Crosshair, semi-automated targeting for electron microscopy with a motorised ultramicrotome

Kimberly Meechan[1, 2*], Wei Guan[3], Alfons Riedinger[4], Vera Stankova[4], Azumi Yoshimura[3], Rosa Pipitone[1], Arthur Milberger[5], Helmuth Schaar[5], Inés Romero-Brey[1], Rachel Templin[1, 7], Christopher J Peddie[3], Nicole L Schieber[1, 8], Martin L Jones[3], Lucy Collinson[3], and Yannick Schwab[1*]

[1]Cell Biology and Biophysics Unit, European Molecular Biology Laboratory (EMBL) Heidelberg, Germany
[2]Collaboration for joint PhD degree between EMBL and Heidelberg University, Faculty of Biosciences
[3]Francis Crick Institute, London, UK
[4]Electronic Workshop, European Molecular Biology Laboratory (EMBL) Heidelberg, Germany
[5]Mechanical Workshop, European Molecular Biology Laboratory (EMBL) Heidelberg, Germany
[6]Electron Microscopy Core Facility, European Molecular Biology Laboratory (EMBL) Heidelberg, Germany
[7]Current address: Ramaciotti Centre for Cryo-EM, Monash University, Melbourne, Australia
[8]Current address: Centre for Microscopy and Microanalysis, University of Queensland, Brisbane, Australia
[*]Corresponding authors: kimberly.meechan@embl.de; yannick.schwab@embl.de

Abstract

Volume electron microscopy (EM) is a time consuming process - often requiring weeks or months of continuous acquisition for large samples. In order to compare the ultrastructure of a number of individuals or conditions, acquisition times must therefore be reduced. For resin-embedded samples, one solution is to selectively target smaller regions of interest by trimming with an ultramicrotome. This is a difficult and labour-intensive process, requiring manual positioning of the diamond knife and sample, and much time and training to master. Here, we have developed a semi-automated workflow for targeting with a modified ultramicrotome. We adapted two recent commercial systems to add motors for each rotational axis (and also each translational axis for one system), allowing precise and automated movement. We also developed a user-friendly software to convert X-ray images of resin-embedded samples into angles and cutting depths for the ultramicrotome. This is provided as an open-source Fiji plugin called Crosshair. This workflow is demonstrated by targeting regions of interest in a series of *Platynereis dumerilii* samples.

1 Introduction

Imaging samples with electron microscopy (EM) is a time consuming process - often requiring weeks or months of continuous acquisition for large samples[1–3]. This means that it is rarely

feasible to acquire image volumes from entire specimens, and we must instead target specific regions of interest. For resin-embedded samples, this process is usually done with an ultramicrotome that allows manual trimming of the block using a razor blade and/or diamond knife, followed by cutting of thin sections at precise locations.

Targeting with an ultramicrotome is a difficult and manual process. The operator must rely on surface features that are visible through the ultramicrotome's binocular microscope. This makes it challenging to target regions deep within a sample, especially when heavy metals make the sample opaque. In addition, the orientation of cutting must be set manually by adjusting three different ultramicrotome axes. This kind of 3D thinking is challenging, taking much time and training to master.

Other studies have improved the precision and ease of targeting by leveraging external 3D maps from light or X-ray microscopy[4–9]. X-ray offers many advantages for EM targeting - for example, it is readily compatible with standard EM sample preparation methods[4, 8, 10, 11], and provides images that highlight similar structures to volume EM (although at a lower resolution)[7, 8, 10, 11]. Laboratory micro-CT systems are becoming ever more popular and can provide an isotropic resolution of about one micron, with scan times in the range of a few hours[5, 12]. In addition, there is increasing access to synchrotrons for X-ray imaging, which can provide resolutions in the range of hundreds or even tens of nanometers, and scan times in the range of minutes (depending on the resolution required)[10–12]. X-ray imaging therefore offers a fast, non-destructive method for obtaining 3D maps of the internal features of a sample for targeting.

Most methods that use light or X-ray microscopy for targeting rely on measuring the depth of regions of interest from the resin block's surface, followed by trimming with an ultramicrotome. As these methods don't compensate for the exact position and orientation of the sample in the ultramicrotome, progress must be checked at regular intervals by iterative rounds of X-ray or light microscopy and trimming, slowing this process down. In addition, these methods focus on the depth of the region of interest, but still require the orientation of cutting to be set entirely manually.

Other studies have sought to bring more automation to the ultramicrotome, to make it easier to use[13–15]. These efforts have mostly focused on the collection of serial sections, with little automation of the initial trimming and targeting process. An exception is Brama et al.[16], in which a miniature fluorescence microscope was integrated into the ultramicrotome, allowing automated tracking of regions of interest during cutting. However, this requires a fluorescent signal to be present within the sample embedded in the resin block, and is therefore not compatible with standard heavy-metal stained specimens. Also, while this technique allows detection and collection of sections from regions of interest, it does not automate the depth or orientation of ultramicrotome cutting.

Here, we created a workflow for semi-automated targeting of regions of interest with a modified ultramicrotome. We introduced automation to two recent commercial systems in the form of motorisation for each of the ultramicrotome axes. This allowed precise and automated angular movement for both systems, as well as translational movement for one system. We also created new software that allows selection of a plane of interest from X-ray images, and automatic

conversion into angles and cutting depths for the ultramicrotome. This is provided as an open-source, user-friendly Fiji[17] plugin called Crosshair.

2 Results

2.1 Automation of ultramicrotome

We modified two of the most recent commercial ultramicrotomes (Leica EM UC7 and RMC PowerTome PC (PTPC)) to add automation of movement (Figure 1 and Supplemental Figure S1). For the UC7, this consisted of three extra motors (one for each rotational axis) controlled by a small Raspberry Pi computer. This system can be controlled via a simple touchscreen interface (developed with Node-RED), allowing the angles of each axis to be precisely set and measured (see methods section 6.2).

For the RMC, motors were added to the same rotational axes, with additional automation of the two translational stage movements. The five motors were controlled by a Trinamic TMCM-6110 6-axis stepper motor driver board. The graphical user interface (GUI) was developed with Labview and installed onto a touchscreen PC, allowing users to control the motions via keyboard/mouse, touchscreen or a joystick (see methods section 6.2).

Both systems were calibrated with special adaptors made to convert a rotational movement into the rotation of the dial of a Thorlabs CRM1PT/M rotation mount, whose vernier scale provides 5 arcmin (0.083 degree) resolution (see methods section 6.2).

2.2 Targeting calculations

Our goal was to allow automatic calculation of the ultramicrotome moves required to cut to a specific target plane. The target plane defined the final position and orientation of the desired block surface (Figure 2A). Thin sections could then be taken for TEM (Transmission Electron Microscopy) or array tomography, or the surface could be imaged directly with volume SEM (Scanning Electron Microscopy).

This targeting problem was broken down into two steps: that of orientation, and that of distance. For the orientation problem, we found a combination of three angles (knife tilt, sample tilt, sample rotation, Figure 1A) that brought the target plane to be vertical, and parallel to the knife (Figure 2B). For the distance problem, we found the depth to cut from the sample surface to reach the target plane (Figure 2C).

The orientation problem required a mathematical description of how the target plane orientation changed as the knife tilt, sample tilt and sample rotation were varied. This was a similar problem to that which must be solved to, for example, orient the 'end-effector' of a robotic arm by varying the angle of its individual joints. In robotics, this is solved by construction of a 'forward kinematics' equation that calculates the end-effector position, given the angles of the joints, and

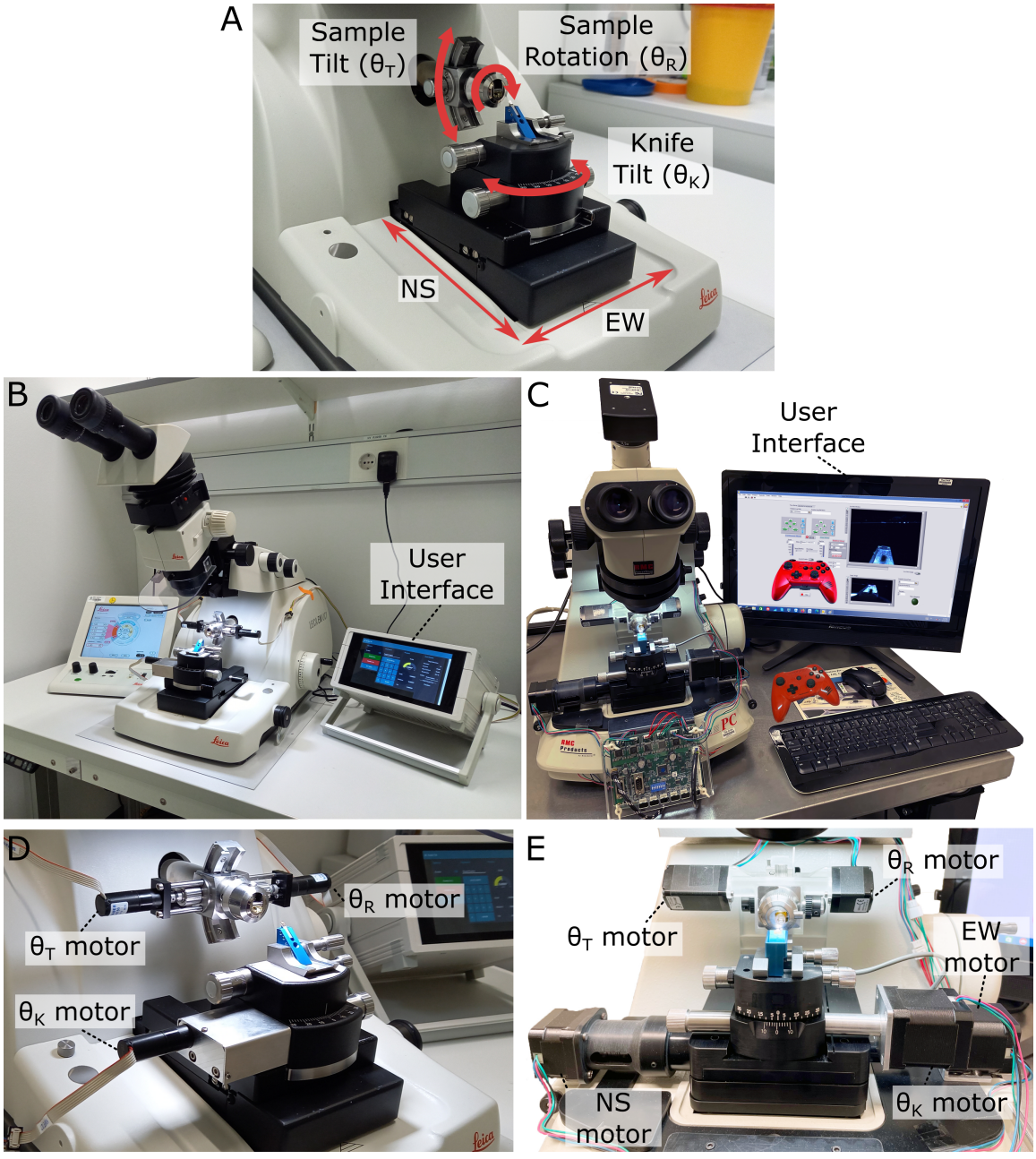

Figure 1: Motorised ultramicrotome. **A** - summary of the main ultramicrotome axes of rotation (sample tilt, sample rotation and knife tilt), and movement (NS/EW), which are common to both Leica and RMC systems. **B** - overview of motorised Leica system. **C** - overview of motorised RMC system. **D** - Zoom of *B*, showing motors attached to each axis. **E** - Zoom of *C*, showing motors attached to each axis.

