PatchPerPixMatch for Automated 3d Search of Neuronal Morphologies in Light Microscopy

Lisa Mais[1], Peter Hirsch[1], Claire Managan[2], Kaiyu Wang[2], Konrad Rokicki[2], Robert R. Svirskas[2], Barry J. Dickson[2], Wyatt Korff[2], Gerald M. Rubin[2], Gudrun Ihrke[2], Geoffrey W. Meissner[2], and Dagmar Kainmueller[1]

[1] Max-Delbrück-Center for Molecular Medicine in the Helmholtz Association (MDC), Berlin, Germany, {firstname.lastname}@mdc-berlin.de
[2] HHMI Janelia Research Campus, Ashburn, VA, USA

Abstract. Studies of individual neurons in the Drosophila nervous system are facilitated by transgenic lines that sparsely and repeatably label respective neurons of interest. Sparsity can be enhanced by means of intersectional approaches like the split-GAL4 system, which labels the positive intersection of the expression patterns of two (denser) GAL4 lines. To this end, two GAL4 lines have to be identified as labelling a neuron of interest. Current approaches to tackling this task include visual inspection, as well as automated search in 2d projection images, of single cell multi-color flip-out (MCFO) acquisitions of GAL4 expression patterns. There is to date no automated method available that performs full 3d search in MCFO imagery of GAL4 lines, nor one that leverages automated reconstructions of the labelled neuron morphologies. To close this gap, we propose PatchPerPixMatch, a fully automated approach for finding a given neuron morphology in MCFO acquisitions of Gen1 GAL4 lines. PatchPerPixMatch performs automated instance segmentation of MCFO acquisitions, and subsequently searches for a target neuron morphology by minimizing an objective that aims at covering the target with a set of well-fitting segmentation fragments. PatchPerPixMatch is computationally efficient albeit being full 3d, while also highly robust to inaccuracies in the automated neuron instance segmentation. We are releasing PatchPerPixMatch search results for ∼30,000 neuron morphologies from the Drosophila *hemibrain* in ∼20,000 MCFO acquisitions of ∼3,500 Gen1 GAL4 lines.
Code: *https://github.com/Kainmueller-Lab/PatchPerPixMatch*
Results: *https://pppm.janelia.org*

1 Introduction

The recent release of a Drosophila melanogaster central brain connectome reconstructed from electron microscopy (*hemibrain* [21]), as well as a large resource of MCFO acquisitions of GAL4 line Drosophila central nervous systems (CNSs) [8,10] have paved the way for intersectional approaches [7] to sparsely and repeatably target a vast collection of individual neurons in the Drosophila CNS. To this end, a neuron of interest with known morphology, e.g. as reconstructed by electron microscopy, needs to be identified in MCFO acquisitions of two distinct GAL4 lines. Given such two GAL4 lines, a split-GAL4 line [13] can then be created, which expresses the positive intersection of the respective GAL4 expression patterns.

A key step in this approach is the identification of GAL4 line MCFO acquisitions that label a target neuron of interest. To this end, the NBLAST approach [6] has been shown to successfully identify neuron morphologies across imaging modalities [22]. However, NBLAST relies on curated reconstructions of target as well as source neuron morphologies as input. Such reconstructions have not been feasible to obtain at scale for GAL4 line MCFO acquisitions. An approach for neuron search that does not require reconstructions of individual neuron morphologies in MCFO images as input is Color-depth maximum intensity projection (MIP) search [11,1]. This approach operates on 2d projection images of MCFO channels, where it computes pixel-wise heuristic matching scores against projections of respective target neurons.

We describe here *PatchPerPixMatch*, an alternative fully automated approach that allows for efficient 3d search of neuron morphologies in GAL4 MCFO acquisitions. Our approach is based on PatchPerPix [9], a deep learning-based instance segmentation approach we have developed in previous work to tackle challenging properties specific to neurons in MCFO imagery, namely large spans of individual neuron instances, and overlaps of multiple instances as caused by partial volume effects. PatchPerPixMatch aims at piecing together a known target neuron morphology from PatchPerPix segmentation fragments. Thus, crucially, PatchPerPixMatch provides built-in robustness against false split errors in automated neuron instance segmentations.

We phrase the PatchPerPixMatch objective formally as a combinatorial optimization problem. We derive upper and lower bounds to the respective energy, and leverage these bounds to engineer an efficient solver. Thus we were able to run $\sim$600 million PatchPerPixMatch searches, namely for $\sim$30,000 target neuron morphologies reconstructed from the hemibrain [21] (version 1.2) in $\sim$20,000 MCFO acquisitions of $\sim$3,500 Gen1 GAL4 lines [10].

We are releasing the PatchPerPixMatch search results on *https://pppm.janelia.org*. Our code is available at *https://github.com/Kainmueller-Lab/PatchPerPixMatch*.

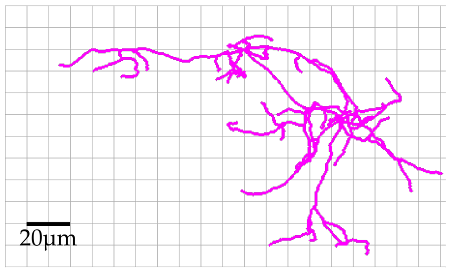
2 Method

Individual neurons in MCFO image data as released in [10] are infeasible to be reconstructed manually at scale. Respective suitable automated instance segmentation methods are scarce due to challenging characteristics specific to neurons in light microscopy, such as wide spans and complex shapes of neurons, as well as overlap of multiple neurons due to partial volume effects. In previous work, we have developed PatchPerPix [9], a deep-learning based instance segmentation method that is able to handle the above characteristics of neurons in light microscopy. PatchPerPix yields all neuron instances in an image in one go, yielding run-times that allow for processing MCFO data at scale.

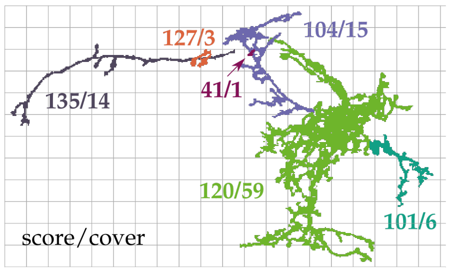
PatchPerPix segmentations of neurons in MCFO image data are not to be considered correct reconstructions: Due to ambiguities in the data that often are hard to resolve locally even by eye, as well as the entailed scarcity of training data, false split- or false merge segmentation errors appear in many neurons. However, PatchPerPix segmentations do abundantly yield accurate reconstructions of large *fragments* of neurons.

The idea behind PatchPerPixMatch is to leverage PatchPerPix segmentations despite their inaccuracies, and thus allow for fully automated search of target neuron morphologies in MCFO data. To this end, PatchPerPixMatch searches for a target neuron morphology by determining a subset of PatchPerPix neuron segmentation fragments that jointly explain the target as well as possible, as illustrated in Figure 1. Thus PatchPerPixMatch achieves inherent robustness to false-split segmentation errors.

The target neuron morphology is assumed to be given in skeletonized form, and in the same reference coordinate system as the segmentation fragments, which can e.g. be achieved via registration of the underlying imagery to a template brain [2]. In a first step, neuron segmentation fragments above a threshold size of $10\mu m$ (bounding box diagonal) are skeletonized [15], and NBLAST scores [6] are computed individually for each fragment vs. the target morphology. In a second step, PatchPerPixMatch aims at finding a subset of PatchPerPix fragments that jointly achieve *high NBLAST scores* and *cover large part of the target*, while also exhibiting *consistent colors* in terms of the three neuron-labelling channels of the underlying MCFO image. This second step is described in detail in Sections 2.1 to 2.3. Efficient processing of both hemispheres of the Drosophila brain is described in Section 2.4. While PatchPerPixMatch is inherently robust to false-split segmentation errors, it can, to some degree, also handle false-merge segmentation errors, as described in Section 2.5.



(a) Target neuron morphology



(b) PatchPerPixMatch fragment selection

Fig. 1: (a) Exemplary target neuron morphology (pC1e, see Resources Table in Figure 8), and (b) PatchPerPix instance segmentation fragments selected to cover the target neuron morphology (from an MCFO acquisition of the known matching GAL4 line R35C10). Each fragment yields a *score* that reflects how well it fits the target morphology locally, as well as a *coverage value* that reflects which percentage of the target the fragment may explain. Section 2.1 describes how we obtain scores and coverages per fragment, and how we derive a respective overall score for a selection of fragments. Sections 2.2 and 2.3 describe how we search for an optimal fragment selection for a given target neuron morphology.

