Supplementary Information

Supplementary Notes

Supplementary Note 1 | Agentic workflow architecture and reproducibility

Supplementary Note 2 | MRI-style transfer network for data augmentation

Supplementary Note 3 | Segmentation network architecture

Supplementary Note 4 | Connectivity-aware white-matter refinement

Supplementary Tables

Supplementary Table 1 | Quantitative assessment of skull-stripping detection and species differentiation in five species

Supplementary Table 2 | Performance benchmarks for skull-stripping and white-matter segmentation pipelines in five species

Supplementary Table 3 | Comparative performance assessment of cortical surface reconstruction pipelines in seen-domain data

Supplementary Table 4 | Skills implemented by four specialized agents in PrimateX

Supplementary Figures

Supplementary Fig 1 | Data curation and quality control for multi-source NHP T1-weighted MRI datasets

Supplementary Fig 2 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (common marmoset) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.

Supplementary Fig 3 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (cynomolgus macaque) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.

Supplementary Fig 4 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (rhesus macaque)

demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.

Supplementary Fig 5 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (chimpanzee) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.

Supplementary Fig 6 | Quantitative validation of segmentation performance across species in the absence of ground truth

Supplementary Fig 7 | Qualitative assessment of segmentation performance on unannotated unseen-domain datasets

Supplementary Fig 8 | Qualitative assessment of segmentation performance on eight previously unseen NHP species

Supplementary Note 1: Agentic workflow architecture and reproducibility

Problem context. Structural brain preprocessing is inherently sequential and prone to error propagation, where inaccuracies in skull-stripping distort downstream tissue segmentation, ultimately compromising cortical surface reconstruction[1]. Key challenges include species-specific template selection, tissue prior customization, and parameter optimization - tasks traditionally guided by expert intuition rather than codified logic. Large language model (LLM)-driven agents offer a transformative approach, which dynamically interpret user intent, discern biologically appropriate processing strategies, and assemble structured workflows without requiring manual parameter specification. Parallel successes include CRISPR-GPT[2], which achieved >99% design accuracy in gene-editing experiments via a state-machine architecture, and CellAgent[3], which automated 92% of tasks in a single-cell RNA-seq benchmark using a three-tiered planner-executor-evaluator framework. A critical barrier to LLM adoption in

scientific workflows is their inherent non-determinism. Systematic evaluations reveal that even at zero-temperature sampling, API-based LLMs reproduce identical outputs merely 22% of the time[4]. This variability stems from GPU floating-point non-determinism, dynamic inference scheduling, and uncontrolled backend updates - all incompatible with reproducible neuroimaging analysis.

PrimateX resolves this through a two-phase architecture that isolates stochastic reasoning from deterministic execution. During the planning phase, the LLM freely explores potential workflows, optimizing for species and task constraints. Once validated, the workflow is compiled into a skill dependency graph and executed through a stateful runtime that disallows runtime extension. This separation preserves the LLM's adaptive strength while meeting scientific rigor requirements.

Agent architecture

PrimateX employs an LLM-based agentic system that converts natural language requests into deterministic, executable neuroimaging workflows. The system leverages DeepSeek-v4-flash as its core reasoning engine - a mixture-of-experts model (284B total parameters, 13B active per token; 1M-token context window) accessed via the official DeepSeek API.

Skill-centric architecture. Departing from flat function-calling approaches, PrimateX implements a hierarchical skill architecture where workflow specification, environment interaction, and execution are governed by modular Agent Skills. This design extends prior tool-augmented language model paradigms[5, 6]. At initialization, the system loads (i) Prior knowledge: curated neuroimaging principles and PrimateX-specific protocol rules; (ii) Skill definitions:

self-contained modules encapsulating metadata, I/O specifications, procedural logic, and resources (e.g., templates, atlases) for discrete operations. To optimize context efficiency, only skill metadata (names, purposes) reside in the base prompt; full implementations are dynamically retrieved when invoked. Complete skill specifications are cataloged in Supplementary Table.4.

Workflow assembly and execution. The agent constructs workflows through three phases: (i) Objective constraint: user prompts are contextualized with session parameters, species data (where available), and execution limits (e.g., hardware, runtime). (ii) Skill retrieval & planning: relevant skills are selected and ordered via explicit dependency graphs, ensuring input/output compatibility and procedural correctness. (iii) Stateful execution: a structured reason-act loop invokes skills sequentially, with the runtime environment providing real-time feedback on input availability, output validity, and execution logs.

The system enforces a fixed ten-stage sequence: 1. species-task routing → 2. preprocessing → 3. skull stripping → 4. ACPC alignment → 5. white-matter segmentation → 6. cortical reconstruction → 7. native-resolution rescaling → 8. FreeSurfer-to-GIFTI conversion → 9. surface-based analysis → 10. report generation. This sequence is encoded as the dependency graph among skills and cannot be extended at runtime. By trading open-ended flexibility for strict procedural boundaries, PrimateX guarantees the reproducibility and auditability required for scientific pipelines - attributes difficult to guarantee under fully dynamic LLM orchestration.

Agent specifications

PrimateX implements a multi-agent system comprising four specialized roles—user-proxy, planner, execution, and analysis - each governing a distinct subset of Agent Skills with rigorously defined input/output schemas. Agents operate under structured system prompts that enforce role-specific constraints on scope, decision logic, and output formatting. Inter-agent communication occurs via standardized JSON messages, ensuring machine-parsable audit trails.

A shared persistent memory maintains session state throughout the workflow, including normalized task specifications, current workflow plans, intermediate execution records, file manifests, and runtime observations from state transitions. This architecture eliminates redundant intent reprocessing, enables seamless role handoffs, and preserves end-to-end provenance.

User-proxy agent. The user-proxy agent translates free-form requests into structured task specifications. Its functions include: parsing natural language objectives, extracting explicit parameters (e.g., input paths, subject IDs, workspace), detecting missing required inputs, flagging species for automated detection when unspecified. Output: a normalized JSON task specification with fields: user_objective, confirmed_parameters, missing_parameters, execution_constraints, and historical_context_summary.

Planner agent. The planner agent maps task specifications to executable plans using the fixed 10- stage PrimateX workflow. Each stage is codified with: input/output requirements, parameter dependencies, and ordered execution constraints. Output: a structured workflow plan with fields: each carrying stage_name, stage_goal, stage_inputs, stage_outputs, and required_parameters; execution_order, species_normalization, and unresolved_blockers.

Execution agent. The execution agent operates as a deterministic state machine. States correspond to computational module invocations; transitions require validation against predefined criteria (e.g., file existence, non-empty masks, expected dimensionality). Upon module failure (malformed code, dependency errors), the system captures error tracebacks, passes context to the LLM for revision, and retries ≤ 8 times before logging failure. Output: a runtime status record with fields: current_stage, execution_status (pending, running, completed, or failed), generated_files, log_location, and runtime_observations.

Analysis agent. The analysis agent performs downstream surface-based analyses on reconstructed outputs, including geodesic/Euclidean surface metrics, surface-based network analysis. Its scope is limited to post-reconstruction quantification, although it does not modify upstream outputs. Output: analysis results with fields: analysis_target, input_surfaces, analysis_type, analysis_status, generated_results, and analysis_summary.

Agent execution walkthrough: cynomolgus macaque example

This section traces a complete execution for a cynomolgus macaque scan (subject ION001), detailing each stage from free-form request to final report. The process unfolds in three phases: initialization, ten-stage processing, and outcome synthesis, demonstrating how the agent architecture achieves reproducible neuroimaging analysis.

Agent initialization. The process begins when a user submits a free-form processing request ("perform structural preprocessing on this scan") accompanied by a T1-weighted image path. The user-proxy agent parses this input, extracting confirmed parameters including the input path, subject identifier, and workspace directory. Notably absent is the species designation, which

triggers the system's auto-detection protocol. The agent generates a structured JSON task specification containing the user's objective, confirmed parameters, missing parameters requiring resolution, execution constraints, and a summary of any relevant historical context from prior sessions.

This specification passes to the planner agent, which maps the task onto PrimateX's immutable ten-stage workflow. The planning process incorporates species-specific parameters drawn from the system's knowledge base. With the execution plan established, the execution agent initializes a deterministic state machine that will govern the subsequent processing. This state machine enforces strict transition rules between stages based on predefined validation criteria.

Stage 1. Species-task routing. The system evaluates the scan against five possible species categories (common marmoset[7], cynomolgus macaque, rhesus macaque, chimpanzee, and human) and determines skull presence (a binary classification). For subject ION001, the classifier identifies the scan as originating from a cynomolgus macaque and detecting skull inclusion, which directs downstream processing through the skull-stripping pathway.

Stage 2. Preprocessing. Standard preprocessing operations then commence, applying NHP-optimized corrections that include structural denoising and bias field adjustment. This is followed by spatial resampling to standardize voxel dimensions and spatial normalization for alignment to a common reference space, ensuring cross-subject comparability. The preprocessed volume is registered to the input catalog alongside the original scan to enable quality assessment.

Stage 3. Skull stripping. The system proceeds to skull stripping using a cynomolgus-specific deep learning segmentation network. This generates both a brain-extracted volume and a corresponding binary mask to subsequent stages.

Stage 4. ACPC alignment. Following successful skull removal, the pipeline performs automated cropping to eliminate non-brain structures in the inferior field of view. This is followed by anterior commissure-posterior commissure (ACPC) alignment through rigid registration to a species-specific reference frame.

Stage 5. White-matter segmentation. The resulting ACPC-aligned volume serves as input for white matter segmentation, where a dedicated network employing connectivity-aware loss functions produces the white matter mask. Quality control protocols evaluate mask quality through connected component analysis and uncertainty estimation.