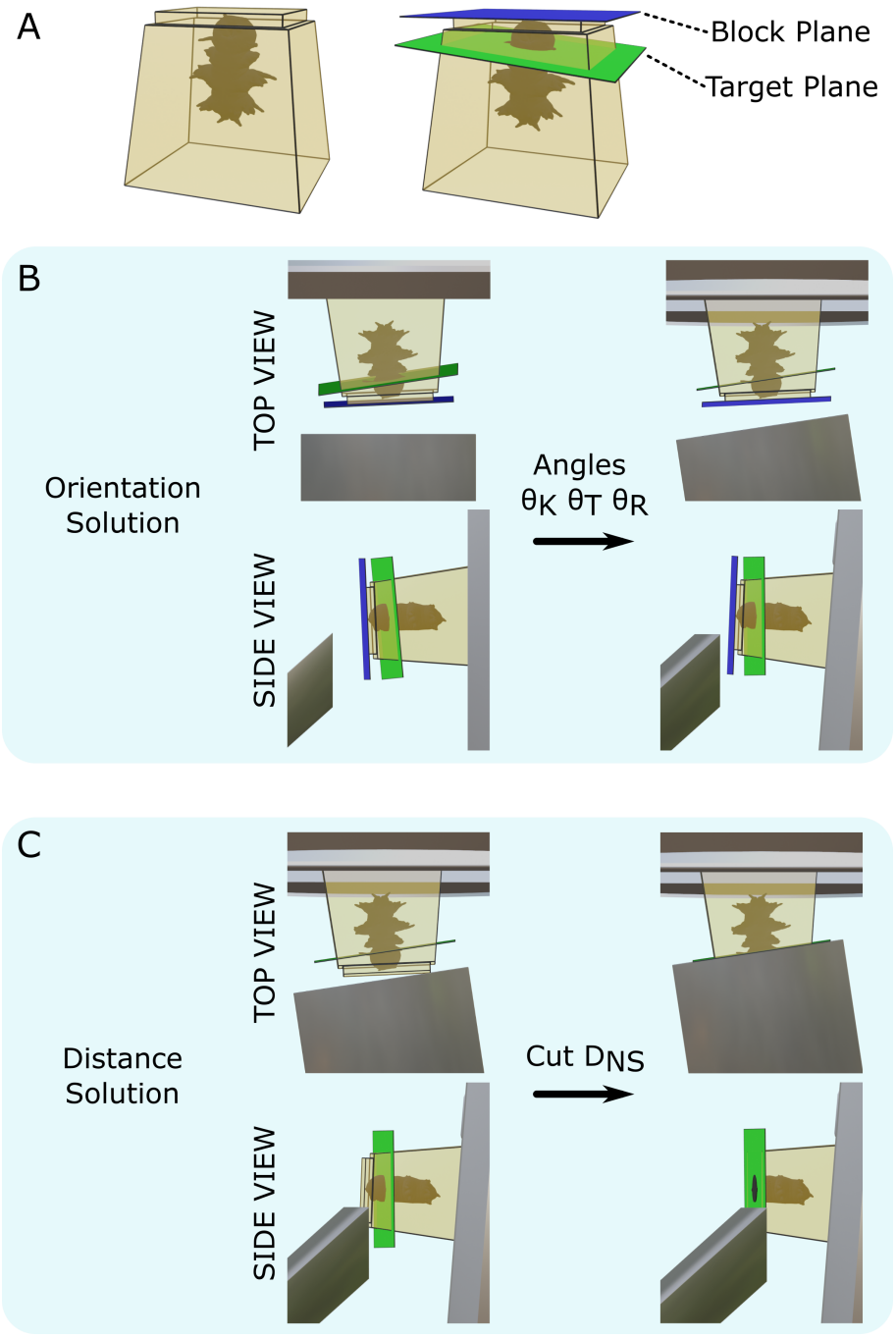

Figure 2: Targeting problem. **A** - Left, diagram of a resin block with a *Platynereis* sample inside. The top surface has been trimmed flat with a diamond knife. Right, same resin block with labelled block plane (i.e. the plane parallel to the flat trimmed surface), and target plane. **B** - diagram of the block in the ultramicrotome before (left) and after (right) the orientation solution is applied. A top view (with diamond knife at the bottom, and sample holder at the top) and side view (with diamond knife on the left, and sample holder on the right) are shown. θ_K is the knife tilt angle, θ_T is the sample tilt angle and θ_R is the sample rotation angle. **C** - diagram of the block in the ultramicrotome before (left) and after (right) the distance solution is applied. Top/side view are the same as in *B*. D_{NS} is the NS cutting distance.

an 'inverse kinematics' equation, that calculates the joint angles, given the desired end-effector position. We therefore applied similar principles, constructing a forward and inverse kinematics equation for the ultramicrotome.

We first constructed the forward kinematics equation to describe the orientation of the target plane, in terms of the three input angles. To do so, we defined orthogonal coordinate frames for each of the ultramicrotome parts (e.g. the knife holder, the sample holder etc.) (Supplemental Figures S5, S6), and calculated the rotations that relate each piece. This is summarised in the pose diagram in Figure 3A. The relation between the sample holder and sample block surface is unknown in this pose diagram. This is because the sample holder does not hold the sample in a fixed orientation, meaning the resin block has a slightly different orientation every time it is placed inside.

To calculate this relation, we therefore required an initial alignment step where the knife is manually aligned to the block face. This is standard procedure for cutting with an ultramicrotome, and so should be familiar to anyone who has used a ultramicrotome previously. In this aligned position, the orientation of the knife and block coordinate frames were the same, allowing the holder to block relation to be calculated (Figure 3B).

This meant that the full forward kinematics equation, describing the rotation from the World coordinate frame to the target plane was:

$$\boldsymbol{F} = \boldsymbol{R_x}(\theta_T)\boldsymbol{R_y}(\theta_R)\boldsymbol{R_x}(-\theta_{IT})\boldsymbol{R_z}(\theta_{IK})\boldsymbol{R_z}(\theta_{to})\boldsymbol{R_x}(\theta_{tr})$$

where $\boldsymbol{R_x}$, $\boldsymbol{R_y}$ and $\boldsymbol{R_z}$ are rotation matrices about the x, y and z axis respectively. θ_T is the sample tilt angle and θ_R is the sample rotation angle. θ_{IT}, θ_{IK}, θ_{to}, and θ_{tr} are constants for a particular targeting run representing the initial sample tilt angle (on alignment), the initial knife tilt angle (on alignment), the offset angle between the block face and the target plane (measured from X-ray), and the rotation angle between the block face and the target plane (measured from X-ray). See Supplemental figure S6 for details of how θ_{to} and θ_{tr} are calculated.

With the forward kinematics equation complete, we then constructed and solved an inverse kinematics equation to determine the required angles for each solution (see methods section 6.3). This gave the solution tilt and knife angles as:

$$\theta_T = \arctan\Big(C_1\cos(\theta_R) + C_2\sin(\theta_R)\Big)$$

$$\theta_K = \arctan\left(\frac{C_3\Big(C_4\cos(\theta_R) + C_5\sin(\theta_R)\Big)}{\left(\sqrt[2]{{C_3}^2 + \Big(C_4\sin(\theta_R) - C_5\cos(\theta_R)\Big)^2}\right)|C_3|}\right)$$

where C_{1-5} are constants for a particular targeting run (see methods for details). This means that, as expected, there are many possible solutions for each targeting setup. In fact any point on these lines (Figure 3C) is a possible solution. Note that there will be some situations where no solution is possible due to the constraints of the ultramicrotome axes e.g. in Figure 3D, the sample tilt of some solutions exceeds the maximum sample tilt of 20 degrees. In these situations, the sample would need to be re-mounted, or re-trimmed to aim for an angular difference within the ultramicrotome constraints.

For the distance problem, we calculated the distance between the block surface and target plane (from the X-ray images), and compensated for the current knife angle to give the true cutting distance:

$$D_{NS} = \frac{D_P}{\cos(\theta_K)}$$

where D_{NS} is the NS distance (cutting distance), D_P is the perpendicular distance between the target plane and the furthest surface point, and θ_K is the knife angle (see methods section 6.4 and Supplemental Figure S7).

2.3 Crosshair targeting workflow

The entire workflow is summarised in Figure 4, with details in the methods (section 6.7). In brief, the resin block was first trimmed manually close to the sample, with a rectangular face in its surface. This block face (which can be any shape with four corners, and straight lines in between) provided the reference to define the block coordinate frame (Figure S5E).

Next, the sample was imaged with X-ray at the highest resolution possible (around one micron isotropic voxel size is feasible on lab-based micro-CT systems). From this X-ray image, two planes were defined - one on the flat block surface, and one on the target plane using the Crosshair plugin (see the Crosshair section below and Figure 2A). Then, the corners of the block surface were marked in the software, as these were used to determine where cutting would begin and the distance required. Finally, the orientation was decided i.e. which edge of the block would face upwards in the ultramicrotome.

Next, we moved to the ultramicrotome. The zero position of the sample tilt and knife tilt axes were checked (see methods), to ensure all angular measures began from the correct position. Then, the resin-embedded sample was inserted, and the knife and block face manually aligned. This aligned position was defined as the zero position for the sample rotation, to simplify the targeting calculations.

The aligned sample and knife tilt angles (θ_{IT} and θ_{IK}) were then precisely read from the motorised ultramicrotome system, and entered into the targeting calculations. This allowed all solutions to be calculated, and one selected by the ultramicrotome user. For example, the user may want to minimise the knife and sample tilt angles to make it easier to collect thin sections after targeting. Finally, the ultramicrotome was precisely set to the three solution angles using the motors, and the solution distance cut.