2.1 Objective

In formal terms, given a set of fragment IDs $F \subset \mathbf{N}$, and a vector of respective indicator variables $\mathbf{x} = (x_1, \ldots, x_{|F|}) \in \{0,1\}^{|F|}$ that encodes fragment selection, the goal is to find a fragment selection that minimizes

$$E(\mathbf{x}) := -\sum_{f \in F} x_f \cdot c_f \cdot \left(s_f - \lambda \sum_{g \in F} x_g \cdot cdp_{f,g} \right), \tag{1}$$

where c_f measures which fraction of the target neuron morphology a fragment f covers, s_f measures how well it scores in covering this fraction, and $cdp_{f,g}$ serves to penalize color differences between fragments (with weight λ). For convenient derivation of bounds to the objective (to follow in Section 2.2), the energy (1) can be re-written and refined as a sum over the points of a point cloud representation of the target, denoted via a set of point indices $T \subset \mathbf{N}$:

$$E(\mathbf{x}) = -\sum_{t \in T : \exists f \in F : c(f,t)=1 \wedge x_f = 1} w_t \cdot \left(s_{\hat{f}(\mathbf{x},t)} - \lambda \cdot \sum_{g \in F : x_g = 1} cdp_{\hat{f}(\mathbf{x},t),g} \right), \tag{2}$$

where w_t weighs the influence of an individual target point, $c(f,t) \in \{0,1\}$ indicates if fragment f covers target point t, and $\hat{f}(\mathbf{x},t)$ selects the best fragment to cover a target point t in case of multiple contenders.

In the following we give our specific definitions for each ingredient of (2), namely for weights w_t, coverage $c(f,t)$, best fragment $\hat{f}(\mathbf{x},t)$, score s_f, and color difference penalty $cdp_{f,g}$.

Target Point Influence w_t: Weights on target points are intended to increase the influence of stereotyped parts of a neuron's morphology, i.e. parts with characteristic location as well as orientation, like e.g. wide-spanning projections, while downgrading the influence of e.g. small twigs in dendritic arbors, where orientation is less likely to be stereotyped [4]. To this end, given a skeleton representation of the target, we prune it by removing terminal branches that are shorter than a threshold length of $10\mu m$, repeatedly for 3 iterations. Then, we assign a "branch length" to each remaining point of the target morphology, as follows: All points in terminal branches are assigned their respective branch's length. Subsequently, the points that have already been assigned a length are ignored, and the procedure is iterated with the remaining morphology, until all points are processed. Finally, the point weights are normalized, such that $\sum_{t \in T} w_t = 1$.

Coverage $c(f,t)$: For an individual fragment f, we define the set of target points it covers, $\{t \in T : c(f,t) = 1\}$, as specified in Algorithm 1. The algorithm assumes a point cloud representation of the fragment as well as the target as inputs. Note that target- and fragment point cloud representations are expected to exhibit comparable sample point distances. We refer to the target point cloud as $P := \{p_t \in \mathbf{R}^3 : t \in T\}$, and to the point cloud for fragment f as $Q := \{q_i \in \mathbf{R}^3 : i \in \{1, ..., N\}\}$.

Algorithm 1 considers the set Q of all fragment points jointly, with the desired effect of putting one large fragment at an advantage as opposed to many small fragments, as

Algorithm 1: Target Point Coverage by a Fragment

```
Initialize target point coverage, $\forall t \in T : c(f,t) := 0$;
Initialize a running set of fragment points, $R := Q$;
repeat
    for each $p_t \in P$ do
        Determine the closest fragment point within threshold distance,
        $q^{closest}(p_t) := \text{argmin1}_{q \in R : ||p_t - q|| < thresh} ||p_t - q||$
    for each $q \in \{q^{closest}(p_t) : t \in T\}$ do
        Mark up to 6 target point(s) to which it is closest as covered:
        $\forall t \in \text{argmin6}_{t \in T : q = q^{closest}(p_t)} ||p_t - q|| : c^{new}(f,t) := 1$;
        if $|\{t \in T : q = q^{closest}(p_t) ||p_t - q||\}| > 6$ then
            $R := R \setminus \{q\}$
    coverageIncreased:=True;
    if $\{t \in T : c^{new}(f,t) = 1\} \subset \{t \in T : c(f,t) = 1\}$ then
        coverageIncreased:=False;
    else
        Update target point coverage: $\forall t : c(f,t) := \max(c(f,t), c^{new}(f,t))$
until coverageIncreased=False;
```

explained by the example sketched in Figure 2. Algorithm 1 restricts the number of target points a fragment point can cover to some maximum number to avoid over-coverage, as exemplified in Figure 3a. Over-coverage would otherwise be caused, e.g., by fragment end points that are closest to many target points within threshold distance. We empirically determined a restriction to six target points to yield visually plausible coverage. The *repeat* loop in Algorithm 1 avoids putting large fragments at a disadvantage as opposed to many small fragments, as exemplified in Figure 3b: E.g., a spurious side branch of a fragment could be closest to many target points, only a restricted number of which will be determined as covered. Our proposed looping gives the thus-excluded target points a chance to still be covered by other fragment points.

Best Fragment $\hat{f}(\mathbf{x}, t)$: Given two fragments, the sets of target points they cover are not necessarily disjoint. To ensure that each target point contributes at most once to the objective (2), for each target point, we select one fragment that covers it "best". While seemingly straightforward to select the best-covering fragment for a target point by means of best NBLAST score, this choice has the drawback of favoring small fragments, as high NBLAST scores are abundantly obtained "by chance" by small fragments. Thus, to avoid false-positive solutions assembled from large amounts of small fragments, we instead employ as selection criterion the (weighted) amount of the target a fragment can explain, thus favoring larger, more characteristic fragments:

$$\hat{f}(\mathbf{x}, t) := \underset{f \in F : c(f,t)=1 \wedge x_f = 1}{\text{argmax1}} \sum_{t' : c(f,t')=1} w_{t'},$$

where we denote $\text{argmax1} := \min(\text{argmax}(.))$ for convenience to include the case of non-uniqueness of argmax.

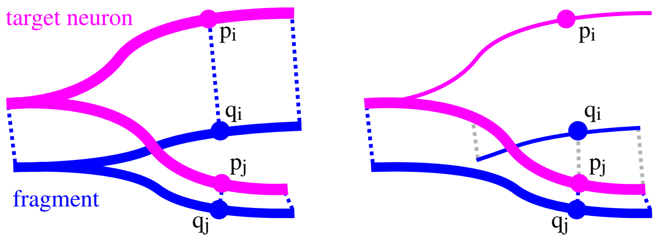


Fig. 2: Algorithm 1, which serves for determining which portion of a target neuron morphology (pink) a neuron segmentation fragment (blue) covers, favors larger fragments: If a fragment is whole (left), q_j covers p_j, and q_i covers p_i because p_j is already "taken". If instead the fragment is split into two parts (right), both q_i and q_j cover p_j as it is the closest target point for both of them, and therefore the upper part of the target neuron will not be covered, indicated by thin (as opposed to bold) magenta line. (Blue dotted lines indicate desired matches, grey dotted lines indicate obsolete matches.)

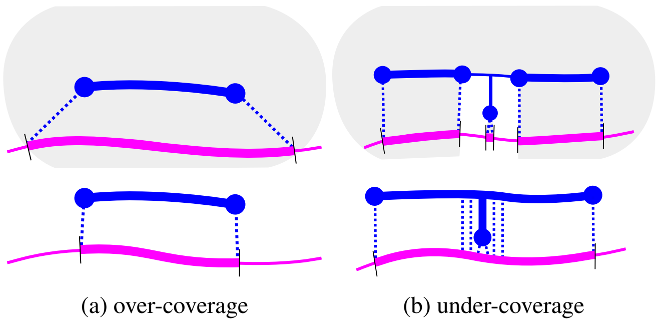


Fig. 3: (a) Top: Without the constraints imposed by Algorithm 1, a fragment (blue) would cover an oversized portion of a target neuron, namely all target points within threshold distance (shaded area / bold magenta line). Bottom: Restricting the number of target points a single fragment point can cover results in coverage of an appropriately-sized target region. (b) Top: In case of a small side-branch, avoiding over-coverage as in (a) can lead to under-coverage, as many target points may have the same branch end-point as their closest point. Bottom: To avoid such under-coverage, Algorithm 1 loops over the point matching process, each time removing fragment points whose coverage had to be restricted in the previous iteration, until coverage does not increase any more.

Fragment Score s_f: As an additional means to reduce the impact of small fragments (and finally only use them as "hole fillers"), we do not use the plain NBLAST score $nblast_f \in [-1, 1]$ of a fragment as score s_f in (2). Instead, first, we cap the normalized NBLAST score at 0.5 to dampen the impact of small non-distinctive fragments that may easily receive very high scores. Second, we adjust the capped score according to the amount of target a fragment can (at most) explain, thus defining a positive, "coverage-adjusted" score, as follows:

$$s_f := (\min\{nblast_f, 0.5\} - 1) \cdot \alpha_f + 2 \tag{3}$$

with

$$\alpha_f := \sigma(-20 \cdot c_f) + 1, \text{ and } c_f := \sum_{t \in T : c(f,t)=1} w_t.$$

With this, α_f approaches 1 for large fragments, thus yielding a capped and positive-shifted NBLAST score, $\min\{nblast_f, 0.5\} + 1$. Instead, for small fragments, α_f approaches 1.5, thus yielding a score that is smaller than the capped and positive-shifted NBLAST score (as α_f is multiplied to the *negative-shifted* NBLAST score).