Stage 6. Cortical surface reconstruction. Cortical reconstruction is initiated after volumetric registration to the species-matched NMT v2 anatomical space, with the CHARM or D99 atlases used for volumetric and subcortical parcellation. The pipeline then generates both white matter and pial cortical surfaces in native and standardized coordinate spaces.

Stage 7. Native-resolution rescaling. These surfaces then undergo native-resolution rescaling to restore original anatomical dimensions, with concurrent recomputation of vertex-wise morphometric properties including cortical thickness, sulcal depth, and curvature measures.

Stage 8. FreeSurfer-to-GIFTI conversion. For compatibility with downstream analysis tools, the system converts reconstructed surfaces and associated metrics to GIFTI format. Following surface reconstruction, cortical surfaces are independently registered to a species-appropriate

surface template (Yerkes19 for macaque specimens), followed by resampling to the fsLR32k standard surface space to facilitate cross-subject and cross-species comparisons.

Stage 9. Surface-based analysis. The analysis agent then performs quantitative surface-based measurements, computing geodesic and Euclidean distance maps alongside surface network topology descriptors. These analyses operate strictly on the final reconstructed representations without modifying source data.

Stage 10. Report generation. The system compiles all stage artifacts - including input/output file paths, execution status logs, runtime observations, and quality metrics - into a comprehensive participant-level report. This report documents topological evaluations of white matter masks and reconstructed surfaces, quantifying Euler numbers, defect counts, hole frequencies, and surface self-intersections as key quality indicators. The resultant documentation ensures complete auditability from initial request through final output generation.

Supplementary Note 2: MRI-style transfer network for data augmentation

Problem context. Multi-site NHP neuroimaging data exhibit substantial appearance variability arising from heterogeneous scanner hardware (field strength, gradient performance, coil geometry), acquisition parameters (resolution, TR/TE/TI, averaging), and animal positioning. While human neuroimaging has benefited from large multi-site initiatives like UK Biobank and ABCD that established traveling-phantom protocols and statistical harmonization methods (ComBat, CovBat)[8, 9], no equivalent standards exist for NHP imaging. The PRIME-DE consortium has documented this variability across 22+ sites[10], with the PRIME-RE community

identifying it as a critical barrier to cross-site reproducibility[11]. The PrimateX dataset reflects this diversity, incorporating scans from 1.5 T and 7.0 T systems (Bruker, Siemens, GE, and United Imaging) with coil configurations ranging from dedicated 8-channel phased arrays to repurposed 12-channel knee coils.

Rather than employing post-hoc harmonization[8, 9], which requires site labels during inference and risks over-correction of biologically relevant variance when covariates are incomplete[12], PrimateX adopts a synthetic augmentation approach. The system generates novel image appearances by transferring style characteristics between real scans while preserving anatomical content, effectively expanding the appearance space encountered during training. This strategy promotes learning of site-invariant features without compromising biological variability. While contrast-agnostic methods like SynthSeg[13], SynthMorph[14], and SynthStrip[15] sample tissue intensities from random Gaussian distributions, PrimateX's transfer mechanism grounds the augmented data in empirically observed scanner appearances from actual primate cohorts, maintaining fidelity to real-world acquisition conditions.

Architecture design. The architecture of PrimateX addresses a fundamental limitation of standard slice-by-slice style transfer methods, which often produce inter-slice discontinuities that manifest as banding artifacts in coronal and sagittal reformats when reassembled into 3D volumes. To preserve inter-slice anatomical continuity, the network processes each slice within the context of its neighbors by taking as input a three-channel stack of adjacent slices [k-1, k, k+1] for target slice k. Following the StyTr2 framework[2 16], the architecture employs separate Transformer-based encoders for processing content and style representations. Each input slice is first partitioned into non-overlapping patches that are linearly projected into token

embeddings before being processed by their respective encoder. Notably, the content encoder incorporates a Content-aware Positional Encoding (CPE)[16] module that robustly encodes spatial relationships, a critical feature given the wide range of native voxel spacings in non-human primate data (0.27mm for common marmoset to 0.9mm for chimpanzee).

The architecture then uses a Transformer-based decoder to integrate content and style representations through cross-attention, where content embeddings serve as queries while style embeddings provide keys and values. The decoder output is projected through a convolutional reconstruction head to produce the final style-transferred 2D slice. Importantly, only the central slice (k) is retained from this process, with all transformed slices being reassembled along the original slice axis while preserving the volumetric integrity through maintenance of the original spatial metadata including orientation, voxel dimensions, and the affine transform.

Loss function. The style transfer objective combines a content fidelity term $L_{content}$ with a style matching term L_{style}, weighted by coefficients λ_c and λ_s, respectively:

$$L_{style\text{-}trans}=\lambda_c L_{content}+\lambda_s L_{style}$$

where $L_{content}$ enforces structural preservation, L_{style} governs appearance transfer, $\lambda_c=15$ and $\lambda_s=8$.

Content fidelity ($L_{content}$) measures the perceptual difference between the generated output slice G and the original content slice C through an ImageNet-pretrained VGG-19 network feature extractor[17]:

$$L_{content}=\sum_{l\in L}\left\|\phi_l(G)-\phi_l(C)\right\|_2^2$$

Style matching (L_{style}) adapts the style from the reference slice S by aligning the channel-wise mean ($\mu(\cdot)$) and standard deviation ($\sigma(\cdot)$) of feature activations in an Adaptive Instance Normalization (AdaIN) formulation[18]:

$$L_{style}=\sum_{l\in L}\left(\left\|\mu\left(\phi_l(G)\right)-\mu\left(\phi_l(S)\right)\right\|_2^2+\left\|\sigma\left(\phi_l(G)\right)-\sigma\left(\phi_l(S)\right)\right\|_2^2\right)$$

The feature extractor $\phi_l(\cdot)$ uses intermediate layers of a pretrained VGG-19 network, with the selected layer set $\mathscr{L}$={2,3,9,11,17} corresponding to conv1_2, relu1_2, pool2, relu3_1 and relu3_4 in PyTorch's implementation. These layers capture low-level textural and contrast features (e.g., noise patterns, intensity distributions) while avoiding deeper semantic representations (conv4/conv5) that are biased toward natural image classification and poorly transfer to MRI. Unlike the original Gram-matrix formulation[17], we adopt AdaIN-style statistics[18] to maintain independence from spatial dimensions - critical for handling varying resolutions. The choice of $\lambda_c > \lambda_s$ prioritizes anatomical fidelity over stylistic alteration, reversing the standard Gatys trade-off to ensure medically interpretable outputs. Both coefficients were empirically determined via grid search on a held-out validation set and kept consistent across all species.

Training. The network was trained with the Adam optimizer with an initial learning rate 5×10^{-4}, a batch size of 1, and up to 80,000 iterations. Volumes were processed along the axial (z-axis) direction to preserve anatomical continuity while computing the loss solely for the central slice - neighboring slices served only as contextual regularizers without directly contributing to gradient updates. To ensure the downstream segmentation network robustly handles cross-site variability, we dynamically sampled style references from a curated pool of training-set scans

representing each acquisition site. This pool strictly excluded held-out test volumes (never used as content or style inputs) and external data from the PRIME-DE and NCBR cohorts (Extended Data Tables 2 - 3), ensuring no leakage during model selection or evaluation. Notably, style transfer was applied exclusively to non-human primate data, as human HCP scans retained their native acquisition appearance due to their standardized protocol, which inherently avoids cross-site harmonization needs.

Supplementary Note 3: Segmentation network architecture

Problem context. Volumetric brain segmentation requires reconciling two competing demands: precise localization of fine anatomical boundaries (e.g., tissue interfaces) and integration of long-range spatial context (e.g., whole-brain structural relationships). This challenge is compounded in cross-site and cross-species settings, where heterogeneity in image acquisition and anatomy amplifies the risk of model under-specification. Conventional architectures struggle to balance these requirements efficiently - convolutional approaches excel at local feature extraction but scale poorly to global interactions, while Transformer-based methods impose prohibitive computational costs for high-resolution 3D data.

PrimateX addresses these limitations through a U-shaped encoder-decoder architecture whose design is dynamically tailored to dataset properties via the nnU-Net framework[19]. The encoder employs a hybrid strategy to resolve the local-global trade-off: in early stages, Hierarchical Gated Convolution Network (HGCN) modules preserve fine-scale spatial relationships critical for boundary delineation, while deeper stages deploy Mixture-of-Experts

Mamba (MoE-Mamba) blocks to model cross-region dependencies. This latter component synergizes two recently established paradigms - conditional computation through expert routing[20] and linear-complexity state-space sequence modeling[21] - to efficiently capture global context without sacrificing adaptability to input variability.

The MoE-Mamba synthesis draws inspiration from language modeling (e.g., BlackMamba[22]) but adapts it to neuroimaging's unique constraints. By routing tokens through specialized experts conditioned on latent features, the model dynamically allocates capacity to handle site-specific artifacts or interspersed morphological patterns, paralleling the success of personalized MoE in brain tumor segmentation (P-MoLE[20]). Crucially, state-space operations maintain tractable memory usage for whole-brain volumes, avoiding the quadratic overheads of self-attention. Cross-species differences are managed at the pipeline level via separate networks (see Methods), allowing the MoE components to focus on intra-species and acquisition-related variance. Together, this hierarchical specialization - from local HGCN processing to globally aware MoE-Mamba transformations - enables robust segmentation across heterogeneous datasets without compromising computational feasibility.