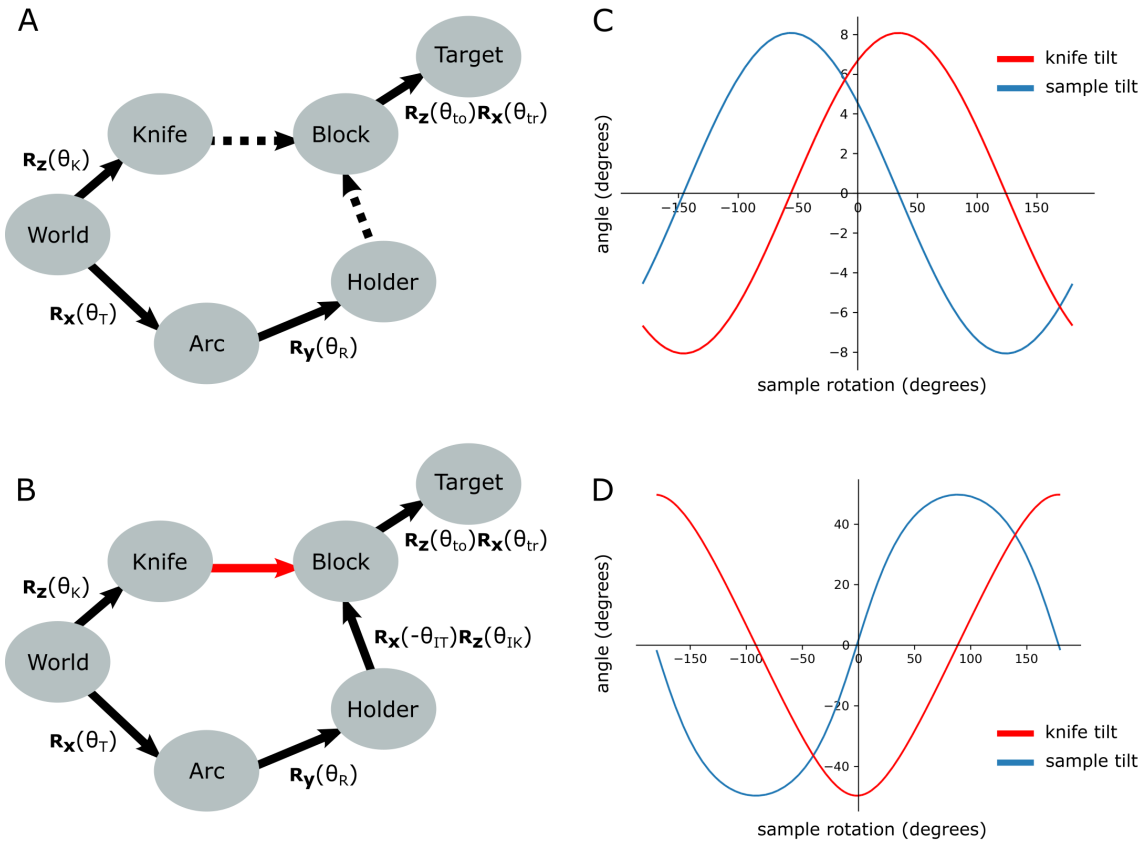

Figure 3: Pose diagrams and solutions. **A** - pose diagram showing relations between coordinate frames. $\boldsymbol{R_x}$, $\boldsymbol{R_y}$ and $\boldsymbol{R_z}$ are rotation matrices about the x, y and z axis respectively. θ_T is the sample tilt angle, θ_R the sample rotation angle, θ_K the knife angle, θ_{to} the target offset angle, and θ_{tr} the target rotation angle. Dashed arrows are unknown relations. **B** - pose diagram when the knife is aligned to the block face. Here, the knife coordinate frame is in the same orientation of the block (red arrow), and therefore we can infer the Holder to Block relation that was unknown in *A*. θ_{IT} is the initial sample tilt angle and θ_{IK} is the initial knife tilt angle, when the knife and block are aligned. We define θ_R to be zero at this aligned orientation. **C** - graphs of knife and sample tilt solution for all sample rotation angles. This used values of $\theta_{IK} = 10$, $\theta_{IT} = 10$, $\theta_{to} =$ -3.3 and $\theta_{tr} = 5.4$. **D** - solution graphs with more extreme initial angles. $\theta_{IK} = 10$, $\theta_{IT} = 10$, $\theta_{to} =$ -60 and $\theta_{tr} = 5.4$.

At this point, thin sections could be taken for TEM (transmission electron microscopy), or array tomography, or the block could be transferred for volume imaging in a FIB-SEM (Focused Ion Beam Scanning Electron Microscope) or SBF-SEM (Serial Block Face Scanning Electron Microscope).

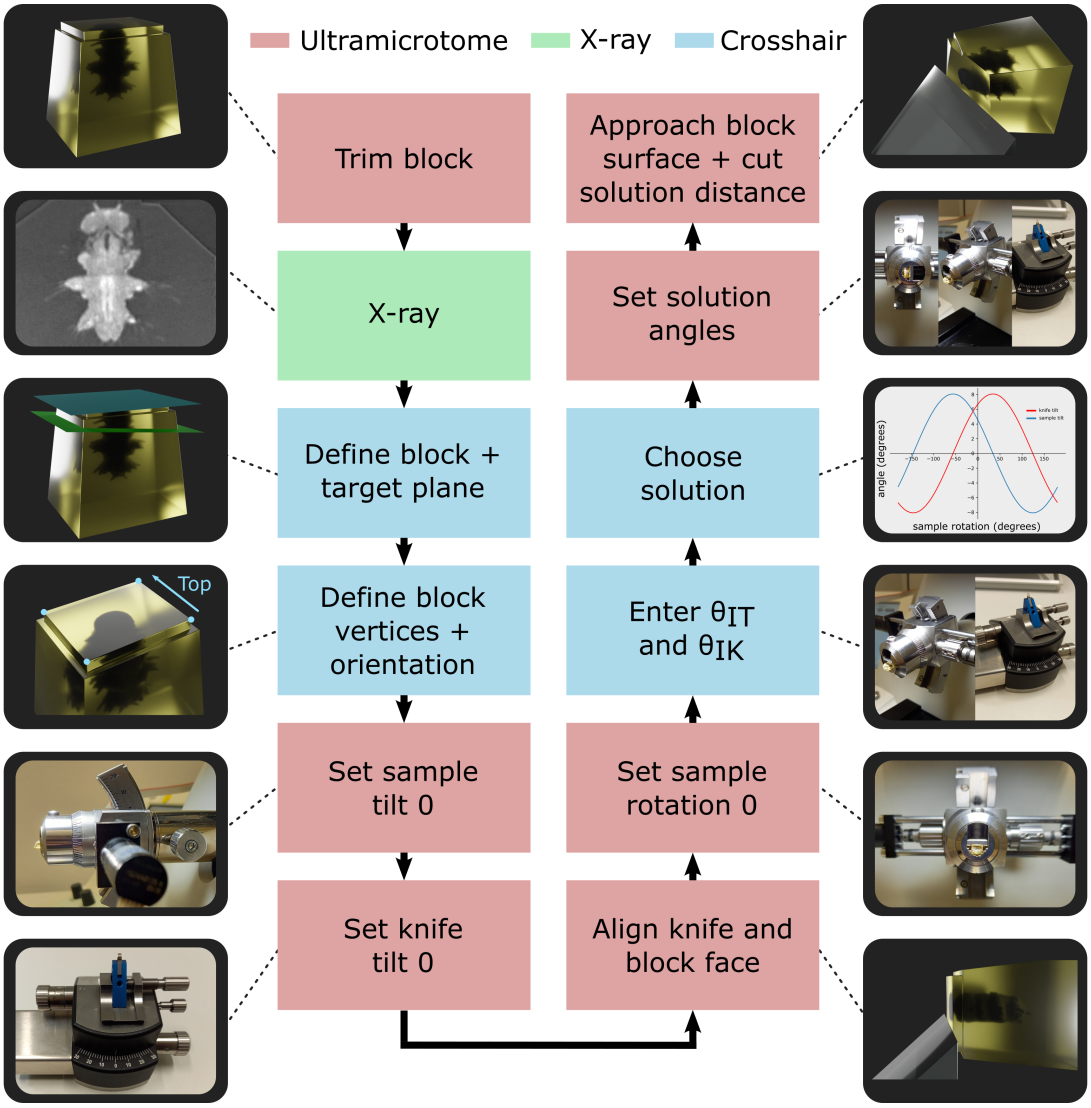

Figure 4: Targeting workflow. Summary of targeting workflow steps with representative images / 3D renders.

2.4 Crosshair

To make this workflow accessible to all, we developed the open-source Fiji[17] plugin Crosshair (`https://github.com/automated-ultramicrotomy/crosshair`) which allows user-friendly measurement of X-ray images, and access to the calculations described above (Figure 5 and Supplemental Figure S8). It uses BigDataViewer[18] to allow browsing of large images in arbitrary

2D slices, and the ImageJ 3D Viewer[19] to allow browsing in 3D.

Crosshair allows the planes and points required for the targeting workflow (Figure 4) to be created and viewed interactively. It also allows the orientation of the block to be marked, and visualised in 2D and 3D.

To make selection of solutions easier, Crosshair also has a 'microtome mode' which allows a simple representation of the ultramicrotome to be manipulated in 3D (Figure 5B). This has the constraints of the ultramicrotome built in, and displays a volume rendering of the X-ray imaged sample at the appropriate orientation. In this way, users can interactively move sliders in the user interface and view the orientation of the ultramicrotome and sample in real time. This also outputs the corresponding angles and cutting distance for each solution.

Finally, Crosshair also has a 'cutting mode' which allows the predicted cutting progression to be viewed in 2D and 3D (Supplemental Figure S8C). In this way, users can choose a solution and then easily view which structures will be visible at which cutting distances. This makes it simple to choose an appropriate solution for their targeting problem.

2.5 Accuracy measures

To test the accuracy of this workflow, we targeted a series of 6 day old *Platynereis dumerilii* samples. *Platynereis* is a marine annelid, that is an established model system for development, evolution and neurobiology[20–22]. As they can be bred quickly and easily in the lab, and have well-defined tissue morphology, we decided to use them as a test case for targeting regions of interest. We set our target plane on each sample to cut through both of the anterior dorsal cirrus that protrude from either side of the *Platynereis* head.

All samples were targeted with the workflow described above - five on the motorised Leica system (samples a-e), and five on the RMC system (samples f-j). After targeting, samples were X-ray imaged again, and registered to the pre-targeting scan. This registration was completed with another Fiji plugin we created - RegistrationTree - that is also freely available on github: `https://github.com/K-Meech/RegistrationTree`. This plugin is a wrapper around two existing registration software, BigWarp[23] and elastix[24, 25], allowing easy passing of images from one software to the other, and smooth use of very large images with elastix. It is standalone, so it can also be used for general registration of large images, independent of the Crosshair workflow.

Once the images were registered, we calculated a series of angle and distance accuracy measures using the built-in features in the Crosshair plugin (Figures 6-7, and see methods section 6.9).

Comparison of the target plane and trimmed block surface showed that all samples hit their target of cutting through both anterior dorsal cirrus, and provided similar *Platynereis* cross-sections to those predicted from the X-ray (Figure 7). The accuracy measures gave a mean angular error of 0.9 degrees, a mean absolute solution distance error of 3.1 microns, and a mean point to plane distance error of 3.1 microns (see methods section 6.9 for details of these different measures). Note that there will be some error in these distance measures due to slight

