# Next Steps: Detecting Worm Retractions and Freezing in Trial Clips

## Integrating Ideas from Tierpsy, OpenWorm Toolbox, and ezTrack

We will leverage proven techniques from **Tierpsy Tracker**, the **Open Worm Analysis Toolbox**, and **ezTrack** to design our feature-extraction pipeline. Tierpsy provides algorithms for **worm tracking and locomotion classification** (forward vs. backward movement, pauses) which we can emulate[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length)[[2]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Motion%20Mode%3A%20vector%20indicating%20if,frame%20that%20animal%20is%20in). The Open Worm Analysis Toolbox offers a framework to compute detailed **behavioral features** from worm trajectory data (it’s a Python port of Schafer Lab’s Worm Analysis Toolbox)[[3]](https://pypi.org/project/open_worm_analysis_toolbox/#:~:text=The%20Open%20Worm%20Analysis%20Toolbox,4). Meanwhile, ezTrack contributes a straightforward method for detecting **freezing** behavior based on motion thresholds[[4]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%E2%80%99s%20Freeze%20Analysis%20Module%20allows,order%20to%20set%20thresholds%20which). By combining these insights, we can build a pipeline that accurately identifies each **retraction** and **freezing** event in our trial clips, along with key latency metrics relative to the stimuli (CS and US).

## Worm Tracking: From Clitellum to Head

To detect subtle behaviors, we first need reliable **tracking of the worm’s movement** within the ROI. A key strategy is to track the worm’s **head position relative to its clitellum** (the clitellum is a thick band around segment ~30, nearer the anterior/head end)[[5]](https://christopherwren.weebly.com/uploads/6/5/0/0/6500204/earthwormbehaviors.pdf#:~:text=8,The%20first). This gives us a fixed body reference point. We will:

* **Isolate the Worm in Each Frame:** Use background subtraction or a static *reference frame* approach (as in ezTrack’s location module) to extract the worm’s silhouette every frame[[6]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%20Location%20Tracking%20Module,come%20directly%20from%20ezTrack%20output). This involves taking the difference between the current frame and a reference (e.g. a background or an initial frame without the worm’s movement) to locate the worm’s body.
* **Skeletonize or Contour-Track:** Once we have the worm’s outline, we can fit a skeleton or at least identify the worm’s endpoints. Tierpsy’s approach would be to compute the worm’s centerline and endpoints for each frame. We may implement a simpler version: find the worm’s head by looking for the end that moves or tapers, and track the clitellum by detecting the distinctive thicker segment or using a relative position along the skeleton. The clitellum’s position (since it’s a visible swelling near the head) helps maintain consistent orientation — ensuring we know which end is the head in each frame[[5]](https://christopherwren.weebly.com/uploads/6/5/0/0/6500204/earthwormbehaviors.pdf#:~:text=8,The%20first).
* **Track Head and Clitellum Over Time:** We will generate time-series data for the head position and clitellum position (e.g. their coordinates, or at least the distance between head and clitellum). By tracking the head-to-clitellum distance frame-by-frame, we can observe extensions vs. retractions of the head. Using the clitellum as an anchor is important so that if the worm changes direction, we don’t mis-label a forward movement as a “retraction.” In essence, the clitellum provides a consistent reference for the worm’s body, allowing us to measure **head movements relative to the body** rather than in absolute terms. This minimizes false positives – we won’t count the worm’s forward movements or random thrashing as “retraction” since those won’t show a *head pulling back toward the clitellum*.

By the end of this tracking step, for each frame of a trial we should know the worm’s head position (and orientation), enabling detection of specific behaviors in subsequent steps.

## Detecting Retraction Events

With the worm’s head trajectory in hand, we can identify **retraction** events – instances where the worm quickly pulls its head backward (a likely response to stimuli). To systematically detect these:

* **Define a Retraction Kinematically:** We consider a *retraction* equivalent to a backward movement of the worm’s head relative to its prior direction. Borrowing Tierpsy’s criteria for backward locomotion, we can define a retraction as **movement in the tail-direction sustained for at least 0.5 seconds and covering >5% of the worm’s body length**[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length). In practice, this means if the head-to-clitellum distance shortens by a significant amount within a short time window, it counts as a retraction. We will tune the exact threshold (e.g. percentage of body length, and time window) based on our worm’s size and video frame rate, but Tierpsy’s 0.5 s / 5% length is a proven starting point[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length).
* **Scan Each Trial for Retractions:** Using the above rule, we’ll scan through the head movement data of each trial clip. Each time the worm’s head retracts (e.g. a rapid decrease in head–clitellum distance), we mark a **retraction event**. There could be multiple retractions in one trial (for example, a worm might withdraw, then extend again and withdraw once more). All such events within the ROI will be detected and time-stamped (frame number or timestamp within the clip).
* **First Retraction After CS:** A particularly important measure for conditioning experiments is the **latency of the first retraction after the Conditioned Stimulus onset**. Since we know the frame when the CS (tone) starts (from our trial metadata), we will identify the **earliest retraction event that occurs after CS onset** in that trial. This gives the **CR latency** (conditioned response latency) for that trial. If the worm retracts during the CS period, that’s likely a learned response to the tone. We’ll record the frame/time of this first post-CS retraction. If no retraction happens during the CS at all, we note that as well (could indicate no response or a freeze instead).
* **First Retraction After US:** Similarly, we’ll capture the **latency of the first retraction following the Unconditioned Stimulus** onset. In paired trials, the US (e.g. a shock or reinforcement) might elicit an immediate withdrawal reflex. We measure from the known US start frame to the first retraction after that. This is essentially the **unconditioned response latency**. Even if a worm already retracted during the CS, the US might cause an additional reaction – but if the worm was already withdrawn, it might not retract again. We’ll have to handle that logic (e.g. if a retraction already occurred just before the US, we might either report “already retracted” or measure if there’s a secondary response).
* **Retractions in CS-only Trials:** For control trials where **no US is delivered** (unpaired CS-only trials), we still want to measure what the worm “would have done” when the US would normally occur. We know the expected timing of the US (for example, if in paired trials the US comes at 6 s into the clip, then in an unpaired trial we use 6 s as a reference point). We will check if the worm performs any retraction at or after that time point in the CS-only trial. Any retraction occurring in that window (had a US been present) will be noted. This will allow comparisons of spontaneous or baseline retraction behavior versus the true US-evoked responses. Essentially, we treat the “start of would-be US” as another reference marker in CS-only trials to time any withdrawal responses.

All the retraction events will be output with their timing. For instance, we might produce a list per trial: “Retraction events at frames X, Y, Z; first post-CS at frame A; first post-US at frame B,” etc. This satisfies the requirement of enumerating **each retraction in the ROI** and the special cases (first after CS, first after US, etc.).

## Detecting Freezing Periods

Next, we need to identify **freezing** behavior – periods where the worm **stops moving** (or moves below a minimal threshold) for a sustained duration. In the context of defensive behavior (like a worm responding to a stimulus by freezing), we define freezing operationally as *cessation of detectable motion*. To detect freezing episodes:

* **Motion Index Per Frame:** We will calculate a quantitative **motion index** for each frame or each short time bin. One approach (inspired by ezTrack’s Freeze Analysis module) is to count the number of pixels that changed from the previous frame beyond a certain grayscale threshold[[4]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%E2%80%99s%20Freeze%20Analysis%20Module%20allows,order%20to%20set%20thresholds%20which). If only a very small fraction of pixels are changing, the animal is essentially still. We can also incorporate the worm’s tracked movement: e.g., use the speed of the worm’s head or the overall displacement of the worm’s body between frames as a measure of motion.
* **Threshold for Freezing:** Following standard definitions, we’ll set a threshold on that motion index below which the worm is considered **immobile**. For example, ezTrack allows users to set a threshold number of pixels; if fewer than that number of pixels change between frames, it’s considered “zero movement”[[4]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%E2%80%99s%20Freeze%20Analysis%20Module%20allows,order%20to%20set%20thresholds%20which). We can calibrate this to our videos (worms might require a more sensitive threshold since their movements are small). Additionally, we require that motion stays below this threshold for a minimum time (the **minimum freeze duration**). Tierpsy’s definition of a “pause” is no movement for at least 0.5 s[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length), whereas in rodent studies freeze is often ≥1–2 s of no movement. We might choose, say, ~0.5 s or 1 s as the minimum freeze length, given worms might not pause for extremely long unless truly freezing.
* **Identifying Freezing Bouts:** We will slide through the video frames, and whenever the motion index falls below the threshold and stays low for the set duration, we mark a **freezing bout**. Each freezing bout has a start time (when the worm stopped moving) and an end time (when movement resumed). For example, if from frame 100 to frame 150 the worm had essentially no movement, that’s a 50-frame freeze (~1.67 s at 30 fps). We’ll log **each freezing event in the ROI** per trial. It’s important to double-check these against the video: we can use the visualizations like those in ezTrack (where the algorithm can overlay which frames were scored as freezing vs moving) to ensure our threshold aligns with human judgment[[7]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=pixels%20whose%20grayscale%20change%20exceeds,and%20click%20cropping%20tool%20allows). We may adjust the sensitivity so that small head oscillations or noise don’t falsely break a freeze.
* **Context of Freezing (during CS or after):** Once we have all freezing bouts, we can analyze when they occur. Often in conditioning, a freeze might occur *instead of* a retraction (e.g. an alternative defensive response). We will particularly note if a worm **freezes during the CS** presentation or right after the US. For instance, a worm might respond to the tone by freezing (no movement) rather than retracting – that would be an important conditioned response. Or a worm might freeze after a shock (post-US freezing). Our output for each trial can list all freeze episodes and we can flag if any overlap with the CS period or immediately post-US period.