Architectural formulation. The encoder follows a hierarchical strategy - early stages use HGCN blocks (stacked gated convolutions) for local structural modeling, while deeper stages employ MoE-Mamba blocks for long-range context. Each MoE-Mamba layer integrates condition computation and state-space efficiency. A two-layer MLP gating network ($w=\mathrm{Gate}(x)\in R^K$) selects among K experts (K=3), adapting to input variations:

$$y=\sum_{k=1}^{K} w_k \cdot \mathrm{Expert}_k(x)$$

where $\mathrm{Expert}(x)$ denotes the state-space transform. The Mamba operation provides $O(n)$ complexity for a volume of n voxels (compared with $O(n^2)$ for self-attention) while maintaining long-range spatial context that is critical for species with large brain volumes (A 0.7 mm isotropic MRI scan of the whole human brain encodes data in ~3.8M voxels).

De-duplication regularizer. To prevent expert collapse (where multiple experts converge to identical functions, negating the benefit of conditional gating), a de-duplication regularizer penalizes pairwise expert similarity:

$$L_{div} = \frac{2}{K(K-1)} \sum_{i<j} \left(\cos\left(\theta_i, \theta_j\right)\right)^2$$

where θ_i denotes the parameters of expert i and $\cos(\cdot,\cdot)$ is cosine similarity computed over flattened parameter vectors. Minimizing L_{div} drives experts toward distinct spatial and contextual representations.

Emergent specialization. Expert utilization is not explicitly via hard load-balancing losses (as in Switch Transformer[23] or GShard[24]). Instead, expert usage is governed by data-dependent routing and gradient flow during training. In practice, this leads to heterogeneous expert utilization, where experts receiving consistently low routing mass undergo limited parameter updates. Combined with routing and diversity regularization, this behavior encourages functional specialization while preventing complete collapse, allowing the model to adapt its effective capacity to species- and sample-dependent anatomical complexity.

Supplementary Note 4: Connectivity-aware white-matter refinement

Problem context. Standard voxel-wise segmentation losses (e.g., Dice, cross-entropy) treat each voxel independently, failing to penalize structural fragmentation. This is especially problematic in white-matter segmentation, where thin tracts (e.g., the vertical occipital fasciculus, anterior commissure) may be correctly classified voxel-wise but remain topologically disconnected - resulting in anatomically implausible segmentations.

Here PrimateX introduces a connectivity-aware refinement strategy that explicitly enforces anatomical coherence in white-matter segmentation. Building on topology-aware frameworks like clDice[25] (which enforces centerline agreement) and skeleton recall loss[26] (which prioritizes under-represented structures), our approach adapts these principles to 3D white-matter segmentation, incorporating species-specific anatomical considerations to optimize connectivity preservation.

Skeleton-informed loss. Soft skeletons for both the predicted probability map (s_p) and ground truth mask (s_γ) are computed using an iterative min-/max-pooling procedure that approximates morphological thinning while maintaining differentiability[25]. Implementation employs a maximum of T = 10 iterations, with early termination triggered when the mean change between successive skeleton estimates falls below $\epsilon=10^{-4}$, ensuring computational efficiency without compromising skeleton quality. Connectivity supervision is then enforced through a skeleton-guided objective:

$$L_{skel}=1-\frac{2\,T_{prec}(s_p,y)\,T_{sens}(s_y,p)}{T_{prec}(s_p,y)+T_{sens}(s_y,p)}$$

$$T_{prec}(s_p,y)=\frac{\sum(s_p\odot y)+\epsilon}{\sum s_p+\epsilon}\qquad T_{sens}(s_y,p)=\frac{\sum(s_y\odot p)+\epsilon}{\sum s_y+\epsilon}$$

where $\sum$ denotes summation over all voxels; $\odot$ denotes the element-wise (Hadamard) product; and ϵ is a small constant for numerical stability. L_{skel} encourages the predicted skeleton to lie within the reference mask and the reference skeleton to be covered by the prediction, thereby penalizing fragmented or disconnected white-matter predictions.

Our loss combines standard volumetric segmentation terms (Dice/Tversky) with a skeleton-aware component (L_{skel}). While the former optimizes tissue overlap, L_{skel} explicitly preserves white-matter topology by penalizing discontinuities in centerline predictions. We adapt the clDice framework to 3D neuroimaging by replacing its 2D morphological operations with differentiable 3D min-/max-pooling, enabling direct volume-wise skeletonization during training.

Uncertainty-guided resampling. Model confidence varies across training samples - volumes with high certainty provide limited gradients for refining boundary detection and thin WM-tract segmentation. To prioritize informative cases, PrimateX employs adaptive resampling weighted by prediction uncertainty, sharpening focus on ambiguous regions critical for connectivity accuracy. Given a training volume x, let Ω_x denote the set of all voxels in x, $v \in \Omega_x$ denote a voxel index, and $\hat{P}(v) \in [0,1]$ denote the predicted WM probability at voxel v. We defined (i) target voxels:

$$T_x = \{v \mid v \in \Omega_x, \hat{P}(v) \geq 0.5\}$$

and (ii) boundary voxels:

$$B_x = \{v \mid v \in \Omega_x, |\hat{P}(v) - 0.5| < \delta\}$$

where $\delta = 0.1$ denotes the half-width of the decision boundary band.

Voxel-wise uncertainty was quantified via predictive entropy:

$$H(v)=-\sum_{c\in\{0,1\}}\hat{P}_c(v)\log\left(\hat{P}_c(v)+\epsilon\right)$$

where $\hat{P}_c(v)$ is the predicted probability that voxel v belongs to class c (WM or background), and $\epsilon=10^{-10}$ is a small constant for numerical stability.

For each training volume, per-sample uncertainty was computed separately for the target and boundary streams:

$$U_T(x)=\frac{1}{|T_x|}\sum_{v\in T_x}H(v)\qquad U_B(x)=\frac{1}{|B_x|}\sum_{v\in B_x}H(v)$$

where larger values indicate greater predictive uncertainty. The target stream ($U_T(x)$) was designed to capture thin-tract uncertainty while the boundary stream ($U_B(x)$) was designed to capture partial-volume and species-specific contrast differences ambiguity. The most uncertain samples (top 20%) were identified for the target and boundary streams, respectively. Samples are dynamically reweighted by their selection frequency: (i) Normal: unselected in either stream (1×weight); (ii) Single-stream: top 20% uncertain in either target or boundary stream (5×); (iii) Dual-stream: top 20% in both streams (10×). This strategy concentrates gradients on challenging WM patterns while preserving generalization by retaining all samples.

Training objectives. The segmentation network is optimized via a multi-task objective function that integrates voxel-wise classification accuracy, boundary preservation, and cross-species anatomical consistency.

(i) Skull-stripping objective: $L_{skull}=L_{seg}+\lambda_{MoE}L_{MoE}+L_{div}+\lambda_{ds}L_{ds}$

where L_{seg} combines Dice loss and regional Tversky loss. While deep supervision employed geometrically decaying weights with normalized $1/2^i$ scaling across decoder levels (λ_{ds}). The Dice loss is defined as:

$$L_{Dice}=1-\frac{2\sum(p\cdot y)+\epsilon}{\sum(p^2)+\sum(y^2)+\epsilon}$$

After the volume is partitioned into eight octant sub-volumes, the regional Tversky loss is defined as:

$$L_{Tversky}=1-\frac{\sum(p\cdot y)+\epsilon}{\sum(p\cdot y)+\alpha\sum(p\cdot \bar{y})+\beta\sum(\bar{p}\cdot y)+\epsilon}$$

where $\bar{p}=1-p$ and $\bar{y}=1-y$, with base $\alpha=0.3$, $\beta=0.4$, and adaptive region-wise weights are computed from local false-positive/false-negative statistics. This regional decomposition ensures that anatomically challenging regions (e.g., orbital frontal cortex where NHP extracranial musculature confounds boundary detection, inferior brain near the foramen magnum) receive appropriate loss weighting even when they constitute a small fraction of total brain volume.

L_{MoE} regularizes the gating distribution toward a uniform prior via KL divergence[27]:

$$L_{MoE}=KL(U\ //\ \bar{\mathbf{w}})=\sum_k \frac{1}{K}\cdot\log\left(\frac{1/K}{\bar{w}_k+\epsilon}\right),\qquad \bar{\mathbf{w}}=\frac{1}{B}\sum_{b=1}^{B}\mathbf{w}^{(b)}$$

where $\bar{w}$ is the gating distribution averaged across the B samples in a minibatch and $w^{(b)}$ is the per-sample soft routing distribution. Regularizing the batch-average (rather than each per-sample $w^{(b)}$) during the early stages of training is essential: it encourages all K experts to receive non-negligible aggregate utilization during training while leaving each individual routing decision free to specialize. The KL regularization weight is gradually reduced during

training, recovering the load-balancing intent[27, 28] without enforcing the hard top-k constraint of Switch Transformer or GShard. Combined with L_{div}, it promotes both expert diversity and balanced utilization without removing data-dependent specialization.

L_{div} is a de-duplication regularizer that penalizes pairwise cosine similarity between expert parameter representations, preventing expert collapse without enforcing explicit load balancing. L_{ds} is the deep-supervision term that applies the segmentation objective at each decoder scale, with supervision weights decaying exponentially from fine to coarse, thereby prioritizing fine-resolution structural learning.

(ii)White-matter segmentation objective: $L_{WM}=L_{Dice}+L_{Tversky}+\lambda_{MoE}L_{MoE}+L_{div}+\lambda_{ds}L_{ds}+L_{skel}$

where L_{skel} enforces skeleton-informed connectivity preservation, extending the skull-stripping objective with white-matter tract continuity constraints. The regional Tversky loss used $\alpha = 0.3$ and $\beta = 0.4$, while deep supervision employed geometrically decaying weights with normalized $1/2^i$ scaling across decoder levels (λ_{ds}).