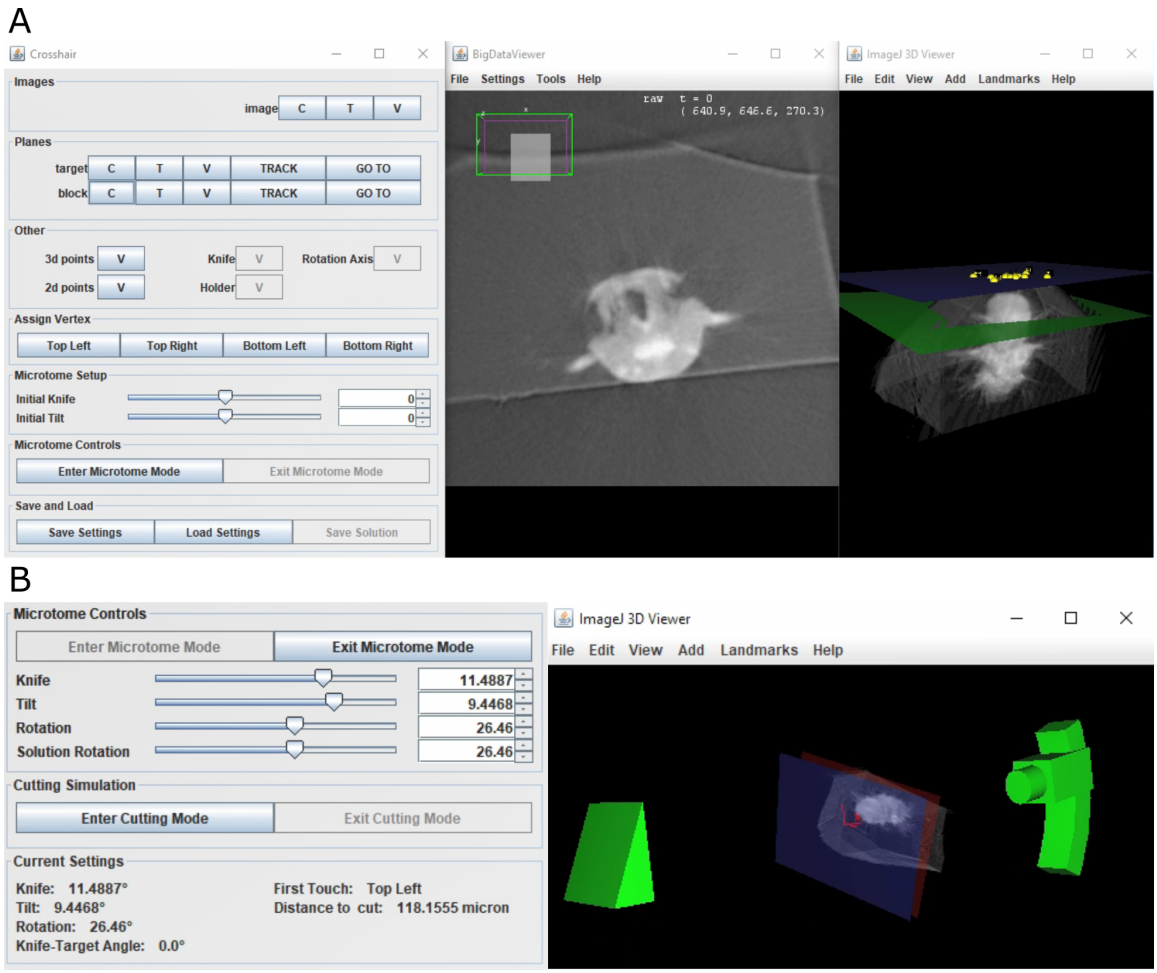

Figure 5: Crosshair. **A** - Crosshair plugin user interface, with an example *Platynereis dumerilii* X-ray image. Left - controls. Middle - BigDataViewer window showing 2D cross-section. Right - ImageJ 3D viewer showing volume rendering with block and target planes (blue and green respectively). The green target plane is the same as the 2D slice shown. **B** - left, microtome mode controls and solution display. Right, corresponding 3D representation of ultramicrotome and sample.

inaccuracies in the registration process - estimated at around 2 microns error (see methods section 6.9 and Figure S9). Given that our X-ray voxel size is only one micron, there is a limit to how precisely such small distances can be measured. This could be improved in future by moving to higher resolution X-ray imaging from synchrotrons, which can provide resolutions in the range of hundreds of nanometers.

Note that there were also some differences between the mean targeting accuracy reached with the Leica and RMC systems. The Leica system had a mean angular error of 1.1 degrees, mean absolute solution distance error of 4.5 microns, and a mean point to plane distance error of 4.8 microns. The RMC system had a mean angular error of 0.7 degrees, mean absolute solution distance error of 1.6 microns, and a mean point to plane distance error of 1.3 microns. There are a number of factors that could have contributed to this difference. Firstly, the samples tested with the Leica system were X-ray imaged with a different micro-CT system to those tested on

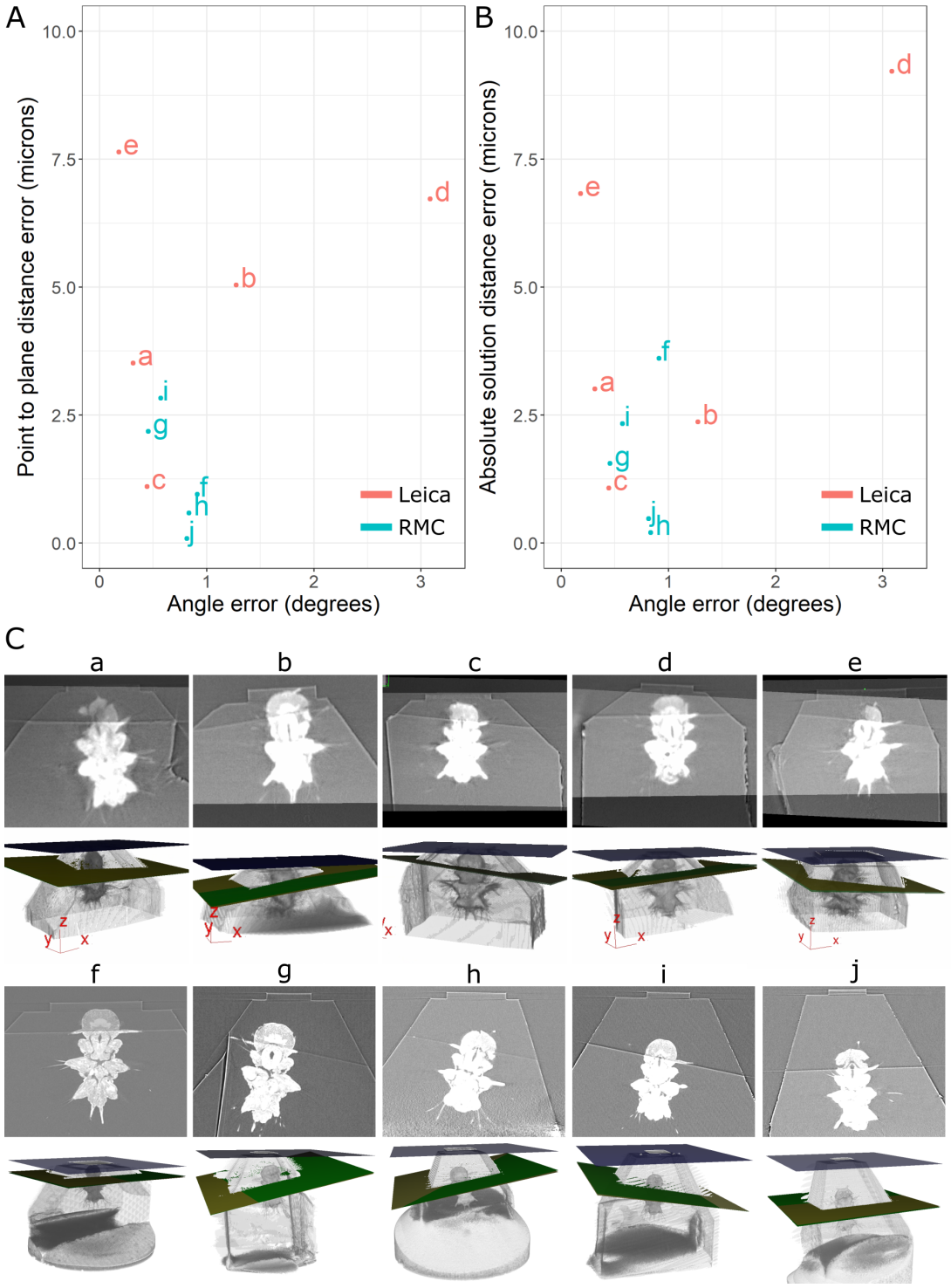

Figure 6: Accuracy test results. **A** - Graph of point to plane distance error vs angle error for all 10 samples (a-j). **B** - Graph of absolute solution distance error vs angle error. **C** - For each sample (a-j), two images are shown. Top - registered before and after X-ray images. Bottom - snapshot of the 3D viewer in Crosshair. In the 3D view, the block surface is shown in blue, the target in green, and the plane reached in yellow. In most cases, the green and yellow planes are too close together to distinguish well. The axes shown in some of these images is the 3D viewer coordinate system.

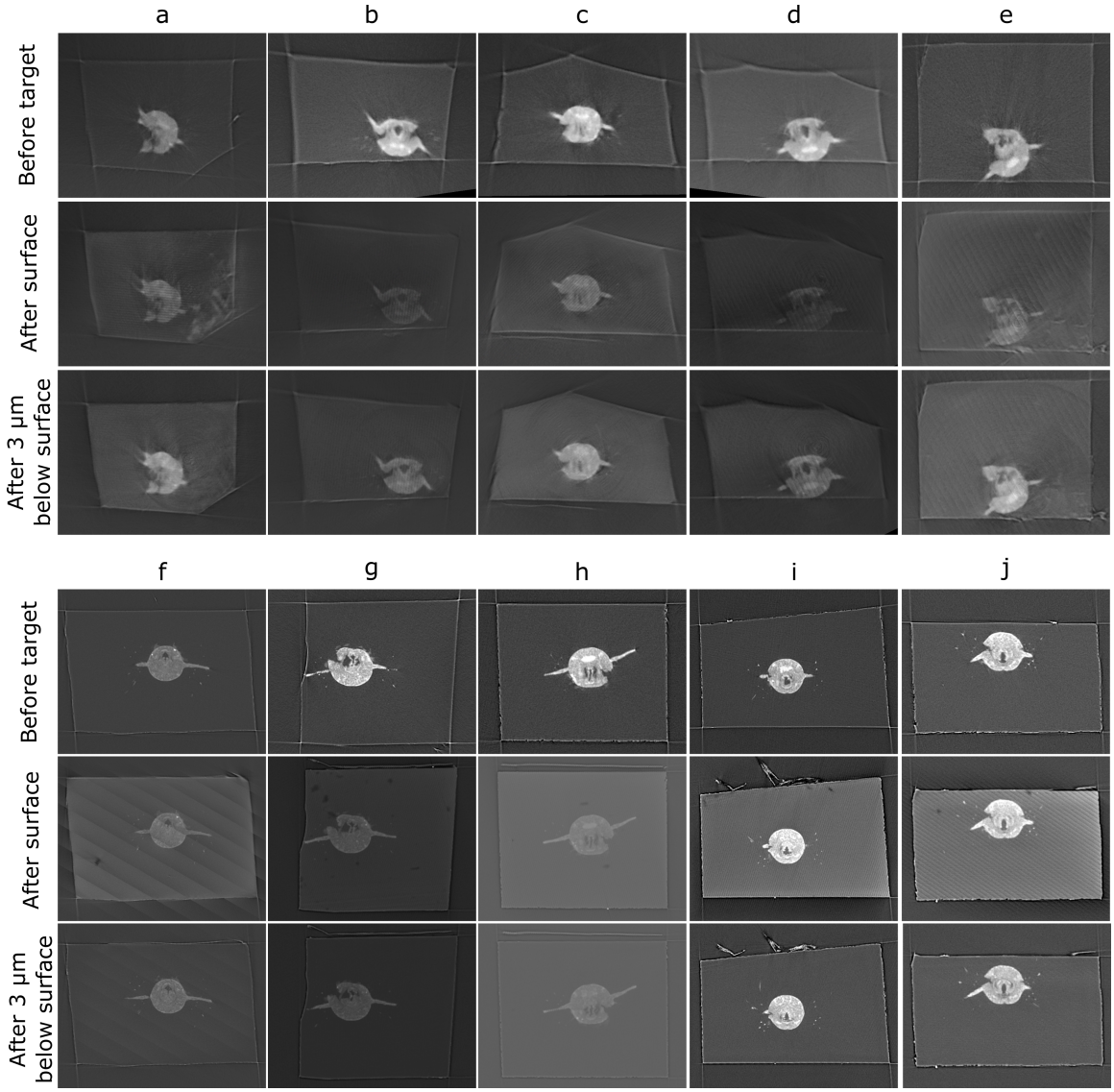

Figure 7: Comparison of target plane (from the before scan), to the block surface (after scan). Each sample (a-j) has a column with three images. Top row - the target plane from the before targeting scan. Middle row - the block surface from the after targeting scan. Bottom row - cross-section parallel to block surface, but 3 micrometre inside the block. The middle row images can be quite dark / noisy as they slice right along the surface of the block, so the bottom row is provided so the cross-section can be seen more easily.

the RMC system (see methods section 6.9.2). While all were scanned with the same voxel size (1 micron), the amount of detail that could be seen inside the *Platynereis* was quite different (Figure S10). Therefore, the higher quality of the X-ray scans for samples f-j (tested with RMC) may have contributed to the improved average targeting accuracy. Secondly, there are still a number of manual steps in this workflow (such as the manual alignment of knife and block face), that can allow human error to contribute to the accuracy. Samples a-e and f-j were targeted by different researchers, so this may also contribute to the differing average accuracy. Finally, the Leica and RMC systems use different hardware (e.g. motors) which will also contribute to their differing average accuracy.

