Cycle-consistent Multi-graph Matching for Self-supervised Annotation of C. Elegans

Supplementary Material

Table of Contents

A. Runtimes

All runtime experiments were conducted on a machine equipped with an AMD Ryzen Threadripper PRO 5975WX CPU (32 cores, 3.60 GHz) and 256 GB of RAM. For comparability and consistency, all experiments were performed using single-threaded processing without parallel jobs.

We present runtimes for all steps involved in our proposed unsupervised MGM pipeline in Tab. A.1. The alignment step encompasses all 100 training worms and 100 test worms. The Bayesian optimization parameter learning was performed on a subset of $N_{\text{learn}} = 15$ training worms. Additionally, we report the runtimes for the three different MGM solver modes (see Sec. 2.2), when applied to solve the worm multi-matching problem over all 100 training worms. Finally, the evaluation runtimes include atlas construction, matching the atlas to each of the 100 test worms, and performing the final evaluation.

Table A.1. Runtimes of the unsupervised MGM pipeline.

Process	**Duration (h)**
Alignment (Sec. 3.4)	8
Parameter learning (Sec. 3.2)	81
MGM Solver (*Direct*)	2
MGM Solver (Sync. sparse)	10
MGM Solver (Sync. dense)	10
Atlas building & evaluation	0.3

We further detail the runtime of the MGM solver modes in Tab. A.2. While synchronization-based methods solve the MGM problem in under an hour, they require a preprocessing step involving the solution of $\binom{100}{2} = 4950$ pairwise GM problems, which constitutes the most time-consuming part of the process.

Table A.2. Runtimes of the three MGM solver modes when solving the worm multi-matching problem over all 100 training worms. Durations are given in h:min.

	Direct	**Sync. sparse**	**Sync. dense**
Pairwise GM	–	09:36	09:36
MGM Solving	02:11	00:04	00:46
Total	02:11	09:40	10:22

In contrast, the *direct* mode does not require this costly initial step. Consequently, the *direct* solver is overall nearly five times faster than the synchronization-based modes. However, solving the pairwise problems in parallel could in theory significantly reduce their runtime.

We attribute the difference between the *sparse* and *dense* synchronization mode mainly to an inefficient implementation. The software was written to solve sparse problems in particular and currently does not handle the many zero cost assignments of Eq. (7) well. Infinite cost assignments of sparse problems are only modelled implicitly by not storing a cost value for these assignments. This is highly beneficial to make sparse problems tractable, however, many optimizations that can be assumed for dense problems are not applied.

B. Cost learning

We provide details of Sec. 3.2 where the covariance and problem size parameters for our unsupervised MGM pipeline are learned based on a subset of $N_{\text{learn}} = 15$ training worms.

Learned parameters. The model size parameters learned from $N_{\text{learn}} = 15$ training worms of the `200worms` dataset are given as follows:

$$K_{\text{min}} = 14, \quad \tau_{\text{cen}} = 1.95, \quad \tau_{\text{rad}} = 0.15 \tag{8}$$

The learned covariances are given by

$$\Sigma^{\text{cen}} = \begin{pmatrix} 130 & 0 & 0 \\ 0 & 43 & 0 \\ 0 & 0 & 28 \end{pmatrix}, \quad \Sigma^{\text{rad}} = \begin{pmatrix} 156 & 0 & 0 \\ 0 & 200 & 0 \\ 0 & 0 & 174 \end{pmatrix} \tag{9}$$

and

$$\Sigma^{\text{off}} = \begin{pmatrix} 172 & 0 & 0 \\ 0 & 32 & 0 \\ 0 & 0 & 44 \end{pmatrix} \tag{10}$$

Unassignment cost constant. To avoid trivial empty matchings, which would result in a minimal cycle loss of zero but would fail to produce meaningful solutions, the

Table C.1. Evaluating ablations on the final model without relearning. Keeping learned covariances constant, we investigate the impact on the final pre-atlas and atlas accuracy when linear *center* (λ^{cen}), *radii* (λ^{rad}) or quadratic *offset vector* (λ^{off}) costs are discarded. Notably, including radii costs leads to no improvement in the linear model. This can be explained by numerous manually curated cell segments in our training and test data, with a constant cell-independent segment radius. Thus the latter has little to no influence on the model. In addition, shown by the *single re-alignment* ablation, we tested the effect of the second re-alignment step on the training worms, which is performed after cost learning. Skipping it, leads to only slightly worse results.