Color Difference Penalty $cdp_{f,g}$: A color is assigned to an individual fragment via K-Means clustering of the respective MCFO image voxels' three channel values. We empirically set $k = 2$, and set the fragment color $col_f \in \mathbf{R}^3$ to the mean of the larger cluster. Fragment colors for an individual MCFO acquisition are then normalized. We aim at a penalty that is small for differences in intensity, and large otherwise. More specifically, we follow the empirical observation that channel intensities within one neuron may fluctuate individually, though the channels that do fluctuate tend to do so in concordant direction. Hence we define the color difference penalty, based on color difference $cd_{f,g} := col_g - col_f \in \mathbf{R}^3$ and a color difference threshold cdt, as follows:

$$cdp_{f,g} := \begin{cases} 0.1 \cdot \max\limits_{i=1}^{3} |cd_{f,g,i}| & \text{if } \forall i : cd_{f,g,i} < cdt \vee \forall i : cd_{f,g,i} > -cdt. \\ |\max\limits_{i=1}^{3} cd_{f,g,i} - \min\limits_{i=1}^{3} cd_{f,g,i}|, & \text{otherwise.} \end{cases} \tag{4}$$

Thus, a small penalty is assigned if color channels fluctuate in concordant direction (where concordance is determined with slack cdt), whereas a larger penalty is assigned otherwise. The weight λ in (2) is set to adjust the color difference penalty for the number of selected fragments, and to weigh color difference penalties against scores. We empirically set $\lambda(\mathbf{x}) := 0.5/(-1 + \sum_{g \in F} x_g)$.

Note that the PatchPerPixMatch energy (2), given our definitions of all of its components, can be modelled as a Markov random field (MRF), where nodes represent fragments, and binary labels represent fragment selection. However, the factorization of the respective probability distribution contains not just 2nd order terms (due to the color difference penalites), but also higher ($>$ 2nd) order terms due to the dependency of best fragments $\hat{f}$ and weight λ on the overall fragment selection $\mathbf{x}$. Consequently, generic efficient approximate solvers for 2nd order binary MRFs, like e.g. QPBO [14], do not apply.

2.2 Bounds

For efficient optimization of the PatchPerPixMatch objective (2), we define computationally cheap upper and lower bounds, as follows:

Upper bound to the minimum energy:

$$ub := -\max_{f \in F} s_f \cdot c_f \tag{5}$$

Our upper bound constitutes the minimal energy achievable with a single fragment solution.

Lower bound to the minimum energy:

$$lb := -\sum_t w_t \cdot \max_{f \in F : c(f,t)=1} s_f \ , \tag{6}$$

Our lower bound is obtained by picking the highest-scoring of all fragments per target point and ignoring the color difference penalty. This is a valid lower bound because

$$\forall t, \mathbf{x} : \max_{f \in F : c(f,t)=1} s_f \geq s_{\hat{f}(\mathbf{x},t)},$$

and furthermore, the color difference penalty is always positive, i.e., $\forall f, g : cdp_{f,g} \geq 0$.

Lower bound to the energy of a candidate solution $\mathbf{x}$*:*

$$lbs(\mathbf{x}) := -\sum_t w_t \cdot \max_{f \in F : c(f,t)=1 \wedge x_f = 1} s_f \ , \tag{7}$$

analogous to the lower bound to the minimum energy.

2.3 Solver

For efficient optimization of the PatchPerPixMatch objective (2), we first determine a set of candidate solutions heuristically via agglomerative clustering [20] of fragments by color difference. For each candidate solution, we then discard fragments whose total color difference penalty outweighs their score. We continue to the next candidate solution early if a lower bound (7) to the energy of the candidate surpasses a running upper bound to the minimum energy. The solver is specified in detail in Algorithm 2.

2.4 Efficient processing of both brain hemispheres

Due to fly brain symmetry, most target neurons may be found in either hemisphere of a segmented brain [12]. PatchPerPixMatch aims at finding the best match across hemispheres. To this end, we mirror the target neuron in the reference coordinate frame. For efficient processing of both hemispheres, we first determine the computationally cheap upper and lower bounds (5) and (6) for each hemisphere individually. If the lower bound determined for side I exceeds the upper bound determined for side II, side I can be discarded. Otherwise, we proceed with side I, and use the energy of its best solution as initial upper bound for side II.

Algorithm 2: PatchPerPixMatch Solver

```
As set of fragments F, use all f with score $s_f > 0.5$ ;
Initialize running upper bound to the minimum energy, $ubr := ub, \mathbf{x}_{ubr} := \mathbf{x}_{ub}$, as
  defined in (5) ;
Determine a set of candidate solutions heuristically via agglomerative clustering of
  fragments by color difference;
for each candidate solution $\mathbf{x}$ do
    Initialize lower bound to energy of candidate solution, $lbsr(\mathbf{x}) := lbs(\mathbf{x})$, as
      defined in (7);
    if $lbsr(\mathbf{x}) \geq ubr$ then
        continue to next candidate solution
    repeat
        Compute best fragments $\hat{f}(\mathbf{x}, t)$ for all t;
        Update lower bound $lbsr(\mathbf{x}) := -\sum_{f \in F: x_f = 1} s_f \cdot \sum_{t \in T: f = \hat{f}(\mathbf{x}, t)} w_t$;
        if $lbsr(\mathbf{x}) \geq ubr$ then
            continue to next candidate solution
        dumped:=False;
        for each selected fragment f do
            Compute $\sum_{g: x_g = 1} cdp_{f,g}$;
            if $\lambda \cdot \sum_{g: x_g = 1} cdp_{f,g} > s_f$ then
                Dump fragment, i.e. set $x_f := 0$;
                dumped:=True;
        Compute energy $E(\mathbf{x})$ of remaining solution;
        if $E(\mathbf{x}) < ubr$ then
            Update $ubr := E(\mathbf{x}), \mathbf{x}_{ubr} := \mathbf{x}$;
    until dumped=False;
```

2.5 Robustness to false merge segmentation errors

PatchPerPixMatch naturally handles false split errors in the underlying segmentation by seeking a combination of segmentation fragments to best cover a target. False merge errors, on the other hand, may lead to false misses of matching neurons. This is mainly due to the fact that false merge fragments necessarily yield poor NBLAST scores, as a false merge fragment by definition cannot be a subset of any target morphology.

To nevertheless increase robustness of PatchPerPixMatch against false merge segmentation errors, we proceed as follows: We compute a second set of NBLAST scores, for which we prune each fragment to the part that lies within $20\mu m$ of the target. False merge fragments may thus yield decent NBLAST scores. We perform the whole PatchPerPixMatch pipeline again for this second set of NBLAST scores, where we re-use previously computed coverage values c_f to avoid redundant computation.

Pruned fragments necessarily yield higher or equal NBLAST scores as compared to the original fragments. Hence PatchPerPixMatch energies of solutions obtained in the ”pruned” pipeline run cannot be directly compared with energies of solutions obtained in the ”vanilla” pipeline run. Thus we rank the lists of matches yielded by the two pipeline runs independently, in terms of PatchPerPixMatch energy. To yield one final ranked list while accounting for the fact that energies are not directly comparable across runs, we merge the lists from the two pipeline runs *by rank* (and not by energy). Specifically, we add 10.5 to the ranks yielded by the pruned run to achieve uniqueness and to slightly favor the vanilla run, then sort the merger of both lists by rank, and finally remove duplicate matches with lower ranks from the merged list.

3 Results

We ran PatchPerPixMatch for ∼30,000 target neuron morphologies from the hemibrain [21] version 1.2, selected by means of size and quality tags via the *neuPrint* tool [5] publicly available at *https://neuprint.janelia.org*. We searched for these target neuron morphologies in ∼20,000 MCFO brain acquisitions of sparse and medium density Generation 1 GAL4 lines (Gen1 MCFO Phase 1 Categories 2 and 3 as published with [10], available at *https://gen1mcfo.janelia.org*, registered to the JRC2018 Unisex template [2]).

Bulk quantitative analysis. We evaluated the accuracy of PatchPerPixMatch quantitatively on 10 target neuron morphologies and respective 47 known matching GAL4 lines [17,19,18], as listed in Appendix A, Figure 8. For each known matching GAL4 line, we determined two different ranks: (1) The rank of the best-matching MCFO acquisition of the GAL4 line among all ∼20,000 searched MCFO acquisitions (termed *sample rank*), and (2) the rank of the best-matching MCFO acquisition of the GAL4 line among all ∼3,500 best-matching MCFO acquisitions per GAL4 line (termed *line rank*).

The median sample rank of an MCFO acquisition of a known matching line is 23. The median respective line rank is 18. I.e., for half of the known matching lines, the best-matching respective MCFO acquisition ranks at or above 23 among all ∼20,000 samples, and at or above 18 among all ∼3,500 best-matching samples per line. In terms of sample ranks, best-matching MCFO acquisitions of the known matching lines are ranked among the top 1000, top 500, top 150, and top 50 samples for 42, 40, 36, and 26 of the 47 known matches, respectively. In terms of line ranks, best-matching MCFO acquisitions of the known matching lines are ranked among the top 500, top 150, and top 50 lines for 42, 38, and 30 of the 47 known matches, respectively.

