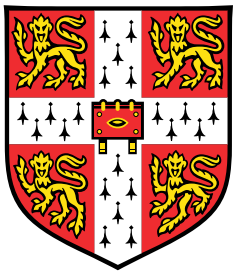
# Hardware, software and applications in super-resolution microscopy



**Marcus Fantham**

Supervisor: Prof. Clemens F. Kaminski

Department of Chemical Engineering and Biotechnology
University of Cambridge

This dissertation is submitted for the degree of
*Doctor of Philosophy*

St. Catharine's College October 2018

I would like to dedicate this thesis to my loving parents ...

# Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Marcus Fantham
October 2018

# Acknowledgements

And I would like to acknowledge ...

Clemens

LAG group

MNG and QI group, particularly Miranda for her friendship

Collaborators, especially Michelle and Edward

Friends - parents - and family who have looked after me during the last 3 years, particularly Helene and Ellie, Catz choir, the top floor crew.

# Abstract

This is where you write your abstract ...

# Table of contents

# List of figures

# List of tables

# Chapter 1

# Introduction

## 1.1 Breaking the diffraction limit

Our understanding of the biology of life is limited to what we can observe. The human visual system is limited to a spatial resolution of ~100 µm [CITE!]. The invention of high power microscopes, generally attributed to van Leeuwenhoek in 1660, and pioneering experiments by Robert Hooke [cite micrographia] and Swammerdam [cite ?], revealed that the building blocks of life are invisible to the naked human eye.

Lens technology continuously improved through the 17$^{th}$ and 18$^{th}$ Centuries, revealing smaller and smaller objects, until in 1873 Abbe showed that the diffraction of light through a lens places a fundamental limit on the size of objects which can be resolved. The famous Abbe diffraction limit, shown in Equation 1.1, states that the minimum separation distance, $d$, at which two objects can be resolved decreases with the wavelength of light, $\lambda$, but increases with the lens' acceptance angle, $\alpha$, and refractive index of the medium between the lens and the object, $n$. The wavelength range of visible light is 400 nm–700 nm, and the maximum acceptance angle of a lens approaches 90°. The refractive index of air is 1.0, although special immersion oils can be used to increase $n$ to ~1.5. Using Equation 1.1, we then calculate that the maximum resolution of a microscope lens is ~200 nm.

$$d = \frac{\lambda}{2n\sin(\alpha)} \tag{1.1}$$

A number of technologies which are not based on optics exist to greatly surpass the resolution of optical microscopes, including transmission electron microscopy and scanning electron microscopy. However due to sample preparation requiring either freezing or a metal coating, these techniques are not compatible with live cell biology.

Furthermore, the invention of fluorescent labelling gives optical microscopy a distinct advantage over any other microscopy technique in terms of specificity. Sophisticated biochemistry, based on antibody chemistry, genetic expression, and other biotechnologies, can be used to label specific cellular compartments or organelles with fluorescent molecules. When these fluorescent molecules are illuminated with a certain wavelength of light, they absorb photons, exciting electrons to a higher energy state. The electrons lose some energy through so-called vibrational states, then as the electron falls back to the ground state photons are emitted at a red-shifted wavelength - where the wavelength shift is proportional to the energy lost through vibrational states, as per the Plank-Einstein relation $E = hf = hc/\lambda$.

Fluorescent labelling has a number of unique applications. Firstly, since the light used to excite fluorescence is blocked by filters before it reaches the observer, only fluorescence emission light is seen through the microscope. This creates a bright image of the compartment of interest against a black background, giving a high signal-to-noise ratio. Furthermore the specificity of labelling means that other parts of the cell which are not of interest for a given experiments are invisible, further enhancing the observability of the labelled compartment. Finally, if two or more fluorescent labels are used with non-overlapping fluorescent spectra, then imaging in multiple channels can be used for co-localisation studies, for example to confirm that a certain protein interacts with a certain organelle.

The unique advantages of florescent labelling have motivated researchers in the last two decades to invent optical methods to image at resolutions below the diffraction limit. Whilst it remains impossible to capture an individual image beyond the diffraction limit, digital cameras and intensive computational reconstruction has enabled techniques which sacrifice temporal resolution for enhanced spatial resolution. This is the principle behind techniques based on Photoactivated Localisation Microscopy (PALM) and Stochastic Optical Reconstruction Microscopy (STORM): in each image, only a small fraction (<1 %) of fluorophores are emitting light. Assuming that two adjacent fluorophores do not emit simultaneously, the centre of the diffraction pattern is calculated as the true location of the molecule; when this is applied to a dataset of 1000+ images, a super-resolution image is constructed.

STORM is able to achieve a spatial resolution of ~10 nm–20 nm; however it takes around 1 min–5 min to generate such an image. Dynamic events in live cell biology cannot be captured, and so a faster technique must be used for such experiments.

Structured Illumination Microscopy (SIM), as refined for super-resolution by Gustafsson [cite], utilises 9 raw images to reconstruct a resolution-enhanced image up to twice the diffraction limit of the lens. Using state-of-the-art lenses and cameras, this can produce images with a spatial resolution of <100 nm, at a video rate of 11 Hz.

## 1.2 Cell biology for microscopy

The tools described in Chapters 2 and 3 find applications in cell biology, which are detailed in Chapters **??** and 5. Since this thesis covers a broad range of themes, from engineering through computer science to biology, this section is included to provide a basic understanding of cell biology necessary for appreciating the findings detailed in the applications chapters.

Figure 1.1 shows a generic animal cell.

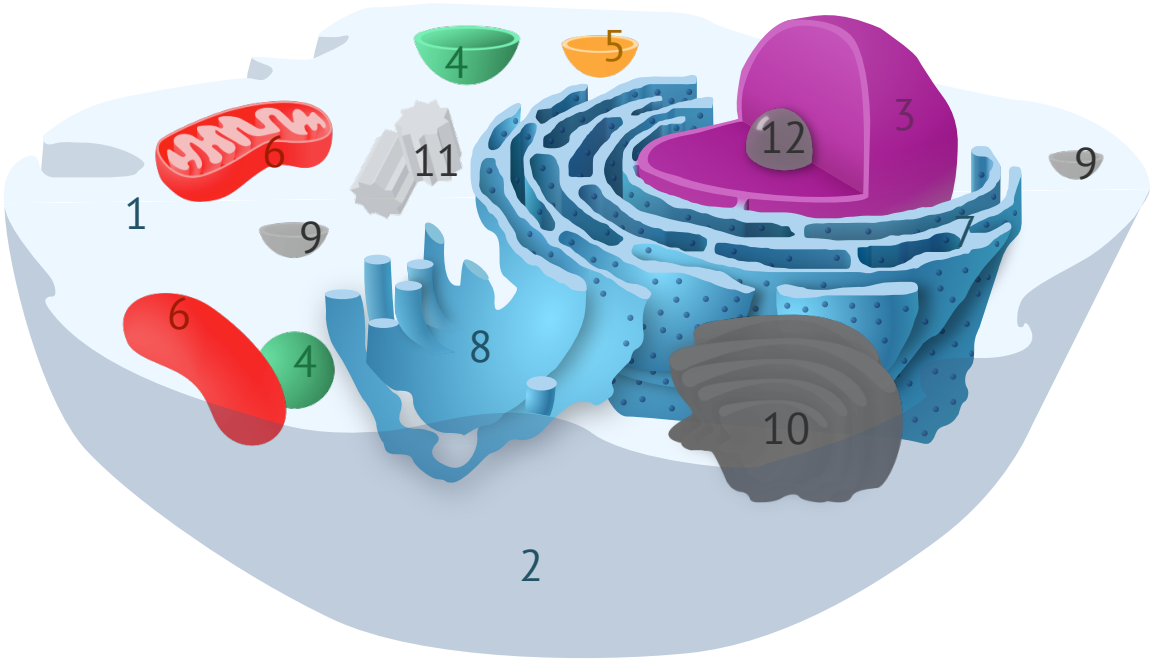



**Figure 1.1:** A generic animal cell, adapted from Wikipedia[1], but with the organelles discussed in this thesis highlighted in colour. Numbered annotations are as follows:

1. Cytosol
2. Cell membrane
3. Nucleus
4. Lysosome
5. Endosome
6. Mitochondria
7. ER network
8. ER sheets
9. Vesicle
10. Golgi apparatus
11. Centrosome
12. Nucleolus

I have included a diagram of the cell, with the compartments relevant to this thesis highlighted.

## 1.3 Structure and aims of this document

Since October 2015, I have been working in the Laser Analytics Group building tools and applying them to answer biological questions.

Finally, this document provides a memento to myself of my PhD experience. I am proud of what I have achieved alongside my collaborators, and have included some of my favourite quotes at the start of each chapter. I hope these do not come across as arrogant or self-praising; they simply serve as a reminder to myself that others appreciate my hard work and that the effort has undoubtedly been worth it.

# Chapter 2

# LAG SIM: A versatile, user-friendly structured illumination microscope

"The LAG SIM works good now and I really love it!"

*Email from Meng Lu (Molecular Neuroscience Group, Cambridge), 26-06-2018*

## 2.1 Introduction

### 2.1.1 Background

Epifluorescent, widefield microscopy use a field of light to illuminate a labelled sample. Floresecnet emmission light is then emitted from any fluorescent molecule located within the microscope's field of view. Fluorescent molecules above and below the focal plane of the lens will also receive illumination light and fluoresce, causing an out-of-focus blur on the image with intensity [$\frac{1}{z^2}$ ??] where $z$ is axial distance from the focal plane.

Out-of-focus light can be removed by lacing a pinhole in the excitation and emission path, and scanning the small spot of light across the sample to build up an image one pixel at a time. This technique, invented in 1955 and called confocal microscopy, physically blocks out-of-focus light, making 3D reconstructions of samples possible. The cost of this optical sectioning is a slow acquisition speed due to the nature of a point-scanning technique.

A faster way of removing out-of-focus light can be achieved with total internal reflection fluorescence (TIRF). In this scheme light is projected into the outer edge of the back-aperture of a specialised lens, such that it emerges from the lens steeper than the critical angle required for reflection, rather than refraction, from a glass interface. Although this means that no light passes above the microscope slide, energy from the evanescent is able to couple into molecules close to the coverglass, inducing fluorescence. TIRF illuminates molecules within

100 nm of the coverglass without being obscured by out-of-focus light, with the disadvantage that details deeper into the sample cannot be observed.

Sheppard [1990 + others] and Wilson [1984, 1997] showed that optical sectioning can be achieved computationally if the illumination light is modulated with a structured pattern. Illuminating the sample through a grating to produce a 2D sinusoidal illumination pattern, and moving the pattern through 3 evenly-spaced phase steps, produces 3 images described by Equation 2.1, where $i = \{1,2,3\}$ for three phase steps, $W$ is the sample fluorescence produced by a widefield image, $m$ the modulation index, $t$ the modulation frequency, $x$ a lateral spatial dimension in the sample plane, and $phi_i = \left\{0, \frac{2\pi}{3}, \frac{4\pi}{3}\right\}$ the phase steps of the illumination pattern.

$$I_i = W\left(1 + m\cos\left(tx + \phi_i\right)\right) \tag{2.1}$$

To remove the out-of-focus unmodulated component from the reconstructed image $I_R$, we borrow from communication theory and use either square-law detection shown in Equation 2.2, or heterodyne detection shown in Equation 2.3.

$$I_R = \left((I_1 - I_2)^2 + (I_2 - I_3)^2 + (I_1 - I_3)^2\right)^{\frac{1}{2}} \tag{2.2}$$

$$I_R = \left|I_1 + I_2 \exp\left(\frac{2\pi j}{3}\right) + I_3 \exp\left(\frac{4\pi j}{3}\right)\right| \tag{2.3}$$

### 2.1.2 Resolution-doubling SIM

The widefield image $W$ produced by a fluorescent microscope is made up of the underlying sample fluorescence, $S$, convolved with the microscope point spread function (PSF), $P$. When light from the sample plane passes through a lens, the finite aperture means that a small point of light is spread into a larger spot in the image plane. This means that two independent points of light which are too close together cannot be resolved; that is, the lens acts as a low pass filter for spatial frequencies. This can be seen in Figure 2.1a, where spatial frequencies cut off above the diffraction limit.

The same structured illumination microscopy (SIM) used for optical sectioning can be re-purposed for surpassing the diffraction limit. First discussed by Lucosz [1963], and then more explicitly described by Heintzmann [1998 in proceedings] for sinusoidal illumination patterns, the first experimental result showing 2D isotropic resolution doubling with 9 raw image acquisitions was published by Gustafsson in 2000. Heintzmann shows that if we take the Fourier transform of the acquired images shown in Equation 2.1, we obtain the set of images shown in Equation 2.4, where ˆ-symbols represent 2D Fourier transforms of their

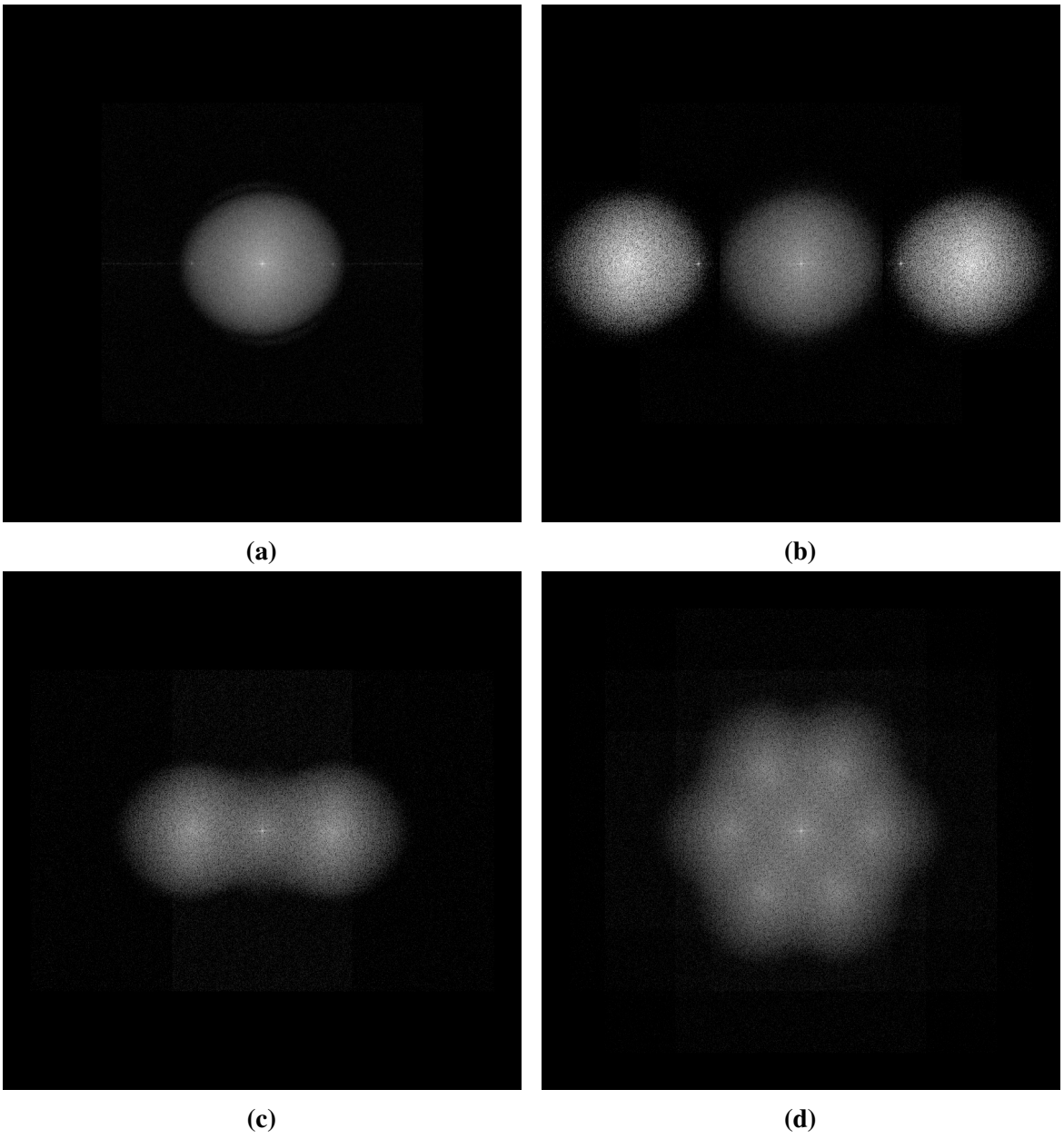


**Figure 2.1:** (a) shows the 2D Fourier transform of a raw SIM image, where the 3 delta peaks from the sine wave illumination pattern can be clearly seen as bright spots. (b) shows the separated components, after extraction by Equation 2.5. (c) shows the components shifted into the correct position in Fourier space, ready for inverse Fourier transforming - but this would only provide resolution enhancement in the x-direction. (d) shows the full Fourier space reconstruction generated from 3 rotations of the SIM pattern, requiring a total of 9 images. The full Fourier space reconstruction has an OTF which has local peaks and is not rotationally symmetric, necessitating the Wiener filtering scheme described in Equation 2.6.

equivalent variables, and $k_x$ is the Fourier variable of $x$, that is $f(x) \Leftrightarrow \hat{f}(k_x)$. Note that $W$ has been replaced with its underlying components $P \otimes S$, where $\hat{P}$, the Fourier transform of the PSF, is known as the optical transfer function (OTF).

$$\hat{I}_i = \hat{P}\hat{S} \otimes \left( \delta(k_x) + \frac{m}{2} e^{j\phi_i} \delta(k_x + t) + \frac{m}{2} e^{-j\phi_i} \delta(k_x - t) \right) \tag{2.4}$$

Each acquired raw image contains contributions from each of 3 shifted Fourier components. Remembering $i = \{1,2,3\}$, these Fourier components can be separated into individual images by solving the set of simultaneous equations for $\hat{S} \otimes \delta(k_x)$, $\hat{S} \otimes \delta(k_x + t)$, and $\hat{S} \otimes \delta(k_x - t)$ with matrix inversion, as shown in Figure 2.1b and Equation2.5 [Wicker 1998?]. Assuming phase steps are all equal, performing a 3D Fourier transform of the phase-stepped raw images stacked in the 3rd dimension produces the same result. [Gustafsson 2000?]

$$\begin{bmatrix} \hat{S}(k_x) \\ \hat{S}(k_x + t) \\ \hat{S}(k_x + t) \end{bmatrix} = \begin{bmatrix} 1 & \frac{m}{2}\exp(-0j) & \frac{m}{2}\exp(0j) \\ 1 & \frac{m}{2}\exp\left(-\frac{2j}{3}\right) & \frac{m}{2}\exp\left(\frac{2j}{3}\right) \\ 1 & \frac{m}{2}\exp\left(-\frac{4j}{3}\right) & \frac{m}{2}\exp\left(\frac{4j}{3}\right) \end{bmatrix}^{-1} \begin{bmatrix} \hat{I}_1 \\ \hat{I}_2 \\ \hat{I}_3 \end{bmatrix} \tag{2.5}$$

To complete the reconstruction, the separated Fourier components must be placed at in the correct position in Fourier space, as determined by $t$. Assuming the sinusoidal illumination pattern is generated by the same lens used for imaging the sample, the highest frequency sine wave which can be generated will correspond to delta peaks at the edge of the microscope's support in Fourier space, shown in Figure 2.1a. When the components are shifted into the appropriate location, the support and resolution of the microscope doubles in the $x$ direction, as shown in Figure 2.1c.

To achieve isotropic 2D resolution doubling, the sinusoidal illumination pattern must be rotated to cover more area in Fourier space. Typically a total of three rotations are used, at 60 deg and 120 deg to the original pattern orientation. Performing the reconstruction procedure on all 9 images produces a Fourier space, as shown in Figure 2.1d, gives the desired isotropic resolution doubling.