Optimization. All networks were optimized using Adam (learning rate: 1×10^{-4}, weight decay: 3×10^{-7}) for up to 300 epochs. We implemented a dual-network architecture per species - one dedicated to skull stripping and another for white-matter segmentation - resulting in 10 distinct parameter sets (5 species × 2 tasks). Input volumes were preprocessed as described in Methods (§Volume preprocessing). For all data, we applied nnU-Net[19] spatial augmentation scheme during training. NHP data additionally received offline style-transfer augmentation (Supplementary Note 2) to account for cross-site variance, while human data (from the protocol-

standardized HCP) bypassed this step, though still undergoing standard online spatial/intensity transformations.

Supplementary Table 1 | Quantitative assessment of skull-stripping detection and species differentiation in five species

Task	Species	n	Balanced ACC	AUROC	F1	Recall
Species classification	Common marmoset	292	0.990 ± 0.004	1.000 ± 0.000	0.990 ± 0.004	0.980 ± 0.009
	Cynomolgus macaque	124	0.989 ± 0.006	1.000 ± 0.000	0.964 ± 0.012	0.983 ± 0.012
	Rhesus macaque	566	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	Chimpanzee	12	0.957 ± 0.041	0.999 ± 0.001	0.872 ± 0.075	0.915 ± 0.083
	Human	400	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
Task classification	With skull	713	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	Without skull	713	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

Supplementary Table 2 | Performance benchmarks for skull-stripping and white-matter segmentation pipelines in five species. Sub-tables correspond to (a) common marmoset, (b) cynomolgus macaque, (c) rhesus macaque, (d) chimpanzee and (e) human. Bold values indicate the best-performing method, and * indicates $p < 0.05$ for comparisons between PrimateX and the best-performing competing method for each metric.

(a) Common marmoset

Task	Methods	Dice	HD95	ASSD	NSD@0.5 mm	Speed
Skull stripping	FSL	0.779±0.043	5.223±1.296	1.571±0.301	0.241±0.024	**0.219±0.001**
	MedSAM2	0.808±0.035	6.640±1.687	1.677±0.346	0.371±0.049	0.285±0.047
	nBEST	0.964±0.013	0.766±0.240	0.261±0.092	0.875±0.089	2.858±0.034
	DeepBet	0.973±0.005	0.556±0.088	0.196±0.034	0.938±0.024	0.778±0.031
	PrimateX	**0.979±0.009**	**0.481±0.146**	**0.161±0.065**	**0.948±0.033**	0.618±0.059
WM segmentation	FSL	0.929±0.018	0.548±0.142	0.155±0.029	0.950±0.013	85.361±2.559
	MedSAM2	0.673±0.023	3.824±0.170	0.707±0.017	0.741±0.017	4.813±0.310
	nBEST	0.711±0.033	3.785±0.067	0.622±0.027	0.815±0.019	35.215±1.112
	DeepBet	0.910±0.008	0.362±0.072	0.154±0.010	**0.980±0.007**	6.051±0.251
	PrimateX	**0.944±0.030**	**0.331±0.062**	**0.094±0.050**	0.980±0.017	**3.608±0.817**

(b) Cynomolgus Macaque

Task	Methods	Dice	HD95	ASSD	NSD@0.5 mm	Speed
Skull stripping	FSL	0.687±0.120	15.603±9.758	5.143±2.739	0.207±0.094	**0.110±0.011**
	MedSAM2	0.934±0.021	5.035±2.726	1.229±0.469	0.471±0.109	0.258±0.116
	nBEST	0.970±0.008	0.982±0.238	0.444±0.105	0.695±0.140	1.381±0.248
	DeepBet	0.978±0.004	0.852±0.159	0.332±0.055	0.803±0.092	0.337±0.073
	PrimateX	**0.984±0.004***	**0.616±0.148***	**0.252±0.060***	**0.881±0.064***	0.732±0.047
WM segmentation	FSL	0.842±0.029	1.831±0.403	0.475±0.088	0.749±0.109	61.632±6.657
	MedSAM2	0.681±0.033	4.442±0.850	0.981±0.112	0.557±0.090	**1.795±0.224**
	nBEST	0.809±0.013	1.656±0.365	0.481±0.035	0.774±0.096	20.519±2.096
	DeepBet	0.871±0.010	0.818±0.117	0.308±0.022	0.786±0.100	2.939±0.433
	PrimateX	**0.932±0.012***	**0.500±0.000***	**0.156±0.020***	**0.922±0.084***	2.233±0.418

(c) Rhesus Macaque

Task	Methods	Dice	HD95	ASSD	NSD@0.5 mm	Speed
Skull stripping	FSL	0.518±0.041	37.940±3.793	11.935±1.536	0.104±0.020	**0.065±0.014**
	MedSAM2	0.493±0.028	44.406±8.572	16.059±2.935	0.017±0.012	0.105±0.006
	nBEST	0.969±0.002	1.044±0.061	0.513±0.025	0.689±0.031	0.547±0.008
	DeepBet	0.984±0.001	0.655±0.096	0.291±0.011	0.939±0.014	0.112±0.005
	PrimateX	**0.994±0.004***	**0.500±0.000***	**0.104±0.076***	**0.990±0.013***	0.710±0.004
WM segmentation	FSL	0.881±0.014	1.067±0.143	0.254±0.026	0.887±0.018	139.544±7.236
	MedSAM2	0.709±0.061	10.979±9.421	1.851±0.802	0.800±0.035	2.410±0.212
	nBEST	0.794±0.020	6.406±0.838	0.896±0.136	0.863±0.018	18.677±0.852
	DeepBet	0.834±0.020	0.767±0.082	0.355±0.025	0.877±0.021	2.531±0.145
	PrimateX	**0.956±0.024***	**0.500±0.000***	**0.087±0.041***	**0.982±0.009***	**1.977±0.192**

(d) Chimpanzee

Task	Methods	Dice	HD95	ASSD	NSD@0.5 mm	Speed
Skull stripping	FSL	0.780±0.343	25.192±5.187	10.876±18.960	0.134±0.095	**0.126±0.077**
	MedSAM2	0.532±0.021	58.732±6.765	19.897±20.355	0.027±0.005	0.450±0.190
	nBEST	0.942±0.004	2.970±0.340	1.588±0.118	0.038±0.022	0.871±0.062
	DeepBet	0.986±0.002	1.086±0.215	0.441±0.436	0.561±0.059	0.218±0.008
	PrimateX	**0.990±0.002**	**0.900±0.000**	**0.332±0.065**	**0.675±0.059**	0.703±0.172
WM segmentation	FSL	0.835±0.114	4.451±4.564	0.953±0.713	0.473±0.191	58.475±5.542
	MedSAM2	0.431±0.213	42.049±22.889	14.578±7.400	0.001±0.001	5.081±0.553
	nBEST	0.742±0.082	6.160±4.304	1.458±0.597	0.355±0.112	19.888±0.429
	DeepBet	0.840±0.049	3.983±3.771	0.889±0.436	0.439±0.049	**2.672±0.267**
	PrimateX	**0.920±0.023**	**1.086±0.215**	**0.352±0.149**	**0.704±0.028**	2.969±0.245

(e) Human

Task	Methods	ICC(2,1)	ICC(3,1)	Pearson's	SSIM	Speed
Skull stripping	FSL	0.998	0.999	0.999	0.991±0.002	0.062±0.003
	PrimateX					0.896±0.063
WM segmentation	FSL	0.998	0.999	0.999	0.965±0.003	35.357±2.509
	PrimateX					1.960±0.277

Supplementary Table 3 | Comparative performance assessment of cortical surface reconstruction pipelines in seen-domain data. Bold values indicate the best-performing method, and * indicates $p < 0.05$ for comparisons between PrimateX and the best-performing competing method for each metric.

Species	Methods	Euler Number	Defects	Holes
Common marmoset	FreeSurfer	-53.100±30.274	55.100±30.274	27.550±15.137
	NHPPipeline	-28.700±16.968	30.700±16.968	15.350±8.484
	PrimateX	**-27.200±17.464**	**29.200±17.464**	**14.600±8.732**
Macaque	FreeSurfer	-114.158±19.686	116.158±19.686	58.079±9.843
	NHPPipeline	-41.421±13.379	43.421±13.379	21.711±6.690
	PrimateX	**-10.316±5.493***	**12.316±5.493***	**6.158±2.746***
Chimpanzee	FreeSurfer	-46.286±17.748	48.286±17.748	24.143±8.874
	NHPPipeline	-26.143±19.317	28.143±19.319	14.071±9.659
	PrimateX	**-15.000±8.218***	**17.000±8.218***	**8.500±4.109***

Supplementary Table 4 | Skills implemented by four specialized agents in PrimateX

