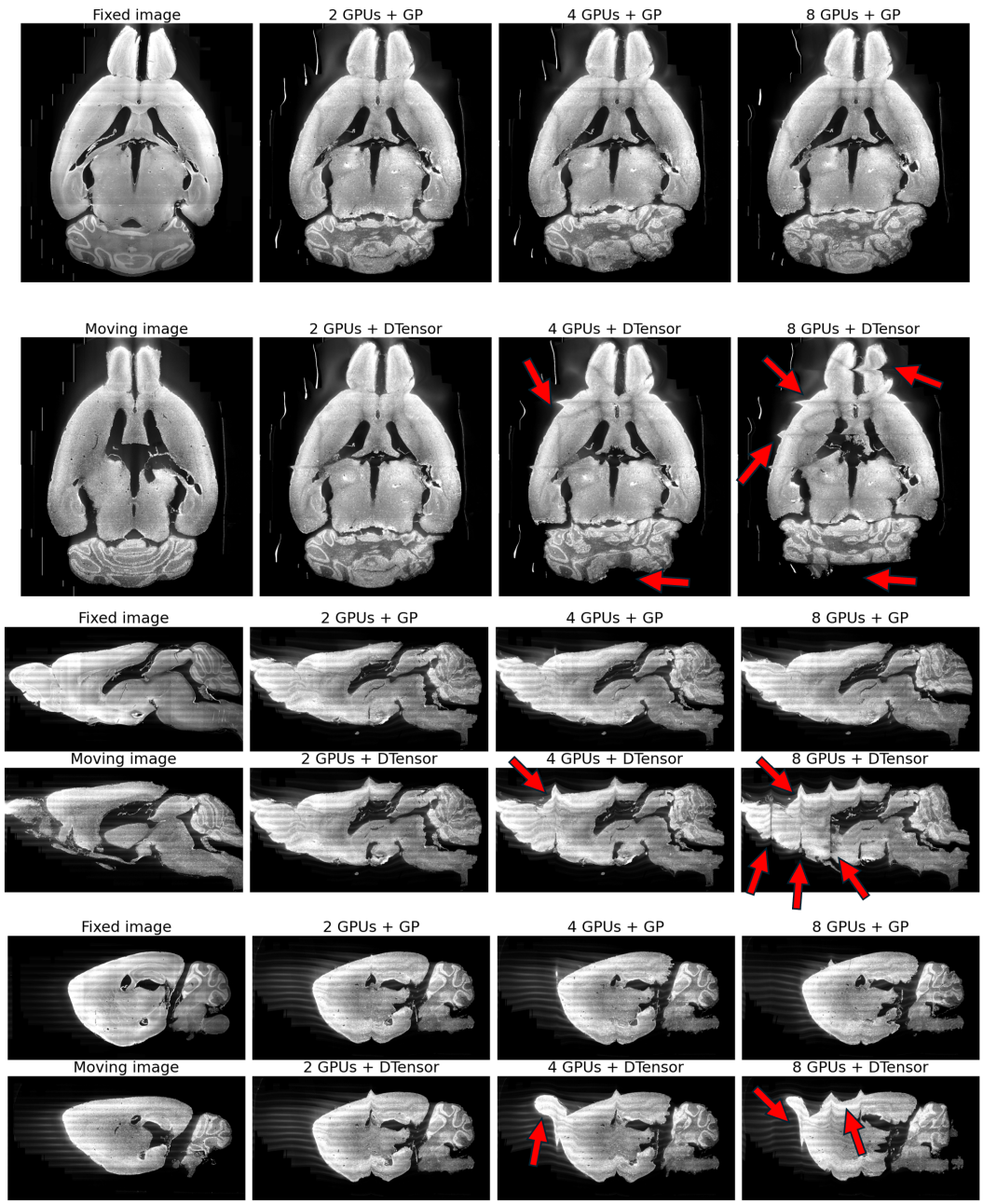


Figure 10: Qualitative ablation of GP synchronization in FFDP on the fMOST mouse brain dataset. Red arrows highlight regions affected by incorrect boundary effects due to no synchronization.

for three commonly used configurations. We further plot the validation performance across these settings with respect to Wall clock time in Fig. 12. Our fused operations demonstrate efficiency with fast convergence while reducing memory usage.

E.1 Registration to a 100 micron ex-vivo brain MRI volume

In this section, we describe the parameters used for the registration of a 250μm in-vivo T1-weighted MRI volume described in Lüsebrink et al. (2017) to the 100μm ex-vivo brain FLASH volume described in Edlow et al. (2019). First, we perform an multi-scale affine registration at 3mm, 2mm, 1mm, 500μm resolutions for 500, 250, 100, 100 iterations respectively, using the Fused Mutual

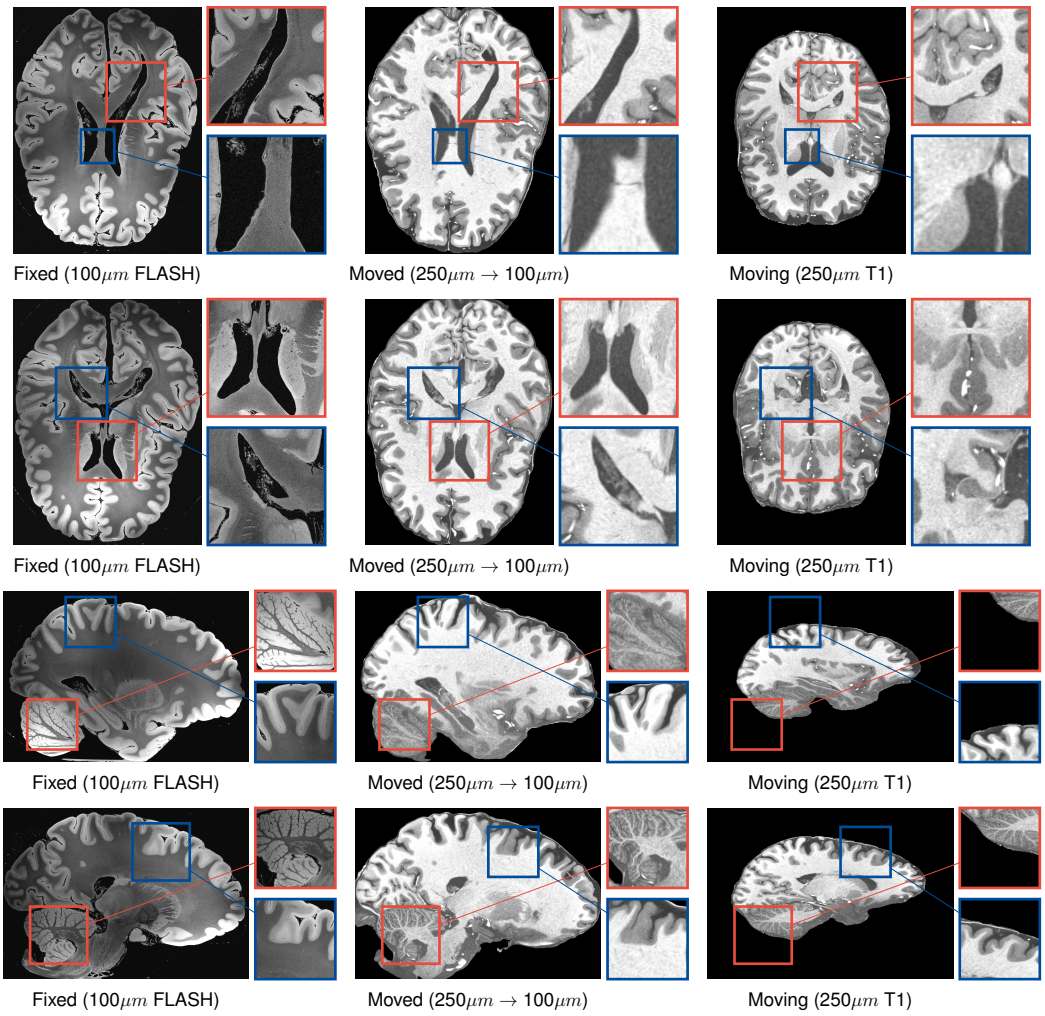


Figure 11: Qualitative comparison of registration results from 250μm T1 (Lüsebrink et al., 2017) to 100μm ex-vivo FLASH (Edlow et al., 2019). Intricate structures like cerebellar white matter and GM-WM interfaces are not very discernable at 1mm, but can be aligned at 100μm with our method.

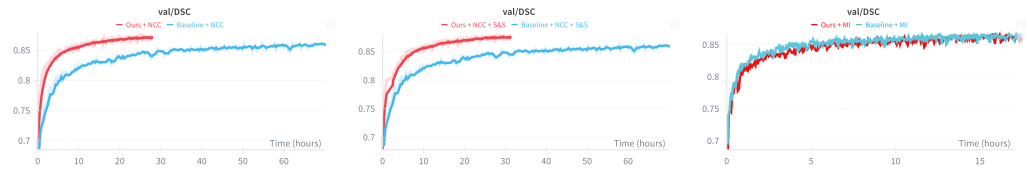


Figure 12: Ablation on TransMorph training runtime with and without our fused operations. For LNCC, our method converges in about 30 hours, while the baseline converges in about a week.

Information loss. This step takes about 12 seconds to run on a single NVIDIA A6000 GPU. The second step was to run multi-scale deformable optimization with scales 3.2mm, 1.6mm, 0.8mm, 0.4mm, 0.2mm, 0.1mm (scale factors of 32, 16,8, 4, 2, 1) for 250, 100, 100, 100, 50, 20 iterations respectively, using the fused LNCC loss. This step took about 58 seconds on 8 NVIDIA A6000 GPUs. Qualitative results are shown in Fig. 11.

Table 2: Qualitative Comparison of Methods. We compare the methods on qualitative features such as GPU support, multimodal capabilities, ability to run for unequal sizes of fixed and moving images, non-standard image sizes, whether the model can work with full context for larger images, whether the model supports multi-GPU training, and whether the model supports arbitrary loss functions. Deep learning methods support multimodal registration only if they are trained on multiple modalities. CLAIRE requires the image sizes to be divisible by the number of GPUs, and does not support arbitrary loss functions. Our method supports all of the above features, leading to a seamless experience for users with minimal data preprocessing overhead.

Method	GPU support	Multimodal	$N \neq M$	Non-std sizes	Full Context	Multi-GPU	Supported Similarty functions
Deep learning	✓	✓?	✗	✗	✗	✗	Fixed at training
ITK-DReg	✗	✓	✓	✓	✗	✗	ITK-filters
CLAIRE	✓	✗	✗	✓?	✓	✓	MSE only
Ours	✓	✓	✓	✓	✓	✓	Any

F An efficient Fused Local Normalized Cross Correlation loss

Local Normalized Cross Correlation (LNCC) loss is ubiquitously used throughout the image registration literature(Liu et al., 2024c; Avants et al., 2008b; 2009; ANTsX; Jena et al., 2024a; Wu et al., 2024; Hu et al., 2024; Wu et al., 2022; Zhao et al., 2019a;b), owing to its robust behavior to unimodal and multimodal images alike. This operation is a key memory-bound bottleneck in image registration pipelines. Few approaches have been proposed to provide improved implementations (Jia et al., 2025; Chen et al., 2022), but we note that these implementations are still memory intensive and thus not scalable. We address this bottleneck by analytically deriving a fused implementation that is memory efficient and scalable.