The final step for this reconstruction procedure is to inverse Fourier transform the reconstructed Fourier space image, producing an image with double the equivalent widefield resolution.

### 2.1.3 Refining the reconstruction algorithm

The 2D OTF of a lens is a rotationally symmetric function which gradually reduces to zero as spatial frequency increases. A line profile through the OTF of an ideal lens in Figure 2.2a

shows that the higher the spatial frequency, the less resolving power the microscope has. In a practical set-up, noise further limits the resolution at high frequencies, such that signal-to-noise ratio decreases as spatial frequency increases.

The signal-to-noise ratio has a particularly dramatic effect in SIM. Since each frequency component $\hat{S}_i$ is multiplied by the microscope OTF, the shifting shown in Figure 2.1b results in an overall OTF which does not gradually decrease, but rather has peaks at certain frequencies, as shown in Figure 2.2b. These peaks translate directly to frequency peaks in the signal-to-noise ratio, therefore the noise pattern in the reconstructed image is no longer white noise with equal power at all frequencies. Furthermore, it can be seen in Figure 2.1d that the reconstructed SIM OTF is not rotationally symmetric. These two aspects combine to cause characteristic hexagonal noise artefacts in reconstructed SIM images.

To reduce these artefacts, Fourier components are reconstructed through a de-noising algorithm. The most common practice is to combine the frequency components through a generalised Wiener filter, as shown in Equation 2.6 [Gustafsson 2008], to produce the reconstructed image $\hat{I_R}$. Note that this equation now describes the general 2D or 3D case, where $\mathbf{k}$ is a vector of Fourier variables and $\mathbf{p}$ is a vector in the direction of the sinusoidal illumination pattern for each direction $d$. $w^2$ is a constant, adjusted empirically depending on the level of noise in the image, and $A(\mathbf{k})$ is an apodisation function.

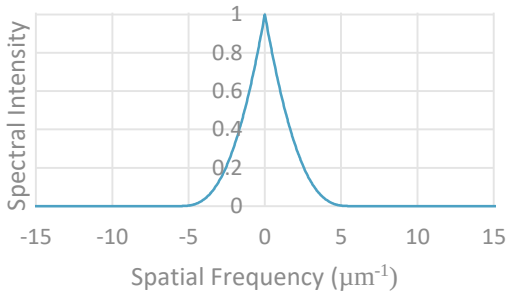
$$\hat{I_R} = \frac{\sum_{d,i} \hat{P}^*(\mathbf{k}+\phi_i\mathbf{p}_d)\,\hat{S}_{d,i}(\mathbf{k}+\phi_i\mathbf{p}_d)}{\sum_{d,i}\left|\hat{P}(\mathbf{k}+\phi_i\mathbf{p}_d)\right|^2 + w^2} A(\mathbf{k}) \tag{2.6}$$

If $A(\mathbf{k}) = 1$, this reconstruction scheme removes the decaying characteristic of the OTF, instead making a flat top-hat filter shown in Figure 2.2c which suddenly cuts off at the doubled resolution limit. This can also be seen in Figures 2.2e and 2.2f, where the Fourier space image now has equal spectral intensity across the reconstructed OTF. However, because the Fourier transform of a top-hat function is a sinc function, this causes ringing artefacts in the reconstructed image. The apodisation function $A(\mathbf{k})$ is therefore chosen as a filter which decays from the 0 frequency to the new resolution limit, for example a Gaussian filter. [cite gross JLS ER paper]

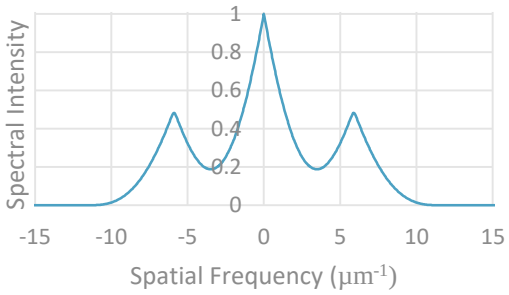
More recent research [cite cite cite] suggests improvements to generalised Wiener filtering, and a detailed discussion of alternative filtering schemes follows in Section 2.4 of this chapter.

### 2.1.4 Aims for LAG SIM

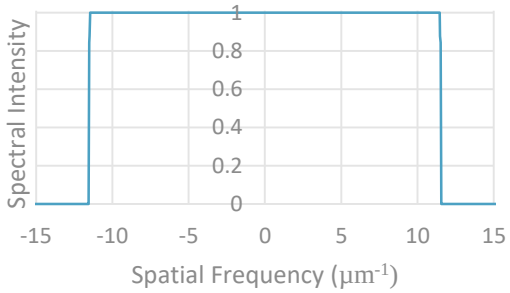
When I arrived in the Laser Analytics Group (LAG) a SIM microscope designed by Laurie Young was reaching the end of its practical life. Issues with device degradation, detailed in



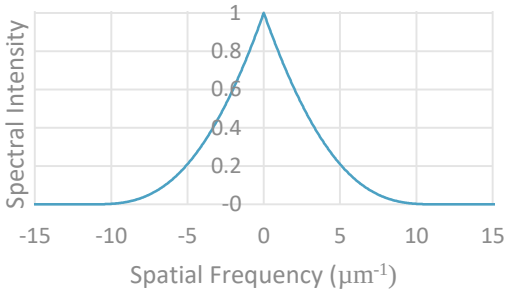

**(a)**



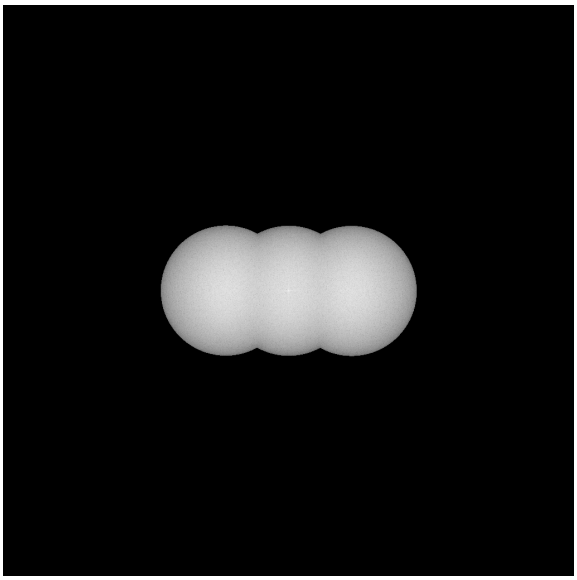

**(b)**



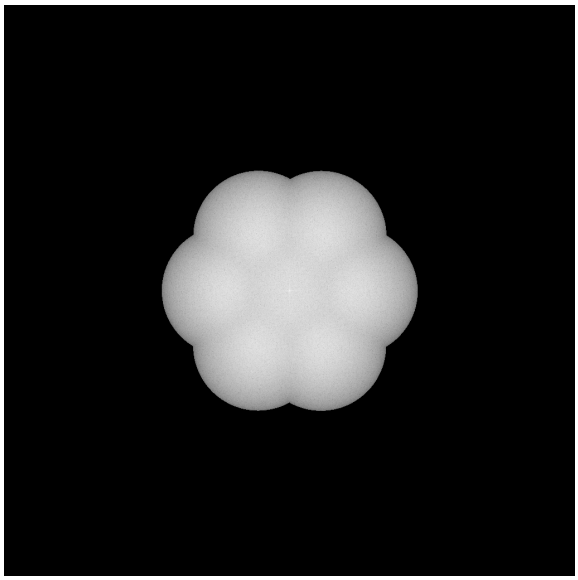

**(c)**




**(d)**



**(e)**



**(f)**

**Figure 2.2:** (a) shows the rotationally-averaged OTF of an ideal lens, which is equivalent to a cross-section through the 2D OTF due to rotational symmetry. After SIM reconstruction, (b) shows that local peaks appear in the OTF, due to shifting in Fourier space; furthermore, the OTF is no longer rotationally symmetric, producing characteristic SIM artefacts. The Wiener filtering scheme described in Equation 2.6 produces a flat OTF shown in (c), with equal power at all frequencies; however the sharp cutoff produces ringing artefacts after Fourier transforming into real space. (d) shows the apodisation filter $A(\mathbf{k})$ which is applied to the OTF to reduce ringing artefacts, and closely resembles the widefield OTF shown in (a). (e) and (f) show the same shifted Fourier components as Figures 2.1c and 2.1d, but reconstructed through a Wiener filter to remove local peaks in the OTF.

Section 2.2.2, limited the imaging speed to 0.1 Hz. Furthermore, a complicated ensemble of software required an expert user to operate the microscope.

A physical relocation of the laboratory in 2016 provided the ideal opportunity to rebuild the microscope and the associated control software. The first priority was to restore the microscope to 11 Hz imaging speed. Once this was achieved, a list of several other aims were devised:

1. Provide an easy method to switch between optical sectioning SIM and resolution-doubling SIM in TIRF (Sections 2.2 and 2.3).
2. Split the fluorescence emission light at the output of the microscope to facilitate simultaneous capture of multiple colour channels (Section 2.2.1).
3. Re-write the control software to make the microscope user-friendly enoguh for a non-expert user to operate unsupervised (Section 2.3).
4. Design reconstruction software to allow users to quickly reconstruct artefact-free images without detailed knowledge of reconstruction algorithms (Section 2.4).

As well as detailing solutions to these challenges, this chapter also presents a showcase of various biological experiments performed with LAG SIM. Section 2.6 can therefore be used a collection of case study examples for anyone wishing to run similar studies.

## 2.2 SIM hardware

### 2.2.1 Optical path design

As part of his PhD work from 2012-2016, Dr. Laurie Young designed and built a SIM microscope in the Laser Analytics Group [2]. The current SIM setup in the Laser Analytics Group ('LAG SIM') is based heavily on this original design, but with several modifications which were implemented by me when the Group moved location in February 2017.

The optical path diagram is shown in Figure 2.3, and works as follows: laser light is generated by one of three lasers at a wavelength of 488 nm, 561 nm, or 640 nm; its polarisation is aligned with the direction of the structured illumination pattern by a Pockels Cell and achromatic quarter-wave plate; it passes through a beam expander so that it fully covers the SLM display; it reflects and diffracts off the SLM display, which is displaying a binary diffraction grating; it passes through a 'beam minimiser' (a beam expander in reverse) so that it is an appropriate size for the microscope lenses; it passes through a spatial filter at a Fourier plane, which removes all but the ±1 diffraction orders from the SLM diffraction; it reflects off of a dichroic mirror; it passes into the microscope's tube lens; it reflects off of a

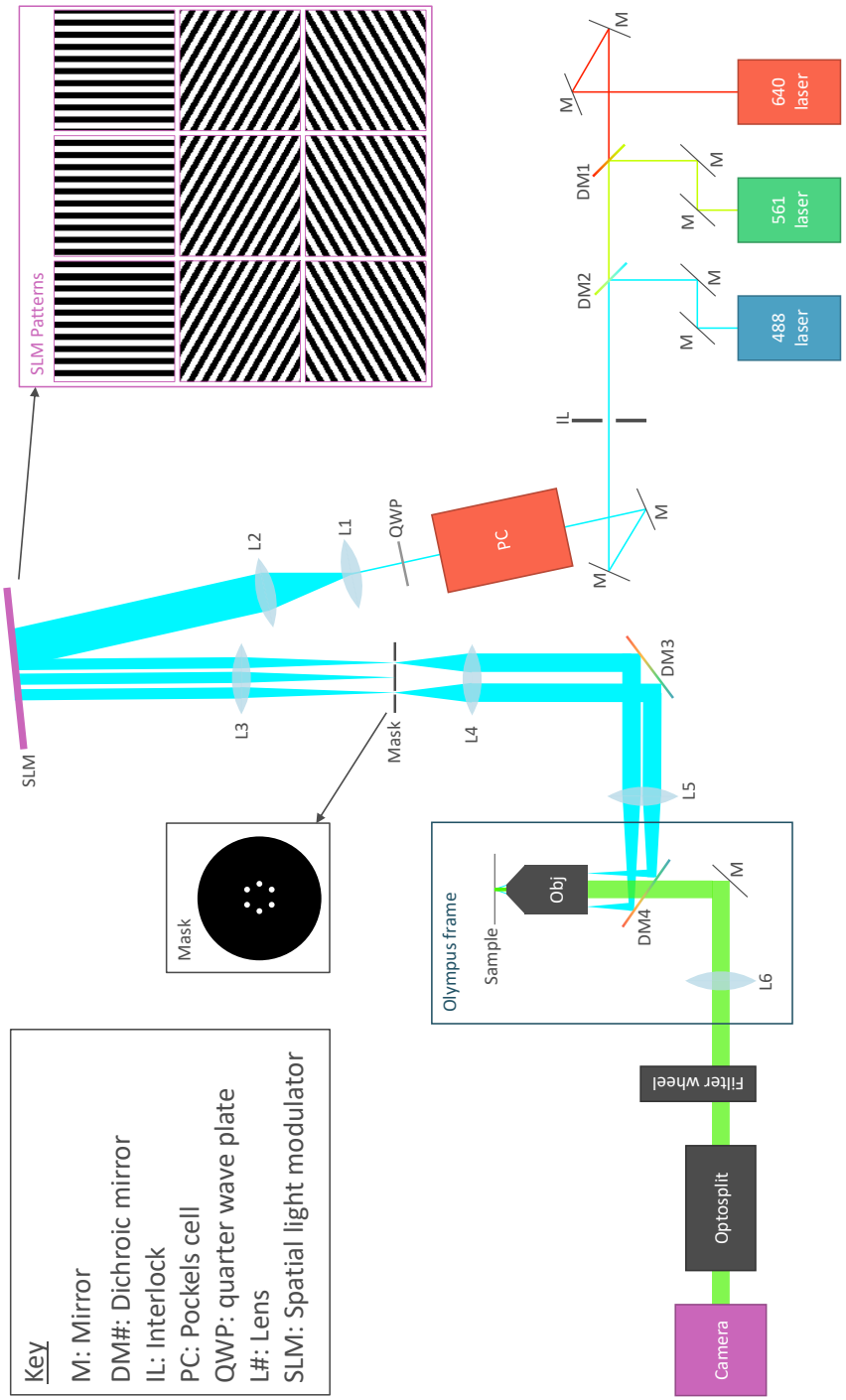



**Figure 2.3:** The optical layout of the SIM aligns light from one of 3 lasers onto an optical path. The light passes through a Pockels cell and quarter wave plate for polarisation rotation, before a square-wave pattern is applied by a spatial light modulator. The patterned light is passed through a spatial mask in the Fourier plane to produce a sinusoidal pattern, which is relayed onto the sample. The objective lens collects fluorescent emission light, which is filtered to remove any reflected excitation light before reaching the sCMOS camera.

second dichoric mirror, which is perfectly matched to the first; and finally it passes into the objective lens and is focussed onto the sample.

This effectively images a low-pass-filtered version of the SLM's diffraction grating onto the sample. The spatial filter means that the binary grating becomes a sinusoidal pattern, with the distance between peaks set by the period of the diffraction grating shown on the SLM display and the demagnification of the system from SLM to sample. An equivalent way of conceptualising this is to picture the two filtered beams of light entering the back aperture of the objective lens, such that they produce an interference pattern when they meet at the sample plane. The result is a sinusoidal illumination pattern at the sample plane, the orientation and period of which can be set by a binary pattern shown on the SLM.

This SIM is a fluorescence microscope, so that (to a reasonable approximation) the amount of light emitted from a point in the sample is proportional to the amount of light illuminating that point, but the emission light is at a different wavelength to the illumination light. In the LAG SIM microscope, fluorescent emission light travels back through the objective lens, through the dichroic mirror and towards the camera to record a 2D image. Before it reaches the camera, it is filtered in one of two ways, to remove any reflected illumination light. The first option is to use a filter wheel, which contains three filters appropriate for removing the illumination light and can be switched via a serial command from the computer. The second option is to use the Optosplit III from Cairn Research, which uses filter cubes to split the emission light into three paths which are imaged onto separate areas of the camera, facilitating simultaneous imaging of three colour channels.

The Optosplit facilitates a significant increase in speed for multicolour imaging. In this setup, the fastest speed at which the camera can reliably image without dropping frames is 10 ms. A SIM reconstruction requires 9 images, giving a total exposure time of 90 ms per channel; however switching channels with the filter wheel adds an additional 100 ms as the wheel rotates in the next filter and settles. For a 3-colour image using the filter wheel, the fastest SIM imaging rate is therefore 570 ms per frame, or 1.75 Hz. Using the Optosplit is over 6 times faster, giving the full 11.1 Hz imaging rate even for multicolour images.

Finally, an autofocus system between the microscope frame and filter wheel works with the z-control of the microscope stage to maintain a constant focus on the sample. This is useful in two scenarios. Firstly, when a timelapse sequence is captured over a number of hours, the microscope lens drifts downwards away from the sample. Without the autofocus system, this would result in a loss of focus; however by automatically moving the stage as the lens drifts, focus can be maintained over a period of days. Secondly, if a large field of view is imaged, focus can be lost due to the cover glass not being perfectly flat. Again, the autofocus system holds focus over this full field of view, allowing large areas to be imaged.

### 2.2.2 Pockels Cell

In the SIM excitation path, a Pockels Cell is used to rotate the polarisation of the illumination light so that it is aligned with the sinusoidal SIM pattern.

In SIM, it is important that the modulation contrast is as large as possible, that is, that the troughs of the sinusoidal illumination pattern are as dark as possible. When the 9 raw images are computationally recombined, the signal-to-noise ratio of high-spatial-frequency components is directly proportional to the modulation contrast [3].

Modulation contrast of fluorescence emission light can be degraded by a number of sources, including out-of-focus light and aberrations in the imaging system, which is unavoidable in thick samples. Furthermore, if we do not take care with the polarisation of the illumination light, modulation contrast is degraded due to the effects of a high numerical aperture lens.

For two waves to interfere, they must have the same polarisation. [says this on Wikipedia "Wave interference", need better reference.] As light passes through a high numerical aperture lens, the polarisation state which is parallel to the meridional plane (p-polarised) is rotated; whereas the polarisation state perpendicular (s-polarised) is unaffected. To ensure that light from the two back-aperture spots arrive at sample plane with the same polarisation, it is therefore necessary that the light is entirely s-polarised.

If the meridional plane is defined at an angle of 0° around the $\phi$ axis, aligned with the back-aperture spots when the SLM is showing a vertical SIM grating pattern, the incident light should be linearly polarised at $\phi = 90°$ so ensure s-polarised light. However, for 2D isotropic resolution enhancement, the SIM pattern must also be rotated to $\pm 60°$. For these patterns, the spots are no longer aligned with the meridional plane defined for $\phi = 0°$, thus the $\phi = 90°$ polarisation does not represent s-polarised light for these patterns. In order to maintain s-polarised light, thereby ensuring maximum pattern contrast, the polarisation must be rotated for each orientation of the SIM pattern.

[Need an image in here showing a lens, the meridional plane, two spots from the SIM pattern, and how they rotate for a different SLM pattern)

The original SIM system, detailed in reference [2], used a liquid crystal variable retarder (LCVR) from Meadowlark Optics [cite old magazine] to rotate the polarisation to align with the SLM's grating orientation. This suffered an unexpected problem where the laser beam damaged the liquid crystal material leaving a visible spot on the LCVR and no longer retarded the light, so the system no longer rotated polarisation. The product line has since been discontinued. As an interim solution, an achromatic half-wave plate was mounted in a serial-controlled rotation stage in place of the LCVR. This setup successfully rotated the polarisation and ensured good modulation contrast, but was only able to image at 0.1 Hz.