3 Discussion

The motorised ultramicrotome systems and targeting workflow presented here provide a user-friendly solution for targeted EM. The workflow allows X-ray images of resin-embedded samples to be converted into a set of three angles (sample tilt, sample rotation and knife tilt) and a cutting distance for the ultramicrotome. These moves can then be precisely executed with the two motorised ultramicrotome systems we developed to reach a target plane. All measurement steps are integrated into the open-source Fiji plugin Crosshair, making it easily accessible to all. Across 10 targeting runs, the workflow provided a mean angular error of 0.9 degrees and a mean distance error of 3.1 microns. Once the X-ray images have been acquired, the Crosshair measurements and trimming can be done in 1-2 hours, depending on the sample. This is a significant speed up where, in our experience, non-assisted targeting from X-ray may take 1-2 days for complex samples.

There are a number of factors that limit the accuracy of the current system - the first is the resolution of the X-ray images. All X-ray imaging in this paper was done with lab-based micro-CT systems, that offer a maximum resolution of around one micron isotropic voxel size. As all measurements are based on these images, this voxel size limits the achievable accuracy, and contributes to our registration uncertainty. A future direction would be to combine Crosshair with higher resolution X-ray imaging from a synchrotron. Synchrotrons are extremely powerful sources of X-rays, and can provide resolutions in the range of hundreds or even tens of nanometers. In recent years, there have been a number of studies that combine X-ray imaging from a synchrotron with EM[7, 10, 11] - demonstrating the power of combining the large field of view and fast scan times of X-ray, with the high resolution of targeted EM acquisition. As this combination becomes ever more popular, it is more important than ever to have user-friendly targeting solutions like Crosshair.

Another accuracy limitation, are the manual steps that remain in this targeting workflow. While we automate the motorised movements, and angle/distance calculations, a number of steps still require manual use of the ultramicrotome, and will therefore vary in accuracy depending on how experienced the user is. For example, the setting of the zero positions of the knife and sample tilt, alignment of the knife and block face, and estimation of when cutting begins are all done manually. This will therefore introduce some variation due to human error.

Each of these manual steps should be minimised or eliminated in future work to improve accu-

racy. For example, the zero for the sample tilt could be set by precise measurement of the arc piece, rather than by eye. Also, it would be useful to find a more automated way to set the knife zero (although this is more complex as the knife is removed and replaced each time, and different knives are used for different purposes).

The manual alignment of the knife and block face could be removed, if another method was found to determine the relation between the block surface and the sample holder. For example, the sample could be X-ray imaged directly inside the sample holder, although this is challenging in many micro-CT systems due to its large size. Alternately, a thin pin could be designed that would fit in the micro-CT, and also fit tightly inside the sample holder at a fixed orientation. This is very similar to how samples f-j were mounted for targeting (see methods section 6.9.1 and Figure S11). Each of these samples was mounted to a thin aluminium pin at the start of the targeting process, and all trimming steps completed with a Gatan 3View Rivet holder that allows the pin to slot tightly inside. To eliminate the manual alignment step, this pin/holder would have to be adapted so the pin only slots inside in a fixed orientation. For example, by adding a groove to the side of the pin, that fits a corresponding projection on the sample holder. This would allow the relation between the block surface and sample holder to be calculated directly from the X-ray images alone.

The final manual step is assessing when the knife has reached the block surface. Currently, this must be assessed visually by examining the block through the ultramicrotome's binocular microscope. If cutting starts from one corner, it is easy to cut some distance before debris are visually detected on the knife (especially for less experienced ultramicrotome users). This results in errors in the final distance cut. Automation of this step is difficult, and would require hardware improvements that allow the absolute distance between the knife and block surface to be known.

Moving away from improving the workflow's accuracy, it would also be interesting to expand the workflow's scope in future work. For example, Crosshair's calculations should be applicable to any 3D map (not only X-ray), and so could be expanded to targeting from 3D light microscopy. This would be useful for targeting fluorescently labelled structures during CLEM (correlative light and electron microscopy) as in Ronchi et al 2021[9]. Also, Crosshair could be expanded to allow multiple modalities to be used together - for example, displaying registered light microscopy and X-ray images of the same sample.

In addition, Crosshair could be expanded to automate hitting not only a target plane, but a specific location within that plane. For example, for the *Platynereis*, we may cut to a target plane that gives a cross-section through the entire head, and then automatically trim the block sides to leave only a certain sub-structure of interest. This is useful for methods such as FIB-SEM that cannot image large block surfaces. This would require precise, automated East-West movement of the knife (as already implemented on the RMC system), and expansion of the targeting calculations and Crosshair plugin.

In summary, this targeting workflow offers a first step towards fully automated targeting of structures within resin blocks for EM. By making it simpler, and faster, to target a specific plane of interest we can achieve higher throughput, and shorter overall acquisition times. Also, by providing a user-friendly software for this process, we lower the barrier for newcomers to

get started with 3D targeting. By providing the motorised ultramicrotome plans and software fully open-source, we hope that this work will enable more EM labs to rapidly target regions of interest for EM.

4 Acknowledgments

We thank the Electron Microscopy Core Facility (EMBL) for access to the instrumentation for electron microscopy and the Arendt lab (EMBL Heidelberg) for providing the *Platynereis dumerilii* samples. We also thank Christian Tischer (Centre for Bioimage Analysis, EMBL Heidelberg) for his help with creating the Fiji plugins, and the Mechanical Workshop at The Francis Crick Institute for their mechanical expertise and practical assistance during the development.

5 Competing interests

The authors declare that no competing interests exist.

6 Materials and Methods

6.1 Data availability

All X-ray images are available on EMPIAR with id 'EMPIAR-11055': `https://www.ebi.ac.uk/empiar/EMPIAR-11055/`. All files for the accuracy measures tests (Crosshair settings and solutions, accuracy measures, registration metadata, registration accuracy measures) and calibration of the angular rotation of the ultramicrotomes are available on BioStudies with id 'S-BSST845': `ebi.ac.uk/biostudies/studies/S-BSST845`. Diagrams, CAD files and software for the two ultramicrotome systems are available in the following github repositories: `https://github.com/automated-ultramicrotomy/embl-system` and `https://github.com/automated-ultramicrotomy/crick-system`. The Crosshair Fiji plugin is available with tutorials and documentation at the following github repository: `https://github.com/automated-ultramicrotomy/crosshair`, along with a video tutorial on YouTube for the Leica system: `https://www.youtube.com/watch?v=OM712ECthGg`.

6.2 Ultramicrotome systems and calibration

6.2.1 Leica system

The motorised Leica system was designed as a simple add-on to any existing Leica UC7 (or similar ultramicrotome). This means it can easily be added/removed when it is required. Only

one modification was made to the ultramicrotome itself - a hole was drilled into the cutting arm at a precisely vertical position (Supplemental Figure S1A). This fits a small pin that was added to the back of the arc piece (Supplemental Figure S1B) to keep the arc piece precisely vertical.

A new sample holder/arc and knife holder were purchased from Leica, and modified with FAULHABER DC motors for each axis (Figure 1B, D). These motors have their own driver module that uses serial communication RS232 (12V). A cable with a converter (RS232 to UART) is used to communicate with a Raspberry Pi running a Node-RED (`https://nodered.org/`) program for user interface and motor control. The Raspberry Pi is connected to a Raspberry Pi 7 inch touchscreen display, and a StromPi3 battery. The battery is designed to hold up the shutdown process for one second and allow Linux to shut down properly, avoiding damage to the SD memory card that holds the operating system. The setup uses a desktop power supply (that provides 12V), and connects to the StromPi3 battery module. The StromPi3 then provides the 5V needed for the Raspberry Pi. All electronics are housed in a case with an adjustable arm for good screen visibility / ease of use. This allows the motor positions to be controlled and monitored easily by interacting with the touchscreen. The Node-RED user interface 'flow' json is available here: `https://github.com/automated-ultramicrotomy/embl-system/blob/main/node-red/node-red-flow.json`. Once Node-RED is installed, the flow can be loaded by selecting 'import' from the menu, and providing this json file. This also requires the following nodes to be installed: node-red-dashboard (`https://flows.nodered.org/node/node-red-dashboard`), and node-red-node-serialport (`https://flows.nodered.org/node/node-red-node-serialport`).

Diagrams, CAD files and software for this system are available on the following github repository: `https://github.com/automated-ultramicrotomy/embl-system`.

6.2.2 RMC system

To motorise the existing RMC PTPC ultramicrotome, we replaced the manual operational knobs with Trinamic stepper motors and KHK gear sets. For each gear set, one gear was mounted on the shaft originally holding the knob, and another gear was mounted onto the stepper motor. There are five stepper motors and gear sets used in total, to motorise the sample tilt/rotation on the sample holder, knife tilt and stage NS/EW translations. Each stepper motor was chosen based on the consideration of both its maximum torque and its weight. For example, the motor to tilt the sample on the sample holder needs to be reasonably powerful as it is holding both the motor to rotate the sample and itself, which means it can not be too heavy at the same time. The motor mount on the sample holder was 3D printed to reduce the weight and other motor mounts were manufactured in the Francis Crick Institute's mechanical workshop.

The retrofitted five motors were controlled by a Trinamic TMCM-6110 6-axis stepper motor driver board, which was linked via USB to a touchscreen PC. The graphic user interface (GUI), as shown in Figure S1D, was developed with Labview and installed onto this PC, allowing users to control the motions via keyboard/mouse, touchscreen or even a joystick. The stepper motors can provide 51,200 microsteps per full 360^{o} revolutions, which is equivalent to an accuracy of up to 0.00703125^{o} per microstep.

Diagrams, CAD files and software for this system are available on the following github repository:

`https://github.com/automated-ultramicrotomy/crick-system`.

6.2.3 Calibration of Leica and RMC system

In order to calibrate the number of degrees rotated by these stepper motors via different gear sets (not only gear sets for retrofitted motors but also internal gears in the original RMC and Leica sample holder and knife holder to conduct the rotation), therefore different gear ratios, we designed special adaptors to convert the rotational movement of the sample holder and the knife holder to the rotation of the dial of a Thorlabs CRM1PT/M rotation mount, whose vernier scale provides 5 arcmin (0.083 degree) resolution. We rotated/tilted the sample and knife clockwise and anticlockwise traveling the whole range, then recorded the microstep numbers and readings from the rotation mount with roughly equal intervals. A linear regression was performed to finally work out the microstep numbers of the angle resolution, which we set to 0.1 degree to match the vernier scale used (Figures S2, S3, S4).