Definition of LNCC loss. Given two images F and M, and a radially symmetric averaging convolution filter W such that $\sum_k w_k = 1$, we define the Local Normalized Cross Correlation (LNCC) loss as:

$$\mathcal{L} = \frac{1}{N}\sum_i n_i \quad , \quad n_i = \frac{A_i^2}{B_i C_i + \epsilon} \tag{3}$$

where

$$\mu_i^F, \mu_i^M = \sum_k w_{ik} F_k, \sum_k w_{ik} M_k \tag{4}$$

$$A_i = \sum_k w_{ik}(F_k - \mu_i^F)(M_k - \mu_i^M) \tag{5}$$

$$B_i = \sum_k w_{ik}(F_k - \mu_i^F)^2 \tag{6}$$

$$C_i = \sum_k w_{ik}(M_k - \mu_i^M)^2 \tag{7}$$

Here, we use overloaded notation $w_{ik} = w_{(i-k)} = w_{(k-i)} = w_{ki}$ due to radial symmetry of w. We can expand Eqs. (5) to (7) as follows:

$$A_i = \left(\sum_k w_{ik} F_k M_k\right) - \mu_i^F \mu_i^M = \mu_i^{FM} - \mu_i^F \mu_i^M \tag{8}$$

$$B_i = \left(\sum_k w_{ik} F_k^2\right) - (\mu_i^F)^2 = \mu_i^{F^2} - (\mu_i^F)^2 \tag{9}$$

$$C_i = \left(\sum_k w_{ik} M_k^2\right) - (\mu_i^M)^2 = \mu_i^{M^2} - (\mu_i^M)^2 \tag{10}$$

Algorithm 2 outlines a vanilla PyTorch implementation of the LNCC loss function. The computational overhead of the algorithm arises due to many intermediates stored in high-bandwidth memory

Algorithm 2 Vanilla PyTorch LNCC implementation

Require: F (input image), M (reference image), w (window size), $reduction$ (reduction type)

1: Define a radially symmetric convolution filter W of size $w \times w \times w$ with $\sum W[i] = 1$
2: Define $\texttt{state} = (F, M, F^2, M^2, FM)$ ▷ Elementwise operations
3: **Compute** $\texttt{state} = W * \texttt{state}$ ▷ Convolution
4: Get $\mu_F = \texttt{state}[0]$ ▷ Local mean of F
5: Get $\mu_M = \texttt{state}[1]$ ▷ Local mean of M
6: Compute $\sigma_F^2 = \texttt{state}[2] - \mu_F^2$ ▷ Local variance of F
7: Compute $\sigma_M^2 = \texttt{state}[3] - \mu_M^2$ ▷ Local variance of M
8: Compute $\sigma_{FM} = \texttt{state}[4] - \mu_F \cdot \mu_M$ ▷ Local covariance of F and M
9: Compute LNCC $= \frac{\sigma_{FM}^2}{\sigma_F^2 \sigma_M^2 + \epsilon}$ ▷ Add small ϵ to avoid division by zero
10: **if** `reduction` == `NONE` **then**
11: **return** LNCC
12: **else**
13: Compute loss: Loss $= 1 -$ mean(LNCC)
14: **return** Loss
15: **end if**

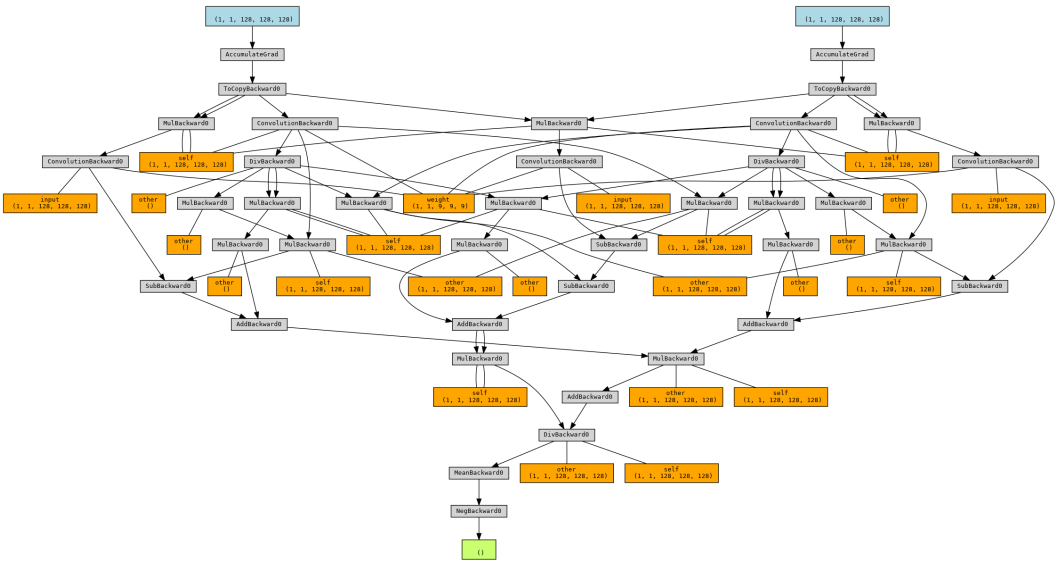


Figure 13: Computational graph of the vanilla PyTorch implementation of the LNCC loss function. Blue nodes denote the input images, Orange nodes denote intermediate tensors that are stored in HBM, Gray nodes denote operations on the computational graph, and Green node denotes the final loss. Orange nodes are the primary memory bottleneck.

(HBM). Specifically, the quantities $W * \texttt{state}, I^2, J^2, IJ, \sigma_I^2, \sigma_J^2, \sigma_{IJ}, \mu_I^2, \mu_J^2, \mu_I\mu_J, \sigma_I^2\sigma_J^2, (\sigma_I^2\sigma_J^2 + \epsilon), \sigma_{IJ}^2, \sigma_{IJ}^2/(\sigma_I^2\sigma_J^2 + \epsilon)$ are all stored as intermediate tensors, each of size N, totalling a $16N$ memory overhead in addition to storing `state`. The computational graph of the vanilla PyTorch implementation is shown in Fig. 13. During the backward pass, the backprop algorithm computes the gradient with respect to each of these variables costing an additional $16N$ memory overhead. A `torch.compile` implementation fuses some of the arithmetic, but leaves a lot of room for improvement (see Fig. 7). We present an algorithm that only requires an additional intermediate variable `state` of size $5N$, saving upto $27N$ memory.

F.1 An efficient fused LNCC implementation

During the forward pass, we initialize a `state` variable of size $5N$. To minimize HBM reads from F and M, we write a fused kernel to initialize the `state` variable using only one HBM read from F and M. The code in Line 4-9 are elementwise operations, and can be fused into another kernel. The

forward pass therefore consumes only $5N$ additional memory. The pseudocode for the efficient fused LNCC implementation is shown in Algorithm 3.

Efficient Backward Pass In a vanilla PyTorch implementation, the gradients are computed for each intermediate variables in the reverse order in the computational DAG shown in Fig. 13. Typically, our implementation would also require defining the backward pass by computing the gradients with respect to the intermediate variables, and then propagating them to the input images. However, we derive the backpropagation with respect to I and J, given the gradient $g_i = \frac{\partial L}{\partial n_i}$ to avoid calculating intermediate gradients. Using the chain rule, we have:

$$\frac{\partial L}{\partial F_k} = \sum_i \frac{\partial L}{\partial n_i}\frac{\partial n_i}{\partial F_k} \tag{11}$$

$$= \sum_i g_i \left(\frac{2A_i}{B_i C_i}\frac{\partial A_i}{\partial F_k} - \frac{A_i}{B_i^2 C_i}\frac{\partial B_i}{\partial F_k}\right) \tag{12}$$

$$\tag{13}$$

which can be simplified to:

$$\frac{\partial \mu_i^F}{\partial F_k} = \frac{\partial \mu_i^M}{\partial M_k} = w_{ik} \tag{14}$$

$$\frac{\partial A_i}{\partial F_k} = \frac{\partial \left(\sum_k w_{ik} F_k M_k - \mu_i^F \mu_i^M\right)}{\partial F_k} = w_{ik}\left(M_k - \mu_i^M\right) \tag{15}$$

and

$$\frac{\partial B_i}{\partial F_k} = \frac{\partial \left(\sum_k w_{ik} F_k^2 - (\mu_i^F)^2\right)}{\partial F_k} = 2w_{ik}(F_k - \mu_i^F) \tag{16}$$

Substituting these results to Eq. (12) we have:

$$= \sum_i g_i \left(\frac{2A_i}{B_i C_i}\left(w_{ik}\left(M_k - \mu_i^M\right)\right) - \frac{A_i^2}{B_i^2 C_i} 2w_{ik}(F_k - \mu_i^F)\right) \tag{17}$$

$$= \sum_i \frac{2g_i A_i}{B_i C_i} w_{ik}\left[M_k - \frac{F_k A_i}{B_i} + \mu_i^F \frac{A_i}{B_i} - \mu_i^M\right] \tag{18}$$

Using the property $w_{ik} = w_{ki}$, and letting $\gamma_i = \frac{2g_i A_i}{B_i C_i}$, we rewrite the previous equation as:

$$= M_k \cdot \left(\sum_i w_{ki}\gamma_i\right) - F_k \cdot \left(\sum_i w_{ki}\frac{\gamma_i A_i}{B_i}\right) + \sum_i w_{ki}\gamma_i\left(\frac{\mu_i^F A_i}{B_i} - \mu_i^M\right) \tag{19}$$

$$= M_k \cdot (w * \gamma)_k - F_k \cdot (w * \gamma_{AB})_k + (w * \gamma_{FM})_k \tag{20}$$

where $\gamma_{AB} = \gamma_i \frac{\mu_i^F A_i}{B_i}$, $\gamma_{FM} = \gamma_i \cdot \left(\frac{\mu_i^F A_i}{B_i} - \mu_i^M\right)$ - and $*$ is the convolution operation. Similarly, the gradient with respect to the moving image M_k is:

$$\frac{\partial L}{\partial M_k} = F_k \left(\sum_i w_{ki}\gamma_i\right) - M_k \left(\sum_i w_{ki}\frac{\gamma_i A_i}{C_i}\right) + \sum_i w_{ki}\gamma_i\left(\frac{\mu_i^M A_i}{C_i} - \mu_i^F\right) \tag{21}$$

$$= F_k \cdot (w * \gamma)_k - M_k \cdot (w * \gamma_{AC})_k + (w * \gamma_{MF})_k \tag{22}$$

where $\gamma_{AC} = \gamma_i \frac{\mu_i^M A_i}{C_i}$, $\gamma_{MF} = \gamma_i \cdot \left(\frac{\mu_i^M A_i}{C_i} - \mu_i^F\right)$. To compute the gradients with respect to F and M, we need to compute five tensors of the γ family, namely $\gamma, \gamma_{AB}, \gamma_{AC}, \gamma_{FM}$, and γ_{MF}. This is followed by performing a convolution with all the tensors, and computing elementwise operations given by Eq. (20) and Eq. (22). The γ family of tensors are simple elementwise operations on the `state` variable, and therefore can be computed by modifying the `state` variable *inplace* to avoid initializing additional HBM memory.