In order to restore the LAG SIM to 11 Hz imaging speed, we purchased a Pockels Cell from Conoptics. Similarly to the LCVR, this allows the retardation of one polarisation axis which, in combination with a quarter-wave plate, allows for rotation of linear polarisation. The level of retardation is controlled by a voltage across the Pockels Cell, restoring polarisation rotation to the LAG SIM with no mechanical movement.

### 2.2.3 Pockels Cell Alignment

To use the Pockels Cell as a polarisation rotator, light entering the cell must be linearly polarised and aligned at 45° to the fast axis of the device. The Pockels Cell from [?SOMEONE] was delivered with a Glan Taylor polariser appropriately aligned in the factory.

To align the Pockels Cell to the rest of the system, the following procedure was devised:

1. Place the Pockels Cell in 5-axis mount
2. Coarsely align so that the beam passes through the centre of the Pockels Cell
3. Using a power meter after the Pockels Cell, rotate the cell until power is maximised
4. Adjust the tip-tilt of the Pockels Cell, maximising power again
5. Insert the quarter-wave plate (QWP) into the beam path mounted in a rotation mount
6. With the power meter after the QWP, rotate the QWP until power is maximised

Although this alignment procedure could be completed with any laser, for LAG SIM it is best to use the 561 laser, as this is the closest to the centre wavelength of the achromatic quarter-wave plate.

The polarisation state of the light was measured to ensure the Pockels Cell was behaving as expected by using a linear polariser in a rotation mount as an analyser before the power meter. The laser emission entering the Pockels Cell was linearly polarised, with an extinction ratio of 1:[???] for the 561 laser. The Glan Taylor polariser theoretically increases the extinction ratio to >1:100 000 [4], although because it is attached to the Pockels Cell in the factory, this extinction ratio cannot be measured. Light exiting the Pockels Cell is circularly polarised; using the analyser found an extinction ratio of [1:1??]. Finally, the QWP restores the light to linear polarisation, but rotated compared to the input as determined by the voltage over the Pockels Cell. Extinction ratio at the output of the QWP was 1:[???].

### 2.2.4 Pockels Cell Voltages

The voltage across the Pockels Cell determines the amount of optical anisotropy in the cell, which in this setup is directly proportional to the polarisation rotation. The voltages required to control anisotropy are of the order of 200 V. To achieve these voltages, an amplifier supplied by [THE COMPANY] takes a 0 V to 2 V [check this] input and provides 200 VV

gain. The amplifier input voltage is generated by a National Instruments Digital to Analog Converter (DAC), which is controlled by a LabVIEW program.

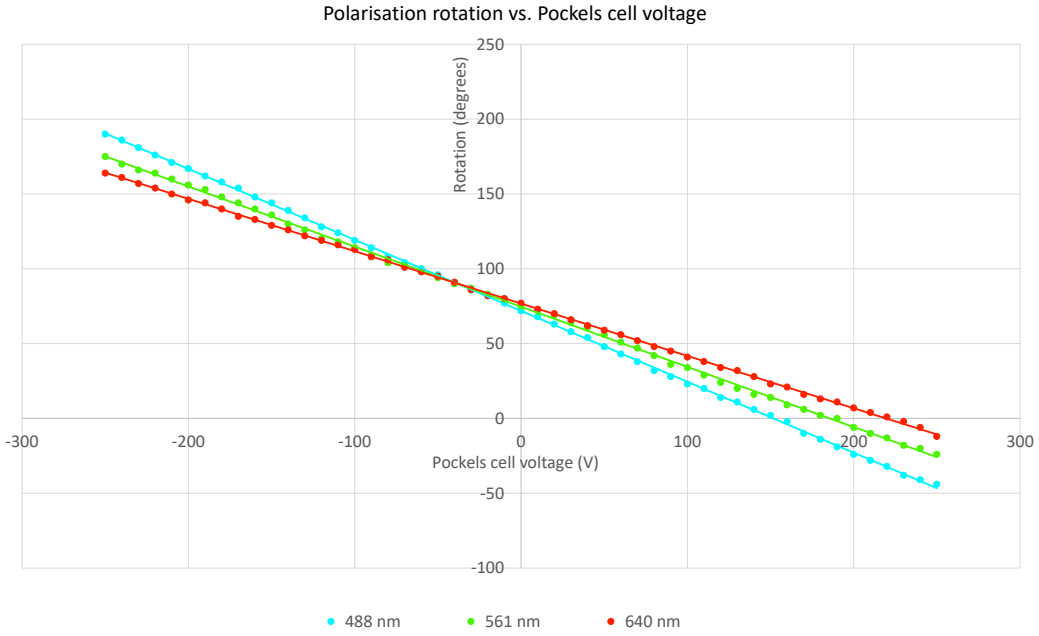



**Figure 2.4:** When used in combination with a quarter wave plate, the voltage across the Pockels cell is proportional to polarisation rotation for each wavelength of laser light. Though the relationship is slightly different for each wavelength, at low voltages the gradients are similar enough that the 561 nm voltages can be used when the Optosplit is being used to image multiple channels simultaneously.

Figure 2.4 shows the relationship between voltage and polarisation rotation for the three laser lines used on the LAG SIM. Two useful properties of the Pockels cell are evident. Firstly, the device is able to provide a full 360° polarisation rotation, more than adequate for the $\pm 60$ deg required to align polarisation with the SIM pattern orientations. Secondly, to achieve a given rotation a similar voltage is required for all wavelengths. This is an especially important property for using the Optosplit, when all lasers are imaging the sample simultaneously.

To calculate the voltages required to achieve maximum pattern contrast at each wavelength and orientation, a program was written in LabVIEW to sweep the Pockels' Cell voltage across the input range in steps of 0.2 V. At each voltage step, 9 SIM images of sub-diffraction beads were recorded, and their contrast was measured by a MATLAB script. The contrast was plotted against voltage, as shown in Figure 2.5. This procedure was repeated for every pattern orientation and every wavelength, producing a table of optimal voltages for each combination, which can be seen in Figure 2.6.

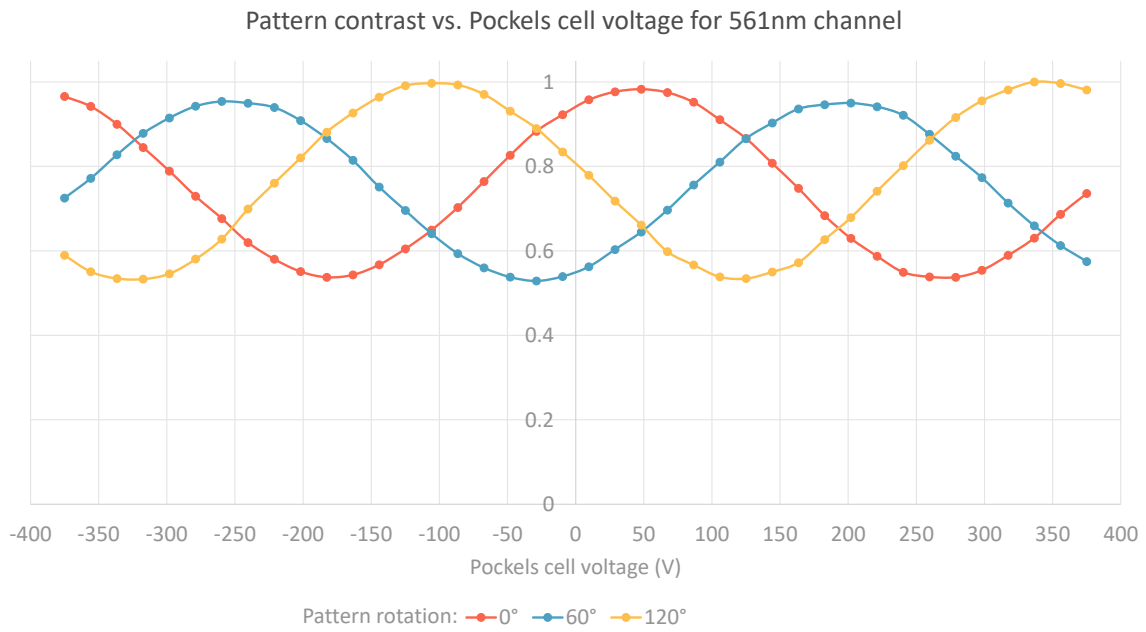



**Figure 2.5:** Pattern contrast was measured on a 100 nm bead sample as the voltage was swept from −375 V to 375 V to find the voltage which gave maximum contrast. The process was repeated for each rotation of the sinusoidal SIM pattern, and for each wavelength of light.

## 2.3 LabVIEW Hardware Control

### 2.3.1 Control requirements

The LAG SIM has a number of hardware components which must all work together to acquire a set of images appropriate for SIM reconstruction. Specifically, the following sequence of events must take place:

1. Set filter wheel to correct position for the excitation and emission wavelengths
2. Set voltage across Pockels Cell for correct polarisation rotation
3. Set SLM pattern to the correct orientation and phase
4. Turn on the laser
5. Open the camera shutter for the exposure time
6. Close the camera shutter
7. Turn off the laser
8. Record the image

Steps 2 to 8 must be repeated for each of the 9 illumination patterns, and the entire process then repeated for additional colour channels if the Optosplit is not being utilised. For a time

series, this entire process must then be repeated for the required number of frames; for a z-stack, the height of the stage must also be adjusted before repeating the process.

The original SIM, detailed in [cite JOVE, again!], used a separate software program for controlling each piece of hardware. This caused a number of problems: firstly, setting up experiments was an complicated process prone to human errors; and secondly, making modifications to the imaging parameters during an imaging session was difficult and time-consuming. When imaging live-cell biology, there is a real impetus to image quickly, as cells are becoming unhealthy and soon dying on the microscope stage.

Furthermore, the complicated system meant that only one person in the LAG had the knowledge to gather images from the SIM. This meant that any biological experiments which would benefit from SIM required my direct supervision, so that many projects were not able to use the SIM, and I personally had little time for other research aims. There was a clear need for a new, unified SIM interface, which could be used by non-expert users with minimal training.

### 2.3.2 LAG SIM Interface

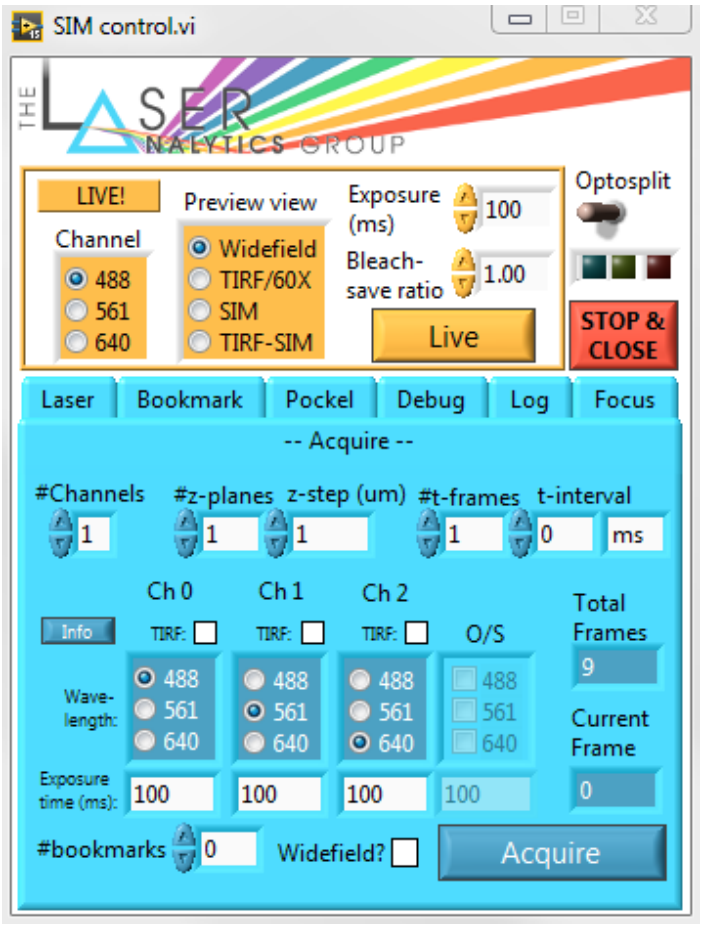
The new interface was built in LabVIEW. Although each hardware component came with its own control software, all components could also be controlled either through serial commands, analog voltages, or transistor-transistor logic (TTL) levels. As a graphical language with hardware control in mind, LabVIEW provides easy-to-use built-in blocks to fulfil each of these functions, as well as a clear visual representation of a sequence of control events. Furthermore it makes designing user interfaces a straightforward process, allowing a programmer to create a user-friendly program for the end user. This made LabVIEW an ideal candidate for building an accessible program for controlling the LAG SIM.

The LAG SIM interface is shown in Figure 2.6. The two key main areas of use are the Live Mode area and the Acquire tab, although for the sake of completeness each tab is detailed in this section.

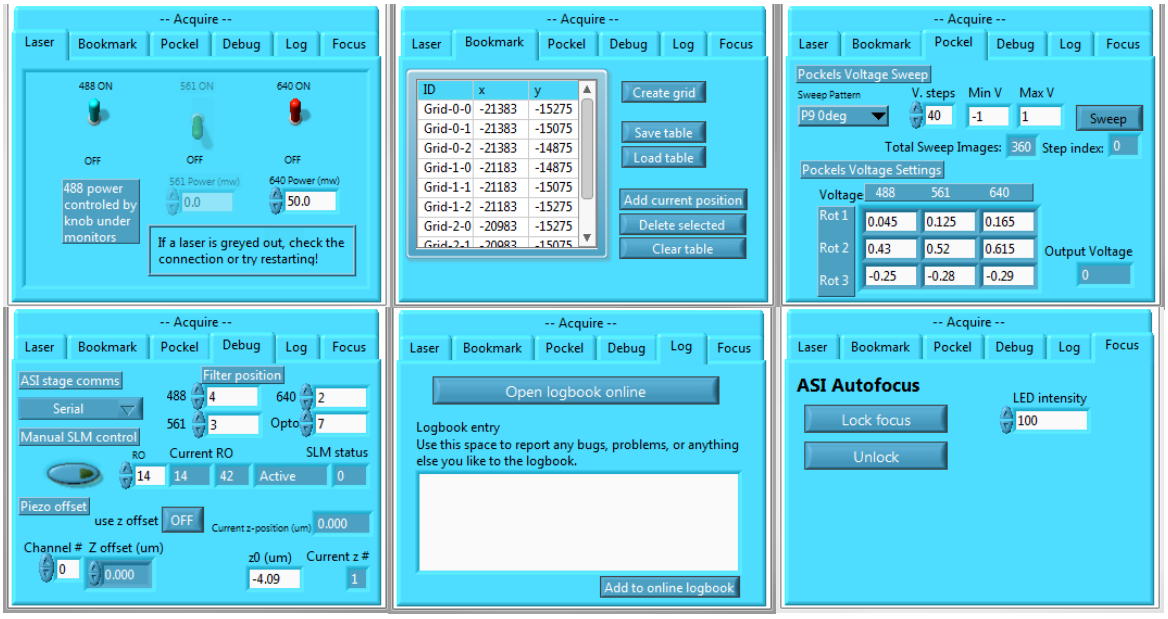
### 2.3.3 Live tab

The live tab is always visible whenever the LAG SIM Control software is open. It works in conjunction with the HCImage camera software provided by Hamamatsu, which provides a live display of the camera chip.

In live mode, the illumination wavelength can be changed with the radio buttons or with the F2, F3 and F4 keys for fast, mouse-free operation. This allows users to quickly identify cells in one channel, for example with a fluorescent nucleus marker, and confirm presence of




**(a)**




**(b)**

**Figure 2.6:** The LabVIEW user interface used to control the LAG SIM has been designed for operation by a non-expert user, with advance tasks hidden in the extra tabs shown in (b).

another marker in the second channel. This fast channel switching minimises photobleaching when panning for healthy cells to image in detail.

Another live-mode feature designed to reduce bleaching is the so-called 'bleach-save ratio.' This is a number between 0 and 1 which switches off the laser for a fraction of the exposure time. By setting the exposure time and bleach-save ratio as small as possible while still being able to find healthy cells, the level of light the sample is exposed to can be minimised. This is especially important in live-cell biology, where excess light can cause phototoxic effects bringing about cell death.

The illumination pattern used for live mode will usually be a pseudo-widefield mode, which emulates widefield illumination by dithering all three phases of a particular orientation over one exposure time of the camera. True epi-illumination is not possible on the LAG SIM due to the spatial mask in the Fourier plane. The user can also choose to preview the SIM pattern by selecting the appropriate radio button; if the stripe pattern is visible, this is a good indication that later reconstruction into a resolution-enhanced image will be successful. If the 1.49 NA TIRF lens is used, then selecting the TIRF radio button sets the SLM pattern such that the ±1 diffraction spots hit the TIRF ring on the objective back aperture, leading to TIRF illumination at the sample. This can be done with the SIM pattern visible or in a dithered mode, as with non-TIRF illumination.

The Optosplit rocker sets the software up for imaging with the Optosplit. [cite] The Optosplit is a device on the output port of the microscope which contains two filter cubes and a set of mirrors so that each channel can be imaged simultaneously onto a separate $512\times512$ area of the $2048\times2048$ camera chip. When the filter cubes are removed, and the Optosplit rocker is in the `OFF` position, the Optosplit has no effect and channels are imaged sequentially, with an appropriate filter for each channel loaded into position by the filter wheel. With Optosplit mode `ON`, however, the live-mode channel selection radio buttons become checkboxes, so that each colour channel can be switched on and off independently. The Optosplit provides much faster imaging speeds, at the potential cost of cross-talk between colour channels. An example of Optosplit use is presented in Chapter 5.

This upper section of the user interface contains two more important components. A laser status indicator shows which lasers are currently exposing, an important safety feature so that users know it is safe to remove their sample after an imaging session. The `STOP & CLOSE` button safely exits the control software, ensuring that the the lasers are switched off, the camera is finished exposing, and the voltage across the Pockels Cell is set to 0 V. To ensure that this exit sequence runs when a user has finished their imaging session, the 'X' close button in the window's title bar is disabled.

### 2.3.4 Acquire tab

Once a healthy cell with the required staining has been located from live mode, it should be captured as a SIM image with the Acquire tab. With the default settings, this will capture a set of 9 raw SIM images following the steps described in Section 2.3.1. Additional settings in this tab allow a set of images to be captured for 3D z-stacks, timelapse imaging, and other multi-image acquisitions.

If the number of z-steps is greater than 1, then the ASI stage will move in the axial direction at the end of every SIM acquisition by the value given in the z-step box. Since the axial resolution of the SIM is 500 nm [check this!!], for a Nyquist-limited z-stack a z-step of 200 nm should be used.

To take a series of images over time, the t-frames should be set to greater than 1. For observing fast, dynamic events, the t-delay should be set to 0 ms. Timelapse imaging, for example capturing an image every minute for an hour, can be achieved by setting the t-delay to 60 s, which will pause the software for the given amount of time at the end of each acquisition.

Multi-channel imaging can be achieved either with or without the Optosplit. If the Optosplit switch is set `OFF`, and the number of channels is set to more than 1, the filter wheel will rotate the correct filter into position after each channel is captured. The exposure time of each raw SIM frame can be set per-channel, and the channel order can be changed with the radio buttons.

Faster imaging can be achieved with the Optosplit. This sets the filter wheel to an empty filter, and filtering is performed with filter cubes in the Optosplit. In this mode, the choice of channels is controlled with checkboxes.

The bookmarks number should be set to the number of bookmarks in the bookmarks table, or a lower number $N$ if the user only wants to capture the first $N$ bookmarked locations from the table. If bookmarks are not to be used, this value should be left at 0 - a value of 1 will move to the first bookmark listed in the bookmark table and capture an image there. Bookmarks are discussed in more detail in Section 2.3.6.