Diagrams / CAD files of the calibration setup are available in these github repositories: `https://github.com/automated-ultramicrotomy/embl-system` and `https://github.com/automated-ultramicrotomy/crick-system`. The tables of values used for the angular calibration are available on BioStudies: `https://www.ebi.ac.uk/biostudies/studies/S-BSST845`.

6.3 Orientation calculation

For this section we use the following symbols:

Variable	Explanation
$\boldsymbol{R_x}$	Rotation matrix about the x axis
$\boldsymbol{R_y}$	Rotation matrix about the y axis
$\boldsymbol{R_z}$	Rotation matrix about the z axis
θ_T	Tilt angle of sample
θ_R	Rotation angle of sample
θ_K	Tilt angle of knife
θ_{IT}	Initial tilt of sample (on alignment)
θ_{IK}	Initial tilt of knife (on alignment)
θ_{to}	Target offset angle i.e. block face to target plane z axis rotation (measured from X-ray)
θ_{tr}	Target rotation angle i.e. block face to target plane x axis rotation (measured from X-ray)

See Figure S6 for details of how θ_{to}, and θ_{tr} are calculated

Constants:

Constant	Full expression
A	$\cos(\theta_{IK} + \theta_{to})$
B	$\sin(\theta_{tr})\sin(\theta_{IK} + \theta_{to})$
C	$\sin(\theta_{IT})\sin(\theta_{IK} + \theta_{to})$
D	$\cos(\theta_{IT})\sin(\theta_{IK} + \theta_{to})$
E	$\cos(\theta_{tr})\sin(\theta_{IK} + \theta_{to})$
F	$\sin(\theta_{IT})\cos(\theta_{tr})$
G	$\sin(\theta_{tr})\cos(\theta_{IT})$
H	$\sin(\theta_{IT})\sin(\theta_{tr})$
I	$\cos(\theta_{IT})\cos(\theta_{tr})$

6.3.1 Forward kinematics equation

The forward kinematics equation was constructed by calculating the rotation matrices that relate each of the individual coordinate frames (Figure S5 and S6). The world coordinate frame was fixed, and defined x as parallel to the east-west movement of the ultramicrotome, y as parallel to the north-south movement, and z as vertical. The other coordinate frames could be imagined as attached to their respective components, moving with them as they move. These relations are summarised in the pose diagram in Figure 3A. We only focused on the orientation, and not relative position, as the position was solved in a separate process (see section 6.4). When the knife and block surface were aligned, the orientation of the knife coordinate frame was the same as the block coordinate frame. This allowed the rotation matrix between the holder and block to be calculated by going the 'long way round' in this diagram i.e. from Holder, to Arc, to World, to Knife, and then to Block (Figure 3B). This was made simpler by defining the sample rotation to be zero at this aligned position.

The full forward kinematics equation, describing the rotation from the World coordinate frame to the target plane was then:

$$\boldsymbol{F} = \boldsymbol{R_x}(\theta_T)\boldsymbol{R_y}(\theta_R)\boldsymbol{R_x}(-\theta_{IT})\boldsymbol{R_z}(\theta_{IK})\boldsymbol{R_z}(\theta_{to})\boldsymbol{R_x}(\theta_{tr}) \tag{1}$$

This equation allowed us to accurately describe the orientation of the target plane for any set of input angles (sample tilt, sample rotation and knife tilt).

6.3.2 Sample tilt solution

The next step was to calculate the required inverse kinematics equation. For us, the constraint to satisfy was that the target coordinate frame's y axis (which was the same as the target plane's normal) must be perpendicular to the world's z axis. This would mean that the plane was vertical and, as the knife cuts vertically, could be reached by the knife (as long as it was within the angle range it could reach).

This was framed mathematically as:

$$(\boldsymbol{F}\boldsymbol{y}) \cdot \boldsymbol{z} = 0$$

where $\boldsymbol{F}$ was the forward kinematics equation defined above, $\boldsymbol{y}$ was the y vector $\left(\begin{smallmatrix}0\\1\\0\end{smallmatrix}\right)$, $\boldsymbol{z}$ was the z vector $\left(\begin{smallmatrix}0\\0\\1\end{smallmatrix}\right)$ and $\cdot$ denoted the dot product. $\boldsymbol{F}\boldsymbol{y}$ gave the vector for the target plane y axis in terms of the world coordinate frame. Constraining this so its dot product with the world z axis must be 0, meant it must be perpendicular to world z i.e. the plane must be vertical.

Using the sympy[26] python package to help with rearrangement this gave:

$$\begin{aligned}\sin(\theta_T)\Big(&\sin(\theta_{tr})\sin(\theta_{IT}) + \cos(\theta_{IT})\cos(\theta_{tr})\cos(\theta_{IK}+\theta_{to})\Big)\\ &+ \cos(\theta_R)\cos(\theta_T)\Big(-\sin(\theta_{IT})\cos(\theta_{tr})\cos(\theta_{IK}+\theta_{to}) + \sin(\theta_{tr})\cos(\theta_{IT})\Big)\\ &+ \sin(\theta_R)\cos(\theta_T)\Big(\sin(\theta_{IK}+\theta_{to})\cos(\theta_{tr})\Big) = 0\end{aligned}$$

As θ_{IT}, θ_{IK}, θ_{to} and θ_{tr} were constants for a particular targeting run, we could replace these with the A-I constants defined above to make the expression easier to work with.

$$(AI+H)\sin(\theta_T) + (-AF+G)\cos(\theta_R)\cos(\theta_T) + E\sin(\theta_R)\cos(\theta_T) = 0$$

$$(-AF+G)\cos(\theta_R)\cos(\theta_T) + E\sin(\theta_R)\cos(\theta_T) = -(AI+H)\sin(\theta_T)$$

$$(-AF+G)\cos(\theta_R) + E\sin(\theta_R) = \frac{-(AI+H)\sin(\theta_T)}{\cos(\theta_T)}$$

$$\tan(\theta_T) = \frac{-AF+G}{-AI-H}\cos(\theta_R) + \frac{E}{-AI-H}\sin(\theta_R)$$

This meant that the final solution for the sample tilt was:

$$\theta_T = \arctan\left(\frac{-AF+G}{-AI-H}\cos(\theta_R) + \frac{E}{-AI-H}\sin(\theta_R)\right) \tag{2}$$

This solution could of course be simplified further by combining these constants together to give something of the form:

$$\theta_T = \arctan\Big(C_1\cos(\theta_R) + C_2\sin(\theta_R)\Big)$$

In practice, we used the form with constants A-I, as this allowed the same constants to be used for this and the knife solution below.

The script for the sympy parts is available in the Crosshair github repository: `https://github.com/automated-ultramicrotomy/crosshair/blob/master/python_scripts/multiple_solutions.py`.

6.3.3 Knife tilt solution

Now we had to find the knife angle that corresponded to a particular sample rotation. For this, we needed the signed angle between global y (i.e. ultramicrotome NS) and local y (the target plane's normal) in the z plane (i.e. the plane with the global z axis as its normal). This was because the knife could only rotate about the z axis, so we only needed the angle within this plane.

This was equal to:

$$\theta_K = \arctan\left(\frac{(\boldsymbol{y} \times (\boldsymbol{F}\boldsymbol{y})) \cdot \boldsymbol{z}}{\boldsymbol{y} \cdot (\boldsymbol{F}\boldsymbol{y})}\right)$$

where $\boldsymbol{y}$ was the y vector $\begin{pmatrix}0\\1\\0\end{pmatrix}$, $\boldsymbol{z}$ was the z vector $\begin{pmatrix}0\\0\\1\end{pmatrix}$, $\cdot$ denoted the dot product, $\times$ denoted the cross product and $\boldsymbol{F}$ was the matrix from the forward kinematics equation defined above in equation 1.

Using sympy[26] to help with rearrangement, and substitution of the same constants A-I from above, this became:

$$\theta_K = \arctan\left(\frac{E\cos(\theta_R) + (AF - G)\sin(\theta_R)}{-E\sin(\theta_R)\sin(\theta_T) + (AF - G)\sin(\theta_T)\cos(\theta_R) + (AI + H)\cos(\theta_T)}\right)$$

This equation gave the corresponding signed knife angle for all values of θ_T and θ_R, but we only wanted the values that are valid given the solution calculated above for the sample tilt. I.e. we only wanted knife angles where the target plane was vertical, and therefore reachable by the knife. Therefore, we substituted in the solution from above (equation 2) for θ_T.

This gave (with some rearrangement / simplification with sympy):

$$\theta_K = \arctan\left(\frac{\Big(AI + H\Big)\Big(E\cos(\theta_R) + (AF - G)\sin(\theta_R)\Big)}{\left(\sqrt[2]{\Big(AI + H\Big)^2 + \Big(E\sin(\theta_R) + (-AF + G)\cos(\theta_R)\Big)^2}\right)|AI + H|}\right) \tag{3}$$

This solution could of course be simplified further by combining these constants together to give

something of the form:

$$\theta_K = \arctan\left(\frac{C_3\Big(C_4\cos(\theta_R) + C_5\sin(\theta_R)\Big)}{\left(\sqrt[2]{{C_3}^2 + \Big(C_4\sin(\theta_R) - C_5\cos(\theta_R)\Big)^2}\right)|C_3|}\right)$$

In practice, we used the form with constants A-I in the Crosshair code.

Therefore, with equation 2 and equation 3 we could find the corresponding sample tilt and knife tilt for any given sample rotation.

The script for the sympy parts is available in the Crosshair github repository: `https://github.com/automated-ultramicrotomy/crosshair/blob/master/python_scripts/multiple_solutions.py`.

6.4 Distance calculation

Once the orientation was solved, we calculated the distance to cut from the sample to reach the target plane. As the cutting direction (NS) may not have been parallel to the target plane normal, this had to be done in two steps. First, the perpendicular distance was calculated from the target plane to the furthest point of the block face, then this was compensated for the offset between this and NS.

When the orientation was solved, we knew that the target plane must be vertical. Therefore, to find the NS distance we only needed to compensate for the current knife angle (Figure S7).

This meant that the distance was:

$$D_{NS} = \frac{D_P}{\cos(\theta_K)}$$

where D_{NS} was the NS distance, D_P was the perpendicular distance between the target plane and the furthest surface point, and θ_K was the knife angle.

6.5 Crosshair

Crosshair is a Fiji[17] plugin written in Java, and is freely available from the github repository: `https://github.com/automated-ultramicrotomy/crosshair`. A user guide is also available: `https://github.com/automated-ultramicrotomy/crosshair/wiki`, along with a video tutorial of the entire workflow demonstrated on the Leica system: `https://www.youtube.com/watch?v=OM712ECthGg`. It can also be easily installed via a Fiji update site (instructions in the github repository).

It builds on top of BigDataViewer[18] for viewing images in 2D, and the ImageJ 3D Viewer[19] for viewing images in 3D. It also builds on a number of functions from Christian Tischer's imagej-utils repository (`https://github.com/embl-cba/imagej-utils`).

6.6 RegistrationTree

RegistrationTree is a Fiji[17] plugin written in Java, and is freely available from the github repository: `https://github.com/K-Meech/RegistrationTree`. A user guide is also available: `https://github.com/K-Meech/RegistrationTree/wiki`. It can be easily installed via a Fiji update site (instructions in the github repository).