Algorithm 3 Fused LNCC Implementation

Require: F (fixed image), M (moving image), w (window size), ϵ (smoothing term)

function FORWARD(F, M, w, ϵ)
  Define convolution filter W of size $w \times w \times w$ with $\sum W[i] = 1$
  `state` $\leftarrow$ `fused_create_interm`(F, M) ▷ Single HBM read: (F, M, F^2, M^2, FM)
  `state` $\leftarrow W *$ `state` ▷ Convolution on all channels
  LNCC $\leftarrow$ `fusedcc_kernel`$(\texttt{state}, \epsilon)$ ▷ Computes Eqs. (8) to (10) followed by Eq. (3)
  return LNCC
end function

function BACKWARD($g = \frac{\partial \mathcal{L}}{\partial n}$, `state`, F, M, W, use_ants_approximation)
  `state` $\leftarrow$ `fused_compute_gamma`$(g, \texttt{state})$ ▷ Computes γ family of tensors *inplace*
  if use_ants_approximation **then**
    `no-op` ▷ ANTs approximation: skip convolutions
  else
    `state` $\leftarrow W *$ `state` ▷ Convolution on all intermediates
  end if
  $\frac{\partial L}{\partial F} \leftarrow M \odot \gamma - F \odot \gamma_{AB} + \gamma_{FM}$ ▷ Eq. (20) computed in fused kernel
  $\frac{\partial L}{\partial M} \leftarrow F \odot \gamma - M \odot \gamma_{AC} + \gamma_{MF}$ ▷ Eq. (22) computed in fused kernel
  return $\frac{\partial L}{\partial F}$, $\frac{\partial L}{\partial M}$
end function

Table 3: Speedup and memory usage of different LNCC backends

N	Method	Forward Time (s)	Forward Speedup	Backward Time (s)	Backward Speedup	Memory (MB)	Memory Reduction (%)
64	Fast LNCC	0.001	2.95	0.003	4.86	21	61.9
	FireANTs	0.003	7.18	0.002	3.07	25	68
	VoxelMorph	0.06	158.76	0.016	24.10	17	52.9
	`torch.compile`	0.003	6.83	0.002	2.30	24	66.7
	Ours	< 0.001	1.00	0.001	1.00	8	0
128	Fast LNCC	0.008	5.88	0.026	34.09	168	61.9
	FireANTs	0.013	9.04	0.008	10.73	200	68
	VoxelMorph	0.482	341.65	0.126	168.33	136	52.9
	`torch.compile`	0.012	8.67	0.007	8.95	192	66.7
	Ours	0.001	1.00	0.001	1.00	64	0
256	Fast LNCC	0.069	6.19	0.204	82.52	1344	61.9
	FireANTs	0.103	9.25	0.294	118.80	2176	76.5
	VoxelMorph	3.905	351.54	3.903	1577.37	1536.2	66.7
	`torch.compile`	0.1	9.02	0.284	114.74	2176	76.5
	Ours	0.011	1.00x	0.002	1.00x	512	0
512	Fast LNCC	0.627	6.56	1.657	98.75	10752	61.9
	FireANTs	0.856	8.95	2.396	142.77	17408	76.5
	VoxelMorph	31.335	327.71	31.665	1887.14	12288.2	66.7
	`torch.compile`	0.829	8.67	2.312	137.80	17408	76.5
	Ours	0.096	1.00	0.017	1.00	4096	0

ANTs gradient approximation. In the ANTs implementation, the gradient computation skips performing the convolution of the γ family of tensors. We implement this as an additional flag that the user can toggle as an option for faster backward passes. All our experiments use this approximation.

F.2 PERFORMANCE

We compare the performance of our fused implementation to various backend implementations. Fig. 7 shows the speedup and memory usage over different image sizes; we tabulate the results here. For this experiment, we initialize two random images of size $N_v \times N_v \times N_v$ and compute the runtime and memory usage for the forward and backward passes. Results are in Table 3. Our implementation consistently achieves upto $6\times$ forward time speedup and $\sim 98\times$ backward time speedup compared to (Jia et al., 2025) and consumes upto 76% less memory than a compiled PyTorch implementation and 61.9% less than a groupwise convolution implementation (Jia et al., 2025).

G A HIGHLY EFFICIENT MUTUAL INFORMATION IMPLEMENTATION

Mutual Information (MI) is one of the most commonly used loss functions for *multimodal* image matching (Chen et al., 2022; Avants et al., 2009; Mattes et al., 2001). Beyond multimodal image matching, MI is a cornerstone operation in computer vision (Isola et al., 2014; Zhao et al., 2019c), contrastive learning (Quan et al., 2024), remote sensing (Liang et al., 2013), graph learning (Peng et al., 2023), ecological and social community interactions (Luo et al., 2021; Corso et al., 2020), and cosmological dynamics (Sarkar & Pandey, 2020). In biomedical imaging and life sciences, MI is used for multimodal image alignment using the assumption that pixels in multimodal images codify some nonlinear function of the underlying tissue type.

Vanilla MI implementation Given images I and J, Mattes MI considers the intensities from the images as samples from probability distributions p_I and p_J that encode some imaging physics. The intensity pairs (I_k, J_k) are considered to be samples from the joint distribution p_{IJ}. If the images are aligned, then I_k and J_k are highly 'predictable' from each other, implying a low conditional entropy $H(I|J)$, or equivalently a large distance from the distribution $p_I p_J$ which models the joint distribution if samples from I and J were independent. This is precisely the mutual information criteria. Since the samples I_k, J_k follow some unknown distributions, we use a kernel density estimator using kernel κ to estimate the empirical distributions of the joint and marginal distributions. To compute empirical MI, the continuous kernel density estimates are discretized into a probability mass function (PMF) with a finite number of bins. The number of bins B is a hyperparameter that is used to define bin centers $b_i \in [0, 1]$ for $i = \{1, \dots, B\}$, assuming that the intensities are scaled to the range $[0, 1]$.

To compute the discrete PMF with autodifferentiation, we compute a Parzen Block $\Psi_I \in \mathbb{R}^{B\times N}$, s.t.$\Psi_I(i, k) = \kappa(b_i - I_k)$. This forms the memory bottleneck in computing the Mattes MI similarity criteria. In the following, we provide a fused implementation that avoids the $O(NB)$ cost of the Parzen Block, making our implementation only $O(1)$ additional HBM overhead.

G.1 IMPLICIT MI IMPLEMENTATION

We implement custom forward and backward passes to compute the joint and marginal histograms p_{IJ}, p_I, p_J from I and J directly, avoiding the $O(NB)$ cost of the Parzen Block. We derive the backward pass first, followed by the forward pass followed by an efficient approximate estimator of the histograms leading to a faster forward pass.

G.1.1 BACKWARD PASS

We are interested in computing the gradients $\frac{\partial L}{\partial I}, \frac{\partial L}{\partial J}$ given $\frac{\partial L}{\partial p_{IJ}}, \frac{\partial L}{\partial p_I}, \frac{\partial L}{\partial p_J}$. We denote $\omega(b_i - I_k) = \frac{\partial \kappa(b_i - I_k)}{\partial I_k}$.

$$\frac{\partial L}{\partial I_k} = \sum_{m,n} \frac{\partial L}{\partial p_{IJ}[m,n]} \frac{\partial p_{IJ}[m,n]}{\partial I_k} + \sum_n \frac{\partial L}{\partial p_I[n]} \frac{\partial p_I[n]}{\partial I_k} \tag{23}$$

$$= \sum_{m,n} g_{IJ}[m,n] \left(\omega(b_m - I_k)\kappa(b_n - J_k)\right) + \sum_n g_I[n] \left(\omega(b_n - I_k)\right) \tag{24}$$

$$= \sum_n \left[g_I[n]\omega(b_n - I_k) + \sum_m g_{IJ}[m,n]\omega(b_m - I_k) \right] = \sum_n \zeta_1[n] + \zeta_2[n] \tag{25}$$

where $\zeta_1[n] = g_I[n]\omega(b_n - I_k)$ and $\zeta_2[n] = \sum_m g_{IJ}[m,n]\omega(b_m - I_k)$.

To compute this backward pass efficiently, we launch $\lceil N/B \rceil$ threadblocks and partition each threadblock in groups of B threads, and compute the partial gradients $\zeta_1[n], \zeta_2[n]$ on each thread. Each group loads the values of I_k, J_k into register memory. we first compute the quantities $\kappa(b_n - I_k), \kappa(b_n - J_k), \omega(b_n - I_k), \omega(b_n - J_k)$ on thread n and use four shared memory arrays to store them. On thread n, we compute the partial gradient $\zeta_1[n] = g_I[n]\omega(b_n - I_k)$ and $\zeta_2[n] = \sum_m g_{IJ}[m,n]\omega(b_m - I_k)$ using a for-loop over the index $m \in \{1, \dots, B\}$. Finally, on each thread we store the value $\zeta_1[n] + \zeta_2[n]$ on shared memory indexed at n, followed by a $O(\log(n))$

parallel sum over partitioned threads to compute the gradient $\frac{\partial L}{\partial I_k} = \sum_n \zeta_1[n] + \zeta_2[n]$. A similar argument is used to compute the gradient over $\frac{\partial L}{\partial J_k}$. This leads to a faster backward pass than the vanilla PyTorch implementation using no additional HBM overhead Fig. 7(b).

Generalization to novel kernels Note that unlike the vanilla implementation, where some choices of κ are more memory intensive than others (for example, the BSpline kernel has $k_P = 14$ versus $k_P = 4$ for the Gaussian kernel), the memory overhead of our implementation does not depend on the analytical form of κ. To generalize the Implicit MI implementation to novel kernels, the user can specify the form of κ and its derivative ω in the forward and backward passes without any additional considerations.

G.1.2 Forward Pass

The forward pass is computed similarly. Note that the individual contributions from I_k, J_k to the joint histogram $p_{IJ}[m, n]$ are $p_{IJ}[m, n] = \kappa(b_m - I_k)\kappa(b_n - J_k)$ for all $m, n \in \{1, \dots, B\}$. The marginal histograms $p_I[n], p_J[n]$ are computed as $p_I[n] = \kappa(b_n - I_k)$ and $p_J[n] = \kappa(b_n - J_k)$ for all $n \in \{1, \dots, B\}$. Similar to the backward pass, we launch $\lceil N/B \rceil$ threadblocks and partition the threadblock in groups of B threads. Each group of B threads loads the values of I_k, J_k into register memory. On thread n, we compute the quantities $\kappa(b_n - I_k), \kappa(b_n - J_k)$ and store them in shared memory. Thread n can add these quantities into the HBM for histogram entries $p_I[n], p_J[n]$ directly. For computing the joint histogram $p_{IJ}[m, n]$, thread n loops over $m \in \{1, \dots, B\}$ and adds the quantities $\kappa(b_m - I_k)\kappa(b_n - J_k)$ into the HBM for histogram entries $p_{IJ}[m, n]$. Since all values of $\kappa(b_m - I_k), \kappa(b_n - J_k)$ are stored in shared memory, this operation is not bottlenecked by slow HBM reads. To avoid HBM write contentions, we write these values into intermediate histogram buffers of sizes $C \times B \times B, C \times B$ (where C is a constant of choice), and sum along the C dimension. However, this is still a relatively slow operation due to computation of $\kappa(b_m - I_k), \kappa(b_n - J_k)$ and making NB^2 HBM writes. We propose an efficient approximate forward pass that launches only N instead of NB threads, and makes only $3N$ HBM writes.