The SIM microscope can be used to capture images in a pseudo-widefield mode, by dithering the SIM pattern as described in Section 2.3.3. This is enabled by ticking the `Widefield` checkbox. This will enabled imaging with a widefield resolution at 100 Hz.

Once the appropriate variables have been set for the acquisition, the total frames is displayed in the `Total frames` box. This number should be entered into the HCImage camera software, to prepare the camera for a high-speed image acquisition. Once the user has pressed `Start` in the HCImage software, the camera buffer is ready to receive images; then pressing `Acquire` in the LAG SIM software starts the acquisition process.

A future version of the software could automate this final step, eliminating the need for the HCImage software. This would require careful access to the camera data stream through LabVIEW.

### 2.3.5 Laser tab

Laser control is integrated into the LAG SIM control software.

At the launch of the software, the controls are greyed-out and disabled. When LabVIEW successfully connects to each laser, its controls become enabled, to switch the lasers on and off and change the power. Lasers are switched off automatically when the software is closed, for added safety when the microscope is unattended.

Having fast and straightforward access to laser control is particularly useful for reducing light dosage in live mode. Power can then be increased for acquisition to obtain the highest possible image quality.

### 2.3.6 Bookmark tab

The LAG SIM LabVIEW program is able to store locations on the microscope slide as bookmarks. If the number of bookmarks in the `Acquire` tab is set to greater than 0, the microscope stage will move before each acquisition to the next bookmarked position in the bookmark table. This is especially useful for imaging a collection of cells over a long period of time.

The SIM user can pan around the microscope slide in `live mode` searching for healthy cells with good fluorescent labelling, and save them to the bookmark table by clicking `Add bookmark`. The location can be returned to by double-clicking on the table entry, or acquired sequentially and repeatedly in the `Acquire` tab.

Furthermore this part of the software contains an option for creating bookmarks in a grid pattern. This can be utilised for imaging large areas, which are then stitched together to form one image, for example the whole mouse brain shown in Figure 2.7.

### 2.3.7 Focus

Using the grid bookmark function to image large fields of view can fail if the coverglass or microscope stage is not perfectly flat, which is usually the case in all practical experiments. To keep the focus locked at the same distance from the sample, the ‘Lock focus’ button begins the autofocus procedure. Assuming the autofocus sensor receives enough reflected signal from the infrared autofocus LED, the sample will be locked a fixed distance from the

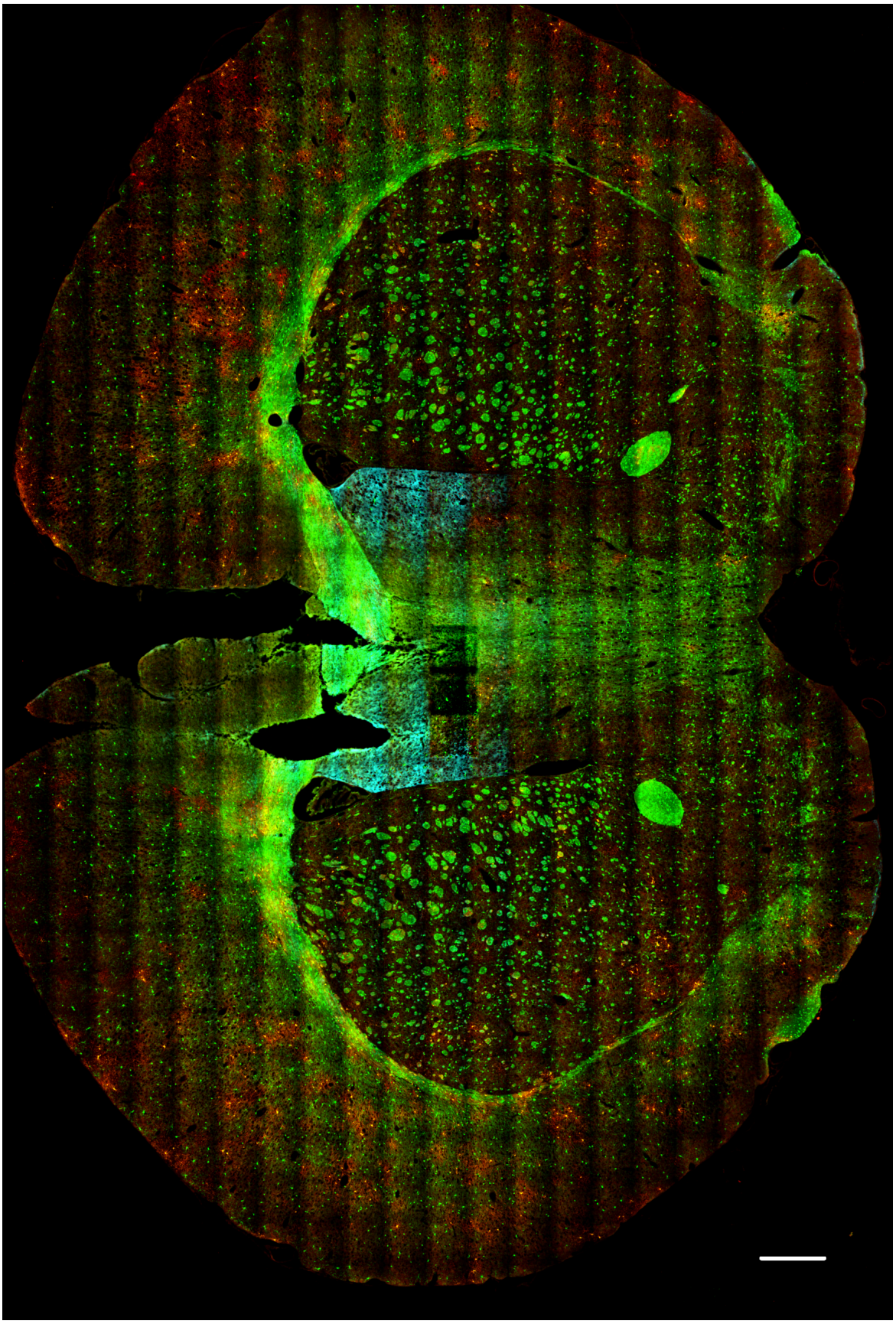


**Figure 2.7:** The bookmark mode offered by LAG SIM can be used to create tiled images, such as this 3-colour image of a mouse brain. Scalebar is 500 µm

objective lens. If not, an error message is returned requesting the user to increase the LED power for a stronger signal. The system can also be used to maintain focus over a time period of hours to days, even as the objective lens drifts away from the sample.

The autofocus is switched off with the 'Unlock focus' button, releasing z-control back to the user.

### 2.3.8 Sweep

The Pockels cell retards one polarisation axis by an amount dependent on the voltage across it. In combination with a quarter wave plate, this allows for linear polarisation rotation necessary for the high pattern contrast required for SIM. The amount of retardation depends on the wavelength of light, therefore different wavelengths require different voltages to achieve the same polarisation rotation.

To find the optimum voltage for each wavelength, functionality is built in to the `Sweep` tab to sweep the voltage from $-1$ V to 1 V, in a number of voltage steps. At each voltage, the SIM pattern's phase is swept and pattern contrast can be measured using a diffraction-limited bead sample. Pattern contrast is calculated with a MATLAB script to produce a graph as shown in Figure 2.5a, showing how pattern contrast changes with voltage.

This gives a coarse estimation of the voltage required to achieve maximum pattern contrast. The sweep limits can then be adjusted either side of the coarse estimation to obtain an accurate voltage, shown in Figure 2.5b.

This process must be repeated for each wavelength, and each rotation of the SIM pattern. Once the process is complete, the optimum voltages can be entered into the table, and will be applied to the Pockels cell during a SIM acquisition.

### 2.3.9 Debug

A number of advanced user options are contained in the `Debug` tab. These extend the functionality of the microscope for non-conventional uses.

The emission filters in the filter wheel change with excitation wavelength for use with commercially available fluorophores. However, some samples - such as semiconducting perovskites - emit around 750 nm when excited with 488 nm light. Unconventional filters can be selected in this tab to image these samples.

Manual SLM control gives the user direct control over the pattern displayed on the SLM. This means specialised patterns, such as a SIM pattern masked by a circle, or even 3 SIM patterns with different orientations overlayed, can be used. These types of patterns are particularly useful for alignment of the system: overlaying 3 orientations of the SIM

pattern produces 6 spots at the Fourier plane, which must be centred around the optical axis; therefore a coarse alignment can be performed using a translucent pinhole alignment tool on the microscope, as shown in Figure 2.8. A fine alignment is then performed using the masked patterns, by ensuring the two circles overlap at the focal plane of the microscope.

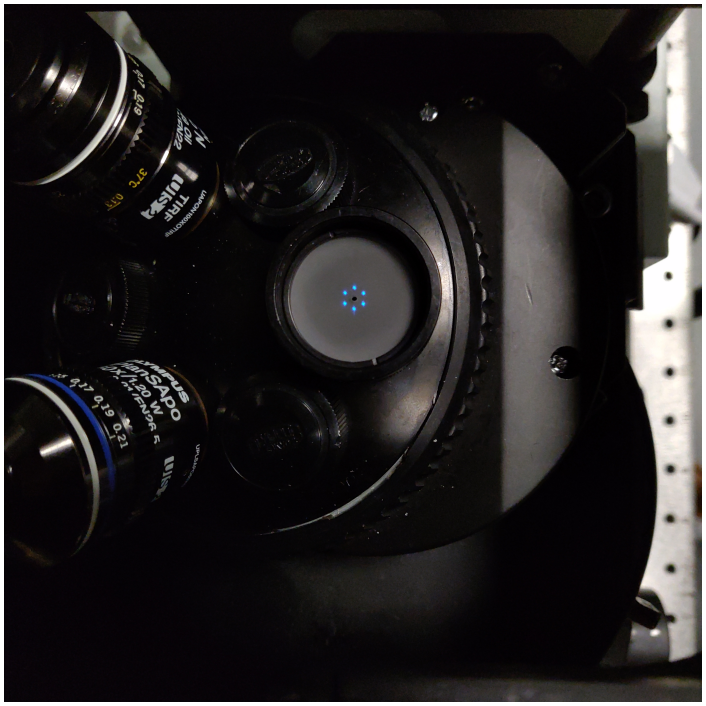


**Figure 2.8:** The 3 orientations of the SIM pattern are overlaid on the SLM, such that 6 spots are produced in the Fourier plane for alignment. The spots should be aligned such that the pinhole is at the centre of the spots.

Finally, the 'Piezo Offset' section adjusts the z-position of the stage for each channel to account for chromatic offset. In practice this offset is very small, on the order of ??100 nm?? and below the diffraction limit of the microscope.

### 2.3.10 Log

The LAG SIM control software logs all activity automatically to Labstep, an online laboratory management system.

When a user opens the LAG SIM software, they will be asked to enter their name for the online logbook. When they press `Start`, the software sends an HTTP request to a Labstep timeline recording the user's name and the start time. Similarly, when the `Stop & Close` the software, their name and finish time is logged to the timeline, along with any errors generated by the software during their use.

Furthermore, every acquisition is also logged to the timeline. This contains metadata relating to the acquisition, for example number of t-frames, z-steps, laser powers, and exposure times. This is especially helpful is the user is later unsure what acquisition parameters were used for a particular image.

The user can also add a customised post to the logbook, for example to record an error with the software. The online Labstep logbook can be opened by clicking the `Open logbook` button, opening the webpage shown in Figure 2.9.

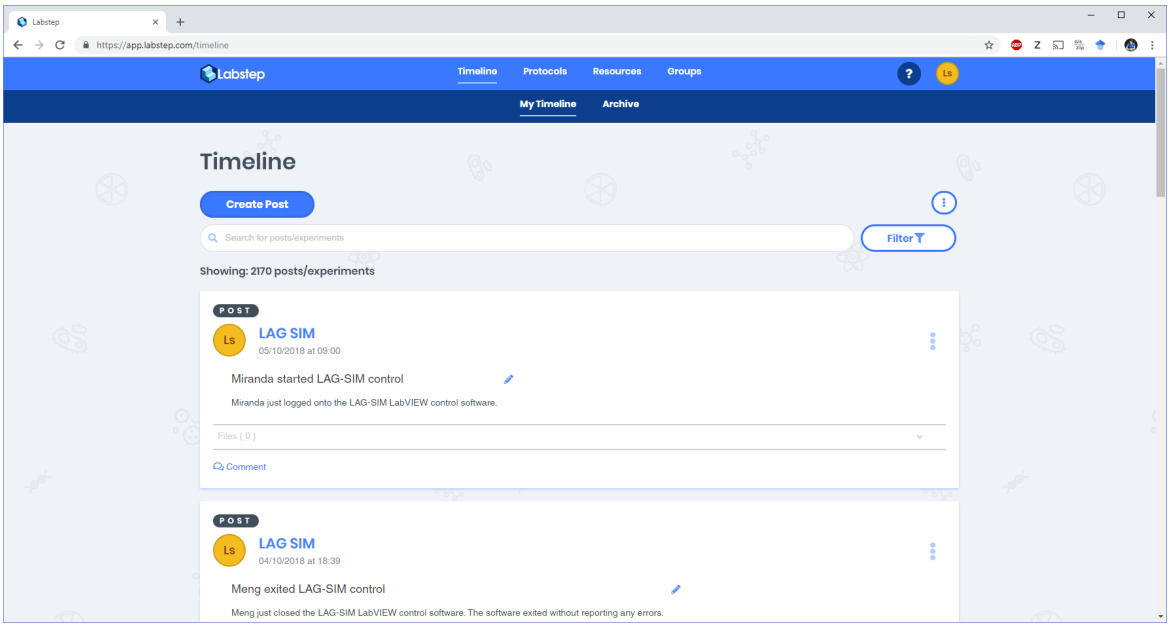



**Figure 2.9:** User activity is automatically logged to an online logbook hosted on Labstep to identify and fix any problems quickly. This figure shows Miranda logged onto the SIM at 09:00, acquired 2 images, then logged off.

Online logging benefits both the microscope user and the developer, allowing problems to be identified and fixed promptly.

## 2.4 Reconstruction Algorithms

Once SIM data is collected, it requires reconstruction to either enhance the resolution, remove out-of-focus light, or a mixture of both.

A basic SIM reconstruction algorithm implements the following steps:

1. Extract pattern parameters (orientation angle and phase shift)
2. Separate Fourier space images into frequency components
3. Weight components in Fourier space by OTF and filter for noise removal

4. Shift frequency components into correct position in Fourier space
5. Inverse Fourier transform for reconstructed image

Additional steps are often added into the basic algorithm to reduce artefacts overall. Raw images may be corrected for consistent intensity and contrast across the dataset, and an additional filtering step for noise removal can be applied before the images are separated in Fourier space. Note also that step 3 can be performed after step 4, but this can lead to additional artefacts in the reconstructed image. Artefacts can furthermore be suppressed by apodisation before the final inverse Fourier transform.

This basic algorithm will provide resolution enhancement, but does not implement optical sectioning for removing out-of-focus light. This requires an additional attenuation step, applied simultaneously with step 3.

Various implementations of the SIM reconstruction algorithm exist. During the course of my PhD, I have utilised and modified three different reconstruction algorithms, and written a plugin for FIJI tailored specifically for LAG SIM and detailed in Section 2.5. Each implementation has its own merits and disadvantages, discussed in the remainder of this section.

### 2.4.1 Lin Shao SIM Reconstruction

The first implementation of the resolution-enhancing SIM algorithm used in the Laser Analytics Group was provided by Lin Shao, who worked in the early days of SIM with Mats Gustafsson. It was developed during his time at Janelia Farm Research Campus, under the supervision of Mats and Eric Betzig. Lin Shao is now at Yale University; although he is contactable, he is not actively maintaining this piece of software.

The program has no graphics user interface, but is called from the command line. Since command line calls can be made from most other programming languages, a custom interface was built in MATLAB for passing in reconstruction parameters, and reconstructing more complex datasets such as multi-channel z-stacks.

The program was not built for widespread distribution, and as such there is no documentation for it. Nor is the program publicly available; requests for a copy of the software have to be made directly to Lin Shao. This means the software is complicated to use, and has not been tested and verified by the wider community.

The algorithm does not implement OTF attenuation, which is used for optical sectioning. This means it cannot remove out-of-focus light, so is not appropriate for reconstructing 3D image stack.

Lin's algorithm is written in native CUDA code to run on NVidia graphics cards. This makes reconstruction very fast - around 1 frame per second on a [SIM COMPUTER GRAPH-

ICS CARD]. However, native CUDA must be recompiled from source on every new computer it is used on, and furthermore because the software has not been updated, it does not work with the latest version of CUDA. Since Lin's implementation is fast becoming obsolete, it is now only installed on one computer in the Laser Analytics Group, which cannot have graphics updates installed.

### 2.4.2 SROS-SIM

Developed by Florian Ströhl in the Laser Analytics Group shortly before I arrived, the SROS-SIM program implements both optical sectioning and resolution enhancement.

SROS-SIM is a MATLAB implementation of the Super Resolution Optical Sectioning algorithm detailed by O'Holleran and Shaw in 2014 [cite]. Optical sectioning removes out-of-focus light, so that 3D images can be reconstructed, for example of a whole cell with mitochondria labelled.

As a set of MATLAB script and functions, the code can be executed on any computer with a MATLAB license. Therefore the implementation is available for all major desktop operating systems, and furthermore is easy to modify. Indeed, I have made modifications to the original code to perform operations on the GPU, bringing a 20X increase in reconstruction speed.

SROS-SIM has the disadvantage of not being publicly available or published and verified by the community. Furthermore the need for an expensive MATLAB license limits who can use the software. Finally, the important step of detecting the illumination pattern parameters is not fully automated, so that the software always requires some user input can cannot be fully automated for a batch reconstruction task.

### 2.4.3 fairSIM

Developed by Marcel Müller and recently published in Nature Methods, fairSIM is an open-source plugin for ImageJ/FIJI, a popular image analysis suite which is also open source and free [5]. The software is well-documented and hosted on Github, so is continually updated and improved by the community.

As a reconstruction algorithm, fairSIM implements resolution enhancement and optical sectioning, with two different noise filters. It provides the user with complete control over the input parameters, and with debugging options enabled will show detailed visual output at each stage of the reconstruction. Although the number of options can be overwhelming for some users, it is powerful and useful software for advanced SIM users.

The major downfall of fairSIM is the user interface. Reconstruction can only be performed on a single-colour, single-frame SIM image. This process requires around 20 clicks, and input parameters must be resubmitted for every reconstruction. This, coupled with the fact that fairSIM does not have GPU support, makes reconstruction very slow, especially for large, multi-colour image sets.

### 2.4.4 LAG SIM

LAG SIM is my personal implementation of the SIM reconstruction algorithm, built using the well-documented fairSIM Java library. It is designed specifically for the Laser Analytic Group's SIM, simplifying the interface for non-advanced users, and implements a unique parameter-finding technique.

It is available publicly as an ImageJ/FIJI plugin. This means that it is free, open source, and stays up-to-date for all users.

The next section describes the LAG SIM plugin in detail, which provides a valuable resource to anyone reconstructing LAG SIM data in the future.

## 2.5 Reconstruction with LAG SIM

Setting SIM reconstruction parameters, particularly for noise filtering and optical sectioning, can be something of a black art. In LAG SIM, building on the fairSIM plugin, there is a choice of 2 filters, which can be used in combination, 2 filter parameters, 2 apodisation parameters, and 2 attenuation parameters, making a high-dimensional parameter space to optimise over. Selecting appropriate values for these parameters requires understanding of what each is for; setting optimum values requires experience and expert skill.

LAG SIM provides a number of user interfaces to assist with parameter optimisation, as well as a number of other tools for batch reconstruction and quick pseudo-widefield previews of the data. The main interfaces are shown in Figure 2.10.

This section contains an explanation of each parameter, and finishes with a workflow for choosing optimal parameters. I hope that this provides a useful resource for anyone reconstructing images from the LAG SIM.