By detecting freezing, we complement the retraction data – together these cover the main defensive responses (active withdrawal vs. passive immobility). The data for freezing will include each bout’s timing (start/end frames or duration) and count of bouts per trial.

## Alignment with Stimulus Onsets

To make the retraction and freezing data meaningful, we need to **align them to the timeline of each trial’s stimuli**. We have the **timestamps/framestamps for CS and US onsets** from our trial metadata (you mentioned having a CSV or chart of these). We will use those as reference points when analyzing each trial:

* **Using CS/US Frame Indices:** When processing a trial clip, we know at which frame the CS started (e.g. frame 30 if there’s a 1 s lead-in at 30 fps) and if applicable, at which frame the US started. We will likely set up our analysis such that time zero (or frame zero) corresponds to CS onset for convenience. In practice, we might trim the 1 s lead and treat the first frame of the tone as time 0. Regardless, having the exact frame indices for CS and US means we can convert any event’s frame number to a **time relative to CS or US**. For example, if the first retraction after CS is detected at frame 45 of the clip and CS started at frame 30, then the latency = 15 frames = 0.5 seconds after CS onset (if 30 fps). We’ll output that as the first-CS-retraction latency. Similarly, for US: if US began at frame 210 and a retraction happened at frame 220, that’s 10 frames (0.33 s) post-US latency.
* **CS-Only Trials (Virtual US Time):** In trials without a US, we will define a “virtual US onset” for analysis purposes – likely at the same interval where a US would normally occur. For instance, if in paired trials the US comes 1 s after CS ends, we use that timestamp in the CS-only trial as a reference. Then we check if any retraction occurred shortly after that point. We might mark, “Retraction at +0.2 s after *would-be US* in CS-only trial” if one occurred. This way, we can directly compare behavior in CS-only vs. paired trials at equivalent time points.
* **Output Data Structure:** The results will be tabulated in a clear format (CSV or similar). Each trial (clip) can have a row in a CSV file, containing: the trial ID, the condition (paired or unpaired), number of retractions, number of freezes, **first retraction latency after CS**, **first retraction latency after US (or would-be US)**, and perhaps total freezing duration or count during CS, etc. We will preserve all these measurements to test our hypotheses about the conditioning (e.g. a trained worm might show a shorter CS->retraction latency or more freezing during CS, etc.). Aligning everything to stimulus onsets ensures we can average or compare across trials properly.

In summary, this alignment step is straightforward since our pipeline will have the **exact frame indices of stimulus events**. It’s mostly about using those indices to calculate timings for the events we detect, and ensuring we handle the absence of US in control trials appropriately.