Ablation	Pre-Atlas Acc.	Atlas Acc.
$\lambda^{\text{cen}} = 1,\ \lambda^{\{\text{rad, off}\}} = 0$	0.953 ± 0.024	0.960 ± 0.021
$\lambda^{\text{rad}} = 1,\ \lambda^{\{\text{cen, off}\}} = 0$	0.260 ± 0.189	0.060 ± 0.011
$\lambda^{\text{off}} = 0,\ \lambda^{\{\text{cen, rad}\}} = 1$	0.953 ± 0.023	0.960 ± 0.021
$\lambda^{\text{off}} = 1,\ \lambda^{\{\text{cen, rad}\}} = 0$	0.967 ± 0.016	0.958 ± 0.021
Single re-alignment	0.963 ± 0.018	0.971 ± 0.015
Full pipeline	0.966 ± 0.016	0.972 ± 0.015

unassignment cost c_0 is crucial. This constant forces the algorithm to establish correspondences by preventing an empty solution. We set $c_0 = 10000$ for cost learning and atlas matching, to enforce as many matchings as possible. For the MGM problem after cost learning, we chose $c_0 = 40$, which results, on average, in one unmatched nucleus per GM problem. By allowing this, we found the resulting atlas to yield slightly better results. However, the best value is very much dependent on the final magnitude of all the cost parameters. As such, we recommend fine-tuning c_0 via a grid search or another Bayesian optimization step on a validation dataset.

C. Additional ablation experiments

In addition to the experiments shown in Tab. 1, we performed ablations on our final model without retraining. The aim is to investigate the contribution of the second re-alignment step and the impact of individual cost terms on the final accuracy. The results are presented in Tab. C.1.

D. Atlas accuracy distribution over test worms.

We illustrate in Fig. D.1 the specific accuracy value of every test worm for both our supervised and unsupervised atlas.

E. Supervised atlas construction

We present our construction of the supervised baseline, which is based on the approach outlined in [11]. Since we have access to the ground truth labels from which we can extract additional information, the supervised approach differs from the unsupervised one in the following aspects:

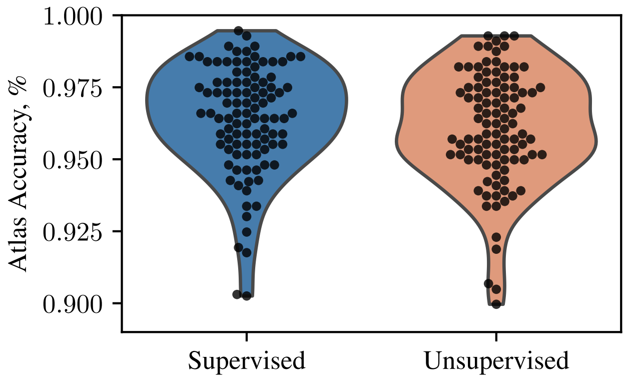

Figure D.1. Distribution and density of accuracies on the 100 test worms for the supervised and unsupervised atlas. Each black dot represents a distinct worm of the test set and its corresponding accuracy. The violin plot behind the black dots shows the approximate shape of the probability density function over all samples.

Alignment. Similar to the unsupervised case, we first preprocess all worms as described in Sec. 3.4, then align all worms to an optimal base worm W. In the supervised case we follow the approach from [12]. We start by fixing each training worm W_i as a target. We then align the nuclei centroids of all remaining worms to W using a linear transform minimizing the total Euclidean distance of corresponding centroids. From all worms aligned to W_i we build a supervised atlas based on W_i. This results in 100 supervised atlases. Now, for each such atlas we compute the average normalized centroid distance to each of the remaining 99 training worms, and average this distance across these. We choose as our supervised baseline atlas the one atlas minimizing the average distance, and finally align all other training worms again by minimizing the total Euclidean distance of corresponding centroids. This ensures that the aligned worms fit as closely as possible to the selected base worm W. Lastly, for the alignment of the test worms, we apply the average affine transformation to each worm in the test set.