Quantitative analysis for the neurons pC1d and pC1e. For the neuron pC1d, a set of known matching GAL4 lines has been determined via exhaustive behavioral screen of nearly all Janelia (”R”) lines [16] and thus exhibits an exceptional level of completeness. Furthermore, an overlapping, yet not identical set of matching lines is also known for the morphologically similar neuron pC1e. These circumstances exclusively allow for a quantitative assessment of false PatchPerPixMatch hits. Note that the ”VT” lines were not included in the behavioral screen and were hence excluded from our respective analysis. Figure 4 shows exemplary morphologies of the pC1d and pC1e neurons. Note the distinctive arbor present in pC1d but not in pC1e.

Appendix B, Figure 9 lists the PatchPerPixMatch line- and sample ranks of the assessed known matching lines for pC1d and pC1e. Appendix B, Figure 10 lists the first

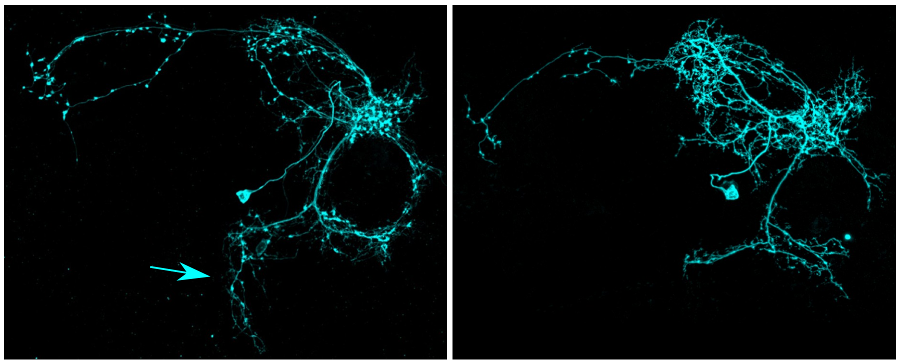

(a) pC1d (b) pC1e

Fig. 4: Exemplary pC1d and pC1e neurons (excerpts from Figure 4 – figure supplements 3D and 4D of [17]). pC1d features a distinctive arbor, see arrow.

50 PatchPerPixMatch hits for pC1d as well as for pC1e. All non-confirmed Janelia line hits were inspected visually and deemed as either plausible, not convincing, or too faint or crowded to judge. For pC1d, out of the 27 Janelia line samples among the first 50 PatchPerPixMatch hits, 16 are samples from confirmed matching lines, 4 are confirmed matches for the morphologically similar pC1e, 2 are suspected matches of the morphologically similar pC1e, 2 are too crowded or faint to judge, and 3 do not look plausible. For pC1e, out of the 28 Janelia line samples among the first 50 PatchPerPixMatch hits, 17 are samples from confirmed matching lines, 2 look plausible albeit not confirmed, 3 are too crowded or faint to judge, and 6 do not look plausible. Two lines that are known to contain pC1e but not pC1d, namely R60G04 and R60G08, are ranked at line ranks 1 and 3 for pC1e. Respective false hits for pC1d rank comparatively lower, namely at line ranks 10 and 12, with 4 and 5 known matching lines, i.e. true hits, ahead of them, respectively.

Qualitative analysis. We inspected PatchPerPixMatch hits visually to assess its potential to reveal matches to end users, as well as to determine typical causes for falsely missing known matches. Figure 5a showcases a true positive PatchPerPixMatch hit for a relatively sparse MCFO sample, which is easily visible by eye in a maximum intensity projection (MIP) of the MCFO image (see row 1). Figures 5b and 5c showcase true positive hits for more challenging MCFO acquisitions, namely for densely clustered and dim neuronal morphologies. These are hardly visible by eye either in the MIP of the MCFO image (row 1), or in color-depth MIPs [11] of the MCFO channel that best labels the matching morphology (rows 4 and 5). However, the hits are revealed by masking the MCFO image with the instance segmentation fragment selection determined by PatchPerPixMatch (rows 2 and 3).

By visual inspection, we found that many of the MCFO acquisitions that rank higher than those of known matching lines appear to contain visually plausible matches (cf. Figure 5d). However, their respective lines have not been subjected to split line creation

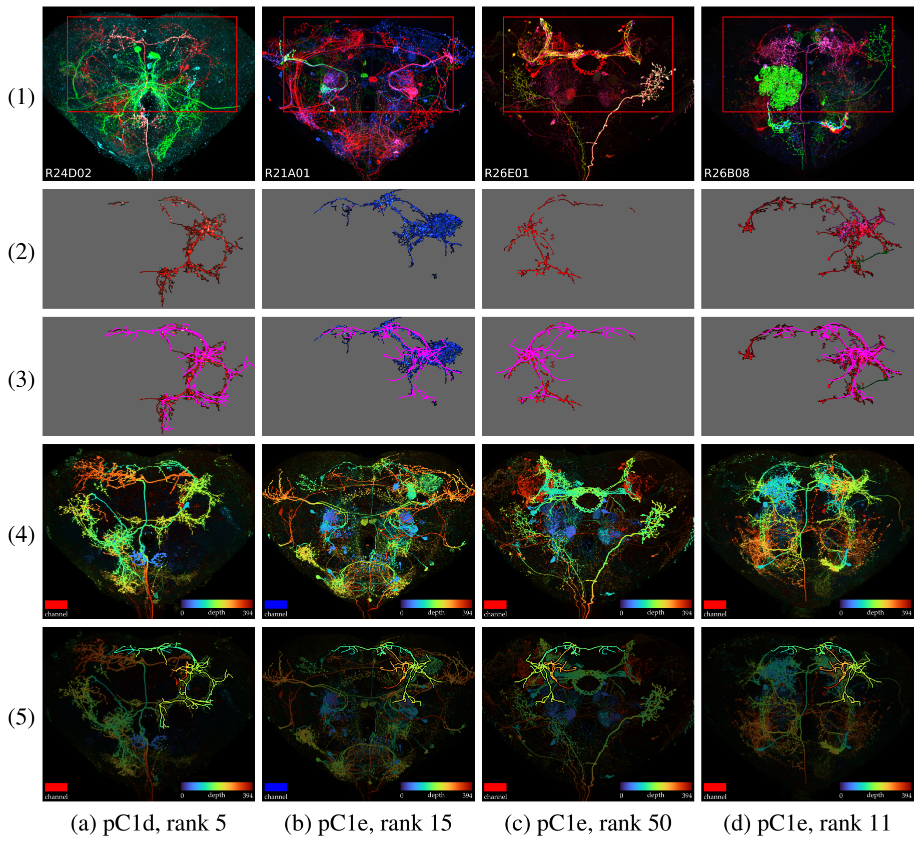


Fig. 5: (a-c) Confirmed true positive and (d) convincing-looking albeit not confirmed PatchPerPixMatch hits for the target neurons pC1d and pC1e. The rows display the following from top to bottom: (1) Maximum intensity projection (MIP) of MCFO acquisition. (2) Section within the red rectangle from (1) masked by selected PatchPerPix fragments, (3) overlaid with skeletonized target, (4) Color-depth MIP [11] of best-matching channel, and (5) the same dimmed and overlaid with color-depth projection of the target. In (a), the target neuron can easily be spotted in rows 1, 4 and 5, whereas in (b) and (c), the PatchPerPixMatch fragment selection is required to reveal respective hits, as shown in rows 2 and 3.