### 2.5.1 Illumination detection

In order to produce an artefact-free reconstruction, the SIM algorithm requires measurement of the illumination pattern to a sub-pixel accuracy. This is necessary for correctly performing the Fourier-space shift.

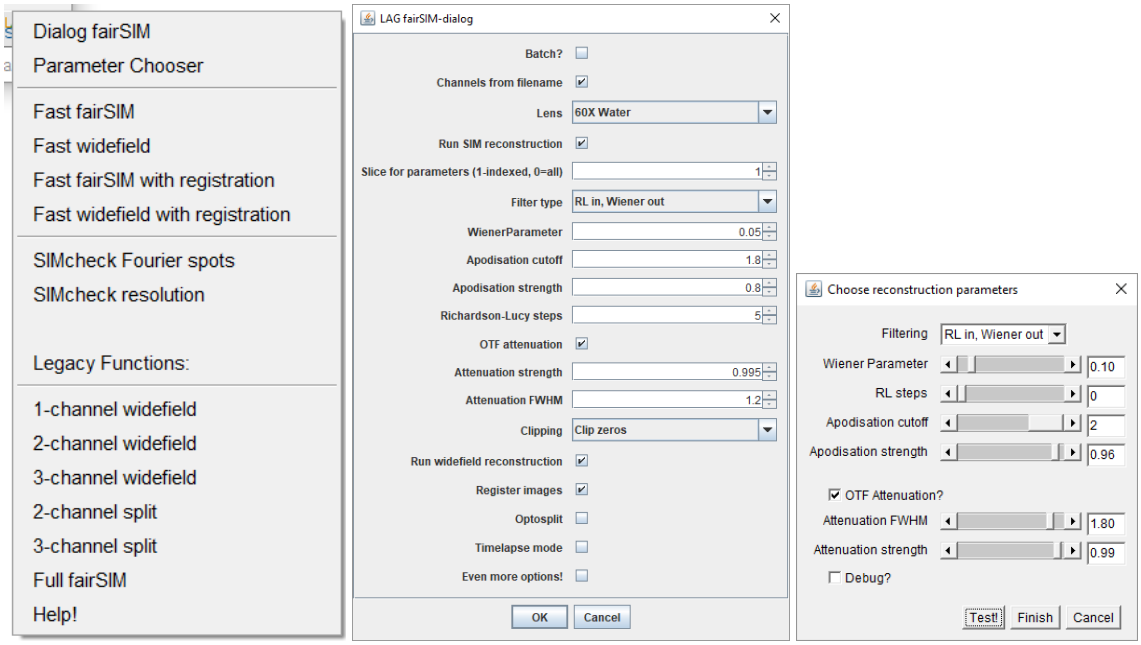



**Figure 2.10:** The LAG SIM FIJI/ImageJ user interface, utilising the fairSIM reconstruction algorithm, is specifically designed for the LAG SIM to make artefact-free reconstruction quick and straightforward. (a) shows the main LAG SIM menu, with the many reconstruction and SIMcheck tools created; (b) shows the main reconstruction interface, facilitating batch reconstruction; and (c) shows the quick parameter tester, for finding the optimal reconstruction parameters.

The LAG SIM reconstruction interface (Figure **??**) allows the user to choose which slice the illumination pattern is measured from. For z-stacks, it is best that the most in-focus slice is used, to get the most accurate measurement. Conversely, for a time-series it is best that slice 1 is used, since it will have the least amount of photobleaching.

Illumination detection with LAG SIM has been designed to require the least amount of user input necessary. The user simply needs to select the objective lens used, and the software will infer the necessary information such as pixel size and numerical aperture. Wavelength can be extracted directly from the filename if the file is saved using the correct naming convention.

### 2.5.2 Wiener filter and apodisation

When the Wiener filtering scheme is used for noise removal, the Wiener parameter and apodisation parameters are used in a complementary fashion.

Noise filtering is particularly important in SIM due to the non-conventional relationship between signal-to-noise ratio and spatial frequency, caused by Fourier-space shifting. A conventional 2D microscope OTF has a peak at 0, and decreases symmetrically, as shown in Figure [?]. As noise is assumed to be constant over all spatial frequencies, this causes a decrease in signal-to-noise ratio at high spatial frequencies. In SIM, a super-resolution image is built up by shifting the OTF in Fourier space. This results in a local maxima in signal-to-noise ratio, as shown in Figure [?]. The extra noise in this region causes swirling hexagonal noise patterns, seen in Figure [?].

Whilst these patterns are simply how noise looks in a SIM image, they can easily be misinterpreted as features. The Wiener filter is a conventional noise filter used in image processing, which successfully removes hexagonal SIM noise as shown in Figure [?]. The 'Wiener parameter' input is able to control how much noise filtering is applied, as demonstrated by Figures [?].

Aggressive noise removal, using high values for the Wiener parameter, is not without consequence. Figure [?] shows that as the Wiener parameter is increased, the OTF of the reconstruction becomes less physical. This results in ringing and negative pixel values around real features in the reconstructed image, seen in Figure [?].

To reduce ringing artefacts, the reconstructed image is passed through an apodisation filter.

Apodisation is a type of low-pass filter, reducing power in higher spatial frequencies. The radius of the apodising filter in Fourier space is controlled by the 'apodisation cutoff' parameter, where a value of 1 corresponds to the cutoff spatial frequency of the objective lens based on its numerical aperture and the emission wavelength of the image. The shape

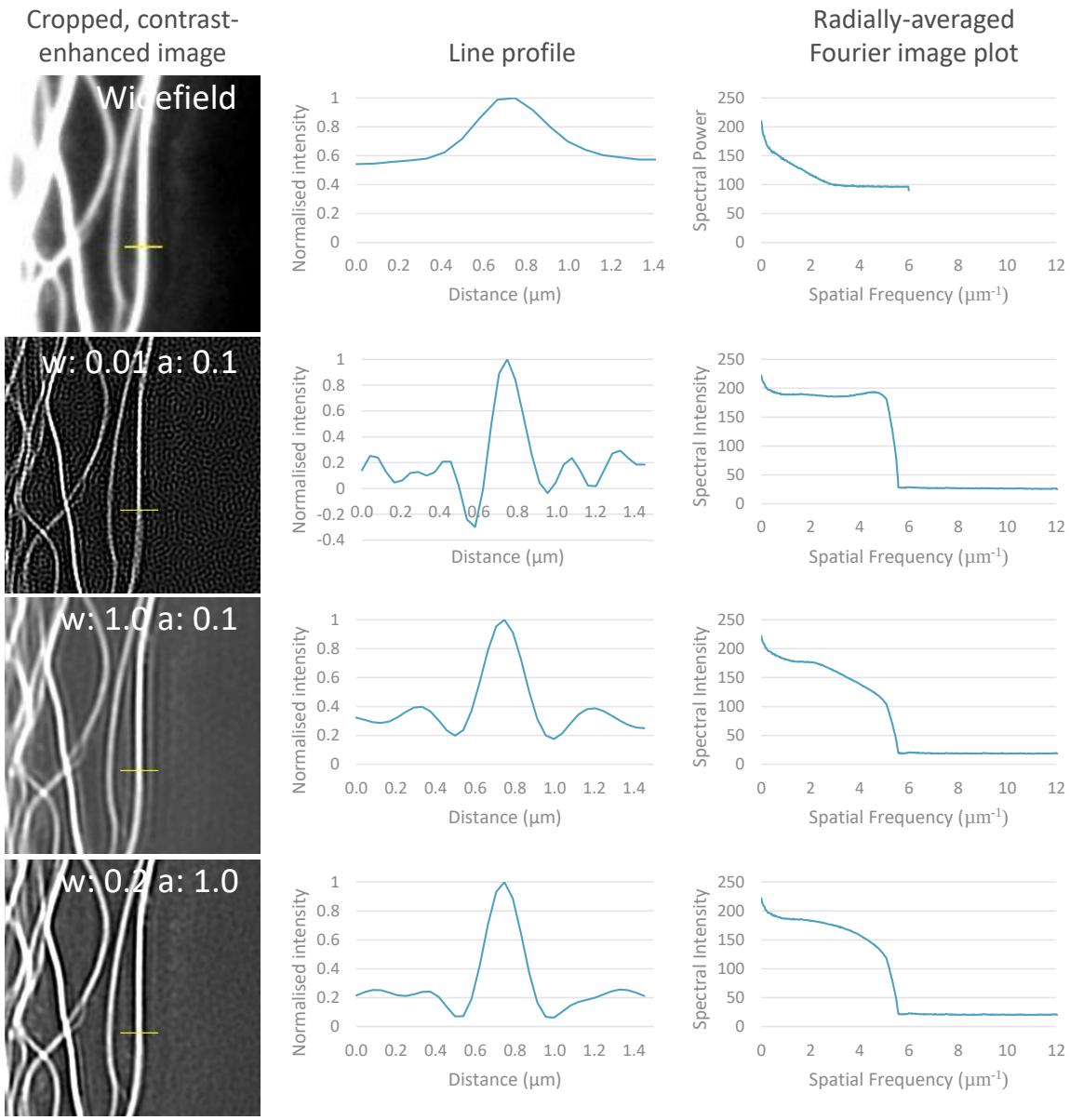



**Figure 2.11:** The left column shows crops from a SIM reconstruction with various values of Wiener parameter (w) and apodisation strength (a), and a widefield image for comparison. Note that image contrast has been greatly enhanced to observe the noise pattern. The middle column shows the line profile through a tubule, showing the noise pattern with a low Wiener parameter, ringing with a high Wiener parameter, and suppression of noise and ringing using apodisation. The right hand column shows the radially averaged Fourier transform plot of each image; note the local peak when a low Wiener filter is used, which boosts noise at this frequency causing the unconventional noise pattern.

of the apodisation filter is controlled by the 'apodisation strength' parameter, where a value of 0 gives a tophat filter, $\frac{1}{\sqrt{2}}$ a triangular filter, and larger values reduce the full-width half maximum.

The apodisation cutoff should be as large as possible, with a maximum value of 2 for full SIM resolution enhancement. Smaller cutoff values reduce the radius of the reconstruction in Fourier space, directly reducing resolution; a value of 1 corresponds to widefield resolution. Furthermore, the apodisation strength should be as small as possible whilst still removing ringing artefacts for maximum resolution.

### 2.5.3 Richardson-Lucy Filtering

An alternative filtering scheme for noise removal is the Richardson-Lucy deconvolution. The Richardson-Lucy filtering scheme has several advantages over Wiener filtering: it guarantees non-negative pixel values, removing the need for an apodisation step; it does not cause ringing in the reconstructed image; and it only takes one parameter LAG SIM.

Richardson-Lucy deconvolution is an iterative process, where each iteration of the algorithm converges on the maximum likelihood solution of an image corrupted with Poisson noise. The number of iterations is set in LAG SIM by the user. More iterations leads to a thinner point spread function, and thus higher perceived resolution, but also leads to noise amplification, which can cause a speckle pattern to appear in the reconstructed image.

This filtering scheme can be used on the final reconstructed image in place of Wiener filtering. It can also be used on the raw data set before reconstruction, reducing noise at the start of the algorithm. Pre-filtering is especially effective at removing the strange SIM speckle, since less noise is shifted and boosted in Fourier space.

Richardson-Lucy input filtering can be used in combination with either the Wiener filter or with more Richardson-Lucy filtering on the reconstructed image. Therefore there are in total 4 filtering schemes offered by LAG SIM: Wiener-output; RL-output; RL-input, Wiener-output; and RL-input, RL-output.

### 2.5.4 OTF Attenuation

In conventional widefield microscopy, light from areas above and below the focal plane are captured by the lens, causing out-of-focus light and effectively reducing the axial resolution of the microscope. When observing the 3D widefield OTF, it is obvious that this is the case. The OTF has no support in the axial direction at low frequencies, which is known as the 'missing cone' problem.

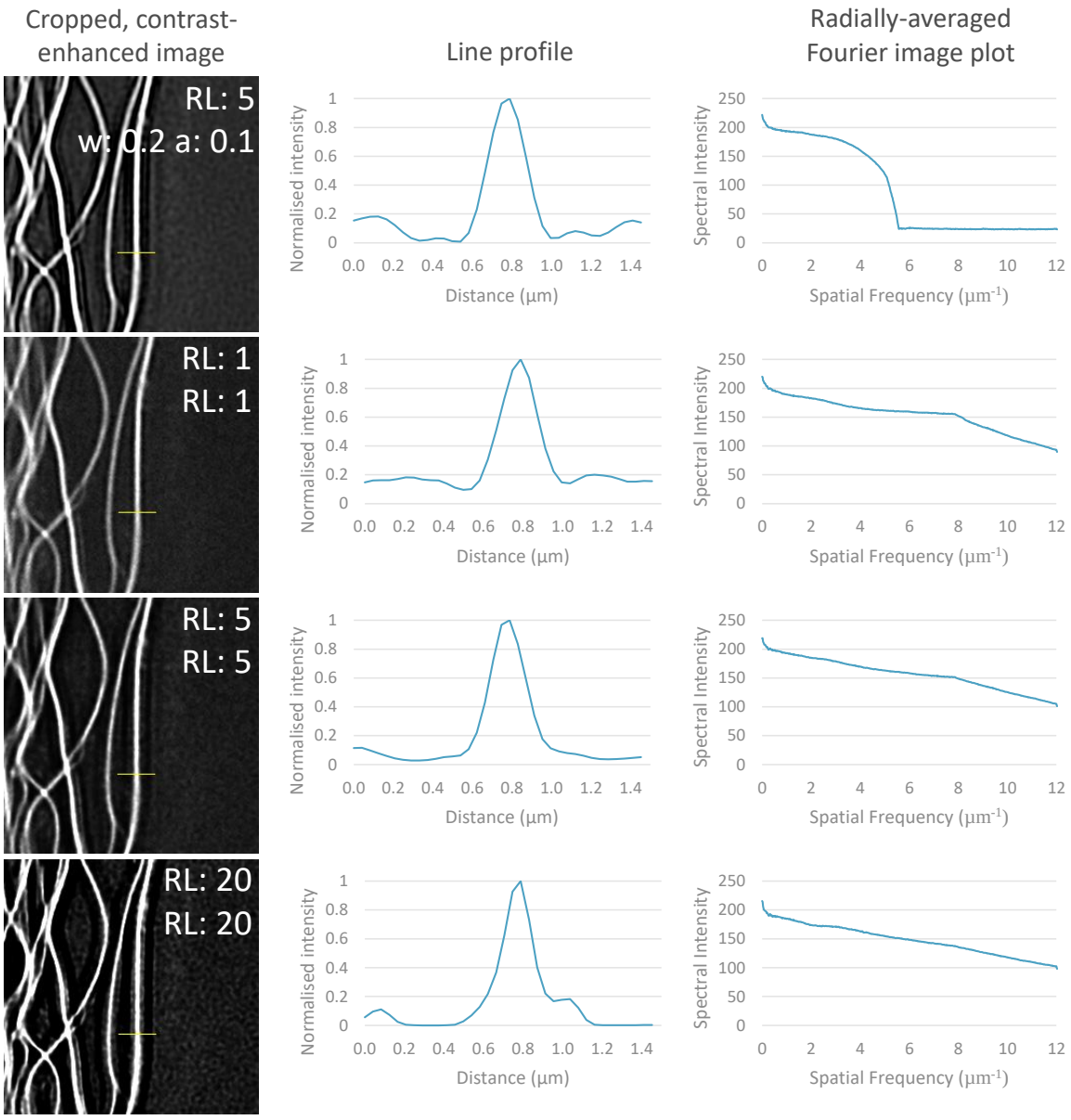



**Figure 2.12:** Richardson-Lucy deconvolution on the raw data reduces noise, leading to fewer artefacts in the Weiner-filter reconstruction. Furthermore using Richardson-Lucy filtering for reconstruction produces a more conventionally shaped OTF than Wiener filtering, which further reduces the number of reconstruction artefacts.

In the Laser Analytics Group's SIM setup, the illumination pattern is only projected onto the sample plane; therefore out of focus light does not show the SIM pattern. Intuitively, one would think this out-of-focus light can be removed by rejecting any part of the image that does not have the SIM pattern, and only showing the in-focus parts of the image with SIM pattern on them. O'Holleran and Shaw show that this is achieved practically by attenuating certain parts of the OTF during reconstruction. [cite]

If the center part of the OTF is computationally removed, out-of-focus light will be removed from the reconstructed image. In conventional widefield microscopy, this would have the side-effect of also removing low-frequency information from the image. In structured illumination microscopy, however, this information can be recovered from shifted components in Fourier space.

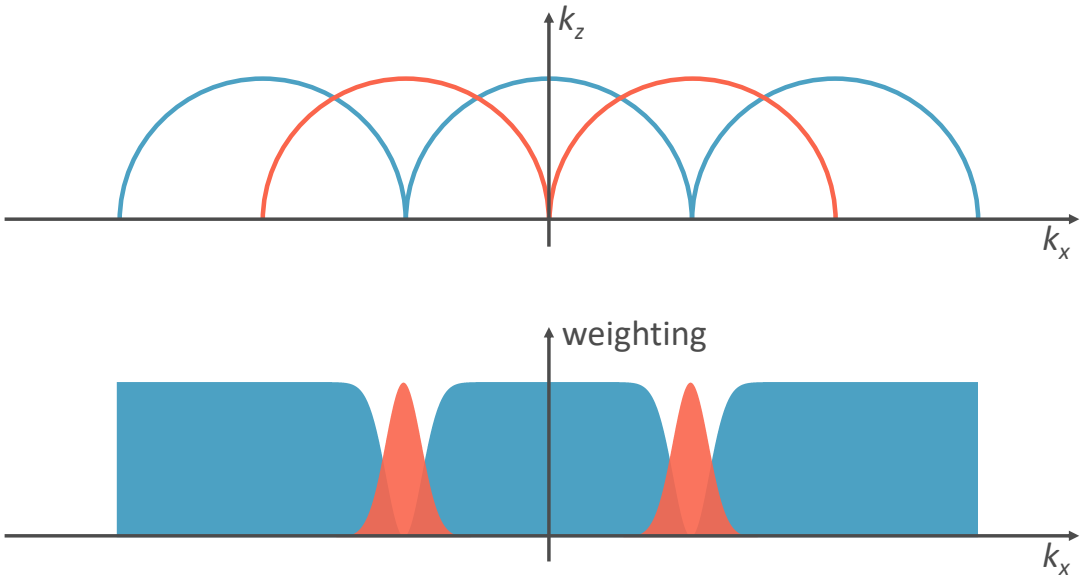



**Figure 2.13:** By suppressing the OTF in areas where it does not have axial support, out-of-focus light is removed. The in-focus light at these spatial frequencies can still be reconstructed as a result of the shifted OTF from SIM reconstruction. Adapted from [6].

Figure 2.13, reproduced from [cite?], shows which parts of the OTF should be attenuated to achieve optical sectioning with no loss of resolution. The optical sectioning is controlled by a Gaussian notch in the first-order passbands, and a complementary Gaussian in the zero-order passband. The width and depth of this passband is controlled in LAG SIM with the parameters 'Attenuation FWHM' and 'Attenuation strength' respectively.

Note that OTF attenuation cannot be used with the Richardson-Lucy output filter.

The water lens is set up for optical sectioning.

Out-of-focus light makes creating 3D reconstructions impossible, as shown in Figure ?. Using optical sectioning to remove this light allows 3D reconstructions of the sample to be created, which can reveal information otherwise unobservable, such as whether a particle is located on the inside or outside of a cell membrane.

## 2.6 Results and Discussion: SIM showcase

LAG SIM has been designed to be a versatile and user-friendly microscope, providing high-speed multi-colour imaging surpassing the diffraction limit with optical sectioning provided computationally or with TIRF. With a number of useful automations, LAG SIM can be used to create timelapse videos, 3D reconstructions, large mosaic images, and for unsupervised imaging of multiple cells at different locations around the coverslip. As a result, the microscope can and has found applications in a wide variety of biological investigations.