Graph matching (GM) hyperparameters. We adopt the sparsity parameters for the atlas-to-worm GM problems from [11]. These are given by $\tau_{\text{cen}} = 8$, $\tau_{\text{rad}} = 12$ and $K = 6$. However, our experiments suggest that the cost weight parameters $\lambda^{\text{cen}}, \lambda^{\text{rad}}, \lambda^{\text{off}}$ reported in [11] are not optimal. We thus finetune the hyperparameters $\lambda^{\text{cen}}, \lambda^{\text{rad}}, \lambda^{\text{off}} \geq 0$ using Bayesian Optimization [16] where our goal is to maximize the average matching accuracy of the baseline atlas w.r.t. to all 100 training worms. The optimal parameters we found after 500 optimization steps are given by

$$\lambda^{\text{cen}} = 0.48, \quad \lambda^{\text{rad}} = 0.34, \quad \lambda^{\text{off}} = 0.81 \tag{11}$$

Improved supervised baseline We decide to use our supervised atlas as a baseline for comparison instead of those provided in [11] and [12], since it achieves overall higher accuracy values:

- [12] apply their methods on the same `200worms` dataset we used for our experiments. We achieve an average accuracy 0.96 ± 0.02 compared to their 0.93 ± 0.11. Note that we used the reported numbers of their single optimal template (i.e. atlas) experiment for comparison because this experiment corresponds to our supervised atlas building pipeline. While [12] achieve higher accuracies in other atlas-to-worm matching methods (0.95 using "5-template before EPC" and 0.97 using "5-template after EPC") those methods do not directly compare, as they rely on evaluating the final accuracy based on the best five atlases instead of just one, or on an Error Prediction and Correction (EPC) approach which involves manual curation (after matching) of nuclei flagged as having a low confidence score.
- [11] apply their methods on a dataset consisting of 30 manually segmented and only partially ground truth annotated worms. We achieve an average accuracy 0.91 ± 0.05 compared to their 0.90 ± 0.07. Note that accuracies for this dataset were obtained by averaging the atlas accuracies in a leave-one-out fashion, i.e. we created 30 atlases where each one of the 30 worms was fixed as test target and the remaining 29 worms were used to build an atlas. The reported accuracies are the accuracies averaged across these 30 atlases.

F. Atlas from StarDist segmentations

The experiments in Sec. 4 use the segmentations of the dataset [12], which are expert-corrected for segmentation errors[2]. In the following, we investigate the impact of using automatically generated segmentations.

Generating automatic segmentations. We apply *StarDist* [24, 25], a state-of the art cell instance segmentation model, to our training and test images. For 3D cell segmentation, *StarDist* provides only a *demo model*, which we used without retraining. Since we assess sensitivity to imperfect segmentations, using a non-optimized model is acceptable. Fig. F.1 visually compares the given ground truth segmentations against the segmentations returned by *StarDist*.

On average, 528 ± 23 nuclei were segmented, which is notably less than the 558 that are present in each imaged worm. The intersection over union (IoU) between the ground truth and the *StarDist* segmentations is 0.40 ± 0.03, which is low but expected due to the rough spherical shape approximations in the ground truth and the lower number of cells segmented by *StarDist*.

[2]As visible in Fig. F.1, all manually placed cell segments of dataset [12] are perfect spheres. Here, *expert-correction* involved only removing incorrect segments and accurately placing nuclei center points. Consequently, these annotations do not constitute a true ground truth w.r.t. the segmentations themselves. Nevertheless, we will continue to refer to this data as the ground truth.