RegistrationTree is a wrapper around BigWarp[23] and elastix[24, 25] registration software. It allows easy passing of images between the two software tools, as well as easing the integration of large images with elastix. Users can build arbitrary trees of affine transformations to register their images together, and visualise/compare these easily. RegistrationTree uses BigDataViewer[18] for 2D visualisation, and BigDataViewer style file formats (HDF5 and N5) to allow very large images to be viewed efficiently on a normal laptop. It builds on top of Christian Tischer's elastixWrapper[27](`https://github.com/embl-cba/elastixWrapper`) and imagej-utils repository (`https://github.com/embl-cba/imagej-utils`). Also, the open-source repositories for image-transform-converters (`https://github.com/image-transform-converters/image-transform-converters`) and bigdataviewer-playground (`https://github.com/bigdataviewer/bigdataviewer-playground`).

6.7 Targeting workflow

The targeting workflow is summarised in Figure 4. We expand on those steps in detail here, in the form of a step by step protocol. They are also described in detail in the Crosshair user guide (`https://github.com/automated-ultramicrotomy/crosshair/wiki`), and in a video tutorial demonstrated on the Leica system: `https://www.youtube.com/watch?v=OM712ECthGg`. Note that there are some differences between the motorised Leica and RMC systems that are highlighted throughout the workflow.

Trim block - Trim the block at the ultramicrotome with a diamond knife. A flat surface must be made that has 4 corners and straight lines in between. Make sure the sides of the block face are trimmed deep enough, so the knife will always touch part of this face first, and not other parts of the block (even if the solution requires a more extreme set of angles). We usually trim about 40-100 microns, depending on the sample.

Ideally, the block surface should be within one ultramicrotome feed length of the target for the greatest accuracy (i.e. 200 microns or less for both the Leica and RMC systems). Every time the cutting feed is reset, the approach of the knife to the block must be repeated, and some accuracy will be lost. If necessary, the distance can be greater though - for example samples h, i and j in this study had cutting depths greater than one feed length (requiring one reset mid-run), and their accuracy was not greatly affected (see Figure 6).

Note that there are a number of different ways to mount the sample for trimming and later X-ray imaging. For example, see the different solutions we used in section 6.9.1.

X-ray - X-ray the block at the highest resolution possible (this will determine how accurate measures made from the X-ray images can be). Ensure the sample and the entirety of the block surface are visible.

Convert the X-ray image to 8-bit (if it isn't already) e.g. in Fiji with Image - Type - 8-bit. The image can also be cropped at this stage if there is a lot of empty area outside the resin block. Open the X-ray in Crosshair, and compare the 3D view/image stack and the sample. If it appears as a mirror image of the sample, then flip the image stack. This can be done e.g. in Fiji with Image - Transform - Flip Vertically, or Image - Transform - Flip Z.

Define block + target plane - Open the X-ray image in Crosshair and set the block and target planes. The target plane can be set using the 'TRACK' button and navigating in the 2D viewer. The block face can be set by placing a series of points on the block surface and fitting a plane to them.

Define block vertices + orientation - Place one point (vertex) on each of the 4 corners of the block face. Then, decide on the orientation the block will have in the ultramicrotome i.e. which side of the block will face up? Mark the points accordingly as 'Top Left', 'Top Right', 'Bottom Left' and 'Bottom Right' in Crosshair.

Once this is complete, it is good practice to 'Save Settings' in Crosshair to make a .json file as a record of the planes and points.

Set sample tilt 0 - Set the sample tilt to zero by eye (just by looking at the scale on the arc piece and trying to get it as close to 0 as possible). For the Leica system, press 'Set Zero' in the 'sample tilt' page of the ultramicrotome user interface. For the RMC system, make a note of the current sample tilt angle.

Set knife tilt 0 - For the knife zero, there is a slightly more complicated procedure than for the sample tilt. We assumed that since the knife is removed and replaced every time, there may be more variation in its position and therefore an extra step may be required to accurately find its proper zero.

Put a blank block into the ultramicrotome sample holder (i.e. just resin with no sample). Set the knife tilt to zero by eye (just by looking at the scale on the knife holder and trying to get it as close to zero as possible). Cut into the blank block at this orientation to make a flat face. Back the knife away, and rotate the sample holder exactly 90 degrees (using the motorised ultramicrotome controls). Align the knife to this face again (only adjusting the knife tilt!). For the Leica system, press 'Set Zero' in the 'knife tilt' page of the ultramicrotome user interface. For the RMC system, make a note of the current knife tilt angle.

Align knife and block face - Replace the blank block with the resin-embedded sample. Make sure it is placed the same way up as indicated in Crosshair.

Align the knife and block face, ensuring the bottom edge of the block is also aligned along

the cutting edge of the knife. Any axis can be freely changed for this step, just as in normal ultramicrotome usage. For the Leica system, the motors can be disabled for this step, and the alignment done manually as with normal ultramicrotome usage. For the RMC system, the motors cannot be disabled, and the motors must be used for this alignment.

Set sample rotation 0 - For the Leica system, set the sample rotation to zero by pressing 'Set Zero' in the 'sample rotation' page of the ultramicrotome user interface. For the RMC system, make a note of the current sample rotation angle.

Enter θ_{IK} and θ_{IT} - Enter the current knife angle and sample tilt angle into Crosshair. For the RMC system, make sure you adjust for the recorded 'zero' position e.g. if the current angle is 10 degrees, and the zero was 2 degrees - then you would enter 8 degrees into Crosshair.

Choose solution - Use the Crosshair 'Microtome Mode' to cycle through all the possible solutions, and choose one to use. The target plane will turn red in the 3D view when there is a valid solution.

Once this is complete, it is good practice to 'Save Solution' in Crosshair to make a .json file as a record of the solution.

Set solution angles - Back the knife away from the block, to ensure there is no risk of hitting it. Using the motors, move each axis to the angles stated in the solution from Crosshair. For the RMC system, make sure you adjust for the recorded 'zero' position of each axis e.g. if the solution angle is 10 degrees, and the zero was 2 degrees - then you would move the ultramicrotome to 12 degrees for that axis.

Approach block surface + cut solution distance - Manually approach the block and get as close as possible to its surface. Any error in this distance (i.e. starting before touching the block, or once it has already been cut into) will contribute to the final error. A good solution is to get as close as possible with the ultramicrotome's NS movement, then set the cutting thickness to some small value (e.g. 70-100nm). Then cutting can be started slowly, carefully watching the knife edge through the ultramicrotome's binocular at high magnification, and stopped as soon as any debris is visible on the knife / any pieces being cut from the block. This ensures the knife is as close as possible to the true surface.

Once complete, cut the distance specified in the Crosshair solution.

6.8 *Platynereis* sample preparation

Platynereis were prepared with a modification of the method from Hua et al.[28], aided by microwave processing (see details in table 1 below). *Platynereis* were flat embedded in thin pieces of resin.

Step No.	Step	Time	Temperature	Microwave Watt	Vacuum
1	2% paraformaldehyde, 2.5% glutaraldehyde in seawater	2 minutes	21°C	100	On
2	2% paraformaldehyde, 2.5% glutaraldehyde in 0.1M cacodylate	2x 14 minutes (2 min on/off cycles)	21°C	100	On
3	0.1M cacodylate	1 immediate. Then 2x40 seconds.	21°C	100	On
4	2% OsO4 in 0.1M cacodylate	2x 14 minutes (2 min on/off cycles)	21°C	100	On
5	2.5% $K_4[Fe(CN)_6]3H_2O$ in 0.1M cacodylate	2x 14 minutes (2 min on/off cycles)	21°C	100	On
6	Water wash	1 immediate. Then 2x40 seconds.	21°C	100	On
7	1% TCH unbuffered	2x 14 minutes (2 min on/off cycles)	40°C	100	On
8	Water wash	1 immediate. Then 2x40 seconds.	40°C	100	On
9	2% OsO4 aqueous	2x 14 minutes (2 min on/off cycles)	21°C	100	On
10	Water wash	1 immediate. Then 2x40 seconds.	21°C	100	On
11	Dehydration series in ethanol (25%, 50%, 75%, 3x100%)	40 seconds each	10°C	250	Off
12	Infiltration series in Durcupan (25%, 50%, 75%, 90%, 3x100%)	3 minutes each.	21°C	150	On
13	100% Durcupan	overnight	21°C	N/A	N/A
14	Polymerisation in oven	48 hours	60°C	N/A	N/A

Table 1: Table of *Platynereis* preparation steps

6.9 Accuracy tests

6.9.1 Initial sample trimming and mounting

Samples were prepared as specified in section 6.8. Individual *Platynereis* were then cut out in small pieces of resin with a razor blade.

For samples a-e, these pieces were glued to larger resin blocks to make them easier to handle in the ultramicrotome. These blocks were then inserted into standard Leica sample holders, and mounted to the ultramicrotome for trimming.

For samples f-j, the small pieces of resin containing *Platynereis* were mounted onto aluminium pins using conductive epoxy glue (ITW Chemtronics). The pin was then inserted into a Gatan 3View Rivet Holder (Gatan 3VRHBM) and mounted to the ultramicrotome (Figure S11A, B) for trimming.

Blocks were trimmed close to the *Platynereis* with a diamond knife, so the block surface for most samples was less than one feed length (200 microns) from the target. A flat surface was trimmed with a rectangular shape as required for Crosshair. Note that samples h, i and j had cutting distances greater than one feed length, but their accuracy was not greatly affected (see Figure 6).

6.9.2 X-ray imaging

Samples a-e were imaged with a Bruker SkyScan 1272, with an isotropic voxel size of 1 micron. The resin blocks were mounted onto small pins with wax for the X-ray, and then removed from these pins for all further steps at the ultramicrotome. The log files with the specific settings for each sample are provided on EMPIAR alongide the images: `https://www.ebi.ac.uk/empiar/EMPIAR-11055/`. Imaging each sample took 2 hours to complete. Fiji was then used to convert the X-ray images to 8-bit, flip them vertically (Image - Transform - Flip Vertically) and crop to remove empty areas outside the resin block.

Samples f-j were imaged with a Versa 510 X-ray microscope (Zeiss). Aluminium pins were mounted to a Versa 510 sample holder (Figure S11C), and imaged at a voltage of 40 kV, power of 3 W. No filter was used, and images were acquired at a binning of 2 with an exposure time of 10 s. Pixel size was chosen to be as close to 1 micron as possible (actual reading varied between 0.9999 and 1.0001 microns for different runs). The field of view thus varied between 1014.9 and 1015.1 microns^2. 1601 projections were captured over the 360 degree tilt range. Imaging each sample took 5 hours and 30 minutes to complete. Fiji was then used to convert the X-ray images to 8-bit, adjust their brightness and contrast, flip vertically (Image - Transform - Flip Vertically) and finally reverse the image stack (Image - Stacks - Tools - Stack Sorter - Reverse).

For all samples, raw and reconstructed X-ray images are available (before and after targeting), as well as the X-ray images used for Crosshair (i.e. after any flipping, cropping etc. was done): `https://www.ebi.ac.uk/empiar/EMPIAR-11055/`.

6.9.3 Crosshair and targeting

The X-ray images were then processed with Crosshair. The block face plane was set by fitting to a series of points placed on the surface. The target plane was set by browsing in 2D to find a plane that cut through both anterior dorsal cirrus that protrude from either side of the *Platynereis* head. The remaining targeting steps were completed as specified in the workflow in section 6.7. For samples a-e, all steps were completed using a standard Leica sample holder. For samples f-j, a Gatan 3View Rivet Holder was used (as specified in section 6.9.1).

All Crosshair settings and solution files are available on BioStudies: `https://www.ebi.ac.uk/biostudies/studies/S-BSST845`.