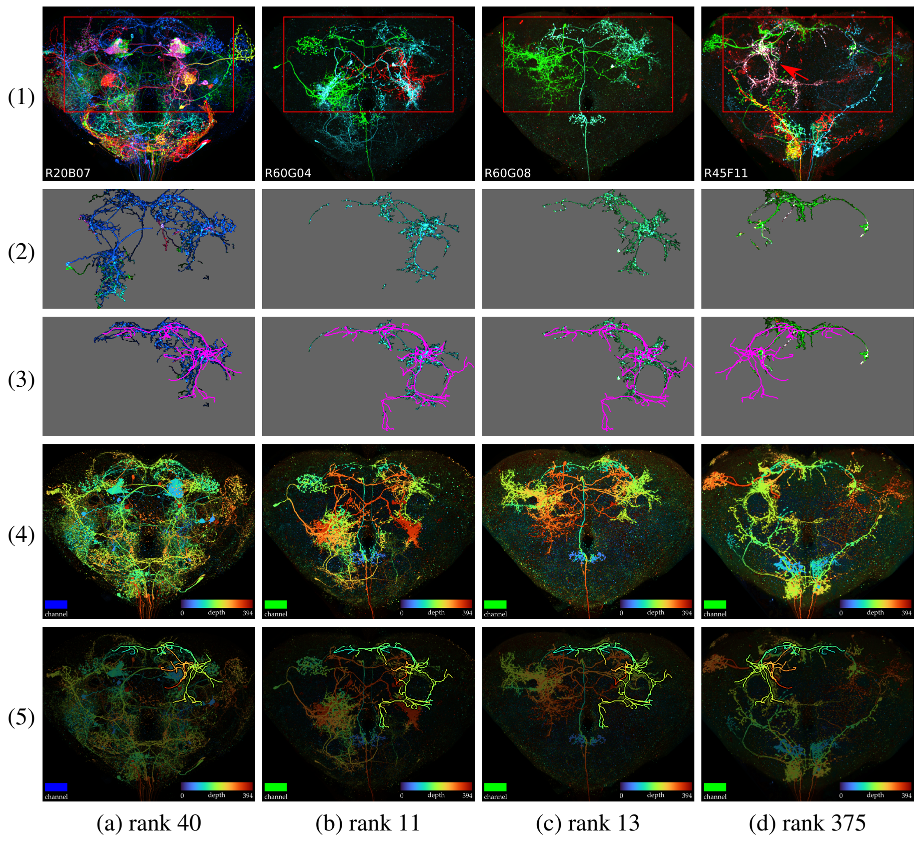


Fig. 6: Failure cases / sources of error for PatchPerPixMatch. (a) False hit for the target neuron pC1e caused by false merge of densely clustered neurons. (b+c) False hits for the target neuron pC1d in lines that are known to exclusively contain pC1e: While most of the target neuron pC1d is covered by high-scoring fragments (resulting in high PatchPerPixMatch scores and ranks), the distinctive arbor of pC1d as compared to pC1e (cf. Figure 4a) is left uncovered. (d) False miss of the target neuron pC1e in an MCFO acquisition where pC1e overlaps with another neuron of different color, causing an observed shift in color. (For a row-by-row description of the visualization, see Figure 5.)

and hence cannot be confirmed to be true matches. On the other hand, some high-ranking MCFO acquisitions appear to be false hits upon visual inspection, some of which can be attributed to false mergers of densely clustered neurons in the PatchPerPix segmentation (cf. Figure 6a). Another potential cause for false hits are neurons with similar morphology, large fractions of which may get covered by high-scoring fragments. Figures 6b and 6c show respective false hits for an extreme case of similar morphologies, namely of the neurons pC1d and pC1e.

Some known matches missed by PatchPerPixMatch can be attributed to drastic shifts in color we empirically observe within some neurons in MCFO acquisitions (cf. Figure 6d). Such shifts may occur in case of overlap with a second neuron with (partly) similar morphology yet different color. Future work will investigate if a modified color difference penalty better handles these cases.

False misses may in general also be caused by the stochastic nature of MCFO labelling, due to which sets of MCFO acquisitions for a GAL4 line are not guaranteed to cover the complete respective GAL4 expression pattern [10]. Hence a neuron labelled in the GAL4 expression pattern may be missing in the respective MCFO image data as opposed to being missed by the PatchPerPixMatch search.

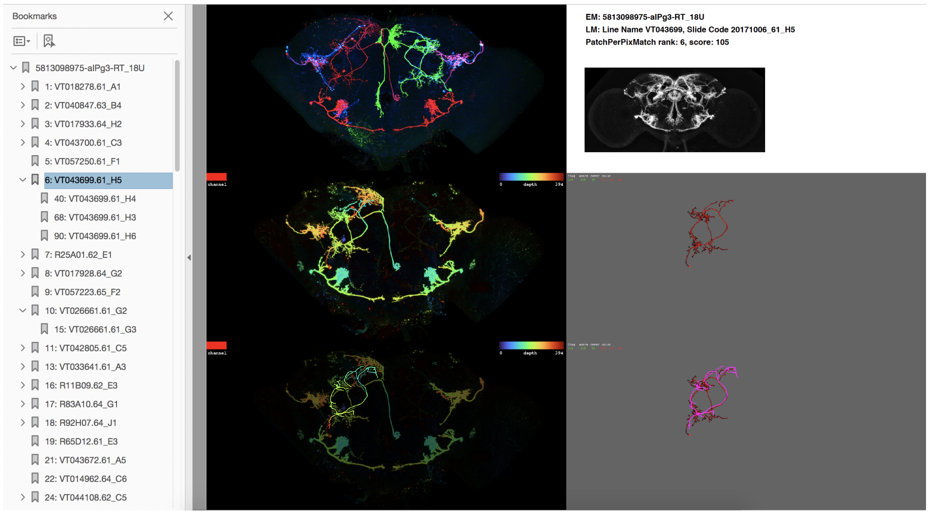
Run-time Analysis and Code Availability. The average run-time for a PatchPerPixMatch search for an individual target morphology over all ∼20,000 segmented MCFO acquisitions was 116 min on a single 3.0 GHz core.

Our code is available at *https://github.com/Kainmueller-Lab/PatchPerPixMatch*.

4 Visualization of PatchPerPixMatch Search Results

Our qualitative analysis reveals, apart from typical sources of error, which kinds of visualization of PatchPerPixMatch hits provide helpful information for the end user to effectively filter out false hits and increase the success rate of split-GAL4 line creation. Figure 7 depicts our thus-derived visualization for an exemplary match of a hemibrain neuron morphology in a GAL4 MCFO acquisition. It shows a PatchPerPixMatch result for a neuron that is easily visible in all panels of the visualization, for the purpose of clearly conveying our visualization.

If the sought target neuron is clearly visible in the maximum intensity projection (MIP) of the MCFO acquisition, then this is considered the best case scenario where split-GAL4 line creation can be done with high confidence. If an MCFO sample labels a higher number of neurons and the target thus cannot be spotted easily by eye in the respective MCFO MIP, it might still be visible in the color-depth MIP panels of the best-matching MCFO channel (as also employed in [11]), which exhibit reduced neuron density and furthermore enable direct comparison of 3d location between target and MCFO fragments. However, if the target is barely visible even in the channel color-depth MIP panels due to faintness or substantial occlusion by other neurons in the same channel, the panels showing MIPs of the fragment-masked MCFO acquisition (with and without the overlaid target) may still reveal a clean hit. These panels may also show more clearly than all others if the potential match misses to cover a characteristic part of the target, as seen e.g. for the false hits of the pC1d neuron in Figure 6b and 6c.



5813098975-aIPg3-RT_18U
1: VT018278.61_A1
2: VT040847.63_B4
3: VT017933.64_H2
4: VT043700.61_C3
5: VT057250.61_F1
6: VT043699.61_H5
40: VT043699.61_H4
68: VT043699.61_H3
90: VT043699.61_H6
7: R25A01.62_E1
8: VT017928.64_G2
9: VT057223.65_F2
10: VT026661.61_G2
15: VT026661.61_G3
11: VT042805.61_C5
13: VT033641.61_A3
16: R11B09.62_E3
17: R83A10.64_G1
18: R92H07.64_J1
19: R65D12.61_E3
21: VT043672.61_A5
22: VT014962.64_C6
24: VT044108.62_C5

EM: 5813098975-aIPg3-RT_18U
LM: Line Name VT043699, Slide Code 20171006_61_H5
PatchPerPixMatch rank: 6, score: 105

Fig. 7: Our proposed visualization of a PatchPerPixMatch search result is composed of:
Top left image: Maximum intensity projection (MIP) of the MCFO sample (all three neuron signal channels).
Mid left image: Color-depth MIP (i.e. projection image with depth encoded as color) of best-matching channel. The channel color is indicated in the top left corner of the image. Color depth encoding is specified in the top right corner of the image.
Bottom left image: Color-depth MIP of best-matching channel, dimmed and overlaid with color-depth projection of target neuron morphology.
Top right panel: hemibrain ID of the target neuron, line- and sample ID of the MCFO sample, PatchPerPixMatch rank (among all ∼20,000 searched MCFO acquisitions) and PatchPerPixMatch score, as well as a MIP of the full expression pattern of the GAL4 line. The latter is supposed to give a sense of the respective density. The PatchPerPix-Match score is defined as $-100 \cdot E(\mathbf{x})$ to yield a positive score (higher is better) with negligible decimal places.
Mid right image: MCFO MIP masked by best-matching PatchPerPix fragments.
Bottom right image: MCFO MIP masked by best-matching PatchPerPix fragments (pruned if the match stems from a pruned run), overlaid with hemibrain body in magenta.
Left side-panel: pdf bookmarks: Matches for a target morphology are sorted by Patch-PerPixMatch ranks, and visualizations of the best 150 matches are assembled into a pdf document, with bookmarks that allow quick access via short summaries of matches (rank: line. sample). If top-150 matches occur in multiple samples of the same line, they are sorted in directly behind the best match of a line.
This Figure is best viewed with zoom / on large screen.

We created ∼30,000 pdf documents, one for each target neuron, visualizing the top 150 PatchPerPixMatch ranks as described in Figure 7. These pdfs are available for download at *https://pppm.janelia.org*. To facilitate annotation efforts, we also provide a spreadsheet for each target neuron morphology, where each row contains the GAL4 line name, MCFO slide code, and PatchPerPixMatch rank (among all ∼20,000 searched MCFO acquisitions) as contained in the respective PDF, sorted by rank.