An approximate histogram estimator Given a kernel κ, we can write $\kappa(b_m - I_k) = \int_t \delta(b_m - I_k - t)\kappa(t)dt = \delta(b_m - I_k) * \kappa$, where δ is the Dirac delta function with the property $\int_{x=-\infty}^{\infty} \delta(x)f(x)dx = f(0)$ for any function f. Using the principle of superposition, we can write $p_I[m] = \frac{1}{N}\sum_k \kappa(b_m - I_k) = \frac{1}{N}\sum_k \kappa * \delta(b_m - I_k) = \kappa * \left(\frac{1}{N}\sum_k \delta(b_m - I_k)\right)$.

In the continuous case, p_I can be obtained *exactly* by calculating the Dirac delta distribution $p_I^{\delta}(b) = \frac{1}{N}\sum_k \delta(b - I_k)$ and convolving it with the kernel κ. However, in the discrete case, this value is inexact. To see this, consider a value I_k that is in bin m, i.e. $\|I_k - b_m\| < \frac{1}{2B}$. The exact value of the PMF due to this sample is $\kappa(b_m - I_k)$. However, the approximate value of the PMF is $\kappa(0)$ since $\delta(b_m - I_k) = 1$ for all $I_k : \|I_k - b_m\| < \frac{1}{2B}$ due to binning, and convolving with κ returns $\kappa(0)$. Since $\|I_k - b_m\| < \frac{1}{2B}$, we can assume that $\|\kappa(0) - \kappa(b_m - I_k)\|$ is small.

To implement this histogram computation efficiently, we launch N threads and in each thread k, compute the bin indices $m^* = \lfloor I_k B \rfloor, n^* = \lfloor J_k B \rfloor$ for each thread, avoiding computation of *soft entries* $\kappa(b_m - I_k), \kappa(b_n - J_k)$ altogether. We simply add 1 to the histogram entries $p_{IJ}[m^*, n^*], p_I[m^*], p_J[n^*]$ in the aggregated histogram buffers, avoiding writing into HBM entries for all $(m, n) \in \{1, \dots, B\}^2$. This reduces the number of HBM writes from $NB^2 + 2NB$ to $3N$. For $B = 32$, this represents $362\times$ less HBM writes. After performing the average, we convolve this histogram with the kernel κ to get the approximate PMF. Since the convolution is done on a B and $B \times B$ sized histograms, this operation is cheap. This implementation leads to faster runtime, consistent performance for both TransMorph and FireANTs (see Table 1).

H Composite Implicit Grid Sampler

I Ring Sampler for scalable distributed interpolation

The random-access nature of deformable interpolation making scaling a difficult challenge for arbitrarily large problem sizes. Given a configuration of sharded images and warp fields across H

Algorithm 4 Grid Sampler Implementation

Require: J_h (moving image shard), $[\mathbf{u}]_j$ (warp field shard), A_h (rescaled affine), t_h (rescaled translation), S_h (diag. scale)

function FORWARD($J_h,\ A_h,\ t_h,\ S_h,\ [\mathbf{u}]_j$)
  `out` $\leftarrow$ `zeros_like`($[\mathbf{u}]_j[0]$)
  for all target voxels (z, y, x) **in parallel (one thread per voxel) do**
    $X \leftarrow (x, y, z)$
    $X_{\text{aff}} \leftarrow A_h X + t_h$ ▷ *affine transform only*
    $X_{\text{disp}} \leftarrow S_h\,[\mathbf{u}]_j[:, z, y, x]$ ▷ *add scaled displacement*
    $X_{\text{src}} \leftarrow X_{\text{aff}} + X_{\text{disp}}$
    `out`$[z, y, x] \leftarrow$ `trilinear_interpolate`$(J_h,\ X_{\text{src}})$ *zero padding at bounds*
  end for
  return `out`
end function

function BACKWARD($g = \frac{\partial \mathcal{L}}{\partial \texttt{out}}, J_h,\ A_h,\ t_h,\ S_h,\ [\mathbf{u}]_j$)
  Initialize $g_{J_h} = 0,\ g_{[\mathbf{u}]_j} = 0,\ g_{A_h} = 0,\ g_{t_h} = 0$
  for all target voxels (z, y, x) **in parallel (one thread per voxel) do**
    Recompute $X,\ X_{\text{aff}},\ X_{\text{disp}},\ X_{\text{src}}$
    Compute tri-linear weights $w_{b_x b_y b_z}$ and $\frac{\partial v}{\partial X_{\text{src}}}$
    Accumulate g_{J_h} into 8 neighbors using $w_{***} \cdot g[z, y, x]$ *(bounds-checked, zero-padded)*
    $g_{[\mathbf{u}]_j}[:, z, y, x]$ += $S_h\ \frac{\partial v}{\partial X_{\text{src}}}\ g[z, y, x]$
    g_{A_h} += $\left(\frac{\partial v}{\partial X_{\text{src}}}\ g[z, y, x]\right) X^\top$
    g_{t_h} += $\frac{\partial v}{\partial X_{\text{src}}}\ g[z, y, x]$
  end for
  return $g_{J_h},\ g_{[\mathbf{u}]_j},\ g_{A_h},\ g_{t_h}$
end function

hosts, neighboring voxels in the sharded warp field can point to pixels in arbitrary regions in the image, illustrated in Fig. 4(a). Moreover, the control points for interpolation can be irregularly distributed across different hosts, illustrated in Fig. 4(b). This makes computation of the interpolated image challenging for displacements that point to pixels between boundaries of different hosts. One approach to avoid this problem is to store the entire moving image on each GPU to compute the interpolated image. However, this approach is impractical once the image size exceeds the memory per GPU. To achieve weak scaling, the HBM overhead per GPU must be proportional to N/H. To alleviate this problem, we propose a ring sampler that avoids the need to store the entire moving image on each GPU by decomposing linear interpolation into partial sums. This produces mathematically correct interpolated images regardless of the nature of the warp field, without storing the entire moving image on each GPU.

I.1 DERIVATION

Consider a d-linear interpolation of an image I defined on Ω using warp coordinates $[\mathbf{u}]_\Omega$ defined on Ω.

$$I = \sum_{b \in \{0,1\}^n} \left(\prod_{k=1}^{n} (1 - \alpha_k)^{1-b_k}\, \alpha_k^{b_k} \right) I\big[i_1 + b_1,\ i_2 + b_2,\ \ldots,\ i_n + b_n\big] \tag{26}$$

where $i_k = \lfloor \varphi(x)_k \rfloor, \qquad \alpha_k = \varphi(x)_k - i_k, \qquad$ for $k = 1, \ldots, d$. Let the individual pixels $I\big[i_1 + b_1,\ i_2 + b_2,\ \ldots,\ i_n + b_n\big]$ be partitioned across H hosts. Since each pixel belongs to exactly one host, we can write $\sum_{h=1}^{H} \mathbb{I}(\mathbf{i} + \mathbf{b} \in [x]_h) = 1$ and multiply with $I[\mathbf{i} + \mathbf{b}]$ to get:

$$I = \sum_{b\in\{0,1\}^n} \left(\prod_{k=1}^{n} (1-\alpha_k)^{1-b_k}\, \alpha_k^{b_k}\right) \left(I\big[\mathbf{i}+\mathbf{b}\big] * \left(\sum_{h=1}^{H} \mathbb{I}(\mathbf{i}+\mathbf{b} \in [x]_h)\right)\right) \tag{27}$$

$$= \sum_{h=1}^{H} \sum_{b\in\{0,1\}^n} \left(\prod_{k=1}^{n} (1-\alpha_k)^{1-b_k}\, \alpha_k^{b_k}\right) \left(I\big[\mathbf{i}+\mathbf{b}\big] * \mathbb{I}(\mathbf{i}+\mathbf{b} \in [x]_h)\right) \tag{28}$$

$$= \sum_{h=1}^{H} I_h \tag{29}$$

where

$$I_h = \sum_{b\in\{0,1\}^n} \left(\prod_{k=1}^{n} (1-\alpha_k)^{1-b_k}\, \alpha_k^{b_k}\right) I\big[\mathbf{i}+\mathbf{b}\big] * \mathbb{I}(\mathbf{i}+\mathbf{b} \in [x]_h) \tag{30}$$

$$= \sum_{b\in\{0,1\}^n} \left(\prod_{k=1}^{n} (1-\alpha_k)^{1-b_k}\, \alpha_k^{b_k}\right) J_h\big[\mathbf{i}+\mathbf{b}\big] \tag{31}$$

where $J_h\big[\mathbf{x}\big] = I[\mathbf{x}]$ if $\mathbf{x} \in [\mathbf{x}]_h$ else 0. Image J_h is therefore *identical* to the sharded image I on host h. Eq. (31) refers to performing trilinear interpolation on the shard I_h (with zero padding) since the sum is only over coordinates that reside in $[x]_h$. This means the warped image in Eq. (26) can be obtained by performing interpolation over the shards individually and adding the warped images together. This is illustrated in Fig. 4(c). Coordinates residing between multiple shards will accumulate partial sums from each sharded image, and no additional consideration is needed for boundary conditions. The communication protocol in this algorithm is similar to Ring Attention (Liu et al., 2024b), where image shards are passed across hosts, and partial results are accumulated into the final result. Our algorithm requires a memory overhead of only N/H to store the sharded image from host $j \neq i$. Our pseudocode is provided in Algorithm 5.

I.2 Implementation Considerations

Rescaling the warp function to sample sharded images Interpolating from sharded images requires one additional consideration. The grid sampler interpolates an image I defined on Ω using warp coordinates $[\mathbf{u}]_\Omega$ defined on Ω. However, the sharded image J_h is defined on the domain Ω_h, and therefore any warped coordinate $\varphi(x) \in \Omega$ must be rescaled to the corresponding coordinates in $\varphi_h(x) \in \Omega_h$. From the implementation standpoint, the leftmost coordinate of J_h is $x^h_{\min}$ when the entire image I is passed to `grid_sampler`. However, when J_h is provided as input to `grid_sampler`, the leftmost pixel of J_h is located at $[-1, -1, \ldots, -1]$ according to PyTorch convention. Since our optimization variables $A, t, [\mathbf{u}]$ refer to locations on Ω, and not Ω_h, we need to rescale these variables appropriately when sampling from J_h.

The rescaling corresponds to a diagonal scaling matrix S_h and translation t_h such that $S_h x^h_{\min} + t_h = x^\Omega_{\min}$ and $S_h x^h_{\max} + t_h = x^\Omega_{\max}$. The resampled warp function to sample from J_h becomes $\varphi_h(x) = S_h(Ax + t + u(x)) + t_h = (A'_h x + t'_h) + S_h u(x)$. where $A'_h, t'_h = S_h A, (S_h t + t_h)$. Therefore, we must sample J_h using the transform $A'_h[\mathbf{x}]_{\Omega_h} + t'_h + S_h[\mathbf{u}]_{\Omega_h}$. In the vanilla grid sampler implementation, the intermediate grid $S_h[\mathbf{u}]_{\Omega_h}$ and its gradient consume another $6N/H$ memory. Combined with the N/H overhead for storing the received image shard, we add a total of $7N/H$ memory overhead, which is less than N for $H \geq 8$, making the algorithm impractical for fewer GPUs (say $H = 4$).