This section presents a small selection of experiments utilising a range of LAG SIM's capabilities. Acquisition and reconstruction methods are provided as a resource for researchers interested in conduction similar experiments with the microscope.

### 2.6.1 Bead alignment slide

For any multicolour SIM experiments, a slide of multicolour subdiffraction beads can be used for measuring and correcting chromatic offset, and allows aberrations to be minimised.

If an appropriate slide is not available, one can be produced as follows:

1. Sonicate 0.1 µm beads for 10 min
2. Dilute beads 1:10 in distilled water, producing a suspension of density [?? ]
3. Dispense 20 µl of the diluted beads onto a coverglass, or 50 µl into an 8-well Labtek
4. Wait for the solution to dry, about 2 h
5. Add mounting medium: immersion oil for use with the 100× oil lens, or water for use with the 60× water lens.
6. Mount the coverglass on a glass slide, or replace the lid on the Labtek and seal with Parafilm to prevent spillages

When the beads slide is viewed on the microscope, aberrations can be minimised by adjusting the correction collar on the objective lens and observing the point-spread function of individual beads. For minimal aberrations, the point-spread function should be symmetrical when defocussing above and below the bead layer.

To measure chromatic offset, a mutli-colour acquisition should be performed with the same settings as the subsequent biological experiment. Reconstruction in LAG SIM for Fiji

should then be performed with the following suggested parameters:

| | |
|---|---|
| **Filter** | RL in, RL out |
| **RL steps** | 5 |
| **OTF attenuation** | false |

After reconstruction, there will be some chromatic offset between each channel, as shown in Figure **??**. After registration and correction, an RGB colour map should produce white beads.

### 2.6.2 Resolution-enhanced optical sectioning of ?? cells

A frequent biological application of microscopy is imaging cultured cells under different treatments or with different genetic modifications. In this particular application, which is described in greater in Chapter 4, we treated the cells with a metal organic framework (MOF) as a drug delivery system. The goal was to observe uptake of the drug into live cells over a period of 24 h.

The MOF was naturally fluorescent in the 488 nm excitation channel. Endosomes were labelled with [?? early endosome stain], and the nucleus was labelled with [?? si dna stain] to visualise the cell.

To ensure MOF had entered the cell, rather than sitting on the cell membrane, it was necessary to view the cells in 3D. The 60× water lens was selected, and operated optical sectioning SIM mode. A z-stack of 3-colour SIM images was captured using the filter wheel, with a z-step of 0.2 µm between each plane. Since fast dynamics were not observed, a 200 ms exposure time was used per raw frame, to ensure a high signal-to-noise ratio.

The z-stack was reconstructed by utilising OTF attenuation for optical sectioning. The following parameters were used for the LAG SIM Fiji plugin:

| | | | |
|---|---|---|---|
| **Filter** | RL in, Wiener out | **RL steps** | 5 |
| **Wiener parameter** | 0.05 | **OTF attenuation** | true |
| **Apodiation cutoff** | 1.7 | **Attenuation FWHM** | 1.2 |
| **Apodiation strength** | 0.8 | **Attenuation strength** | 0.995 |

Images were captured at time points of 30 min, 1 h, 2 h, 6 h, 8 h and 24 h after application of the MOF, to observe that the cells were indeed able to take up MOF. Although an Okolab incubation stage was used to maintain a 5 % $CO_2$, 37 °C environment for the cells, they did not survive on the microscope stage over the full 24 h period, so were returned to the lab

incubator between imaging; therefore, different cells were imaged at each time point. A 3D reconstruction is shown in Figure 2.15.

### 2.6.3 Fast, multicolour imaging of dynamics in ?Meng's cells

During experiments on the dynamics of endoplasmic reticulum (ER), which are detailed in Chapter 5, we observed a strong interaction between ER and other organelles, including lysosomes and tubulin. To develop a deeper understanding of these relationships, we required fast, high-resolution imaging of live cells in multiple colour channels simultaneously.

For high resolution, the 100× oil lens was chosen, although, because we were looking deeper than 100 nm into the cell, it was not operated in TIRF mode. To facilitate simultaneous multi-colour imaging, the Optosplit was employed. Cross-talk between colour channels was minimised by labelling just two organelles per experiment, in the 488 nm and 640 nm excitation channels.

A time-series of SIM images was acquired at a raw exposure time of SI10 ms per frame. These were reconstructed with the following parameters:

| | |
|---|---|
| **Filter** | RL in, RL out |
| **RL steps** | 5 |
| **OTF attenuation** | false |

Work on these observations is ongoing, with a variety of genetic mutations now being applied to cell lines to observe differences in organelle interaction.

### 2.6.4 Timelapse TIRF imaging of ?Anchal's cell membrane

Structure on the cell membrane is difficult to observe in epi-fluorescent microscopy due to out-of-focus light obscuring details on the membrane. LAG SIM's capability to perform TIRF was therefore vital for imaging actin dynamics on the membrane on ??(COS-7?) cells.

In this experiment, ring structures in the actin were found to colocalise with ??protein. By performing time-lapse imaging, and taking a short video every 10 min, we were able to observe that these ring structures produce projections of actin which spread through the cell. The bookmarking feature of LAG SIM was used to record the location of a cells cells distributed across the cover glass, so that many cells could be captured in one imaging session.

The 100× oil lens was chosen for its TIRF capabilities. 2-colour TIRF-SIM imaging was performed sequentially at a frame rate of ??. The following parameters were used for

reconstruction:

| | |
|---|---|
| **Filter** | RL in, RL out |
| **RL steps** | 5 |
| **OTF attenuation** | false |

The concentration of fluorescent molecules was low to ensure the experiment was as physiologically relevant as possible, so Hessian de-noising was performed in MATLAB after the reconstruction.

Further research into these observations is ongoing.

### 2.6.5 Resolution-enhanced optical sectioning for tissue imaging

The versatility of LAG SIM also allows it to perform tissue imaging beyond the diffraction limit. 8 µm-thin slices of mouse brain were labelled to study the myelin sheath of oligodendrocytes. Myelin was labelled in one of two colours, depending on whether the origin of the cell was from the [front or back] of the brain. The end of the each myelin sheath was labelled with a 3$^{rd}$ colour to measure the node of Raniver (that is, the gap between two myelin sheaths), to discover whether there is an observable difference between the two oligodendrocyte types related to disease pathology.

To allow the greatest imaging depth, the lens with the closest refractive index matched to the tissue was chosen; in this case, the 60× water lens. A greater penetration depth could be achieved with a lens better matched to the tissue's refractive index, for example a silicon-oil immersion lens with a refractive index of ???1.4???, if one is available. A z-stack was captured with a step-size of 0.2 µm, to build up a 3D image. Cells were fixed, so a raw frame exposure time of [???] was used to ensure a high signal-to-noise ratio.

To reconstruct the slices, the following parameters were used in the LAG SIM Fiji plugin:

| | | | |
|---|---|---|---|
| **Filter** | RL in, Wiener out | **RL steps** | 5 |
| **Wiener parameter** | 0.05 | **OTF attenuation** | true |
| **Apodiation cutoff** | 1.7 | **Attenuation FWHM** | 1.2 |
| **Apodiation strength** | 0.8 | **Attenuation strength** | 0.995 |

Furthermore, the bookmarking function of LAG SIM was used to create a mosaic pattern, allowing reconstruction of a full brain slice in a single tiled image, as shown in Figure 2.7. Although in the end no statistical difference was calculated between the two cell types, this experiment demonstrated LAG SIM's ability to image tissue samples.

## 2.7 Conclusion

LAG SIM has been restored to its fast imaging speed of 11 super-resolution frames per second through the introduction of a Pockels cell for polarisation rotation. The Pockels cell is a robust, future-proof solution to ensure that the SIM illumination is generated with high pattern contrast. With this successfully in place, the other desirable features of LAG SIM could be addressed.

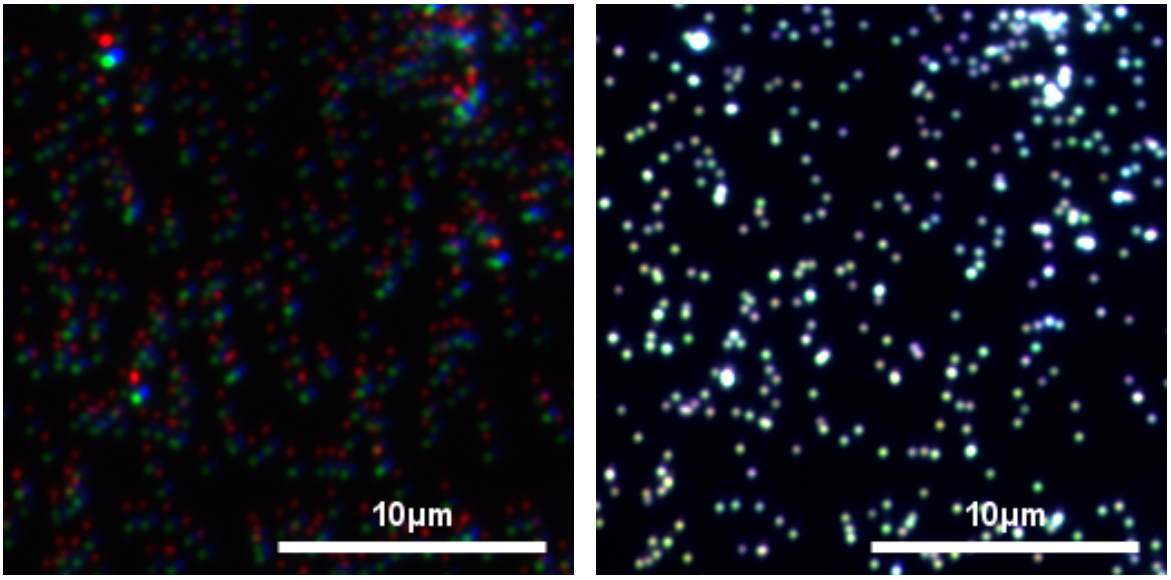
The instrument can now quickly switch between TIRF imaging, resolution-enhanced SIM imaging and SIM optimised for optical sectioning through a combination of well-designed software and straightforward hardware tweaks. This makes it suited to a wide variety of biological imaging experiments, as presented in the SIM showcase.

The addition of an Optosplit, made possible by the implementation of the Pockels cell rather than an LCVR, further enhances LAG SIM's fast imaging capability. Rather than requiring the use of a filter wheel to switch between colour channels, 3 colours can be imaged simultaneously. This removes any mechanical movement from the system, reducing the fastest 3-channel acquisition from 470 ms to the headline 90 ms, a five-fold improvement in frame rate.

Considerable care has been taken to ensure the microscope can be used by non-expert users. The interface is clear and compact, and can be taught to new users in a single imaging session. This means that LAG SIM can be used unsupervised, saving time and allowing for the microscope to perform a diverse range of world-leading biological research.

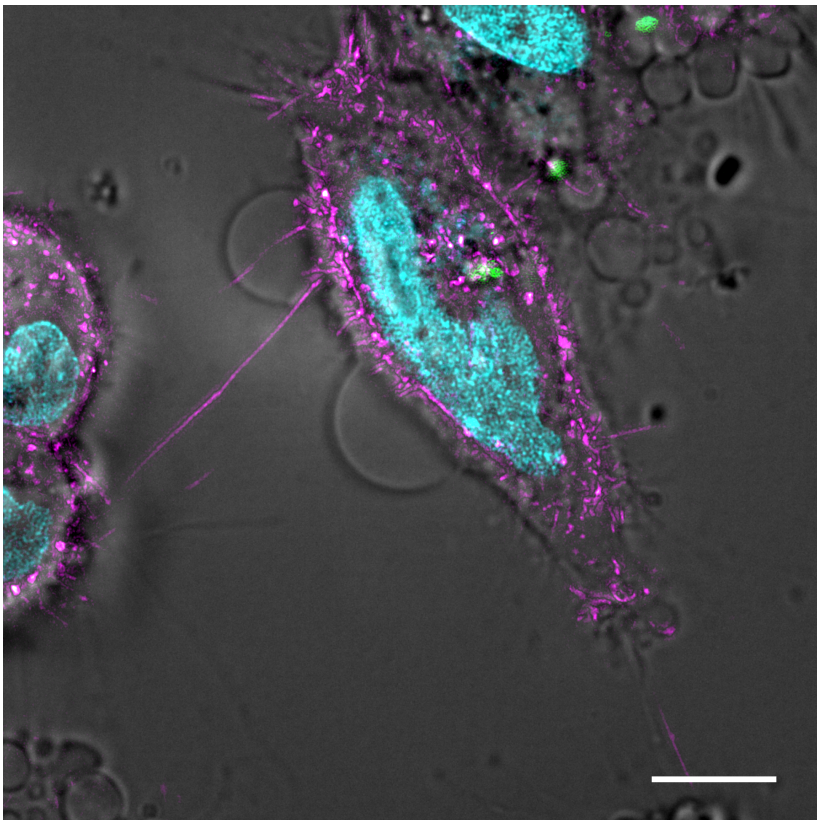
Finally, the LAG SIM Fiji plugin complements the easy-to-use microscope to help users generate artefact-free images. As shown throughout this chapter, reconstruction is a complicated process involving complex mathematics and optical engineering. LAG SIM simplifies this process, so that a user can go from hypothesis to result without the need for a full-time SIM expert.

In addition to the SIM showcase presented here, Chapters 4 and 5 detail two successful experiments facilitated by LAG SIM. Before that, however, Chapter 3 reveals FPBioimage, a user-friendly tool for visualising and sharing 3D volumetric data, such as those obtained through optical sectioning on the LAG SIM.

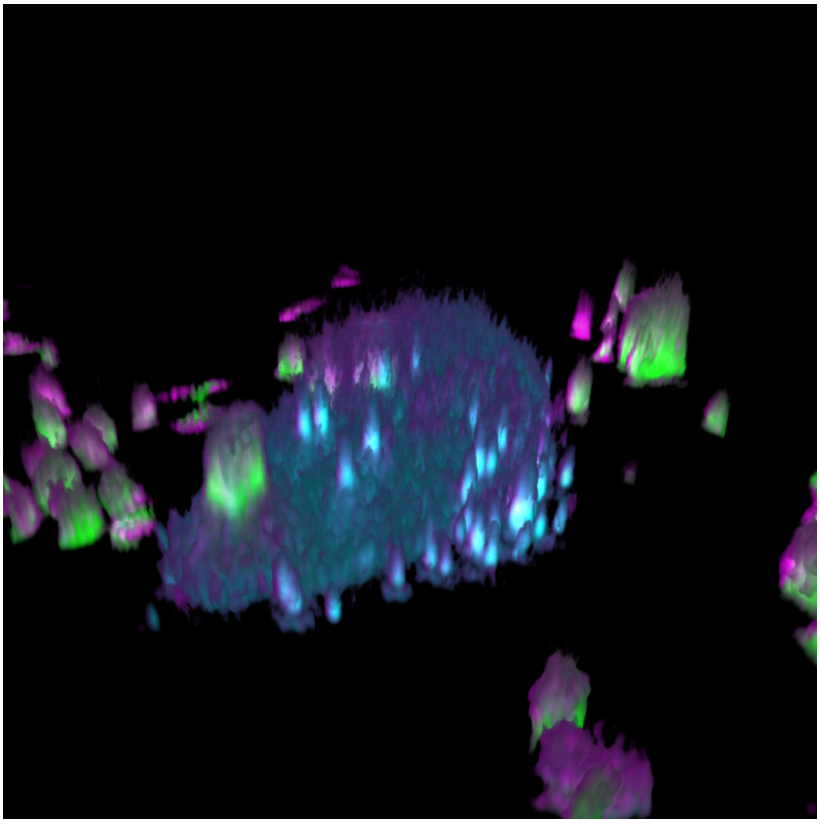



**(a)** **(b)**

**Figure 2.14:** A small chromatic offset between the three SIM colour channels can be seen in (a). When the offset is corrected, as in (b), the beads appear white; the offset parameters can then be used to correct other multi-colour images captured during the same imaging session.