5 Discussion and Conclusion

We have developed PatchPerPixMatch, a fully automated method for efficient 3d search of Drosophila neuron morphologies in light microscopy imagery. PatchPerPixMatch is based on automatic neuron reconstructions, and features built-in robustness to respective segmentation inaccuracies. We release search results for over ∼30,000 neuron morphologies in more than ∼20,000 light microscopy acquisitions of Gen1 GAL4 line MCFO Drosophila brains. This resource serves to complement the 2d projection-based search results from [11,1].

PatchPerPixMatch is applicable to target neuron morphologies from any source, as long as the target is registered to the same reference coordinate frame as the segmentation fragments. Note, if a target neuron morphology stems from light microscopy as opposed to EM, like e.g. the neuron morphologies available in the *FlyCircuit* database [3], the step of pruning small branches of the target neuron (cf. *Target Point Influence* described in Sec. 2.1) may be dispensable. Concerning the suitability of PatchPerPixMatch for user-conducted searches for further target neuron morphologies in Gen1 GAL4 line MCFO brain acquisitions from [10], we expect near-interactive run-times (<10 minutes) on off-the-shelf workstations (~3 GHz, ≥16 cores), deduced from an average single-threaded run-time of 116 minutes as measured in our study.

PatchPerPixMatch includes components that are, to various degrees, learnt from data: PatchPerPix [9] is based on a deep neural network trained for brain neuron instance segmentation, and NBLAST [6] similarity scoring is based on a histogram of atomic features observed for sets of neuron morphologies that include known matches. Consequently, generalizability to new data sources, which potentially follow distributions different from the ones employed in this work, needs to be assessed in respective further studies. E.g., while PatchPerPixMatch is technically not limited to the Drosophila brain, but can also be applied to the ventral nerve cord as is, respective potential distribution differences and their effect on PatchPerPixMatch performance remain to be assessed in future work.

Our released results are based on segmentation fragments obtained with PatchPerPix [9]. Technically, PatchPerPixMatch can take any automated segmentation result of the image data to be searched as input. However, respective segmentation accuracy, and, more specifically, the ratio of false merge- vs. false split segmentation errors, may impact PatchPerPixMatch performance. Future work will aim at end-to-end learning of neuron segmentation plus matching, where we will leverage a set of known matches on top of manual neuron segmentations as training data. Beyond replacing empirically determined parameter values in PatchPerPixMatch with machine-learnt values, such end-to-end learning may help gauge automated segmentation with PatchPerPix [9] towards an optimal balance of split- and merge errors.

Further future work will provide a quantitative comparison of PatchPerPixMatch with the heuristic, 2d projection-based method of Otsuna et al. [11,1], as well as with directly using NBLAST scores [6] to rank PatchPerPix fragments, in terms of their efficiency and effectiveness for finding GAL4 lines known to contain a target morphology. Last but not least, pre-computed PatchPerPixMatch search results will be integrated into the browser-based neuron search tool *NeuronBridge* [1] in a future release.

Acknowledgments. We wish to thank Ruchi Parekh and the Janelia Connectome Annotation Team for providing pre-publication information on known matching GAL4 lines for a series of target neurons reconstructed from the hemibrain. We wish to thank Hideo Otsuna for providing a select list of ∼30,000 hemibrain body IDs as well as their skeletons.

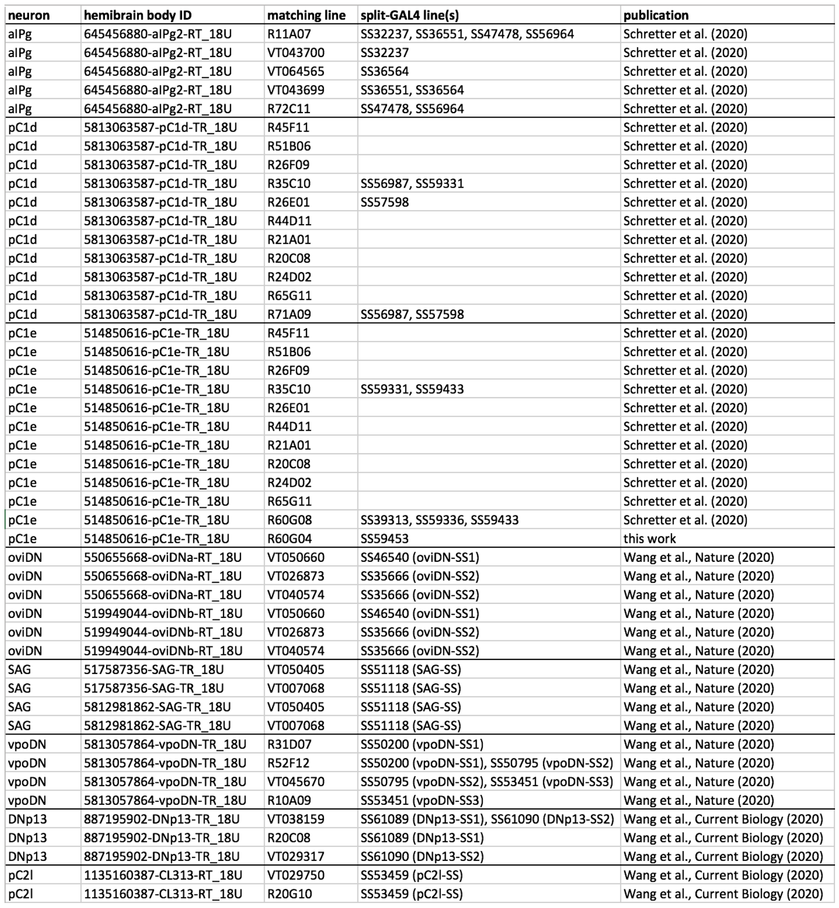
P.H., L.M. and D.K. were funded by the Max Delbrueck Center for Molecular Medicine, and supported by the HHMI Janelia Visiting Scientist Program. P.H. was funded by the MDC-NYU exchange program and HFSP grant RGP0021/2018-102.

References

1. https://doi.org/10.25378/janelia.12159378.v2
2. Bogovic, J.A., Otsuna, H., Heinrich, L., Ito, M., Jeter, J., Meissner, G., Nern, A., Colonell, J., Malkesman, O., Ito, K., et al.: An unbiased template of the drosophila brain and ventral nerve cord. Plos one **15**(12), e0236495 (2020)
3. Chiang, A.S., Lin, C.Y., Chuang, C.C., Chang, H.M., Hsieh, C.H., Yeh, C.W., Shih, C.T., Wu, J.J., Wang, G.T., Chen, Y.C., et al.: Three-dimensional reconstruction of brain-wide wiring networks in drosophila at single-cell resolution. Current biology **21**(1), 1–11 (2011)
4. Chou, Y.H., Spletter, M.L., Yaksi, E., Leong, J.C.S., Wilson, R.I., Luo, L.: Diversity and wiring variability of olfactory local interneurons in the drosophila antennal lobe. Nature Neuroscience **13**(4), 439–449 (Apr 2010)
5. Clements, J., Dolafi, T., Umayam, L., Neubarth, N.L., Berg, S., Scheffer, L.K., Plaza, S.M.: neuprint: Analysis tools for em connectomics. BioRxiv (2020)
6. Costa, M., Manton, J.D., Ostrovsky, A.D., Prohaska, S., Jefferis, G.S.: Nblast: rapid, sensitive comparison of neuronal structure and construction of neuron family databases. Neuron **91**(2), 293–311 (2016)
7. Guo, C., Pan, Y., Gong, Z.: Recent advances in the genetic dissection of neural circuits in drosophila. Neuroscience bulletin **35**(6), 1058–1072 (2019)
8. Jenett, A., Rubin, G., Ngo, T.T., Shepherd, D., Murphy, C., Dionne, H., Pfeiffer, B., Cavallaro, A., Hall, D., Jeter, J., Iyer, N., Fetter, D., Hausenfluck, J., Peng, H., Trautman, E., Svirskas, R., Myers, E., Iwinski, Z., Aso, Y., DePasquale, G., Enos, A., Hulamm, P., Lam, S., Li, H.H., Laverty, T., Long, F., Qu, L., Murphy, S., Rokicki, K., Safford, T., Shaw, K., Simpson, J., Sowell, A., Tae, S., Yu, Y., Zugates, C.: A gal4-driver line resource for drosophila neurobiology. Cell Reports **2**(4), 991–1001 (2012)
9. Mais, L., Hirsch, P., Kainmueller, D.: PatchPerPix for instance segmentation. Lecture Notes in Computer Science **12370**, 288–304 (2020)
10. Meissner, G.W., Dorman, Z., Nern, A., Forster, K., Gibney, T., Jeter, J., Johnson, L., He, Y., Lee, K., Melton, B., et al.: An image resource of subdivided drosophila gal4-driver expression patterns for neuron-level searches. BioRxiv (2020)