To prevent this $6N/H$ additional overhead, we extend the generalized grid sampler as mentioned Appendix 3.1 to sample from a transform of the form $A[x] + t + S[u]$ directly. This computes the value $Su(x)$ directly inside the CUDA kernel, and the backward pass also computes and accumulates the gradient w.r.t. $u(x)$ directly, avoiding the $6N/H$ overhead.

Interleaved communication An important implementation detail is the interleaving of communication and computation in the ring sampler. While we compute the partial moved image aggregate, the next image shard can be fetched asynchronously in the background. This is illustrated in Fig. 14.

Algorithm 5 Ring Sampler Implementation

Require: M_j (moving image shard), $[\mathbf{u}]_j$ (warp field shard), (A, t) (affine transform)

function FORWARD(M_j, $[\mathbf{u}]_j$, (A, t))
  Define $\texttt{moved}_j = 0$
  for $h = 1$ **to** H **do**
    $J_h \leftarrow \texttt{send\_and\_recv}(M_j, h)$ ▷ Send and receive the image shard from offset h
    Compute diagonal S_h, t_h such that $S_h x^h_{\min} + t_h = x^\Omega_{\min}$ and $S_h x^h_{\max} + t_h = x^\Omega_{\max}$
    Rescale affine transform $A_h \leftarrow S_h A, t_h \leftarrow S_h t + t_h$
    $\texttt{moved}_j \leftarrow \texttt{moved}_j + \texttt{grid\_sampler}(J_h; A_h, t_h, S_h, [\mathbf{u}]_j)$ ▷ Avoid computing $S_h[\mathbf{u}]_j$ explicitly
  end for
  return $\texttt{moved}_j$
end function

function BACKWARD($g = \frac{\partial \mathcal{L}}{\partial \texttt{moved}_j}$, $\texttt{moved}_j$, M_j, $[\mathbf{u}]_j$, (A, t))
  Define $g_{[\mathbf{u}]_j} = 0, g_A = 0, g_t = 0, g_{M_j} = 0$
  for $h = 1$ **to** H **do**
    $J_h \leftarrow \texttt{send\_and\_recv}(M_j, h)$ ▷ Send and receive the image shard from offset h
    Compute diagonal S_h, t_h such that $S_h x^h_{\min} + t_h = x^\Omega_{\min}$ and $S_h x^h_{\max} + t_h = x^\Omega_{\max}$
    Rescale affine transform $A_h \leftarrow S_h A, t_h \leftarrow S_h t + t_h$
    if $\texttt{requires\_grad}(M_j)$ **then**
      $g_{inp} \leftarrow \texttt{zeros\_like}(M_j)$
    else
      $g_{inp} \leftarrow \texttt{None}$
    end if
    Compute $\texttt{backward\_grid\_sampler}(g, J_h, A_h, t_h, S_h, [\mathbf{u}]_j, g_{[\mathbf{u}]_j}, g_A, g_t, g_{inp})$
    if $\texttt{requires\_grad}(M_j)$ **then**
      $g'_{M_j} = \texttt{send\_and\_recv}(g_{inp}, -h)$
      $g_{M_j} \leftarrow g_{M_j} + g'_{M_j}$
    end if
  end for
  return $g_{[\mathbf{u}]_j}, g_A, g_t, g_{M_j}$
end function

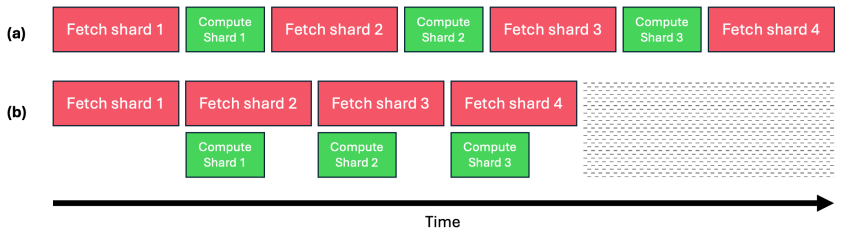


Figure 14: Interleaved communication (red) and computation (green) in the ring sampler. gray denotes time saved by interleaving communication and computation.

I.3 ALTERNATIVE DESIGNS

A naive approach can be to route the coordinate $\varphi(x_i) \in [\mathbf{x}]_j$ to GPU j and retrieve the image coordinate, similar to routing tokens using expert parallelism (EP) used for Mixture-of-Experts (MoEs) (Shazeer et al., 2017; Jordan & Jacobs, 1994). However, this approach has two major drawbacks in our setting. First, due to the deformable nature of φ, the partitioning of coordinates across hosts is generally uneven. In the worst case, a single GPU can receive all $3N$ coordinates leading to an indirect `allgather` operation resulting in OOMs or uneven GPU utilization across hosts. Second, coordinates that point to regions between two multiple image boundaries need to be sent to variable number of hosts, which is non-trivial to implement. These two factors make both the forward and

backward pass implementations cumbersome. Inspired by (Liu et al., 2024b), we propose a distributed ring sampler that decomposes the computation into partial sums, leading to a simple implementation without degraded scaling performance Fig. 8.

J Correctness of Implementation

All code is checked for numerical correctness by comparing the results with PyTorch implementations using unit and integration tests. Code and generated data will be made available to the community.

J.1 Ablation on FireANTs speedup

We run FireANTs with different backends for LNCC and MI loss functions on the OASIS validation set (Marcus et al., 2007; Hering et al., 2022). We measure the end-to-end runtime, peak memory usage (except the fixed and moving images), and Dice score. We ablate on both Greedy and SyN algorithms; in the case of SyN, additional gradients may be required. Results in Table 4 show that our implementation achieves a significant speedup over the baseline implementations. Although the `torch.compile` version of LNCC is faster than other variants, it leads to brittle performance.

Table 4: Extended Results on accelerated registration on FireANTs: Accelerating FireANTs registration with various computation backends and registration algorithms (Greedy and SyN). Our implementations maintain accuracy while substantially reducing runtime and peak memory usage. (Green)/ (Yellow) = best/second; Speedup and memory reduction are computed with respect to our kernels. Our fused kernels maintain accuracy while substantially reducing runtime and peak memory usage.

Algorithm	Method	Backend	Dice Score ↑	Runtime (s) ↓	Memory (MB) ↓	Speedup ↑	Mem. Reduction (%) ↑
Greedy	LNCC	VXM/TM	76.96 ± 3.60	57.08 ± 2.45	1418.5 ± 0.0	113.47	59.29
	LNCC	FastLNCC	76.96 ± 3.60	3.76 ± 0.16	1026.3 ± 0.0	7.48	43.73
	LNCC	FireANTs	72.81 ± 3.87	1.44 ± 0.08	1044.5 ± 0.0	2.87	44.71
	LNCC	`torch.compile`	69.35 ± 4.09	0.82 ± 0.04	860.7 ± 0.0	1.63	32.90
	LNCC	Ours	78.67 ± 3.04	0.50 ± 0.01	577.5 ± 0.0	1.00	0.00
Greedy	MI	PyTorch	75.88 ± 3.45	7.51 ± 0.37	12206.3 ± 0.0	2.59	95.27
	MI	`torch.compile`	75.88 ± 3.45	1.05 ± 0.05	3865.5 ± 0.0	0.36	85.06
	MI	Ours	75.87 ± 3.44	2.90 ± 0.16	577.5 ± 0.0	1.00	0.00
	MI	Ours + `torch.compile`	75.93 ± 3.47	2.95 ± 0.16	657.3 ± 0.0	1.02	12.13
SyN	LNCC	VXM/TM	76.69 ± 2.88	63.57 ± 0.58	1892.0 ± 0.0	65.92	50.05
	LNCC	FastLNCC	76.70 ± 2.88	4.27 ± 0.05	1486.7 ± 0.0	4.43	36.43
	LNCC	FireANTs	74.70 ± 2.93	2.55 ± 0.10	1616.4 ± 0.0	2.65	41.54
	LNCC	`torch.compile`	71.65 ± 3.41	1.46 ± 0.04	1472.0 ± 0.0	1.51	35.80
	LNCC	Ours	78.79 ± 2.82	0.96 ± 0.08	945.0 ± 0.0	1.00	0.00
SyN	MI	PyTorch	76.74 ± 2.58	12.84 ± 0.66	17720.8 ± 0.0	2.96	94.67
	MI	`torch.compile`	76.76 ± 2.58	2.40 ± 0.13	7758.9 ± 0.0	0.55	87.82
	MI	Ours	76.86 ± 2.59	4.34 ± 0.28	945.0 ± 0.0	1.00	0.00
	MI	Ours + `torch.compile`	77.00 ± 2.57	4.56 ± 0.24	1104.5 ± 0.0	1.05	14.44

Table 5: Extended Efficiency Results on faux-OASIS-dataset: Comparison of registration methods across multiple resolutions. Reported metrics include average Dice similarity coefficient (higher is better), wall-clock runtime, GPU cost (measured in GB-hours), relative speedup, and GPU cost reduction with respect to FireANTs + FFDP(Ours). GPU usage (e.g., single GPU, multi-GPU, or CPU) is annotated alongside the cost values.

Resolution	Method	Avg Dice Score ↑	Wall Clock ↓ (10^{-2} Hours)	GPU Cost ↓ (10^{-2} GB-Hours)	Speedup	GPU Cost Reduction (%)
1 mm	TransMorph	0.851 ± 0.016	0.015	0.262^{1}	0.56×	87.81
	VFA	0.851 ± 0.023	0.017	0.216^{1}	0.63×	85.18
	Ours	0.838 ± 0.028	0.027	0.032^{1}	1.00×	0.00
	UniGradICON-IO	0.826 ± 0.022	5.167	58.498^{1}	194.07×	99.95
	FireANTs	0.822 ± 0.032	0.060	0.141^{1}	2.25×	48.83
	UniGradICON-noIO	0.815 ± 0.026	0.067	0.238^{1}	2.50×	86.55
	SynthMorph	0.801 ± 0.022	2.155	99.061^{1}	80.93×	99.97
	Anatomix	0.796 ± 0.035	0.379	2.656^{1}	14.24×	98.80
	CLAIRE	0.776 ± 0.044	0.518	1.389^{1}	19.47×	97.70
	ITK-dreg	0.662 ± 0.055	1.527	1.017^{CPU}	57.37×	–
500 µm	Ours	0.872 ± 0.028	0.109	0.862^{1}	1.00×	0.00
	FireANTs	0.841 ± 0.033	0.270	4.136^{1}	2.48×	48.22
	VFA	0.805 ± 0.044	0.302	3.896^{1}	2.78×	77.87
	CLAIRE	0.779 ± 0.051	25.903	396.169^{1}	238.04×	99.78
	SynthMorph	0.771 ± 0.035	4.068	187.049^{1}	37.39×	99.54
	TransMorph	0.759 ± 0.028	0.198	3.501^{1}	1.82×	75.38
	Anatomix	0.758 ± 0.040	8.837	310.818^{1}	81.21×	99.72
	ITK-dreg	0.699 ± 0.056	41.259	207.466^{CPU}	379.17×	–
	UniGradICON-IO	0.615 ± 0.047	84.538	1072.657^{1}	776.89×	99.92
	UniGradICON	0.610 ± 0.044	0.842	3.545^{1}	7.73×	75.69
250 µm	Ours	0.895 ± 0.029	1.065	47.059^{1}	1.00×	0.00
	CLAIRE	0.809 ± 0.054	1207.536	$159\,046.981^{4}$	1133.84×	99.97
	FireANTs	0.777 ± 0.064	13.588	253.295^{1}	11.73×	81.42
	VFA	0.714 ± 0.066	3.872	49.939^{1}	3.64×	5.77
	SynthMorph	0.690 ± 0.052	32.808	1507.133^{1}	30.80×	96.88
	TransMorph	0.689 ± 0.044	2.597	45.965^{1}	2.44×	−2.38
	Anatomix	0.620 ± 0.031	88.480	3112.015^{1}	83.07×	98.49
	UniGradICON-IO	0.398 ± 0.062	163.812	2539.721^{1}	153.80×	98.15
	UniGradICON	0.359 ± 0.044	7.811	55.057^{1}	7.33×	14.53
	ITK-dreg	0.758 ± 0.046	1363.868	$33\,065.677^{\text{CPU}}$	1280.63×	–