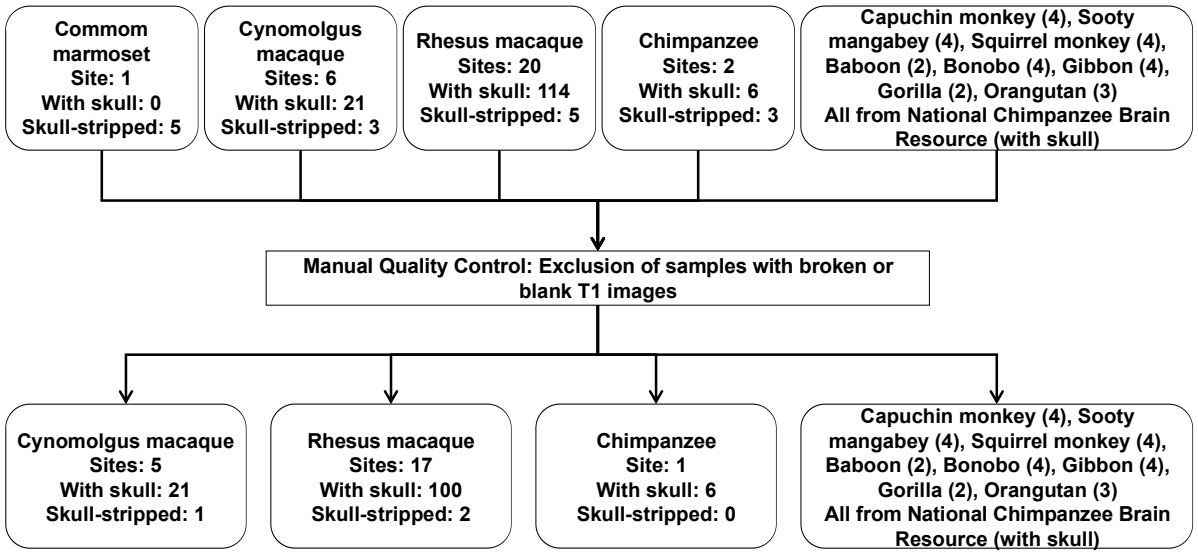
Agent	Skill name	Associated stage	Functional role	Inputs	Outputs
User proxy agent	Query normalization	Agent preamble	Converts free-form processing requests into workflow-compatible structured tasks	Natural-language request	Structured task description
	Objective identification	Agent preamble	Identifies the requested neuroimaging objective (for example preprocessing, segmentation, or full reconstruction)	User request	Workflow objective label
	Parameter extraction	Agent preamble	Extracts confirmed workflow parameters including image path, subject identifier, and workspace directory	User request; interaction history	Parameter
	Missing-parameter detection	Agent preamble	Detects unresolved parameters required for execution	Structured task representation	Missing-parameter feedback
Planner agent	Species-routing delegation	Stage 1	Delegates unresolved species identification to the species-task router	Input T1-weighted image	Species-routing request
	Task-specification serialization	Agent preamble	Emits normalized machine-readable task specifications for downstream orchestration	Validated workflow parameters	Structured JSON task specification
	Canonical pipeline decomposition	Agent preamble	Maps workflow objectives onto the ten-stage canonical PrimateX execution graph	Structured task specification	Ordered execution graph
	Dependency-aware scheduling	Agent preamble	Resolves stage dependencies and execution ordering constraints	Workflow graph	Deterministic stage ordering
	Species-specific parameterization	Stages 1–8	Assigns species-matched preprocessing, segmentation, and atlas parameters	Species label	Species-specific execution parameters
	Skull-state branch planning	Stage 1	Determines whether skull stripping is required based on router predictions	Skull-state prediction	Conditional execution branch
	Resource allocation planning	Stages 2–10	Associates workflow stages with required modules, models, and runtime resources	Execution graph	Stage-resource mappings

	Stage I/O specification	Stages 2–10	Defines explicit stage-level input/output dependencies	Workflow stages	Structured stage interfaces
	Failure-blocker detection	Agent preamble	Identifies unresolved execution blockers before runtime	Planned workflow	Blocker feedback
	Species-task routing execution	Stage 1	Executes species and skull-state classification on the input T1-weighted volume	Input T1-weighted image	Species prediction; skull-state prediction
	Preprocessing orchestration	Stage 2	Executes structural MRI preprocessing including reorientation, bias correction, denoising, and resampling	Raw T1-weighted image	Preprocessed T1-weighted image
	Skull-stripping execution	Stage 3	Executes species-specific brain extraction and brain mask generation	Preprocessed T1-weighted image	Brain-extracted image; brain mask
	ACPC-alignment execution	Stage 4	Performs ACPC alignment and field-of-view normalization	Skull-stripped image	ACPC-aligned image
	White-matter segmentation execution	Stage 5	Executes connectivity-aware white-matter segmentation	ACPC-aligned image	White-matter mask
	Cortical reconstruction orchestration	Stage 6	Coordinates cortical surface reconstruction using species-matched atlases	White-matter segmentation outputs	White and pial cortical surfaces
Execution agent	Native-space rescaling	Stage 7	Restores reconstructed geometry to native spatial resolution	Reconstruction outputs	Native-resolution surfaces and volumes
	Morphometric recomputation	Stage 7	Recomputes cortical thickness, curvature, and sulcal-depth measurements after rescaling	Native-resolution surfaces	Vertex-wise morphometric metrics
	Surface-format conversion	Stage 8	Converts reconstructed surfaces and metrics into GIFTI-compatible representations	Surface reconstructions	GIFTI surface outputs
	Spherical registration and resampling	Stage 8	Registers surfaces to species-specific templates and resamples to fsLR32k space	GIFTI surfaces	Standardized fsLR32k surfaces
	Runtime-state tracking	Stages 1–10	Records execution timestamps, execution status, and runtime observations at each state transition	Runtime execution events	Structured execution logs
	Intermediate-output verification	Stages 1–10	Verifies expected intermediate outputs after	Stage outputs	Verification status

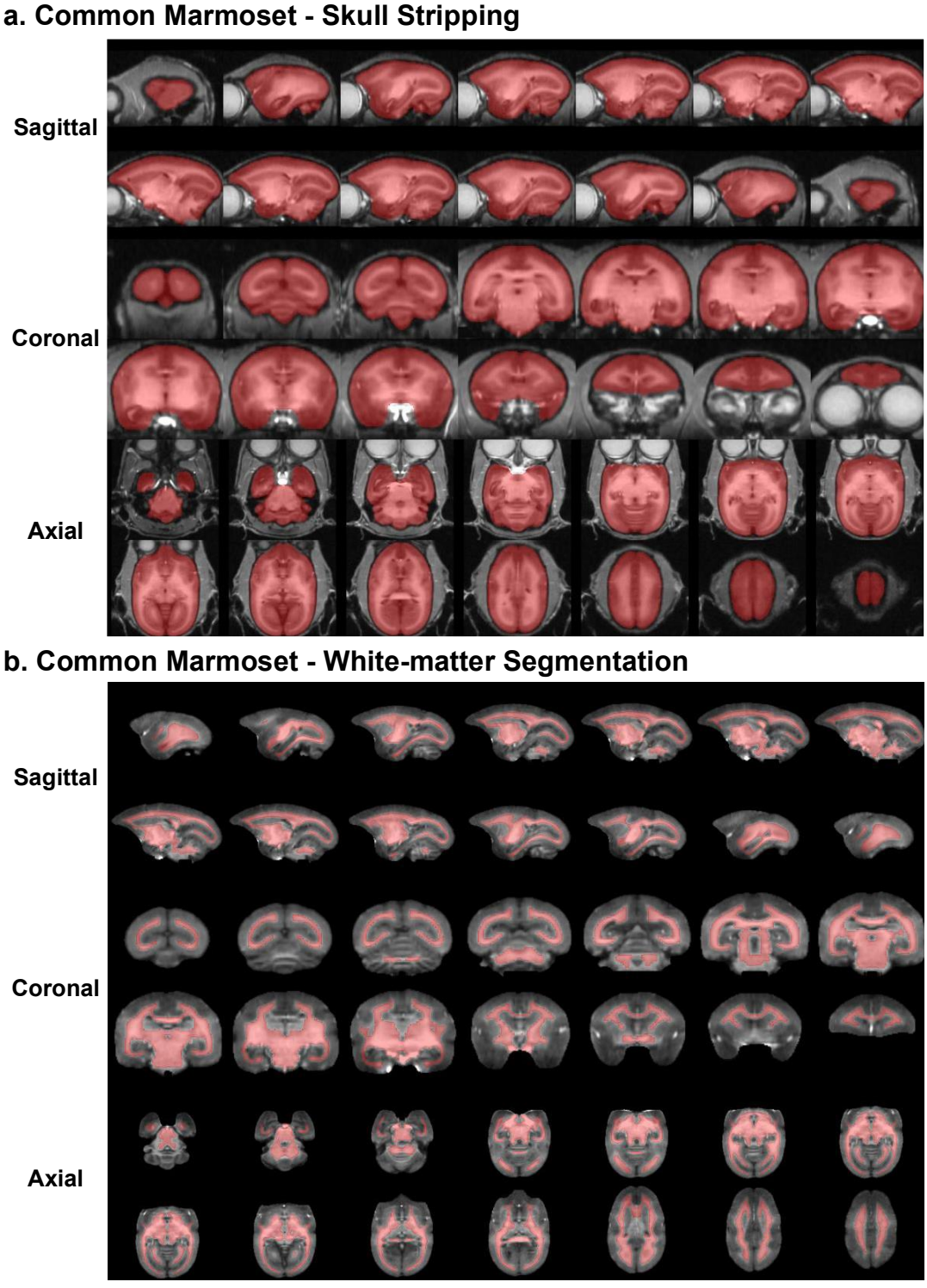
			each stage	
	Quality-flag logging	Stages 3–10	Records stage-specific quality-control metrics and warnings	Stage outputs	Quality-control logs
	Filesystem-state management	Stages 1–10	Maintains standardized subject-level workspace organization	Execution outputs	Organized output hierarchy
	Geodesic-distance analysis	Stage 9	Computes geodesic distance measurements on cortical surface manifolds	Reconstructed surfaces	Geodesic-distance maps
	Euclidean-distance analysis	Stage 9	Computes Euclidean spatial-distance measurements on reconstructed surfaces	Reconstructed surfaces	Euclidean-distance maps
	Surface-network analysis	Stage 9	Computes surface-based network descriptors and graph-level metrics	Surface representations	Network-analysis outputs
	Deterministic result generation	Stage 9	Generates reproducible downstream quantitative outputs from reconstructed surfaces	Surface-analysis tasks	Structured quantitative outputs
Analysis agent	Participant-level report aggregation	Stage 10	Consolidates per-stage artefacts, runtime metadata, and quantitative metrics into participant-level reports	Workflow outputs; QC metrics	Integrated participant report
	Topological quality evaluation	Stage 10	Computes topological quality-control metrics including Euler number, defect count, hole count, and surface self-intersections	White-matter masks; reconstructed surfaces	Surface-topology quality metrics
	Provenance summarization	Stage 10	Summarizes execution traces, runtime observations, and stage-level provenance metadata	Runtime logs	Auditable execution summary