11. Otsuna, H., Ito, M., Kawase, T.: Color depth mip mask search: a new tool to expedite split-gal4 creation. BioRxiv p. 318006 (2018)
12. Pascual, A., Huang, K.L., Neveu, J., Préat, T.: Neuroanatomy: brain asymmetry and long-term memory. Nature **427**(6975), 605—606 (February 2004)
13. Pfeiffer, B.D., Ngo, T.T.B., Hibbard, K.L., Murphy, C., Jenett, A., Truman, J.W., Rubin, G.M.: Refinement of tools for targeted gene expression in drosophila. Genetics **186**(2), 735–755 (2010)
14. Rother, C., Kolmogorov, V., Lempitsky, V., Szummer, M.: Optimizing binary mrfs via extended roof duality. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8. IEEE (2007)
15. Sato, M., Bitter, I., Bender, M.A., Kaufman, A.E., Nakajima, M.: Teasar: Tree-structure extraction algorithm for accurate and robust skeletons. In: Proceedings the Eighth Pacific Conference on Computer Graphics and Applications. pp. 281–449. IEEE (2000)
16. Scherer, D., Müller, A., Behnke, S.: Evaluation of pooling operations in convolutional architectures for object recognition. In: International Conference on Artificial Neural Networks. pp. 92–101. Springer (2010)
17. Schretter, C.E., Aso, Y., Robie, A.A., Dreher, M., Dolan, M.J., Chen, N., Ito, M., Yang, T., Parekh, R., Branson, K.M., Rubin, G.M.: Cell types and neuronal circuitry underlying female aggression in drosophila. Elife **9**, e58942 (2020)
18. Wang, F., Wang, K., Forknall, N., Parekh, R., Dickson, B.J.: Circuit and behavioral mechanisms of sexual rejection by drosophila females. Current Biology **30**(19), 3749–3760 (2020)
19. Wang, K., Wang, F., Forknall, N., Yang, T., Patrick, C., Parekh, R., Dickson, B.J.: Neural circuit mechanisms of sexual receptivity in drosophila females. Nature pp. 1–5 (2020)
20. Ward, J.H.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association **58**(301), 236–244 (1963)
21. Xu, C.S., Januszewski, M., Lu, Z., Takemura, S.y., Hayworth, K., Huang, G., Shinomiya, K., Maitin-Shepard, J., Ackerman, D., Berg, S., et al.: A connectome of the adult drosophila central brain. BioRxiv (2020)
22. Zheng, Z., Lauritzen, J.S., Perlman, E., Robinson, C.G., Nichols, M., Milkie, D., Torrens, O., Price, J., Fisher, C.B., Sharifi, N., et al.: A complete electron microscopy volume of the brain of adult drosophila melanogaster. Cell **174**(3), 730–743 (2018)

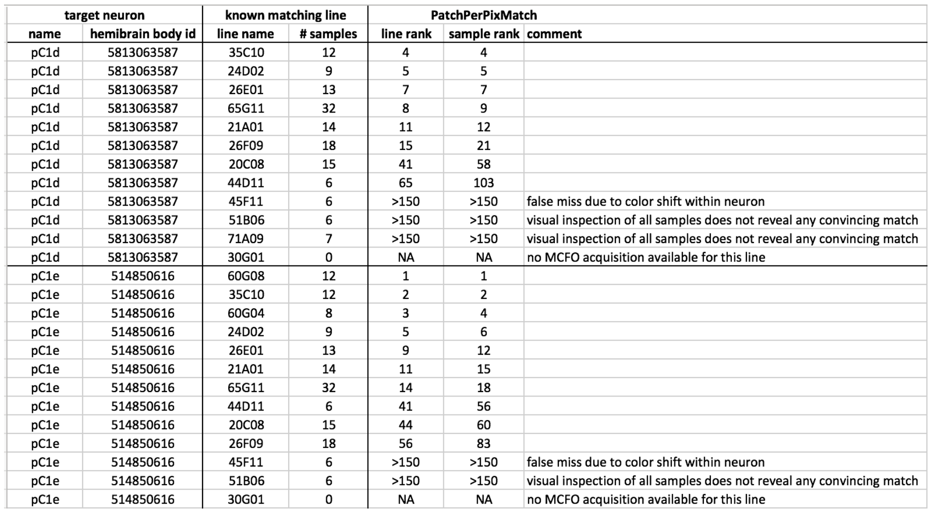
A Resources



neuron	hemibrain body ID	matching line	split-GAL4 line(s)	publication
aIPg	645456880-aIPg2-RT_18U	R11A07	SS32237, SS36551, SS47478, SS56964	Schretter et al. (2020)
aIPg	645456880-aIPg2-RT_18U	VT043700	SS32237	Schretter et al. (2020)
aIPg	645456880-aIPg2-RT_18U	VT064565	SS36564	Schretter et al. (2020)
aIPg	645456880-aIPg2-RT_18U	VT043699	SS36551, SS36564	Schretter et al. (2020)
aIPg	645456880-aIPg2-RT_18U	R72C11	SS47478, SS56964	Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R45F11		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R51B06		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R26F09		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R35C10	SS56987, SS59331	Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R26E01	SS57598	Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R44D11		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R21A01		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R20C08		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R24D02		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R65G11		Schretter et al. (2020)
pC1d	5813063587-pC1d-TR_18U	R71A09	SS56987, SS57598	Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R45F11		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R51B06		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R26F09		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R35C10	SS59331, SS59433	Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R26E01		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R44D11		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R21A01		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R20C08		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R24D02		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R65G11		Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R60G08	SS39313, SS59336, SS59433	Schretter et al. (2020)
pC1e	514850616-pC1e-TR_18U	R60G04	SS59453	this work
oviDN	550655668-oviDNa-RT_18U	VT050660	SS46540 (oviDN-SS1)	Wang et al., Nature (2020)
oviDN	550655668-oviDNa-RT_18U	VT026873	SS35666 (oviDN-SS2)	Wang et al., Nature (2020)
oviDN	550655668-oviDNa-RT_18U	VT040574	SS35666 (oviDN-SS2)	Wang et al., Nature (2020)
oviDN	519949044-oviDNb-RT_18U	VT050660	SS46540 (oviDN-SS1)	Wang et al., Nature (2020)
oviDN	519949044-oviDNb-RT_18U	VT026873	SS35666 (oviDN-SS2)	Wang et al., Nature (2020)
oviDN	519949044-oviDNb-RT_18U	VT040574	SS35666 (oviDN-SS2)	Wang et al., Nature (2020)
SAG	517587356-SAG-TR_18U	VT050405	SS51118 (SAG-SS)	Wang et al., Nature (2020)
SAG	517587356-SAG-TR_18U	VT007068	SS51118 (SAG-SS)	Wang et al., Nature (2020)
SAG	5812981862-SAG-TR_18U	VT050405	SS51118 (SAG-SS)	Wang et al., Nature (2020)
SAG	5812981862-SAG-TR_18U	VT007068	SS51118 (SAG-SS)	Wang et al., Nature (2020)
vpoDN	5813057864-vpoDN-TR_18U	R31D07	SS50200 (vpoDN-SS1)	Wang et al., Nature (2020)
vpoDN	5813057864-vpoDN-TR_18U	R52F12	SS50200 (vpoDN-SS1), SS50795 (vpoDN-SS2)	Wang et al., Nature (2020)
vpoDN	5813057864-vpoDN-TR_18U	VT045670	SS50795 (vpoDN-SS2), SS53451 (vpoDN-SS3)	Wang et al., Nature (2020)
vpoDN	5813057864-vpoDN-TR_18U	R10A09	SS53451 (vpoDN-SS3)	Wang et al., Nature (2020)
DNp13	887195902-DNp13-TR_18U	VT038159	SS61089 (DNp13-SS1), SS61090 (DNp13-SS2)	Wang et al., Current Biology (2020)
DNp13	887195902-DNp13-TR_18U	R20C08	SS61089 (DNp13-SS1)	Wang et al., Current Biology (2020)
DNp13	887195902-DNp13-TR_18U	VT029317	SS61090 (DNp13-SS2)	Wang et al., Current Biology (2020)
pC2l	1135160387-CL313-RT_18U	VT029750	SS53459 (pC2l-SS)	Wang et al., Current Biology (2020)
pC2l	1135160387-CL313-RT_18U	R20G10	SS53459 (pC2l-SS)	Wang et al., Current Biology (2020)

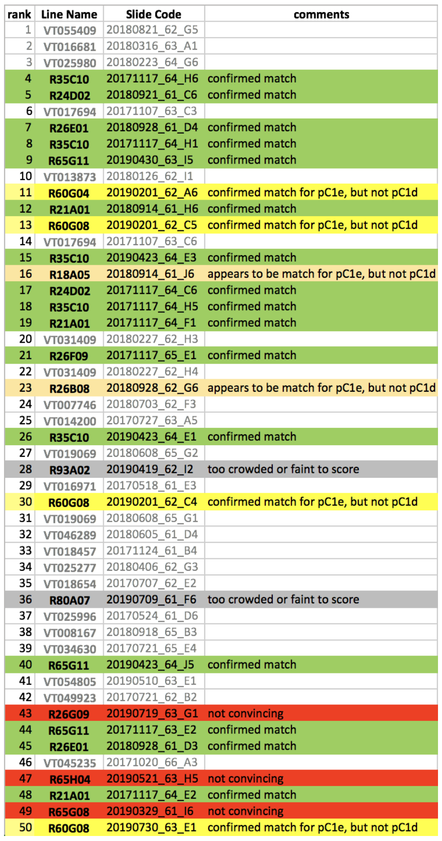
Fig. 8: Known matching lines used in this work, and their origin.