K Additional Details on the simulated ex-vivo brain MRI dataset

In this section, we provide additional details on the synthetic data generation pipeline for the faux-OASIS dataset, followed by baseline configurations, and finally compare performance-efficiency tradeoffs and show qualitative results.

K.1 Synthetic Data Generation Pipeline

To emulate high resolution ($250\mu m$ isotropic) T1 weighted images, we use the standard OASIS validation dataset to generate synthetic images. Our method is inspired by Billot et al. (2023); Dey et al. (2025) to use the labelmaps as a starting point and synthesize images that are faithful to the labelmaps. The synthetic data generation pipeline is illustrated in Fig. 15. Specifically, our pipeline has three stages:

1. **Compute per-label intensity statistics**: For each label, we consider all the intensities in the voxels belonging to the label. We store mean and standard deviation of the intensities for each label computed over the entire OASIS validation set.
2. **Geometry-preserving upsampling of labels**: We use the labelmaps at 1mm isotropic and perform surface-based upsampling to resample the labelmaps with subvoxel accuracy (Sullivan & Kaszynski, 2019).
3. **Intensity painting**: We use the per-label intensity statistics and the voxelized labelmaps at $250\mu m$ isotropic to synthesize the images.

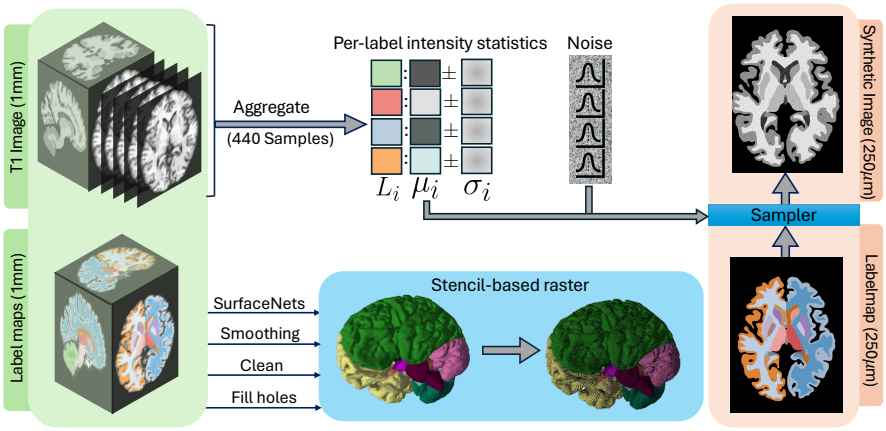


Figure 15: Synthetic data generation pipeline for faux-OASIS. Coarse anatomical labels undergo geometry-preserving upsampling via surface reconstruction, followed by statistical intensity painting to produce high-resolution MR images at 0.25 mm.

Following the generation of $250\mu m$ images, we downsample the images to $500\mu m$ and $1mm$ isotropic to show the effect on performance with downsampled images.

We describe the pipeline in detail below.

Per-label intensity statistics. Contrary to other synthetic data generation pipelinesDey et al. (2025); Billot et al. (2023), we do not want to generate randomized intensities for each image and want to simulate the T1-weighted images. Towards this end, we compute the per-label intensity statistics for all images in the OASIS validation set.

Geometry-preserving upsampling of labelmaps. Given an image volume and labelmap pair (I, L), we upsample the labelmap to $250\mu m$ isotropic $L_\uparrow$. However, naively upsampling the label voxel grid and thresholding typically causes blocky artifacts (Frisken, 2022; Lorensen & Cline, 1998; Schroeder & Tsalikis, 2023), which has led to many sophisticated subvoxel-accurate surface reconstruction algorithms. We use PyVista's SurfaceNets algorithm (Frisken, 2022) to extract surface contours from 3D image label maps. Specifically, an `ImageData` object with labels is converted into *cell data* using `contour_labels` (VTK SurfaceNets) to obtain per-label surfaces S_ℓ that respect voxel geometry and avoid block artifacts with voxel based interpolation. The generated surface is smoothed using a constrained Taubin/Windowed-Sinc smoothing with conservative iterations (typically 16-30, relaxation ≈ 0.5), then use `clean` and `fill_holes` to remove slivers and pinholes while preserving anatomical shape fidelity. The surface S_ℓ is voxelized to obtain a binary mask M_ℓ, then the labelmap $L_\uparrow$ is assembled as

$$L_\uparrow(\mathbf{p}) \;=\; \begin{cases} \ell & \text{if } M_\ell(\mathbf{p}) = 1 \text{ for some } \ell \geq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Finally, image-stencil-based rasterization (`voxelize_binary_mask`) is performed into the target `ImageData` at $t = 0.25$ mm.When surfaces overlap, later labels in the loop take precedence; we process labels in anatomical priority order to ensure critical structures are preserved. All steps for labelmap upsampling are implemented with PyVista/VTK for robustness and reproducibility.

Synthesizing the image. For each label, we fill the voxels with intensities sampled from a normal distribution with the mean and standard deviation of the intensities corresponding to the label.

Given $\{(\mu_\ell, \sigma_\ell)\}$, we synthesize the image by i.i.d. draws within each region:

$$I_{\text{syn}}(\mathbf{p}) \;\sim\; \mathcal{N}\big(\mu_{L_\uparrow(\mathbf{p})},\, \sigma^2_{L_\uparrow(\mathbf{p})}\big), \qquad \mathbf{p} \in \Omega_t.$$

We follow this step with a Gaussian smoothing with $\sigma = 0.75$ voxels to impart local coherence without washing out label edges. Background ($L_\uparrow$=0) is set to zero.

Algorithm 6 High-Resolution MR Synthesis Pipeline

Require: L, spacings $\mathbf{s}$, affine A, stats $\{(\mu_\ell, \sigma_\ell)\}_{\ell=1}^{K}$, target spacing t

Compute $\tilde{N}_x, \tilde{N}_y, \tilde{N}_z$ and $\tilde{A}$ as above
$L_\uparrow \leftarrow 0$ on Ω_t
for $\ell = 1$ **to** K **do**
    $S_\ell \leftarrow \textsc{SurfaceNets}(L{=}\ell)$; smooth & fill holes
    $M_\ell \leftarrow \textsc{Voxelize}(S_\ell, \Omega_t)$
    $L_\uparrow[\text{where } M_\ell{=}1] \leftarrow \ell$ ▷ Assign label to voxelized region
end for
$I_{\text{syn}} \leftarrow 0$
for $\ell = 1$ **to** K **do**
    $U \leftarrow \{\mathbf{p} \mid L_\uparrow(\mathbf{p}) = \ell\}$
    $I_{\text{syn}}[U] \leftarrow \textsc{Normal}(\mu_\ell, \sigma_\ell^2)$ ▷ IID draws
end for
$I_{\text{syn}} \leftarrow \textsc{GaussianBlur}(I_{\text{syn}}, \sigma{=}0.75)$
return $(I_{\text{syn}}, L_\uparrow, \tilde{A})$

Randomization and metadata. All stochastic draws are seeded per subject (seed = `base_seed` + `subject_id`) for exact reproducibility.[2] All outputs are written as NIfTI files with same origin and directions as the original images, but with a voxel spacing of $t = 0.25\,\text{mm}$.

K.2 Baselines

We augment FireANTs (Jena et al., 2024a) with FFDP to enable scalable image registration at high resolutions. The methods and their hyperparameter settings are described below:

- **CLAIRE**(Mang et al., 2019): CLAIRE is a velocity-based diffeomorphic registration framework optimized for distributed GPU/CPU execution via MPI. We use the official repository inside a custom multi-GPU Docker image that adds CUDA-aware Open MPI (v4.0.3; CUDA 11), since the official container supports only single-GPU runs. We launch one MPI rank per GPU and bind each rank to a distinct device via a lightweight wrapper that maps `OMPI_COMM_WORLD_RANK` to `CUDA_VISIBLE_DEVICES`, enabling data-parallel execution across N GPUs. We keep default solver settings, request deformation maps (`-defmap`), and set the continuation parameter `-betacont 7.75e-04` following the official examples; all other hyperparameters use documented defaults, including the iteration cap (`-maxit 50`). Full-resolution runs use 4 GPUs (4 ranks), while half/quarter resolutions use a single GPU (1 rank).
- **ITK-DReg**(itk): ITK-DReg is a CPU-based, distributed, out-of-memory registration framework built on ITK and `dask.distributed`, formulating registration as block-wise map–reduce. We use the `itk_dreg` pipeline with Elastix in deformable-only B-spline mode: the metric is AdvancedNormalizedCorrelation with three pyramid levels (`NumberOfResolutions=3`, `GridSpacingSchedule=[4,2,1]`), optimized via AdaptiveStochasticGradientDescent with `MaximumNumberOfIterations=500`. We use random sampling with `NumberOfSpatialSamples=5000` (refreshed each iteration). Registration operates in voxel units with `FinalGridSpacingInVoxels=20` and `BSplineTransformSplineOrder=3`. To scale to high resolutions, the fixed image is tiled into 256^3-voxel chunks with 25% overlap per axis; per-block results are reduced to a global displacement field defined on a grid subsampled by a factor of 4. ITK threading is set via `SetGlobalDefaultNumberOfThreads`=24 (reported `GetGlobalMaximumNumberOfThreads`=128).
- **FireANTs + FFDP (Ours)**(Jena et al., 2024a): We use the official repository and scripts, except for our proposed modules (grid sampler, LNCC, and Mutual Information). We perform registration using the multi-scale of $4mm, 2mm, 1mm, 500\mu m$, and $250\mu m$ for $200, 200, 200, 100, 25$ iterations. We also truncate the optimization at $1mm$ and $500\mu m$ resolutions to verify the performance of the method at downsampled resolutions. We use

[2] We use `base_seed` = 2025 in our experiments.

our Fused LNCC implementation with a window size of 7, and a learning rate of 0.5. The smoothing kernels are chosen with a $\sigma_{warp} = 0.5$ pixels, and $\sigma_{grad} = 1.0$ pixels.