1

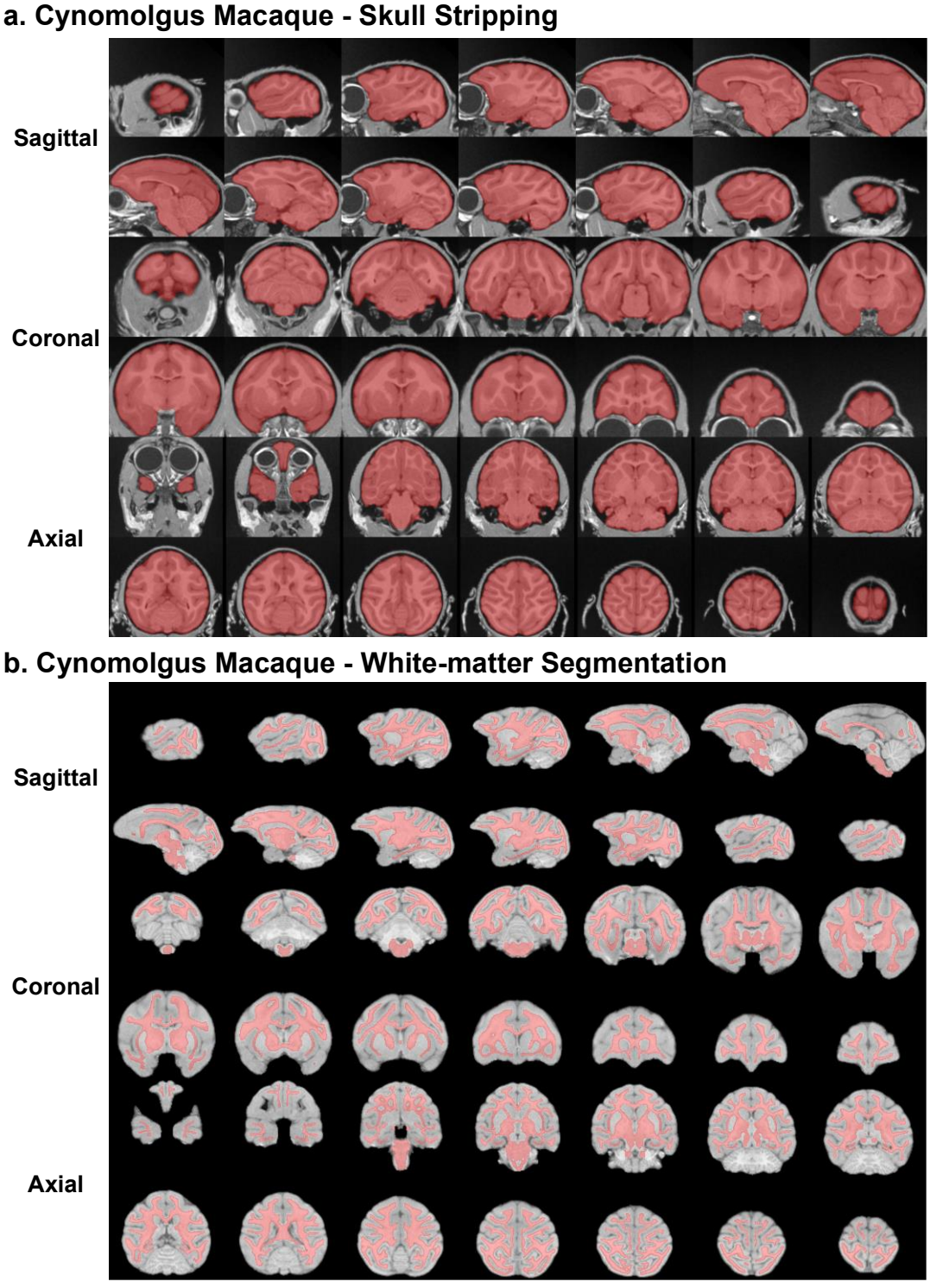
Supplementary Fig 1 | Data curation and quality control for multi-source NHP T1-weighted MRI datasets



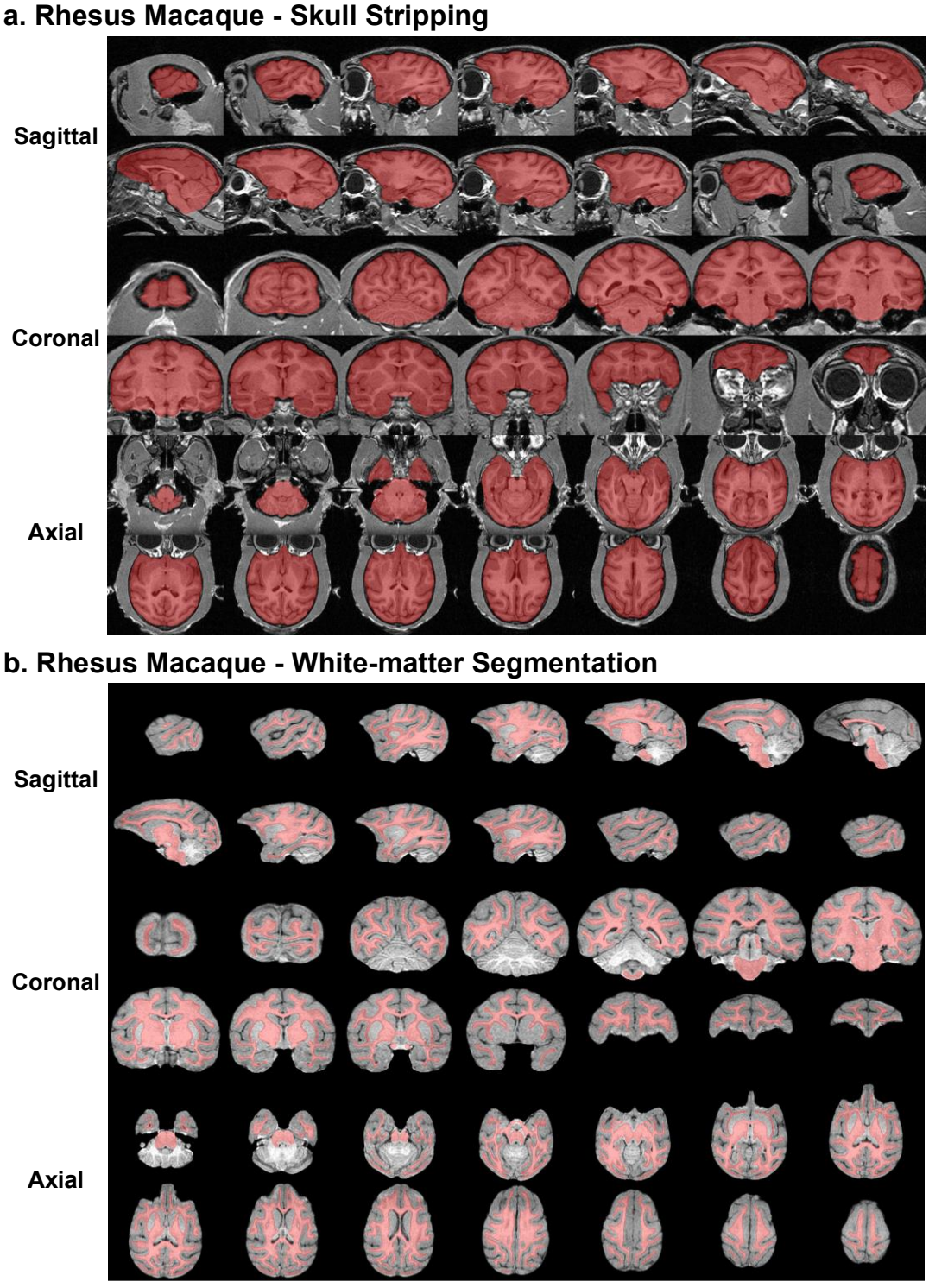
Supplementary Fig 2 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (common marmoset) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.