**(a)**



**(b)**

**Figure 2.15:** 3 colour MOF in cell in 3D. Scalebar in (a) is 20 µm

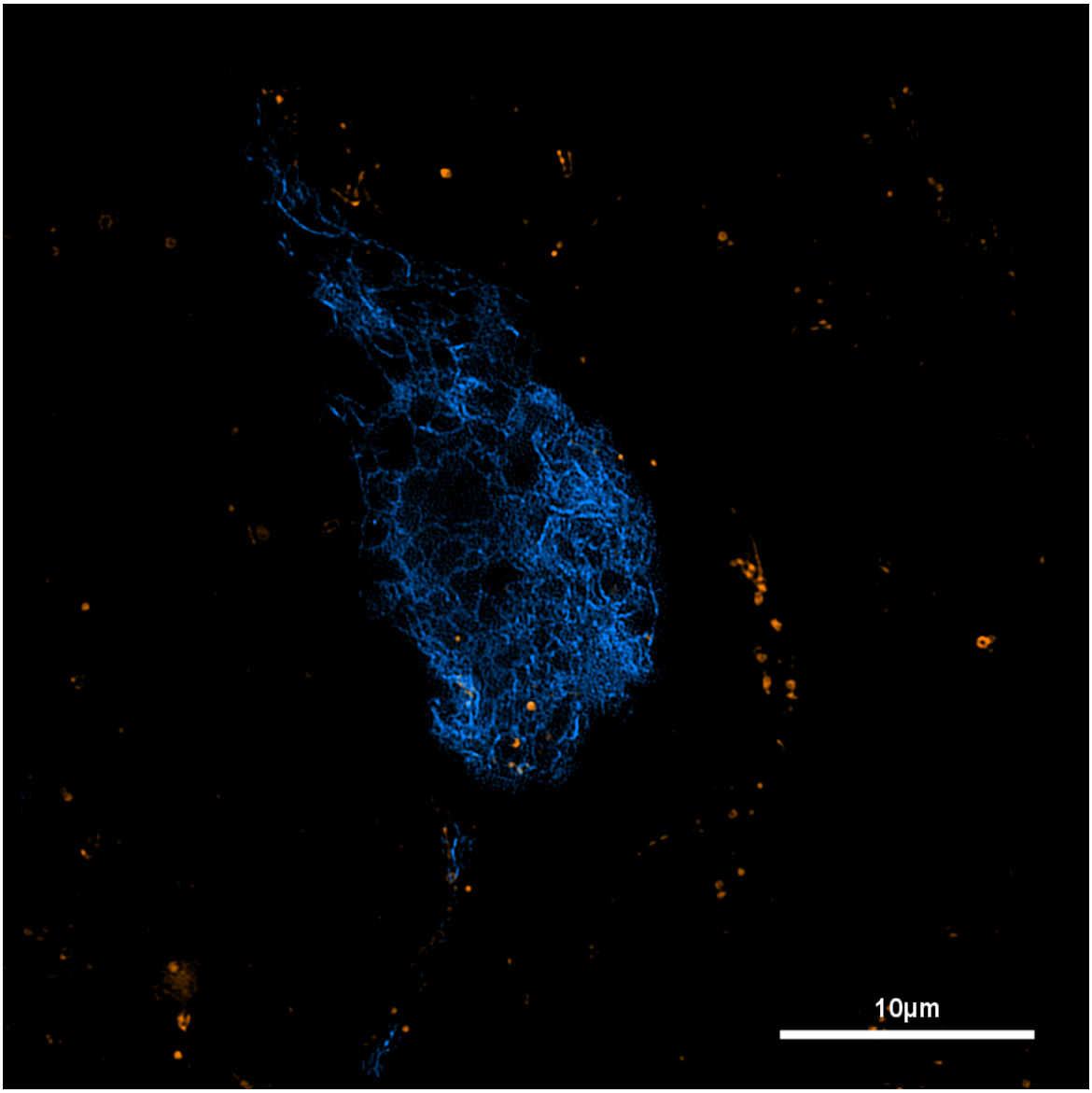
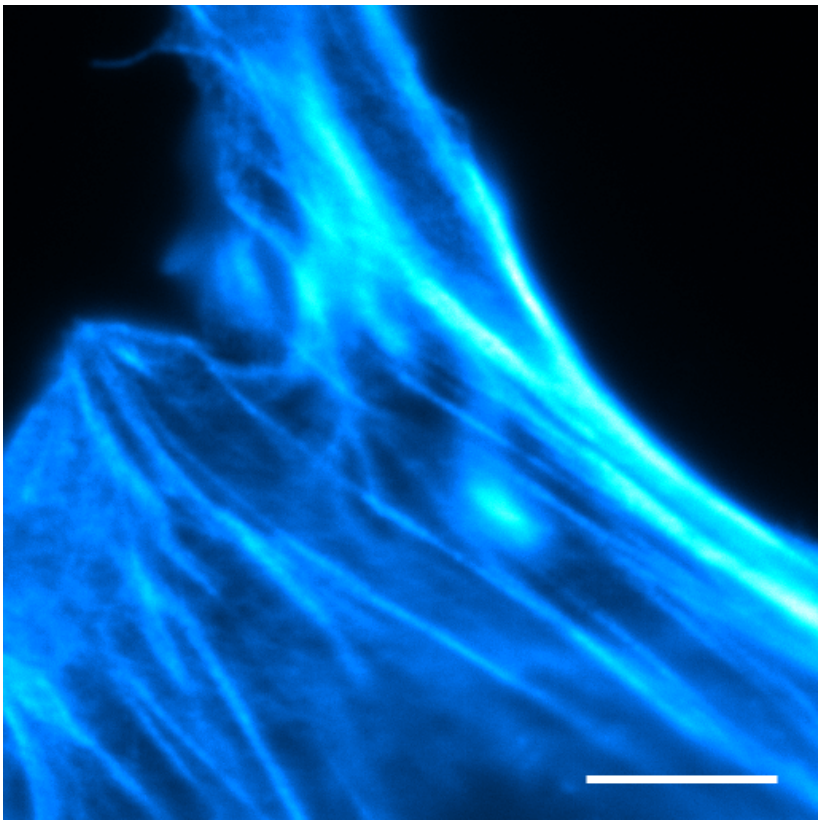
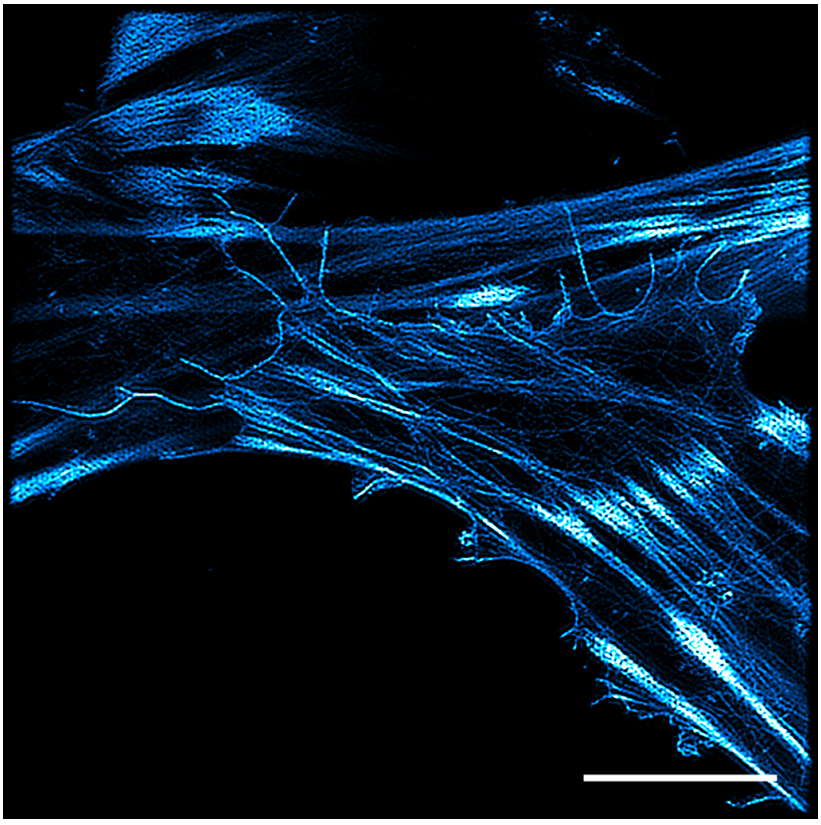



**Figure 2.16:** HEK293 cells with ER is coloured in blue, and lysosomes are coloured in orange. Capturing a timelapse utilising the Optosplit and optical sectioning SIM reveals a strong co-localisation between lysosomes and ER tubules.



**(a)**



**(b)**

**Figure 2.17:** (a) shows actin labelled in COS-7 cells, where out-of-focus light blurs details in the actin. Utilising TIRF-SIM, shown in (b), enhances resolution and removes out-of-focus light. Scalebar is 10 µm.

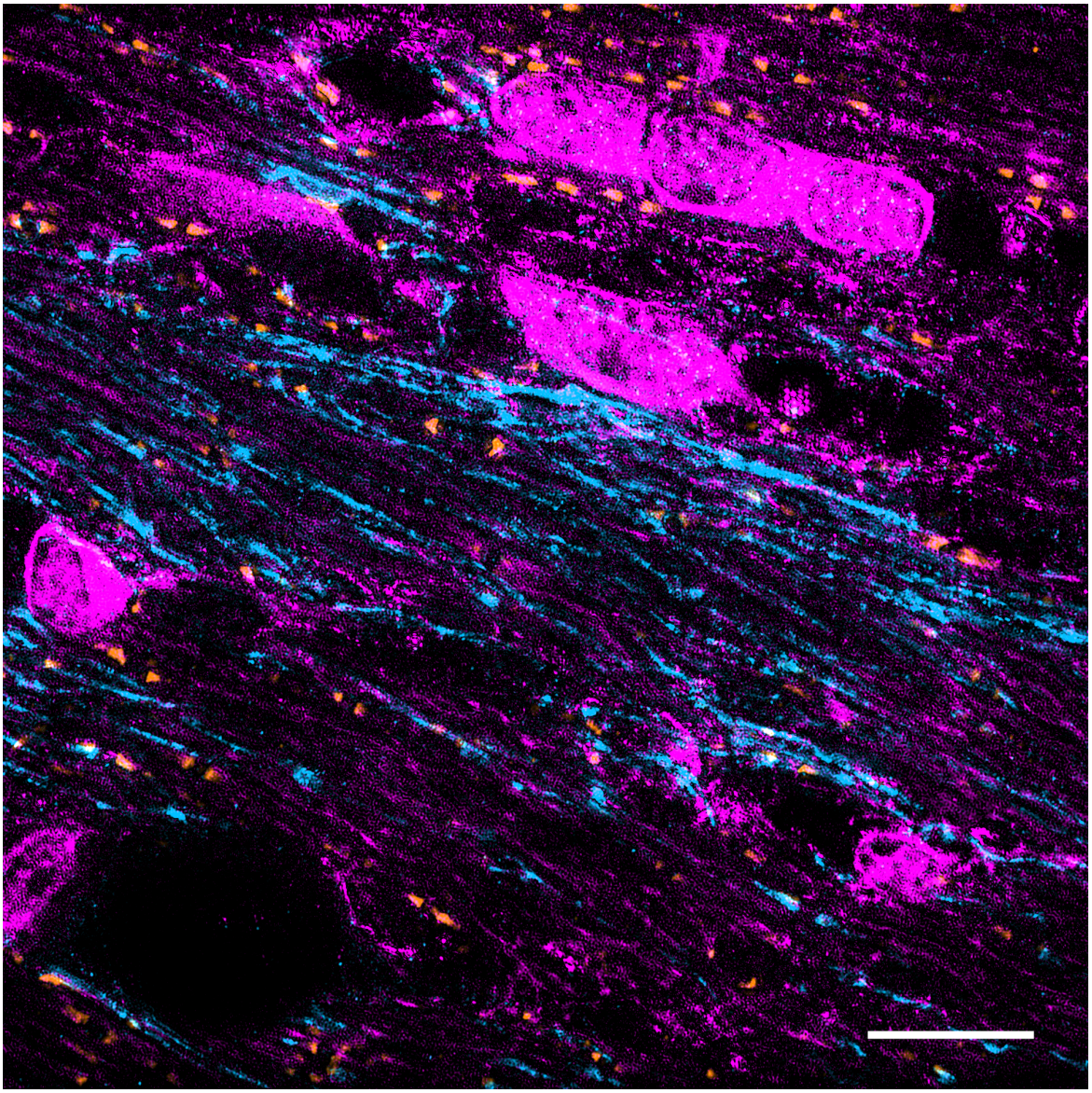


**Figure 2.18:** Oligodendrocytes from the dorsal and ventral region of the brain are coloured in magenta and cyan respectively. The end of the myelin sheath is coloured in orange, and can be used to measure the length of the node of Raniver. Optical sectioning SIM was used on 8 µm brain slices. Scalebar is 10 µm.

# Chapter 3

# FPBioimage: 3D visualisation on the web

"Marcus, hats off; your FPBioimage really rolls."

*Email from Konrad Kölble, 20-06-2018*

## 3.1 Introduction

### 3.1.1 Data Availability

Modern scientific experiments gather data at an ever-increasing rate. This is particularly apparent in 3D microscopy, where techniques such as confocal, optical-sectioning SIM, and light-sheet techniques including selective plane illumination microscopy (SPIM) generate terabytes of volumetric data per hour.

Popular scientific journals require authors to make data available for reproducibility and further analysis.

To give the reader a deeper understanding of 3D data, many authors present it with fly-through videos showing the data from a variety of perspectives, including from within the volume. Whilst this provides significantly more information than static 2D images can provide, the reader is nevertheless unable to interact with the data themselves. They are therefore restricted to just the perspective provided by the author, with no opportunity for personal exploration or analysis. To describe the data as open access should require a link to the volumetric data set so that readers can download it and examine it themselves.

It is perhaps understandable, however, that most published work does not contain links to full volumetric data sets; hosting and maintaining such large volumes of data online incurs prohibitively expensive costs. Repositories specifically designed for this type of data are emerging, such as the Image Data Resource (IDR) from OMERO, but have not been widely

adopted. Many authors therefore simply make a statement that their data is "available upon reasonable request".

In my experience, authors are very willing to accommodate requests for such data. However, receiving and examining 3D volumetric data is not trivial. The large file size means that the common practice is to temporarily upload data to a Dropbox file repository, and share a link to the repository through an email. Once the large data file has been downloaded, specialist image analysis software must be used to render it, requiring expert skill and experience.

As a producer of volumetric data, I faced the data sharing issue myself when one of my collaborators moved to Chicago. We had gathered 3D data utilising optical sectioning on the SIM, as described in Chapter **??**, which now required their specialist biological interpretation. The issue was further complicated since the collaborator had little knowledge of volume rendering software, so could not easily examine the data.

To solve this, I searched for an online application where I could easily upload 3D data to share immediately with my collaborator. Finding that none existed, I created First Person Bioimage (FPBioimage), my own tool to fulfil these requirements.

### 3.1.2 Volumetric rendering

The first demonstration of volumetric rendering was by Pixar in 1987 [**?** ], a year after their aquisition from Lucasfilms by Steve Jobs for $5 000 000 (a bargin considering it was sold to Disney in 2006 for $7 400 000 000 [**? ?** ]). For the first time, a full volumetric rendering of a computerised tomography (CT) scan was displayed.

At SIGGRAPH ’88 [**?** ] Alvy Ray Smith clarified the difference between between volumetric rendering and traditional, geometric rendering. In geometric rendering, an object is described by one or more geometrical shapes, such as a collection of spheres and cubes. To render the object, the surface of the shape is approximated by a mesh of triangular primitives, and the interaction of a virtual light ray with each triangle is calculated.

Volumetric rendering uses a fundamentally different technique. A virtual light ray begins at a pixel on the screen, and progresses through a 3D array of voxels, as shown in Figure 3.1. At equally spaced steps along the ray, the value of the voxel is recorded to build up a ray profile.

Volumetric rendering allows detail within the volume to be visualised. By tuning the opacity of each voxel, details which would be invisible in a surface rendering can be seen in context, through the outer voxels of the model.

The key drawback of volumetric rendering compared to geometric rendering is speed. In the simple volumetric renderer shown in Figure 3.1, a ray is calculated for each pixel on the

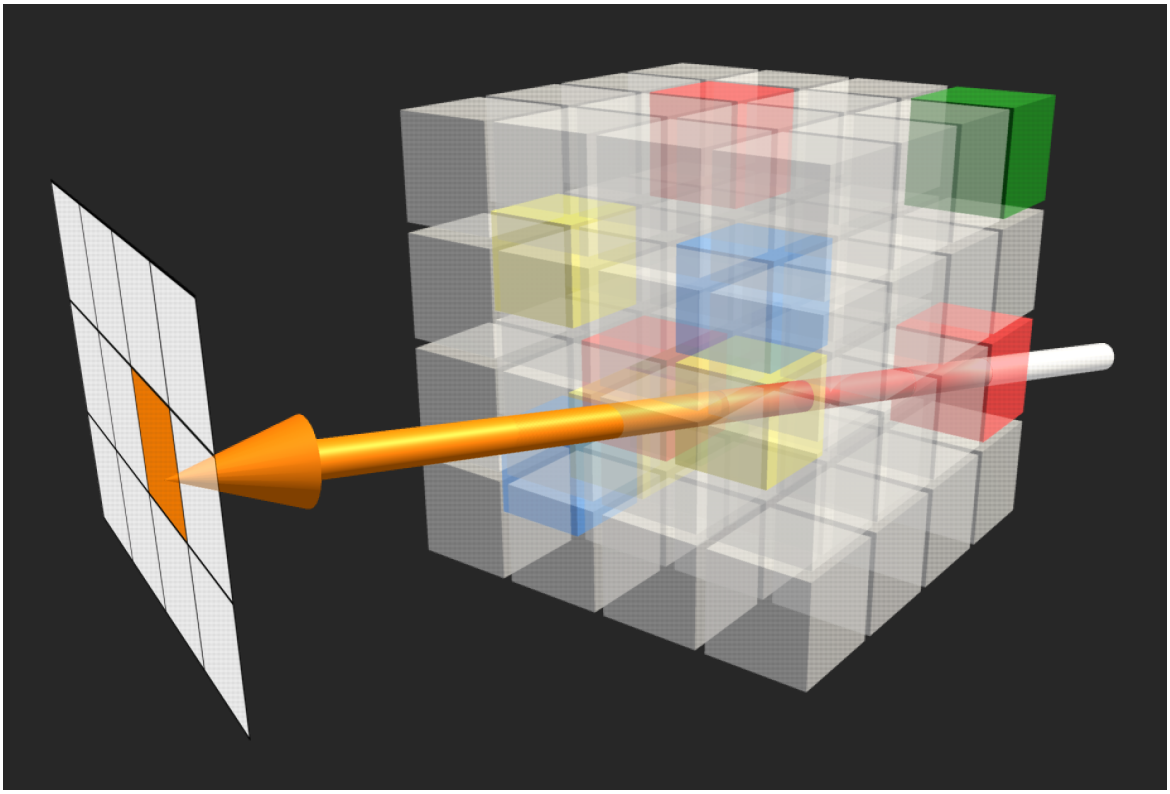


**Figure 3.1:** In a traditional ray marching volumetric renderer, a colourless ray starts at the back of the volume, and picks up colour and transparency at each voxel it passes through. After its journey through the volume, it paints its accumulated colour to the screen pixel it lands on.

screen - 2 073 600 pixels for a 1080p high definition display. If each ray takes 512 samples as it passes through the volume, each rendering of the volume requires 1 061 683 200.

When rendering a single volumetric image, or even a video which is first rendered then presented at a later time, this is not an issue: the rays can computed in parallel across a cluster of computers, and with some patience the images will build up. However, until recently displaying volumetric data at video rate in real time required specialised hardware optimised solely for volumetric rendering. [cite patent??]

The gaming industry has led to a rapid development of devices optimised for performing thousands of simple calculations in parallel. A single graphics processing unit (GPU) contains as many as ??? individual processing cores, which can be used to calculate volumetric rays simultaneously. Combined with intelligent optimisations to the rendering algorithm, GPUs found in devices ranging from laboratory workstations to smartphones now enable real-time volumetric visualisation on personal computers.

### 3.1.3 WebGL

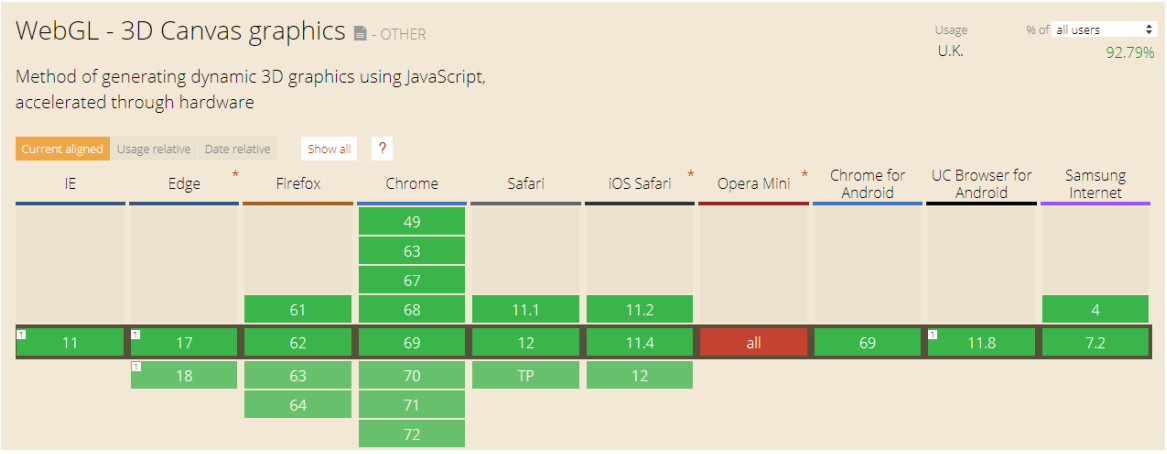
Since the development of highly parallel GPUs, a handful of free and commercial software has been released which provide volumetric rendering capabilities. Popular programs used in the microscope community include ImageJ, Icy, and Imaris; [something and something] are also used in the medical community for visualising patient data. These programs provide advanced offline volumetric rendering, and FPBioimage is not designed to replace them; instead, the key feature of FPBioimage is providing rendering on the web, to facilitate easy and interactive sharing of 3D data.

Creating this tool has only been possible because of the relatively recent invention and widespread adoption of the 'Web Graphics Library' (WebGL), which provides access to a GPU's processing power directly from the web browser. This means that three-dimensional effects on webpages can be rendered to the screen in real time.