## Practical Implementation Steps

With the plan in place, here’s how we can execute it in code (building on our existing repository structure):

* **Frame Processing Pipeline:** For each trial video (8 s clip), read frames (e.g. using OpenCV in Python). We can incorporate this into a new module, perhaps named detect\_track.py as suggested by our project plan[[8]](file://file-RYcKRenYdPQWVUDuBodmJY#:~:text=,target). This module will loop through frames and perform the detection logic.
* Compute a background reference (which could be the first frame before the CS, since at that time the worm might be at rest, or we could use a running average). Then for each frame, do a subtraction: diff\_frame = abs(frame - reference\_frame). Threshold this diff to get a binary mask of moving regions[[6]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%20Location%20Tracking%20Module,come%20directly%20from%20ezTrack%20output). From this mask, either compute the worm’s **center of mass** or outline. This is analogous to ezTrack’s method which found the animal’s centroid from the frame differences[[9]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=reference%20frame%20and%20ROIs%20with,come%20directly%20from%20ezTrack%20output). For us, the centroid could be less useful (if the worm curls, centroid might not move much), so we might want the outline/skeleton.
* If using a skeletonization approach: apply morphological operations to clean the worm’s binary silhouette (fill holes, remove noise). Then use a skeletonization (Medial axis transform) to get a thin centerline. Identify the endpoints of the skeleton (there will be two endpoints for a single worm). Determine which endpoint is the head – since we know where the clitellum is roughly (e.g. if at segment 30, the head is the nearer end to that segment). We might need an initial calibration per worm (e.g. user clicks on head in first frame, or assume the worm’s orientation initially). Alternatively, one could detect the clitellum by looking for the thickest part of the worm’s body in the contour – that could serve as a marker to split head vs tail side. This part may require some image processing tuning (perhaps intensity or width along the worm’s body).
* Track the head and clitellum positions over time. If the worm moves smoothly, a simple nearest-neighbor tracking of these points frame-to-frame will work (the worm isn’t likely to disappear or teleport within the ROI). We store the distance between head and clitellum each frame, and perhaps the velocity (frame-to-frame change in that distance, or in positions).
* **Feature Detection:** In a separate module (e.g. features.py per plan[[8]](file://file-RYcKRenYdPQWVUDuBodmJY#:~:text=,target)), implement the logic to detect retractions and freezing from the tracked data:
* Iterate through the frames (or use the computed motion index vector). Mark frames that satisfy the retraction criteria (head moving backward). This could be done by checking the sign of the head movement: e.g., if head-to-clitellum distance at frame N is *smaller* than it was a few frames earlier (signifying a backward move), and the magnitude of that change > X (threshold), then flag a potential retraction. We might require it to persist for a few frames (to avoid counting one-frame jitter). This yields a list of frames or intervals where a retraction is happening. We can condense consecutive frames into one “event” (e.g. if frames 50–60 show a continuous backward motion, that’s one retraction event starting at frame 50).
* Likewise, compute the motion index (pixel changes or head speed). Mark segments of low motion for freezing. For example, use a sliding window of 0.5 s: if in that window the total movement is below threshold, and this continues for the next windows, then that period is freezing. Record the start and end frame of each freeze.
* As events are detected, check their timing relative to CS/US onsets. For each trial’s data, determine the first retraction after CS onset, etc., by simply comparing event frame indices to the known CS frame. Do the same for post-US.
* **Output Results:** Compile the detected features into a human-readable and analyzable format. A CSV output is ideal: each row = one trial, with columns for each metric (e.g. *FirstRetract\_CS, FirstRetract\_US, RetractCount, FreezeCount, FreezeDurations, etc*). Additionally, we might create an event log (e.g. a CSV with one row per event per trial, listing trial ID, event type, frame/time). This could help if we need to do raster plots or further analysis of when events occur.
* **Validation and Tuning:** It’s crucial to validate the algorithm on a few sample videos. We can take a couple of trial clips and overlay our detected events for sanity check. For example, generate a video playback where retraction events flash a marker on the worm, or freezing periods shade the video background. This will show if our thresholds are sensible. If we notice missed events or false positives, adjust the parameters (motion threshold, etc.). Since we expect relatively consistent behavior (all worms get the same stimuli), we can use one set of parameters for all trials. But if individual variation or lighting differences exist, we remain open to adjusting per session.
* **Leveraging Libraries (optional):** If we encounter difficulties in our own implementation, we have the option to incorporate parts of existing libraries. For instance, we could attempt to use Tierpsy Tracker in batch mode on our clips to get worm skeletons and their “motion mode” per frame[[2]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Motion%20Mode%3A%20vector%20indicating%20if,frame%20that%20animal%20is%20in), then simply parse that output for backward (retraction) and pause (freeze) events. Alternatively, since Open Worm Analysis Toolbox can calculate a rich set of features, we could feed our tracked skeleton data into it to obtain metrics like reversal count, etc., automatically[[3]](https://pypi.org/project/open_worm_analysis_toolbox/#:~:text=The%20Open%20Worm%20Analysis%20Toolbox,4). However, given our project’s scope, a custom solution tailored to our ROI and stimuli timing might be simpler and more transparent. (Using those tools as a benchmark is still a good idea to ensure our results make sense.)