Transferring ground truth labels. To evaluate our unsupervised and supervised approaches with the *StarDist* segmentations, nuclei annotated with labels are required. We transfer these labels from the ground truth segmentation based on cell overlap: Each generated cell segment is matched to the ground truth segment with which it shares the largest voxel intersection. To ensure consistency, we apply the same heuristic we used to assign ground truth labels to an unsupervised atlas (see Sec. 3.3), guaranteeing that each ground truth nucleus is assigned to at most one *StarDist* segment.

The resulting matching yields a labeling of the predicted nuclei by inheriting the labels from the corresponding ground truth nuclei. Using this approach, we annotate on average 503 ± 19 nuclei per worm in the *StarDist* segmentations, leaving approximately 25 nuclei per worm unlabeled.

Despite this, both the supervised and unsupervised atlases are able to recover around 545 distinct nuclei, due to aggregation over all worms during atlas construction.

Cost learning. For the *StarDist* segmentation in the unsupervised setting we have obtained the following parameters:

$$K_{\text{min}} = 16, \quad \tau_{\text{cen}} = 9.08, \quad \tau_{\text{rad}} = 7.30 \tag{12}$$

$$\Sigma^{\text{cen}} = \begin{pmatrix} 176 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 24 \end{pmatrix}, \quad \Sigma^{\text{rad}} = \begin{pmatrix} 65 & 0 & 0 \\ 0 & 188 & 0 \\ 0 & 0 & 10 \end{pmatrix} \tag{13}$$

and

$$\Sigma^{\text{off}} = \begin{pmatrix} 200 & 0 & 0 \\ 0 & 181 & 0 \\ 0 & 0 & 182 \end{pmatrix} \tag{14}$$

In the supervised setting, we obtained the following weights:

$$\lambda^{\text{cen}} = 0.21, \quad \lambda^{\text{rad}} = 1.63, \quad \lambda^{\text{off}} = 1.35 \tag{15}$$

F.1. Accuracy evaluation (Tab. F.1).

As shown in Tab. F.1, the accuracy gap between the supervised and unsupervised models increases substantially when using the *StarDist* segmentation, compared to the ground-truth segmentation results in Tab. 3. Notably, the performance gap between the *Unlearned quadratic* model and the *Full unsupervised pipeline* also widens considerably, clearly demonstrating the benefit of our learning approach and its robustness to imperfect segmentations.

That said, we found the *Pre-processing* step (see Sec. 3.4) to be the most vulnerable component to segmentation errors in the *StarDist* output, and a major contributor to the observed accuracy drop. While the pre-processing aligned worms within 1–5° using ground-truth segmentations, the alignment error increased to nearly 40° with *StarDist* segmentations; see Fig. F.2. Such large misalignment often could not be fully corrected by the subsequent steps based on pairwise GM described in the *Alignments* paragraph of Sec. 3.4.

The supervised pipeline was considerably more robust to these pre-processing issues, as it uses ground-truth correspondences for alignment during training (see Suppl. E). The resulting atlas parameters (15) place the strongest weight on shape and offset features, which are largely insensitive to misalignment, yielding better test accuracy. In contrast, the unsupervised parameters (13)–(14) assign significantly higher weight to absolute coordinates, making them substantially more sensitive to alignment errors and therefore more prone to mismatches during GM.

Additional ablation experiment (Tab. F.1). The variant *Learned* $\lambda^{\{\text{cen, rad, off}\}}$ extends the *unlearned quadratic* model by performing 500 Bayesian optimization iterations to estimate the weights $\lambda^{\{\text{cen, rad, off}\}}$:

$$\lambda^{\text{cen}} = 0.03, \quad \lambda^{\text{rad}} = 1.55, \quad \lambda^{\text{off}} = 0.06. \qquad (16)$$

Analogously to the supervised setting, this model is less sensitive to pre-processing errors, primarily due to the higher weight of the shape term λ^{rad}. Due to that, this simplified model even outperforms our *Full unsupervised pipeline*.