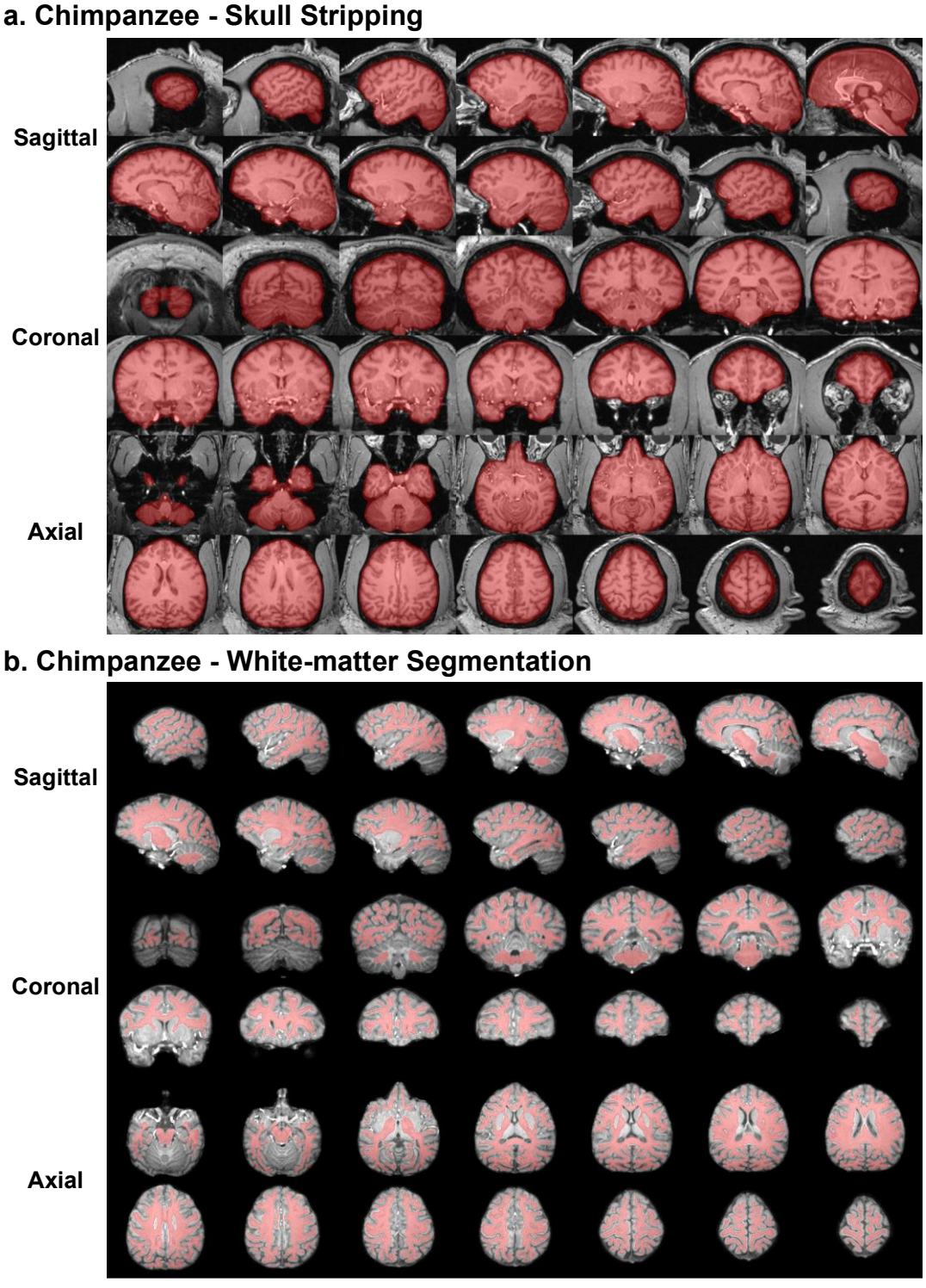
Supplementary Fig 3 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (cynomolgus macaque) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.



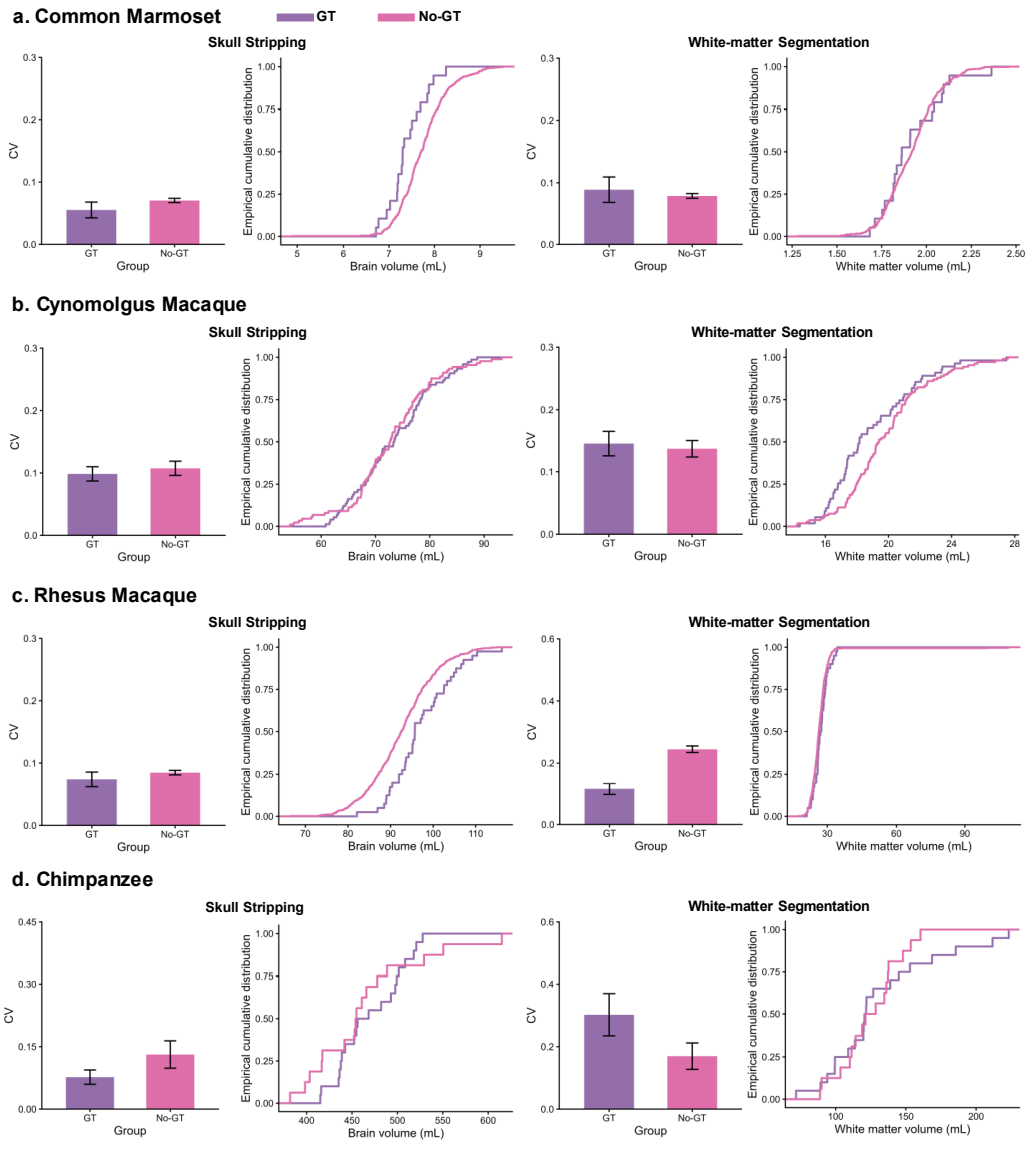
Supplementary Fig 4 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (rhesus macaque) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.



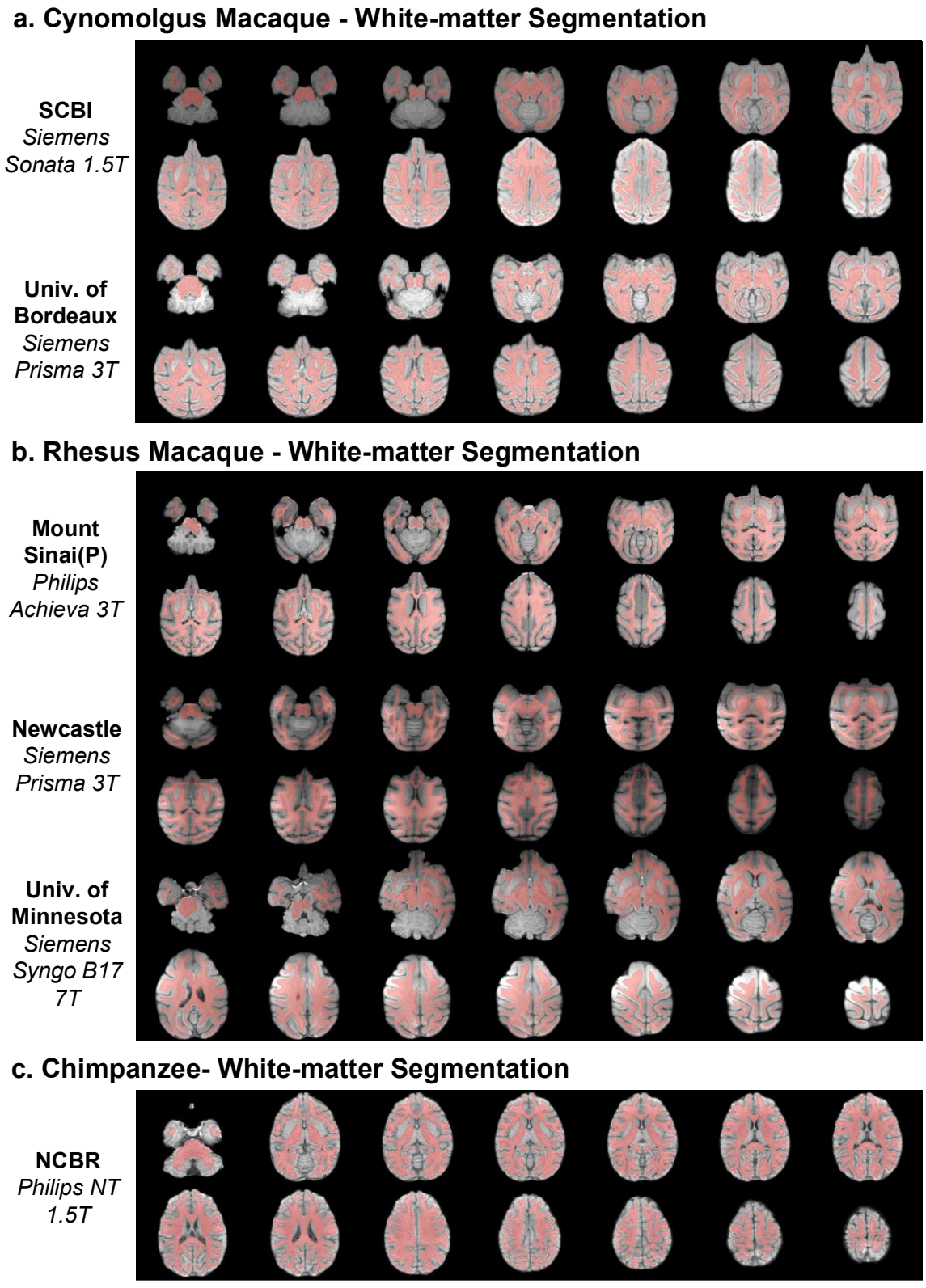
Supplementary Fig 5 | Qualitative assessment of segmentation performance on unannotated seen-domain datasets. Representative examples (chimpanzee) demonstrate model performance under real-world acquisition conditions where manual segmentation labels are unavailable.