We also evaluate against state-of-the-art deep learning methods:

- **SynthMorph**(Hoffmann et al., 2021): SynthMorph uses an acquisition-free synthetic data generation pipeline to train a registration network. We use the default `mri_synthmorph` script provided by the vendor. Since all images are affine-aligned, we use the deformable registration mode `-m deform` with a regularization weight of `-r 0.25`.
- **Vector-Field Attention** (Liu et al., 2024c): Vector-Field Attention (VFA) is a weakly-supervised learning-based method utilizing a novel attention module to retrieve per-pixel correspondence based on feature similarity. We evaluate using the pretrained model (trained on OASIS data with weak label supervision) provided in the official repository.
- **UnigradICON** (Tian et al., 2024): UnigradICON is a foundational registration model by training on a composite dataset consisting of lung CT, knee MRI, Abdomen CT, brain MRI, totalling more than 3 million image pairs, of which 4000 image pairs are sampled per epoch to mitigate data imbalance. The model is trained with a bidirectional similarity loss and an inverse consistency loss. UnigradICON also provides an instance optimization based postprocessing step to improve the registration performance. We use the pretrained model and scripts provided in the official repository, and compare performance with and without the instance optimization step.
- **TransMorph** (Chen et al., 2022): TransMorph is one of the first successful application of transformer-based architectures for image registration, marking a departure from traditional convolutional architectures. Compared to other convolutional architectures, TransMorph demonstrates higher performance under domain shift Jian et al. (2024); Jena et al. (2025; 2024b) among the deep learning methods. We use the pretrained model (`TransMorph-Large` trained on the OASIS dataset) that is provided in the official repository.
- **Anatomix + ConvexAdam** (Dey et al., 2025): Anatomix is a feature extractor that is trained to anticipate strong domain shift at training time and uses contrastive learning to extract domain-agnostic features that mitigate the effect of nuisance factors. Anatomix shows strong results on zero-shot registration on abdomen and myocardium. We use the pretrained model and scripts provided in the official repository.

We also acknowledge Quicksilver (Yang et al., 2017) as a relevant baseline that performs patch-based registration. However, despite our best efforts with containerizing the environment (the dependencies are no longer available or supported on modern hardware), we were unable to run this baseline on our system.

All deep learning methods are tested on $1mm, 500\mu m$, and $250\mu m$ resolutions. On $500\mu m$ and $250\mu m$ resolutions, all methods run out of memory on a single NVIDIA A6000 GPU, and the methods do not provide infrastructure to run on multiple GPUs. We adopt the patch-based registration strategy adopted by the literature on high-resolution registration methods for histology (Wodzinski et al., 2024; Lotz et al., 2015; Liang et al., 2021) as additional baselines with the above deep learning models as registration backends. We choose (Hoffmann et al., 2021; Dey et al., 2025; Tian et al., 2024) as general-purpose deep learning methods to mitigate the effect of domain shift due to patch-based registration at higher resolutions, and (Liu et al., 2024c; Chen et al., 2022) as methods that are trained with weak label supervision on the OASIS dataset to verify performance at 1mm resolution and observe the performance at higher resolutions.

Robust HD90 (Cumulative) Hausdorff distance is a widely adopted boundary-based metric in medical image registration. The conventional definition of HD90 (the 90th percentile Hausdorff distance) simply reports the 90th percentile, but does not provide an average performance for all surface boundaries. In contrast, we employ a modified formulation, which we denote as *cumulative HD90*, designed to provide a more stable and comprehensive estimate. Specifically, rather than selecting the single distance value at the 90th percentile, we compute the mean of all surface distances

up to the 90th percentile. Formally, given sorted distances $\{d_i\}_{i=1}^{N}$, we compute

$$\mathrm{HD}_{90}^{\mathrm{cu}} = \frac{1}{k}\sum_{i=1}^{k} d_i, \quad k = \lfloor 0.9\,N \rfloor.$$

Distances are computed bidirectionally between ground-truth and predicted surfaces using isotropic voxel spacing, and the final HD90 is defined as the maximum of the two directional estimates.

K.3 Additional Results and Discussion

Table 5 shows the performance comparison for all methods at different resolutions. All methods using full-context (CLAIRE, ITK-DReg, FireANTs) show improvement in performance with resolution, while all deep learning methods degrade in performance due to (a) progressive domain shift at higher resolutions, even for models trained on multiple or synthetic data, and (b) unlike image registration for histology slides, volumetric datasets like these require large deformations, and patch-based methods do not provide the context to perform well at higher resolutions. In terms of efficiency, our method is substantially more efficient, both on terms of wall clock time, and the total GPU-hours consumed.

Although CLAIRE proposes a distributed GPU framework, the usage of scaling-and-squaring (which requires performing an integral and its adjoint computation every iteration) and other line search subroutines consume a considerable amount of resources. On the faux-OASIS dataset at full resolution, CLAIRE runs out of memory with 1 and 2 GPUs, and does not work on 3 GPUs due to indivisibility of the image size by 3. So the minimum number of GPUs required to run CLAIRE is 4. Our method runs on a single GPU, but does not require the image sizes to be divisible by the number of GPUs, or any other qualitative constraints, allowing researchers to simply plug in their inputs and run their workflows. For large-scale volumetric image registration problems, our method achieves three orders of magnitude of speedup over CLAIRE while enabling multimodal support and arbitrarily loss functions of choice.

K.4 Qualitative Results

Qualitative results are shown in Figs. 16 to 18. With the exception of CLAIRE, ITK-DReg, and Ours, all methods get progressively worse as the resolution increases.

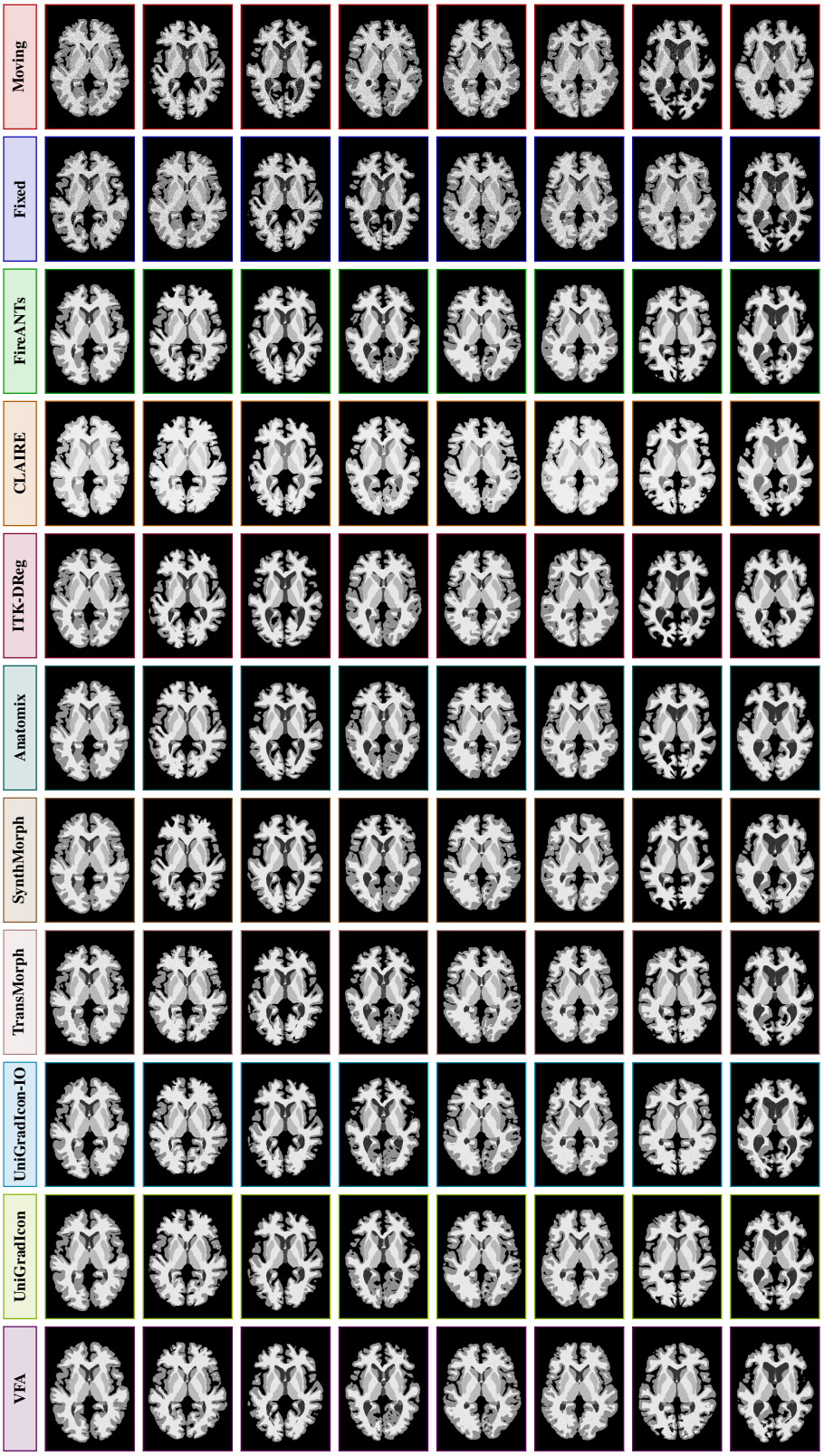


Figure 16: Qualitative comparison of registration results at **1 mm**. Each row corresponds to the moving image, fixed image, or one of the registration methods, with 8 representative slices per row. The comparisons illustrate visual alignment quality and anatomical consistency across methods.