6.9.4 Registration

After targeting, the samples were imaged with X-ray again as described in section 6.9.2. The before and after X-ray images were then registered using our RegistrationTree plugin (see method's section 6.6). For this, the before scan (moving image) was registered to the after scan (fixed image). This used two registration steps - first, a rough point-based registration with BigWarp[23], followed by an intensity-based registration with elastix[24, 25]. Both steps were limited to rotation and translation only. Once complete, this was exported from RegistrationTree as a BigDataViewer xml file containing the required transform in the 'MOVING' space i.e. so it transformed the after scan onto the before scan.

For some of our X-ray images, we saw slight variation in the before and after scans of the same sample (particularly with the Bruker system). We therefore also calculated an approximate error in the accuracy of our registration. This was done by manually labelling 4 corresponding points in each registered pair of X-ray images for all samples. For example, the tips of the *Platynereis* parapodia (appendages used for swimming and crawling). These were placed using BigWarp's features for point placement, and then exported to csv files. The euclidean distance between each pair of points was then calculated for Figure S9. This gave a mean error of 1.9 microns (for the points labelled repeat 1). As there was also some error due to manual placement of these points, we repeated the process twice for the same locations and images (repeats 1 and 2 in Figure S9A). The mean difference between these repeats was 1.1 microns. As the average difference between repeats was less than the error we measured, it implies that this likely comes from error in the registration, rather than inaccuracy of the manual point placement. As they are of a similar size though, it does emphasise the difficulty of accurately measuring these small distances, given that the voxel size of our micro-CT was only one micron.

For all samples, the RegistrationTree and registration accuracy files are available on BioStudies: `https://www.ebi.ac.uk/biostudies/studies/S-BSST845`.

6.9.5 Accuracy Measures

Accuracy measures were calculated using features built-in to the Crosshair Fiji plugin ('Measure Targeting Accuracy' command). This takes the before X-ray image, the registered after X-ray image and the Crosshair settings and solution files and allows user-friendly calculation of accuracy measures. A plane was fitted to the after scan's block surface by manually placing a series of points on its surface (same workflow as fitting the planes in usual Crosshair usage). A point was also placed on the before scan's target plane approximately at the centre of the *Platynereis* cross-section. Then, the 'Save Measures' option was used to save a simple json file containing the measures.

This included three measures - one for the angle error, one for the solution distance error, and one for the point to plane distance error. The angle error was calculated as the absolute angle between the normals of the target plane, and the after scan surface. The distance error was more complex to calculate, as any planes that are not exactly parallel will intersect at some location (i.e. distance = 0). Therefore, we provided two different measures for this: the solution and point to plane distance.

The solution distance calculated the approximate difference between the solution distance, and the distance actually cut (inferred from the Crosshair settings, and after block surface position). It was therefore a measure of how accurately the solution's distance was cut, rather than a direct measure of how close the final surface was to the target plane. This was calculated by determining the shortest distance between the predicted first touch point (on the original block surface), and the after scan block surface. This was then converted to the NS cutting distance, by compensating for the solution knife angle:

$$D_{NS} = \frac{D_P}{\cos(\theta_K)}$$

where D_{NS} was the NS distance, D_P was the perpendicular distance between the after surface plane and the first touch point, and θ_K was the knife angle from the chosen solution.

The final distance error was then:

$$\text{Error} = D_{NS} - D_{Sol}$$

where D_{NS} was the NS distance, and D_{Sol} was the distance from the chosen solution. A positive error indicated that the sample was trimmed too far, while a negative error indicated it wasn't trimmed enough.

Note that these calculations assumed that cutting started from the predicted point. If the angle was so far off that cutting started from another point, this measure would be erroneous. This is something that was checked at the start of each run though, by examining the block and checking that the correct vertex was touched first. This calculation also assumed that the knife angle was the same as stated in the solution. If there was any error in the knife angle setting, then this would also affect this accuracy measure.

To have a more direct measure of how close the final surface was to the target plane, we also

provided the point to plane distance error. This is calculated as the shortest distance between a point on the before target plane (near to the centre of the region of interest), and the after block surface. Note that therefore this measure will vary depending on where this point was placed within the target plane.

See here for the calculation scripts: `https://github.com/automated-ultramicrotomy/crosshair/blob/master/src/main/java/de/embl/schwab/crosshair/targetingaccuracy/AccuracyCalculator.java`. For all samples, all accuracy settings and accuracy measures files are available on BioStudies: `https://www.ebi.ac.uk/biostudies/studies/S-BSST845`.

References

[1] Louis K Scheffer et al. "A connectome and analysis of the adult Drosophila central brain". In: *eLife* 9 (Sept. 2020). Ed. by Eve Marder et al., e57443. ISSN: 2050-084X. DOI: 10.7554/eLife.57443.

[2] Zhihao Zheng et al. "A Complete Electron Microscopy Volume of the Brain of Adult Drosophila melanogaster." In: *Cell* 0.0 (July 2018). ISSN: 1097-4172. DOI: 10.1016/j.cell.2018.06.019.

[3] Hernando M. Vergara et al. "Whole-body integration of gene expression and single-cell morphology". In: *Cell* 184.18 (2021), 4819–4837.e22. ISSN: 00928674. DOI: 10.1016/j.cell.2021.07.017.

[4] Matthia A Karreman et al. "Find your way with X-Ray: using microCT to correlate in vivo imaging with 3D electron microscopy". In: (2017). DOI: 10.1016/bs.mcb.2017.03.006.

[5] Matthia A Karreman et al. "Fast and precise targeting of single tumor cells in vivo by multimodal correlative microscopy." In: *Journal of cell science* 129.2 (Jan. 2016), pp. 444–56. ISSN: 1477-9137. DOI: 10.1242/jcs.181842.

[6] C. Shan Xu et al. "An open-access volume electron microscopy atlas of whole cells and tissues". In: *Nature* November 2020 (2021). ISSN: 0028-0836. DOI: 10.1038/s41586-021-03992-4.

[7] Jacob M Musser et al. "Profiling cellular diversity in sponges informs animal cell type and nervous system evolution". In: *Science* 374.6568 (2021), pp. 717–723. DOI: 10.1126/science.abj2949.

[8] Eric A. Bushong et al. "X-Ray Microscopy as an Approach to Increasing Accuracy and Efficiency of Serial Block-Face Imaging for Correlated Light and Electron Microscopy of Biological Specimens". In: *Microscopy and Microanalysis* 21.1 (2015), pp. 231–238. ISSN: 14358115. DOI: 10.1017/S1431927614013579.

[9] Paolo Ronchi et al. "High-precision targeting workflow for volume electron microscopy". In: *The Journal of cell biology* 220.9 (2021). ISSN: 15408140. DOI: 10.1083/jcb.202104069.

[10] Carles Bosch et al. "Functional and multiscale 3D structural investigation of brain tissue through correlative in vivo physiology, synchrotron micro-tomography and volume electron microscopy". In: *bioRxiv* (Jan. 2021), p. 2021.01.13.426503. DOI: 10.1101/2021.01.13.426503.

[11] Aaron T. Kuan et al. "Dense neuronal reconstruction through X-ray holographic nano-tomography". In: *Nature Neuroscience* 23.12 (2020), pp. 1637–1643. ISSN: 15461726. DOI: 10.1038/s41593-020-0704-9.

[12] Philip J Withers et al. "X-ray computed tomography". In: *Nature Reviews Methods Primers* (). ISSN: 2662-84492662-8449. DOI: 10.1038/s43586-021-00015-4.

[13] Valentina Baena et al. *Serial-section electron microscopy using automated tape-collecting ultramicrotome (ATUM)*. 1st ed. Vol. 152. Elsevier Inc., 2019, pp. 41–67. ISBN: 9780128170182. DOI: 10.1016/bs.mcb.2019.04.004.

[14] Thomas Templier. "MagC, magnetic collection of ultrathin sections for volumetric correlative light and electron microscopy". In: *eLife* 8 (2019), pp. 1–18. DOI: 10.7554/elife.45696.

[15] Timothy J. Lee et al. "Large-scale neuroanatomy using LASSO: Loop-based Automated Serial Sectioning Operation". In: *PLOS ONE* 13.10 (Oct. 2018). Ed. by Konradin Metze, e0206172. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0206172.

[16] Elisabeth Brama et al. "ultraLM and miniLM : Locator tools for smart tracking of fluorescent cells in correlative light and electron microscopy [version 1 ; peer review : 3 approved , 1 approved with reservations]". In: *Wellcome Open Res* (2016), pp. 1–26.

[17] Johannes Schindelin et al. "Fiji: An open-source platform for biological-image analysis". In: *Nature Methods* 9.7 (2012), pp. 676–682. ISSN: 15487091. DOI: 10.1038/nmeth.2019.

[18] Tobias Pietzsch et al. "BigDataViewer: Visualization and processing for large image data sets". In: *Nature Methods* 12.6 (2015), pp. 481–483. ISSN: 15487105. DOI: 10.1038/nmeth.3392.

[19] Benjamin Schmid et al. "A high-level 3D visualization API for Java and ImageJ". In: *BMC Bioinformatics* 11.1 (2010), p. 274. ISSN: 1471-2105. DOI: 10.1186/1471-2105-11-274.

[20] B. Duygu Özpolat et al. "The Nereid on the rise: Platynereis as a model system". In: *EvoDevo* 12.1 (2021), pp. 1–22. ISSN: 2041-9139. DOI: 10.1186/s13227-021-00180-3.

[21] Albrecht Fischer and Adriaan Dorresteijn. "The polychaete Platynereis dumerilii (Annelida): A laboratory animal with spiralian cleavage, lifelong segment proliferation and a mixed benthic/pelagic life cycle". In: *BioEssays* 26.3 (2004), pp. 314–325. ISSN: 02659247. DOI: 10.1002/bies.10409.

[22] Elizabeth A Williams and Gáspár Jékely. "Towards a systems-level understanding of development in the marine annelid Platynereis dumerilii". In: *Current Opinion in Genetics & Development* 39 (Aug. 2016), pp. 175–181. ISSN: 0959-437X. DOI: 10.1016/J.GDE.2016.07.005.

[23] John A Bogovic et al. "Robust registration of calcium images by learned contrast synthesis". In: *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*. 2016, pp. 1123–1126. DOI: 10.1109/ISBI.2016.7493463.

[24] Stefan Klein et al. "elastix: A Toolbox for Intensity-Based Medical Image Registration". In: *IEEE Transactions on Medical Imaging* 29.1 (2010), pp. 196–205. DOI: 10.1109/TMI.2009.2035616.

[25] Denis P. Shamonin et al. "Fast parallel image registration on CPU and GPU for diagnostic classification of Alzheimer's disease". In: *Frontiers in Neuroinformatics* 7.JAN (2014), pp. 1–15. ISSN: 16625196. DOI: 10.3389/fninf.2013.00050.

[26] Aaron Meurer et al. "SymPy: Symbolic computing in python". In: *PeerJ Computer Science* 2017.1 (2017), pp. 1–27. ISSN: 23765992. DOI: 10.7717/peerj-cs.103.

[27] Christian Tischer. *ElastixWrapper: Fiji plugin for 3D image registration with elastix*. Mar. 2019. DOI: 10.5281/zenodo.2602549. URL: https://doi.org/10.5281/zenodo.2602549.

[28] Yunfeng Hua, Philip Laserstein, and Moritz Helmstaedter. "Large-volume en-bloc staining for electron microscopy-based connectomics". In: *Nature Communications* 6 (2015), pp. 1–7. ISSN: 20411723. DOI: 10.1038/ncomms8923.