Details on the pre-processing errors with StarDist segmentation (Fig. F.2). Due to the high cell density in the worm brain, the head–tail alignment performed equally well for the *StarDist* and the ground-truth segmentations. However, the left–right (dorsal–ventral) alignment was substantially less accurate for *StarDist*, which we attribute to its segmentations being noticeably less symmetric. We validated this by manually re-rotating the test worm with the largest accuracy drop about its head–tail axis, which immediately improved its accuracy from 5% to 64%.

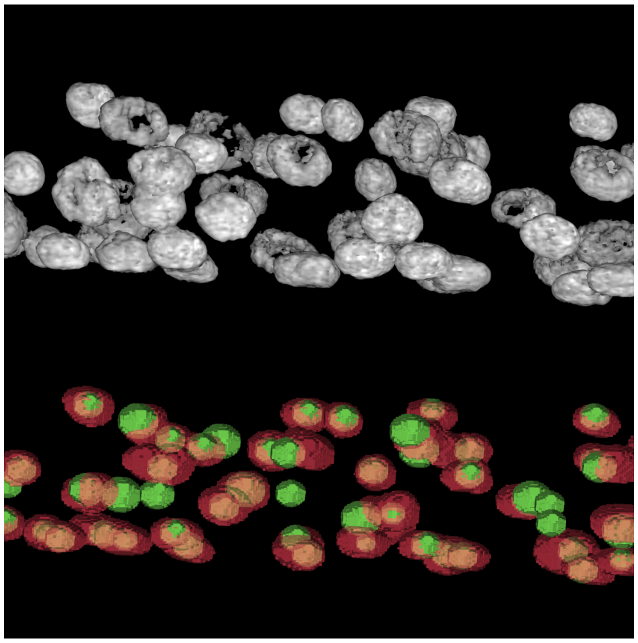
Figure F.1. Excerpt of a raw 3D worm image (top) compared to the two segmentations (bottom): The manual segmentation of dataset [12], showing ground-truth annotations (green) and a segmentation predicted by *StarDist* (red). The manual labels frequently appear as small spherical segments, reflecting the incomplete nature of the manual annotation process, in which nuclei were only marked coarsely rather than fully delineated.

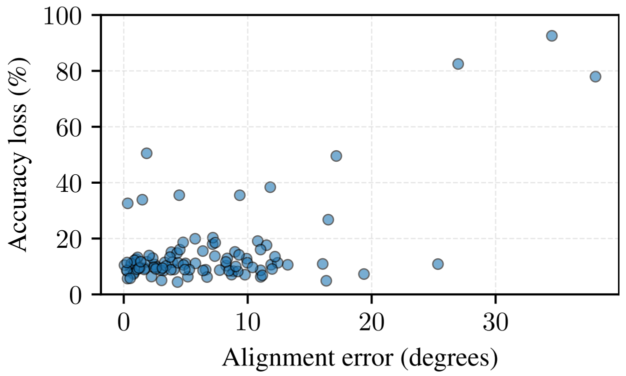

Figure F.2. Relationship between alignment error and accuracy loss. Each point corresponds to one test worm. During pre-processing (Sec. 3.4), worms are rotated about the head–tail axis to maximize left–right symmetry. The x-axis shows the difference between the rotation angles obtained from the ground-truth and the StarDist segmentations. The y-axis shows the resulting drop in matching accuracy when replacing ground-truth segmentation with StarDist, using the respective unsupervised atlases.

Table F.1. Accuracy based on the *StarDist* segmentation for different atlas construction methods. Unsupervised and supervised pipelines constructed and evaluated as in Sec. 4.2.

Method	Pre-Atlas Accuracy (Training data)	Atlas Accuracy (Test data)
Unlearned quadratic	0.668 ± 0.199	0.609 ± 0.192
Learned $\lambda^{\{\text{cen, rad, off}\}}$	$\mathbf{0.858 \pm 0.182}$	0.829 ± 0.062
Full unsupervised pipeline	0.827 ± 0.196	0.809 ± 0.147
Supervised pipeline	—	$\mathbf{0.916 \pm 0.029}$