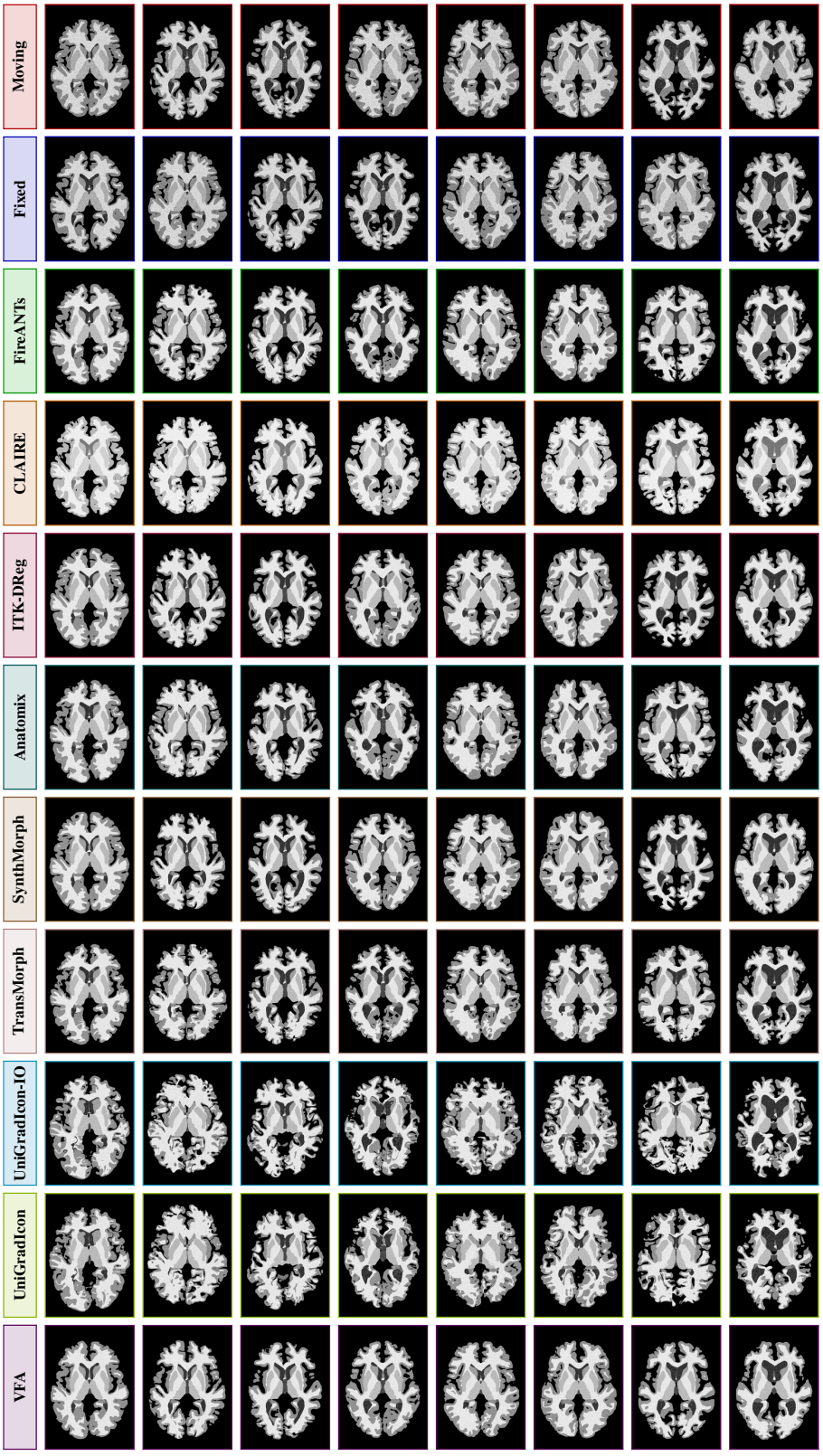


Figure 17: Qualitative comparison of registration results at **500um**. Each row corresponds to the moving image, fixed image, or one of the registration methods, with 8 representative slices per row. The comparisons illustrate visual alignment quality and anatomical consistency across methods.

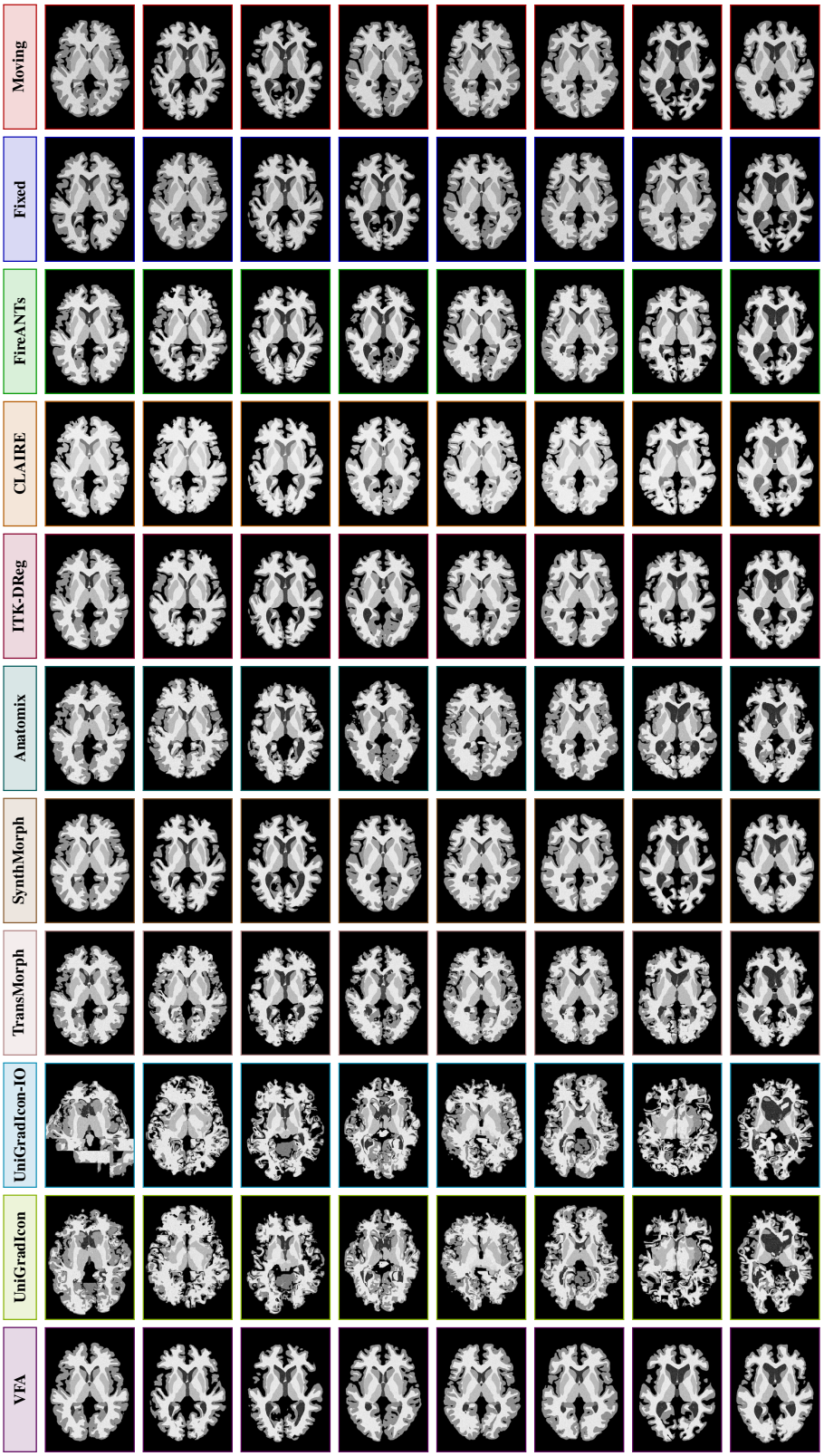
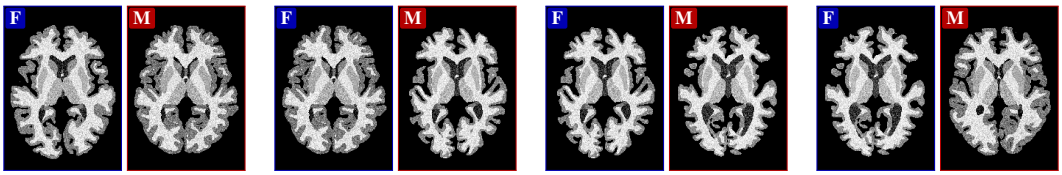
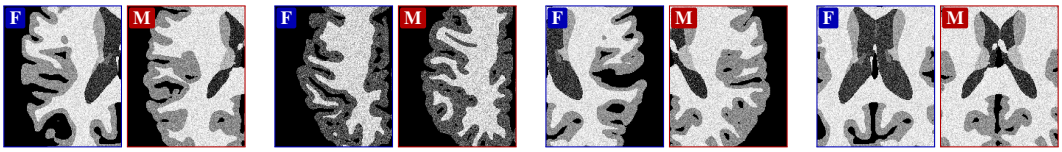


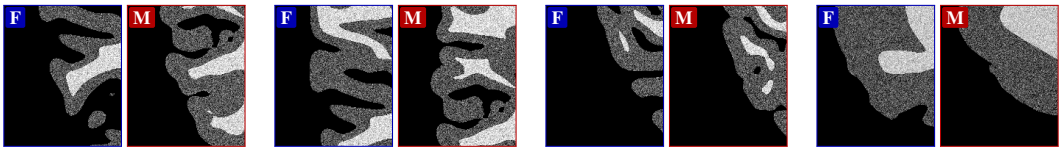
Figure 18: Qualitative comparison of registration results at **250um**. Each row corresponds to the moving image, fixed image, or one of the registration methods, with 8 representative slices per row. The comparisons illustrate visual alignment quality and anatomical consistency across methods.



(a) 1mm isotropic



(b) 500μm isotropic



(c) 250μm isotropic

Figure 19: Patch pairs seen during registration for patch-based methods: For the 1mm isotropic images, there is only a single patch, i.e. the entire image. Deep learning methods utilize the global spatial context to perform accurate registration. At 500μm isotropic, the patches still have large spatial context, but the images are out-of-distribution, leading to *degraded* performance Fig. 5a. At 250μm isotropic, there is no meaningful spatial context and the patches are completely out-of-distribution, leading to poor performance for all patch-based methods.

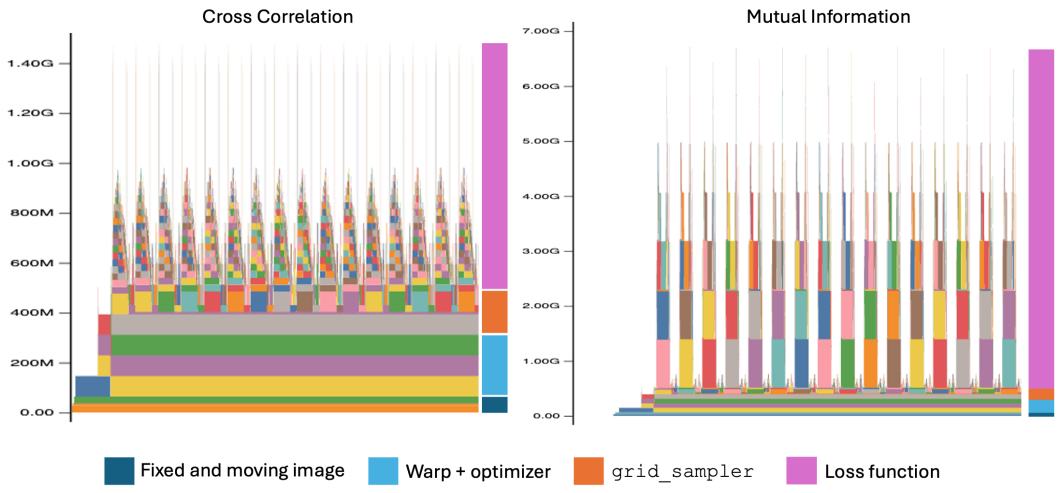


Figure 20: Flamegraph of FireANTs for Cross Correlation (left) and Mutual Information (right) losses on the OASIS dataset. The flamegraph is annotated on the right with colored blocks denoting the memory overheads for the fixed and moving images, the warp field and its optimizer state, the `grid_sampler` operation, and the loss function. Most of the computational overhead is due to the loss function, followed by the `grid_sampler` operation. This motivates the use of fused kernels to eliminate intermediate memory overheads.