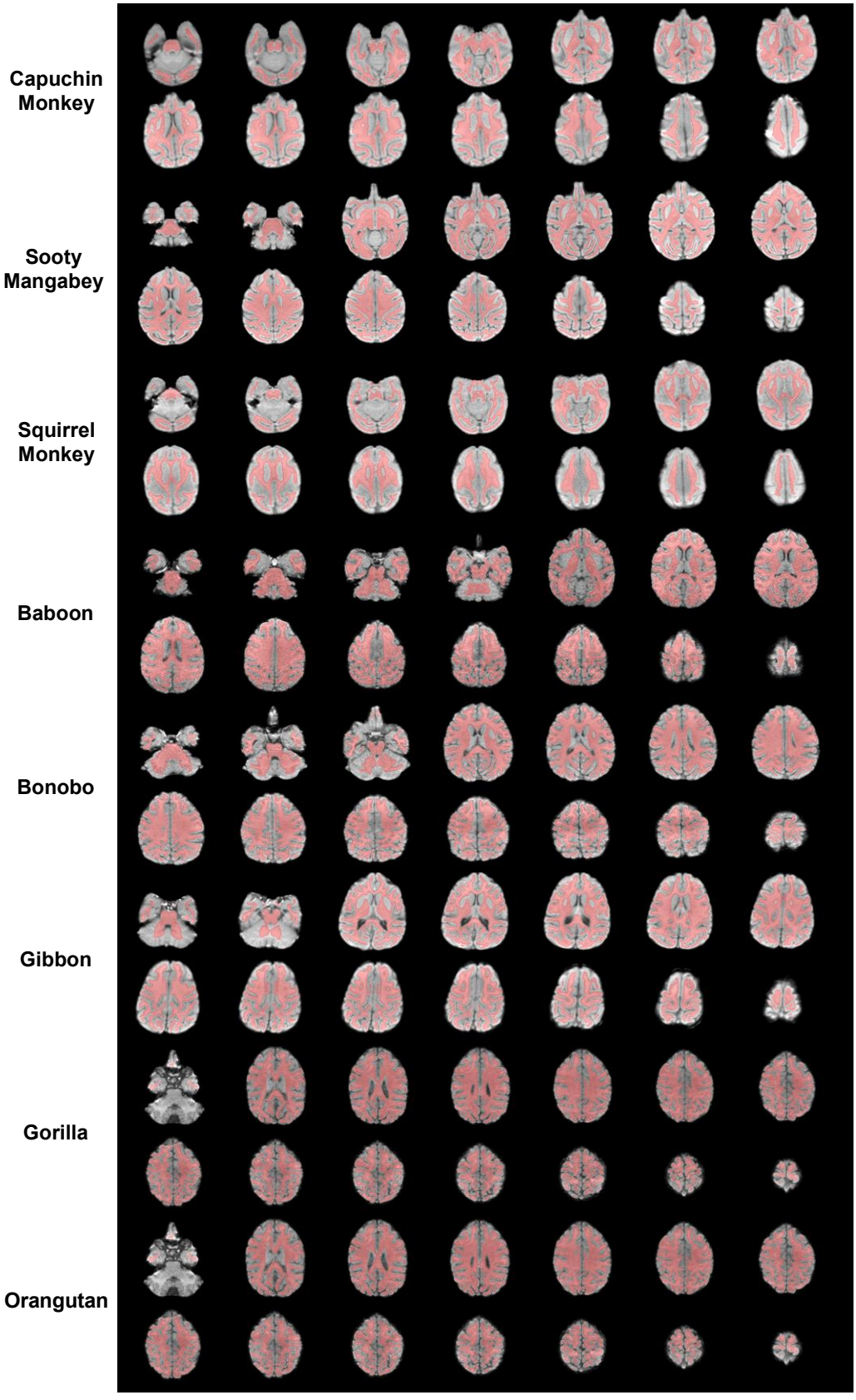
Supplementary Fig 6 | Quantitative validation of segmentation performance across species in the absence of ground truth. (a) Coefficient of variation (CV) analysis comparing volumetric consistency between ground-truth-available (GT) and no-ground-truth (No-GT) cohorts for both skull stripping (orange) and white matter segmentation (blue) across five primate species. (b) Kolmogorov-Smirnov (KS) test results visualized via empirical cumulative distribution function (ECDF) curves, evaluating distributional equivalency of segmented volumes between GT and No-GT conditions.



Supplementary Fig 7 | Qualitative assessment of segmentation performance on unannotated unseen-domain datasets



Supplementary Fig 8 | Qualitative assessment of segmentation performance on eight previously unseen NHP species



References

1. Ren, J., et al. DeepPrep: an accelerated, scalable and robust pipeline for neuroimaging preprocessing empowered by deep learning. *Nature Methods* **22**, 473-476 (2025).
2. Qu, Y., K. Huang, Yin Ming, and et al. CRISPR-GPT for agentic automation of gene-editing experiments. *Nature Biomedical Engineering* **8**, 24-41 (2024).
3. Xiao, Y., et al. Cellagent: An llm-driven multi-agent framework for automated single-cell data analysis. *arXiv preprint arXiv:2407.09811* (2024).
4. Atil, B., et al. Non-determinism of" deterministic" llm settings. *arXiv preprint arXiv:2408.04667* (2024).
5. Schick, T., et al. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems* **36**, 68539-68551 (2023).
6. Yao, S., et al. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629* (2022).
7. Hata, J., et al. Multi-modal brain magnetic resonance imaging database covering marmosets with a wide age range. *Scientific data* **10**, 221 (2023).
8. Fortin, J.-P., et al. Harmonization of multi-site diffusion tensor imaging data. *Neuroimage* **161**, 149-170 (2017).
9. Chen, A.A., et al. Mitigating site effects in covariance for machine learning in neuroimaging data. *Human brain mapping* **43**, 1179-1195 (2022).
10. Milham, M., et al. Accelerating the evolution of nonhuman primate neuroimaging. *Neuron* **105**, 600-603 (2020).
11. Messinger, A., et al. A collaborative resource platform for non-human primate neuroimaging. *Neuroimage* **226**, 117519 (2021).
12. Pomponio, R., et al. Harmonization of large MRI datasets for the analysis of brain imaging patterns throughout the lifespan. *NeuroImage* **208**, 116450 (2020).
13. Billot, B., et al. SynthSeg: Segmentation of brain MRI scans of any contrast and resolution without retraining. *Medical image analysis* **86**, 102789 (2023).
14. Hoffmann, M., et al. SynthMorph: learning contrast-invariant registration without acquired images. *IEEE transactions on medical imaging* **41**, 543-558 (2021).
15. Hoopes, A., J.S. Mora, A.V. Dalca, B. Fischl, and M. Hoffmann. SynthStrip: skull-stripping for any brain image. *NeuroImage* **260**, 119474 (2022).
16. Deng, Y., et al. Stytr2: Image style transfer with transformers. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* 11326-11336 (2022).
17. Gatys, L.A., A.S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. *Proceedings of the IEEE conference on computer vision and pattern recognition* 2414-2423 (2016).
18. Huang, X. and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *Proceedings of the IEEE international conference on computer vision* 1501-1510 (2017).
19. Isensee, F., P.F. Jaeger, S.A. Kohl, J. Petersen, and K.H. Maier-Hein. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods* **18**, 203-211 (2021).
20. Rahman, M.M., M. Trabelsi, H. Uzunalioglu, and A. Boyd. Personalized mixture of experts for multi-site medical image segmentation. *2025 IEEE/CVF winter conference on*

applications of computer vision (WACV) 3172-3184 (2025).
21. Gu, A. and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752* (2023).
22. Anthony, Q., Y. Tokpanov, P. Glorioso, and B. Millidge. Blackmamba: Mixture of experts for state-space models. *arXiv preprint arXiv:2402.01771* (2024).
23. Fedus, W., B. Zoph, and N. Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research* **23**, 1-39 (2022).
24. Lepikhin, D., et al. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668* (2020).
25. Shit, S., et al. clDice-a novel topology-preserving loss function for tubular structure segmentation. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* 16560-16569 (2021).
26. Kirchhoff, Y., et al. Skeleton recall loss for connectivity conserving and resource efficient segmentation of thin tubular structures. *European Conference on Computer Vision* 218-234 (2024).
27. Shazeer, N., et al. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538* (2017).
28. Lee, C.-Y., S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. *Artificial intelligence and statistics* 562-570 (2015).