By implementing these steps, we extend our pipeline beyond just clipping trials to actually **quantifying behavior** in those trials. This will directly address the outputs we need (retractions, freezes, latencies, etc.).

## Leveraging Existing Tools for Validation (Optional)

While our focus is on custom implementation, it’s worth noting how we can borrow and validate with existing toolkits:

* **Tierpsy Tracker for Motion States:** Tierpsy can classify every frame of a worm video as forward, backward, or paused movement[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length)[[2]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Motion%20Mode%3A%20vector%20indicating%20if,frame%20that%20animal%20is%20in). In our context, “backward” frames would correspond to retraction behavior, and “paused” frames to freezing (no movement). We could run Tierpsy on a few clips to see if it flags the same sections of video as backward motion that we identify as retractions. Their criterion (≥0.5 s of movement in tail direction to count as a reversal) is exactly what we’re adopting, so we expect strong agreement[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length). This cross-check can build confidence that our algorithm is picking up genuine events.
* **Open Worm Analysis Toolbox:** This toolbox, being a port of the Worm Analysis from Schafer’s lab, computes hundreds of features including locomotion metrics like reversal frequency, velocities, etc.[[3]](https://pypi.org/project/open_worm_analysis_toolbox/#:~:text=The%20Open%20Worm%20Analysis%20Toolbox,4). If we manage to get our worm’s skeleton data into the toolbox (for example, by formatting our tracking output as a WCON file or similar), it could directly output “reversal events” and “pause duration” for each video. We can use those as an additional validation. Even if we don’t fully integrate it, reviewing the definitions of features in OpenWorm’s docs can guide our implementations (they have well-defined stats for things like “time spent reversing vs pausing”).
* **ezTrack’s Freezing Algorithm:** We will follow ezTrack’s simple but effective approach for freezing detection: counting changed pixels and applying a threshold[[4]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%E2%80%99s%20Freeze%20Analysis%20Module%20allows,order%20to%20set%20thresholds%20which). Since ezTrack is open-source, we could even repurpose its code (written in Python notebooks) for our needs. For instance, ezTrack’s freeze module likely computes a frame-by-frame motion index (they mention plotting motion (blue) vs freezing (gray) traces and saving to CSV[[10]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%20Freeze%20Analysis%20Module,be%20saved%20to%20csv%20files)). We can do something similar and ensure that our freeze scoring correlates with manual observation. Notably, ezTrack was designed for rodents, but as the authors point out, it can be used for *“any experimental species”* by adjusting parameters[[11]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=used%20to%20analyze%20immobility%20in,the%20position%20of%20the%20animal). Worm videos have different scales of movement, so we’ll calibrate the pixel-change threshold accordingly.

Using these tools in parallel to our code will help answer any execution doubts. For example, if there’s a borderline case whether a slight movement counts as freeze or not, seeing how a tool like VideoFreeze or ezTrack would classify it can inform our threshold choice. In the end, however, our pipeline will produce its own outputs consistent with our experiment’s definitions.

## Summary of Expected Outputs

By following the above approach, our pipeline will produce the following outputs for each trial clip:

* **Retraction events:** A list (or count) of all head retractions detected within the ROI during the trial. This includes *when* they occurred (timestamps or frame numbers). We specifically highlight the **first retraction after CS onset** and **first retraction after US onset** (if applicable) as key metrics of interest. These latencies will be used to assess conditioned vs unconditioned response timings. In CS-only trials, any retraction occurring in the post-CS period (especially around the expected US time) will be noted for comparison.
* **Freezing events:** Identification of each freezing bout (period of immobility) in the trial. For each bout, we’ll have its start time, end time, and duration. We can count how many freezing bouts occurred and the total time spent freezing in the trial. We will pay attention to whether freezing coincides with the CS presentation or follows the US. For example, an output might say “freeze from 2.0–3.0 s (during CS)” or “freeze from 6.5–7.5 s (immediately after US)”. Such details could be insightful for understanding the worm’s defensive strategy (e.g. **freezing vs. fleeing**).
* **Frame tracking relative to CS/US:** We will have a clear mapping of frames to the timeline of stimuli. This means each event (retraction or freeze) can be reported as “occurred at X seconds after CS onset” etc. Having these normalized to stimulus onset makes it easier to aggregate data across trials.
* **Data format:** All these results will be compiled into a CSV or similar table. For instance, a single trial’s summary might look like: *Trial 5:* 2 retractions (first@0.8 s after CS, first@0.1 s after US), 1 freeze (duration 1.2 s during CS), etc. We will ensure the format is logical and easy to parse for statistical analysis or plotting.

In conclusion, the next steps involve implementing a robust tracking algorithm (inspired by Tierpsy/OpenWorm) to follow the worm’s head and body, and then applying event detection rules (inspired by Tierpsy’s locomotion criteria and ezTrack’s freeze scoring) to output every instance of retraction and freezing in our trial clips. By tracking from the clitellum to the head, we anchor our measurements to the worm’s anatomy, avoiding spurious detections of retraction that might occur if we only tracked the whole worm’s centroid. And by using frame-differencing and pixel-change thresholds, we can sensitively detect when the worm **truly freezes (stops moving)**[**[4]**](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%E2%80%99s%20Freeze%20Analysis%20Module%20allows,order%20to%20set%20thresholds%20which). All these measures will be aligned to the CS/US timeline, fulfilling the requirement of tying behavioral outputs to stimulus events.

Ultimately, this pipeline will yield a rich dataset: for each trial we’ll know **how many times the worm retracted or froze, and exactly when**, relative to the stimuli. These results will allow us to quantitatively assess the worms’ responses in our conditioning paradigm and move on to the analysis and visualization phase with confidence in the underlying behavioral scoring. [[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length)[[5]](https://christopherwren.weebly.com/uploads/6/5/0/0/6500204/earthwormbehaviors.pdf#:~:text=8,The%20first)

[[1]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Forward%3A%20The%20animal%20is%20moving,of%20its%20length) [[2]](https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/#:~:text=Motion%20Mode%3A%20vector%20indicating%20if,frame%20that%20animal%20is%20in) Association of Two Opposing Responses Results in the Emergence of a Novel Conditioned Response - PMC

<https://pmc.ncbi.nlm.nih.gov/articles/PMC9102977/>

[[3]](https://pypi.org/project/open_worm_analysis_toolbox/#:~:text=The%20Open%20Worm%20Analysis%20Toolbox,4) open\_worm\_analysis\_toolbox · PyPI

<https://pypi.org/project/open_worm_analysis_toolbox/>

[[4]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%E2%80%99s%20Freeze%20Analysis%20Module%20allows,order%20to%20set%20thresholds%20which) [[6]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%20Location%20Tracking%20Module,come%20directly%20from%20ezTrack%20output) [[7]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=pixels%20whose%20grayscale%20change%20exceeds,and%20click%20cropping%20tool%20allows) [[9]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=reference%20frame%20and%20ROIs%20with,come%20directly%20from%20ezTrack%20output) [[10]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=ezTrack%20Freeze%20Analysis%20Module,be%20saved%20to%20csv%20files) [[11]](https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4#:~:text=used%20to%20analyze%20immobility%20in,the%20position%20of%20the%20animal) ezTrack: An open-source video analysis pipeline for the investigation of animal behavior | Scientific Reports

<https://www.nature.com/articles/s41598-019-56408-9?error=cookies_not_supported&code=ead484e6-9a52-44fd-aefb-5f1e34a4f7d4>

[[5]](https://christopherwren.weebly.com/uploads/6/5/0/0/6500204/earthwormbehaviors.pdf#:~:text=8,The%20first) Microsoft Word - Earthworm Behaviors.doc

<https://christopherwren.weebly.com/uploads/6/5/0/0/6500204/earthwormbehaviors.pdf>

[[8]](file://file-RYcKRenYdPQWVUDuBodmJY#:~:text=,target) project\_status\_9-14.json

<file://file-RYcKRenYdPQWVUDuBodmJY>