B Quantitative Evaluation on pC1d and pC1e Neurons



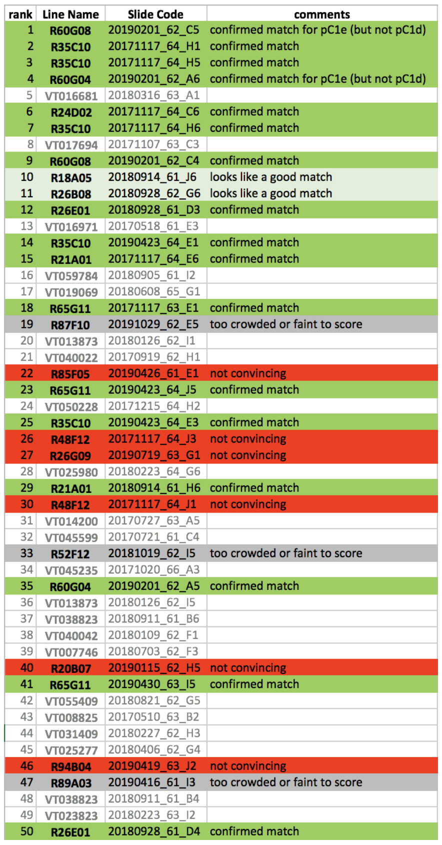
target neuron		known matching line		PatchPerPixMatch		
name	**hemibrain body id**	**line name**	**# samples**	**line rank**	**sample rank**	**comment**
pC1d	5813063587	35C10	12	4	4	
pC1d	5813063587	24D02	9	5	5	
pC1d	5813063587	26E01	13	7	7	
pC1d	5813063587	65G11	32	8	9	
pC1d	5813063587	21A01	14	11	12	
pC1d	5813063587	26F09	18	15	21	
pC1d	5813063587	20C08	15	41	58	
pC1d	5813063587	44D11	6	65	103	
pC1d	5813063587	45F11	6	>150	>150	false miss due to color shift within neuron
pC1d	5813063587	51B06	6	>150	>150	visual inspection of all samples does not reveal any convincing match
pC1d	5813063587	71A09	7	>150	>150	visual inspection of all samples does not reveal any convincing match
pC1d	5813063587	30G01	0	NA	NA	no MCFO acquisition available for this line
pC1e	514850616	60G08	12	1	1	
pC1e	514850616	35C10	12	2	2	
pC1e	514850616	60G04	8	3	4	
pC1e	514850616	24D02	9	5	6	
pC1e	514850616	26E01	13	9	12	
pC1e	514850616	21A01	14	11	15	
pC1e	514850616	65G11	32	14	18	
pC1e	514850616	44D11	6	41	56	
pC1e	514850616	20C08	15	44	60	
pC1e	514850616	26F09	18	56	83	
pC1e	514850616	45F11	6	>150	>150	false miss due to color shift within neuron
pC1e	514850616	51B06	6	>150	>150	visual inspection of all samples does not reveal any convincing match
pC1e	514850616	30G01	0	NA	NA	no MCFO acquisition available for this line

Fig. 9: PatchPerPixMatch line- and sample ranks for known matching lines for the pC1d and pC1e target neurons.



rank	Line Name	Slide Code	comments
1	VT055409	20180821_62_G5	
2	VT016681	20180316_63_A1	
3	VT025980	20180223_64_G6	
4	R35C10	20171117_64_H6	confirmed match
5	R24D02	20180921_61_C6	confirmed match
6	VT017694	20171107_63_C3	
7	R26E01	20180928_61_D4	confirmed match
8	R35C10	20171117_64_H1	confirmed match
9	R65G11	20190430_63_I5	confirmed match
10	VT013873	20180126_62_I1	
11	R60G04	20190201_62_A6	confirmed match for pC1e, but not pC1d
12	R21A01	20180914_61_H6	confirmed match
13	R60G08	20190201_62_C5	confirmed match for pC1e, but not pC1d
14	VT017694	20171107_63_C6	
15	R35C10	20190423_64_E3	confirmed match
16	R18A05	20180914_61_J6	appears to be match for pC1e, but not pC1d
17	R24D02	20171117_64_C6	confirmed match
18	R35C10	20171117_64_H5	confirmed match
19	R21A01	20171117_64_F1	confirmed match
20	VT031409	20180227_62_H3	
21	R26F09	20171117_65_E1	confirmed match
22	VT031409	20180227_62_H4	
23	R26B08	20180928_62_G6	appears to be match for pC1e, but not pC1d
24	VT007746	20180703_62_F3	
25	VT014200	20170727_63_A5	
26	R35C10	20190423_64_E1	confirmed match
27	VT019069	20180608_65_G2	
28	R93A02	20190419_62_I2	too crowded or faint to score
29	VT016971	20170518_61_E3	
30	R60G08	20190201_62_C4	confirmed match for pC1e, but not pC1d
31	VT019069	20180608_65_G1	
32	VT046289	20180605_61_D4	
33	VT018457	20171124_61_B4	
34	VT025277	20180406_62_G3	
35	VT018654	20170707_62_E2	
36	R80A07	20190709_61_F6	too crowded or faint to score
37	VT025996	20170524_61_D6	
38	VT008167	20180918_65_B3	
39	VT034630	20170721_65_E4	
40	R65G11	20190423_64_J5	confirmed match
41	VT054805	20190510_63_E1	
42	VT049923	20170721_62_B2	
43	R26G09	20190719_63_G1	not convincing
44	R65G11	20171117_63_E2	confirmed match
45	R26E01	20180928_61_D3	confirmed match
46	VT045235	20171020_66_A3	
47	R65H04	20190521_63_H5	not convincing
48	R21A01	20171117_64_E2	confirmed match
49	R65G08	20190329_61_I6	not convincing
50	R60G08	20190730_63_E1	confirmed match for pC1e, but not pC1d

(a) pC1d



rank	Line Name	Slide Code	comments
1	R60G08	20190201_62_C5	confirmed match for pC1e (but not pC1d)
2	R35C10	20171117_64_H1	confirmed match
3	R35C10	20171117_64_H5	confirmed match
4	R60G04	20190201_62_A6	confirmed match for pC1e (but not pC1d)
5	VT016681	20180316_63_A1	
6	R24D02	20171117_64_C6	confirmed match
7	R35C10	20171117_64_H6	confirmed match
8	VT017694	20171107_63_C3	
9	R60G08	20190201_62_C4	confirmed match
10	R18A05	20180914_61_J6	looks like a good match
11	R26B08	20180928_62_G6	looks like a good match
12	R26E01	20180928_61_D3	confirmed match
13	VT016971	20170518_61_E3	
14	R35C10	20190423_64_E1	confirmed match
15	R21A01	20171117_64_E6	confirmed match
16	VT059784	20180905_61_I2	
17	VT019069	20180608_65_G1	
18	R65G11	20171117_63_E1	confirmed match
19	R87F10	20191029_62_E5	too crowded or faint to score
20	VT013873	20180126_62_I1	
21	VT040022	20170919_62_H1	
22	R85F05	20190426_61_E1	not convincing
23	R65G11	20190423_64_J5	confirmed match
24	VT050228	20171215_64_H2	
25	R35C10	20190423_64_E3	confirmed match
26	R48F12	20171117_64_J3	not convincing
27	R26G09	20190719_63_G1	not convincing
28	VT025980	20180223_64_G6	
29	R21A01	20180914_61_H6	confirmed match
30	R48F12	20171117_64_J1	not convincing
31	VT014200	20170727_63_A5	
32	VT045599	20170721_61_C4	
33	R52F12	20181019_62_I5	too crowded or faint to score
34	VT045235	20171020_66_A3	
35	R60G04	20190201_62_A5	confirmed match
36	VT013873	20180126_62_I5	
37	VT038823	20180911_61_B6	
38	VT040042	20180109_62_F1	
39	VT007746	20180703_62_F3	
40	R20B07	20190115_62_H5	not convincing
41	R65G11	20190430_63_I5	confirmed match
42	VT055409	20180821_62_G5	
43	VT008825	20170510_63_B2	
44	VT031409	20180227_62_H3	
45	VT025277	20180406_62_G4	
46	R94B04	20190419_63_J2	not convincing
47	R89A03	20190416_61_I3	too crowded or faint to score
48	VT038823	20180911_61_B4	
49	VT023823	20180223_63_I2	
50	R26E01	20180928_61_D4	confirmed match

(b) pC1e

Fig. 10: Annotated first 50 PatchPerPixMatch hits for the pC1d and pC1e target neurons. Confirmed matches for pC1d stem from an exhaustive behavioral screen over all R lines [17]. To this effect the set of known matching R lines for pC1d exhibits a level of completeness. Completeness of known matching lines allows for determining false PatchPerPixMatch hits. To this end we ignored the VT line hits here.