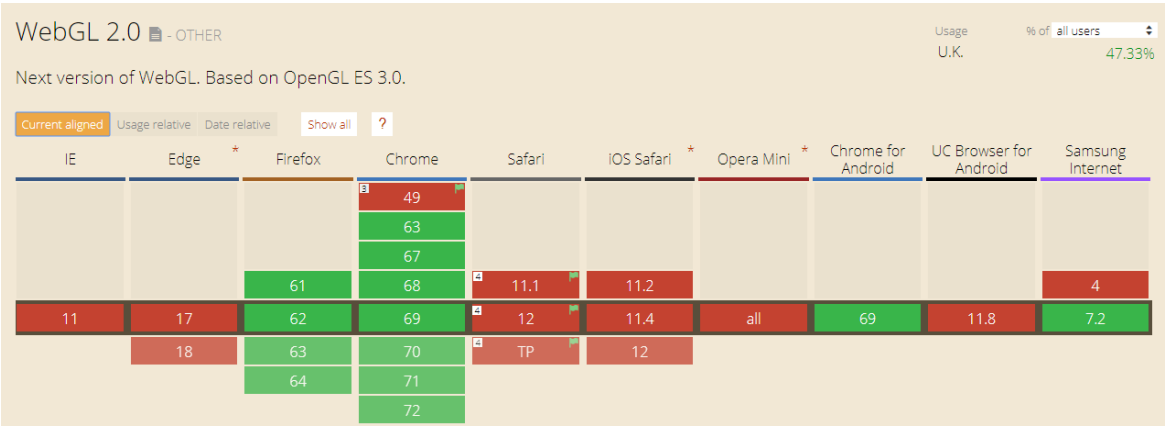
In technical terms, WebGL exposes the `OpenGL ES 2.0` feature set [**?** ]. This is a shader-based application programming interface (API). The API provides a limited number of simple functions which are optimised for running on GPUs in parallel.

The support of WebGL is widespread, [cite canIuse, Figure 3.2, available through HTML5.

A more modern version, WebGL 2.0, exposes the `OpenGL ES 3.0` API, which makes programming for 3D much easier. However, at the time FPBioimage was published, support for WebGL 2.0 was highly limited. Even at the time of writing, WebGL 2.0 is only supported by third party browsers, and cannot be used in Microsoft's Edge or Apple's Safari, limiting it to just 47.33 % of UK web users.




**(a)**




**(b)**

**Figure 3.2:** Screenshots taken from https://caniuse.com[7]. (a) shows that FPBioimage, which is percentage built with WebGL, is supported by the web browsers of 92.79% of UK users. Although WebGL 2.0 makes 3D programming easier, (b) shows that support would drop to 47.33% of users.

To maintain maximum compatibility, volumetric rendering in FPBioimage is implemented entirely in WebGL 1.0. Various strategies for overcoming this restriction are detailed in Section 3.2.1.

### 3.1.4 Aims for FPBioimage

The primary goal for FPBioimage was to create an easy-to-use web application for visualising volumetric data in a web browser. The tool was required to run without requiring any other plugins or downloads, in order that viewing data was a simple process.

After gathering user feedback, a list of desired features was compiled.

1. Rendering options for more advanced users
2. Ability to cut the data open at arbitrary angles and positions (sometimes known as *reslicing*)
3. Offter high resolution screenshots of the data
4. Ability to save a particular view as a bookmark, and share that exact view with others

For researchers sharing their own generated 3D data, more easy-to-use tools were proposed. These had to be compatible with popular image analysis programs, and facilitate fast sharing of volumetric data.

From the first prototype FPBioimage was designed to be user-friendly. The program had to be intuitive so that a user could visualise their data without any training.

## 3.2 Method: software design

This section describes, through the use of diagrams and code snippets, the method by which the volumetric visualisation software was developed.

The primary requirement of FPBioimage was that it must run in a web browser. As mentioned in Section 3.1.1, offline visualisation tools are already available, so the distinguishable feature of this software was the ability to share and publish volumetric data online. For fast development and compatibility with multiple devices, I chose to develop the software in a game engine called Unity [cite]. This gives access to an extensive library of functions for creating dynamic 3D scenes, as well as a language for executing code on the graphics card. When the software is complete, Unity allows the designer to compile the 'game' into compressed HTML/Javascript code to run in a web browser.

### 3.2.1 Fragment shader

**Snippet 3.1:** Fragment shader code for volumetric ray marching

```
1 // Fragment shader
2 float4 frag(frag_input i) : COLOR
3 {
4   // Calculate eye ray (i) intersection with cube bounding box
5   float3 boxMin = { -0.5, -0.5, -0.5 };
6   float3 boxMax = {  0.5,  0.5,  0.5 };
7   float tNear, tFar;
8   bool hit = IntersectBox(i.ray_o, i.ray_d, boxMin, boxMax, tNear, tFar)
      ;
```

```
9   if (!hit) discard;
10  if (tNear < 0.0) tNear = 0.0;
11
12  // Calculate intersection points
13  float3 pNear = i.ray_o + i.ray_d*tNear;
14  float3 pFar  = i.ray_o + i.ray_d*tFar;
15  // convert to texture space
16  pNear = pNear + 0.5;
17  pFar  = pFar  + 0.5;
18
19  // March along ray inside the cube, accumulating color
20  float3 ray_pos = pNear;
21  float3 ray_dir = pFar - pNear;
22  float3 ray_step = normalize(ray_dir) / _Steps;
23
24  float normalised_opacity = _Opacity * length(ray_step);
25  float4 ray_col = 0;
26  float mean_max_voxel = 0;
27
28  for(int k = 0; k < MAX_STEPS; k++){
29    if (k<_Steps){
30      // Get current position
31      ray_pos = pNear + k * ray_step;
32
33      // Clip if behind clip plane or outside unit cube
34      bool clipVoxel = dot(_ClipPlane, float4(ray_pos-0.5f, 1.0f)) > 0.0
            f
35       || !(ray_pos.x > 0.0f && ray_pos.y > 0.0f && ray_pos.z > 0.0f
36        && ray_pos.x < 1.0f && ray_pos.y < 1.0f && ray_pos.z < 1.0f);
37
38      if (!clipVoxel && ray_col.a < 1.0f){
39          // Get colour of the voxel we're passing through
40          float4 voxel_col = sample2D(ray_pos);
41          float mean_col = (voxel_col.r + voxel_col.g + voxel_col.b);
42
43          if (_RenderMode == 0 && mean_col > _Threshold){
44            // Max Intensity Projection
45            if (mean_col > mean_max_voxel){
46              ray_col = voxel_col;
47              mean_max_voxel = (voxel_col.r + voxel_col.g + voxel_col.b)
                    ;
48            }
49
50          } else if (_RenderMode == 1 && mean_col > _Threshold){
```

```
51            // Volumetric ray marching with transparency
52            voxel_col.a *= normalised_opacity;
53            voxel_col.rgb *= voxel_col.a;
54            ray_col += (1.0f-ray_col.a) * voxel_col;
55
56          }
57        }
58      }
59    }
60    ray_col.rgb *= _Intensity;
61      return ray_col;
62  }
```

FPBioimage uses a ray marching technique similar to that presented in Section 3.1.2. The main ray marching loop implementing this is shown in lines Snippet 3.1. This loop is part of the 'fragment shader', which means it runs for every pixel rendered on the display. It is therefore the most important loop in the whole program for efficiency, and any efforts to increase the speed at which this loop executes will be amplified by the number of pixels. We will now examine the loop, and the efficiency modifications which have been made compared to traditional ray marching.

We should first start at lines 40 and 47, and note that there are two render modes: a maximum intensity projection (`_RenderMode 0`) and the volumetric ray marching mode (`_RenderMode 1`) discussed in section 3.1.2. In the maximum intensity projection mode, the ray checks the mean colour of every voxel it passes through, and paints the colour of the voxel with the largest 'mean color' to the screen pixel. Note that mean colour is calculated as the sum of the red, green, and blue colour channels. Strictly we should divide by 3 to calculate the true mean colour value; however, this would mean 2 extra divisions for every loop iteration for every pixel on the screen; removing this step has no effect on the comparison, but gives a noticeable improvement in frames rendered per second.

Before discussing the volumetric ray marching calculation, we should note that in this implementation the ray marches from the screen to the back of the cube - this is the opposite of traditional scheme shown in Figure **??**. The reason for this is, of course, efficiency. As a ray progresses from a pixel through the cube, it becomes more and more opaque, denoted by the alpha value of the colour `ray_col.a`. Once the alpha value reaches 1, any voxels further along the ray's path will not be visible, so there is no reason to look up their colour or add their contribution to the ray. Line 38's second condition is that `ray_col.a < 1.0f` to implement this action. Note that in most programming languages we would break out of the loop at this point, but WebGL 1.0 loops must have a constant number of iterations.

The requirement to have a constant number of iterations also explains the very first condition in the loop on line 29. The variable `k_Steps` sets the depth resolution of the volume. We might expect the `for` loop to finish when `k < k_Steps`, but this would set the depth resolution as a constant, so that there was no option to increase it for powerful machines or decrease it for mobile devices. By looping for `MAX_STEPS` iterations (which is currently set to 512, but could be increased as hardware improves in the future) but performing no calculations if the loop variables `k` is greater than `k_STEPS`, we can still set the depth resolution to integer from 0 to `MAX_STEPS` and gain a performance benefit for low resolutions.

There are two additional conditions which must be met for the volume renderer to add a voxel to the ray. Firstly, shown in line 38, the ray position must be within the unit cube and behind the clip plane. The clip plane can be used to cut open the volumetric model at any angle and position, and is described in more detail in Section **??**. Secondly, the voxel's mean colour intensity must fall about the value of `_Threshold` to contribute to the ray, ensuring that dark voxels around the volume are not visible.

If all the aforementioned conditions are met, then the code in lines 49 - 51 performs the volumetric ray marching step - that is, the colour of the current voxel mixes onto the ray that is passing through it. Firstly, the voxel's opacity is adjusted based on the input variable `_Opacity`, which allows the volume to be rendered with transparency. This new value of alpha is multiplied onto the voxel's colour so that more transparent voxels contribute less intensity to the ray. Finally, the adjusted voxel is added to the ray, with its contribution weighted by the current transparency of the ray.

### 3.2.2 Loading 3D data in WebGL

To render volumetric images using the graphics card, the volumetric data must be loaded onto the graphics card in an appropriate form. WebGL 2.0 provides a `sample3D` class, which is able to store 3D images and return a voxel colour at a requested 3D coordinate. Unfortunately WebGL 1.0 does not have this functionality, and so an alternative scheme based on 2D textures has been devised.

The simplest way to picture volumetric imaging data is as a z-stack of 2D image slices. One option to obtain a voxel colour at a given *xyz*-coordinate would be to store all 2D image slices, and look up the *xy* value of the appropriate slice based on the *z*-coordinate. This is conceptually simple, but inefficient to implement on a graphics card, as each slice must be sent to the graphics card individually, which is not only slow but adds a significant memory overhead to the graphics card loading process. [cite texture atlases?] Instead, I devised a scheme where slices are stored on a large 2D texture atlas in a grid pattern. Then, a

*z*-coordinate corresponds to a specific slice on the grid, and the *xy* components can be added to the bottom-left coordinates of that slice to get the *xyz*-coordinate voxel colour.

A restriction of WebGL 1.0 is a maximum texutre size of 4096×4096. Using the atlas technique, this would support a maximum voxel resolution of 256×256×256. This was found to be inadequate for most researcher's data, so a target of 512×512×512 was set. In order to meet the requirements of WebGL 1.0, this requires 8 2D texture atlases of size 4096×4096.

The 8 texture atlases are filled according to the pseudo-code shown in Snippet 3.2. The code uses a mixture of the modulo operator (%) and division () to determine (a) which atlas to place each volume image slice in, and (b) the location in that image where the slice should be located. Complementary code runs in the WebGL rendering code to read a particular voxel's colour given an *xyz*-coordinate.

**Snippet 3.2:** Pseudo-java code for 3D texture atlas arrangement

```
int sliceWidth = volumeImage.width;
int sliceHeight = volumeImage.height;
int numSlices = volumeImage.depth; // Number of z-slices

int atlasWidth; int atlasHeight;
int numberOfAtlases = 8;

// Pad image size up to next power of 2
int paddedSliceWidth = ceil2(sliceWidth);
int paddedSliceHeight = ceil2(sliceHeight);

int xOffset = (int)Math.floor((paddedSliceWidth - sliceWidth)/2);
int yOffset = (int)Math.floor((paddedSliceHeight - sliceHeight)/2);

// Calculate the xy size of the atlases, making atlases as square as
    possible
int slicesPerAtlas = (int)Math.ceil((float)numSlices/(float)
    numberOfAtlases);
atlasWidth = ceil2(paddedSliceWidth);
atlasHeight = ceil2(paddedSliceHeight * slicesPerAtlas);
while((atlasHeight > 2*atlasWidth) && (atlasHeight > sliceHeight)) {
  atlasHeight /= 2;
  atlasWidth *= 2;
}

// Initialise an array of 8 black 2D textures to become atlases
Image[] atlasArray = new Image[numberOfAtlases];
```

```
26 for (int i=0; i<numberOfAtlases; i++){
27   atlasArray[i] = new Image(atlasWidth, atlasHeight, black);
28 }
29
30 // Fill out atlases with image slices
31 int slicesPerRow = (int)Math.floor((float)atlasWidth/(float)
       paddedSliceWidth);
32 for (int i=0; i<numSlices; i++){
33   int atlasNumber = (int)((float)i % (float)numberOfAtlases);
34   int locationIndex = (int)Math.floor((float)i/(float)numberOfAtlases);
35
36   // Get slice
37   Image slice = volumeImage.getSlice(i);
38
39   // Put slice into atlas at the correct position
40   int xStartPixel = (int)((float)locationIndex % (float)slicesPerRow) *
         paddedSliceWidth + xOffset;
41   int yStartPixel = (int)Math.floor((float)locationIndex / (float)
         slicesPerRow) * paddedSliceHeight + yOffset;
42
43   // The following line should be uncommented for coordinate systems
         that start top-left
44   //yStartPixel = atlasHeight - yStartPixel - paddedSliceHeight + 2*
         yOffset;
45
46   copySubImageToAtlas(sliceTexture, atlasArray[atlasNumber], xStartPixel
         , yStartPixel);
47 }
```

### 3.2.3 Movement and control

It would not make for a particularly exciting real-time volumetric renderer if the user only had one view of the data. Thanks to the Unity library functions, implementing movement is a straightforward and resource-efficient process.

Firstly, as with all offline volumetric rendering programs, the 3D model can be rotate about its $x$-, $y$- and $z$-axes to view a different part of the data. This is achieved in Unity by mapping the volume rendering shader described in Section 3.2.1 onto a cube. The cube is then linked to both the arrow keys and the mouse to provide intuitive manipulation of the volume, as shown in Snippet 3.3. For data whose combination of $xyz$ resolution and $xyz$ voxel size describes a cuboid, the $xyz$ scale of the rendering cube can be adjusted to ensure distortion-free visualisation.

**Snippet 3.3:** C# code using built-in Unity functions to rotate the rendered volumetric data.

```
void FixedUpdate () {
  // This code is attached to the rendering cube, and called once per
     frame

  // Keyboard arrow control
  Vector3 vertRotAxis = transform.InverseTransformDirection(camera.
     TransformDirection(Vector3.right)).normalized;

  float horizontalRot = Input.GetAxis("H2");
  float verticalRot = Input.GetAxis("V2");

  transform.Rotate(vertRotAxis, verticalRot, Space.Self);
  transform.Rotate (Vector3.up, horizontalRot, Space.World);

  // Mouse click control
  if(Input.GetMouseButton(0)){
    // Rotate with the mouse
    Vector3 vertRotAxis = transform.InverseTransformDirection(camera.
       TransformDirection(Vector3.right)).normalized;

    float horizontalRot = Input.GetAxis("Mouse X");
    float verticalRot = Input.GetAxis("Mouse Y");

    transform.Rotate(vertRotAxis, verticalRot, Space.Self);
    transform.Rotate (Vector3.up, horizontalRot, Space.World);
  }
}
```

The 'FP' in FPBioimage actually stands for 'First Person,' and refers to the control the user has over the scene's camera. Rather than just viewing the rotatable render cube from one perspective, as in common in offline volumetric renderers, FPBioimage allows the user to change the position and look-direction of the camera. This is implemented as shown in Snippet 3.4. The `H1` axis is mapped to the A and D keys, and the `Z1` axis to the W and S keys. This, combined with the functionality of using the mouse to change the camera's look-direction, will make control of the camera very familiar to anyone who has played first person perspective computer games. Moving the camera allows the user to create the type of dynamic fly-bys sometimes seen in publication videos in real time, providing a highly immersive visualisation of the data.

**Snippet 3.4:** C# code using built-in Unity functions for moving the camera in a first-person manner.

```
void FixedUpdate(){
  // Translation
  transform.position += Input.GetAxis ("V1") * Camera.main.transform.up
      * Time.deltaTime;
  transform.position += Input.GetAxis ("H1") * Camera.main.transform.
      right * Time.deltaTime;
  transform.position += Input.GetAxis ("Z1") * Camera.main.transform.
      forward * Time.deltaTime;
  transform.position += Input.GetAxis ("Mouse ScrollWheel") * Camera.
      main.transform.forward * Time.deltaTime;

  // Rotation
  if (!Input.GetMouseButton(1)) {
    // This is the 'first person' mouse mode
    rotationX += (Input.GetAxis ("Mouse X")) * mouseSpeed;
    rotationY += (Input.GetAxis ("Mouse Y")) * mouseSpeed;
    rotationY = ClampAngle (rotationY, minimumY, maximumY); // Stops
        camera doing backflips!

    Quaternion xQuaternion = Quaternion.AngleAxis (rotationX, Vector3.up
        );
    Quaternion yQuaternion = Quaternion.AngleAxis (rotationY, -Vector3.
        right);

    transform.localRotation = startRotation * xQuaternion * yQuaternion;
  }

  // Camera movement with mouse
  if (Input.GetMouseButton (1)) {
    transform.position -= Input.GetAxis("Mouse X") * Camera.main.
        transform.right * Time.deltaTime / Screen.width;
    transform.position -= Input.GetAxis("Mouse Y") * Camera.main.
        transform.up * Time.deltaTime / Screen.height;
  }
}
```

### 3.2.4 Bookmarking

FPBioimage implements a bookmarking function to save a particular view with an annotation. The bookmarked view can then be shared with others as a web link, or restored on the same computer at a later time.

To implement the bookmark saving and restoring into the user interface requires several distinct stages depending on the user's input. An elegant method for reliably moving the user between these stages is a state machine. The state machine controlling the bookmark saving and restoration procedures is shown in Figure 3.3, and can be described as follows:

**State 0** The default state, where all states eventually return to. Nothing regarding bookmarking is shown on the screen. Pressing 'B' on the keyboard will start the bookmark creation process; pressing a number key will start the bookmark restoration process. Any other key does not change the state.

**State 1** The bookmarking textbox pops up requesting the user to enter a number key to save the new bookmark in that position. Pressing a number key will take the user to the next stage of bookmark creation; pressing 'Escape' will close the textbox and return to the default state.

**State 2** The user can enter a text annotation into the text box. Pressing 'Escape' will immediately close the textbox and return to the default state; pressing 'Return' (also known as 'Enter') will save the bookmark as described below and move to State 3. Note that an empty annotation is accepted.

**State 3** This state displays information or annotation text. If arriving from State 2, it will show a confirmation message; if arriving from State 4, it will show the annotation text saved alongside the bookmark being restored.

**State 4** This state simply checks that the volumetric model has loaded, to ensure no error is thrown if trying to restore a bookmark on an uninitialised volume. The state machine continues to wait in State 4 if the volume is loading, so any requested bookmark will be loaded immediately when the volume is ready.

Internally, the bookmarks are encoded as JavaScript Object Notation (JSON) strings, converted to a base-64 string, and then saved to the web browser's 'Local Storage'. Saving to Local Storage requires the C# program to interface with the web browser through Javascript. Unity provides the function `ExternalEval(string jscode)` to complete this task, where `jscode` is a string containing the JavaScript code to be executed. Once the bookmark has been encoded, it is saved to the web browser as shown in lines 1 and 2 of Snippet 3.5. The bookmark is saved to Local Storage with a unique prefix followed by the bookmark's number, so that it can be restored at a later time.

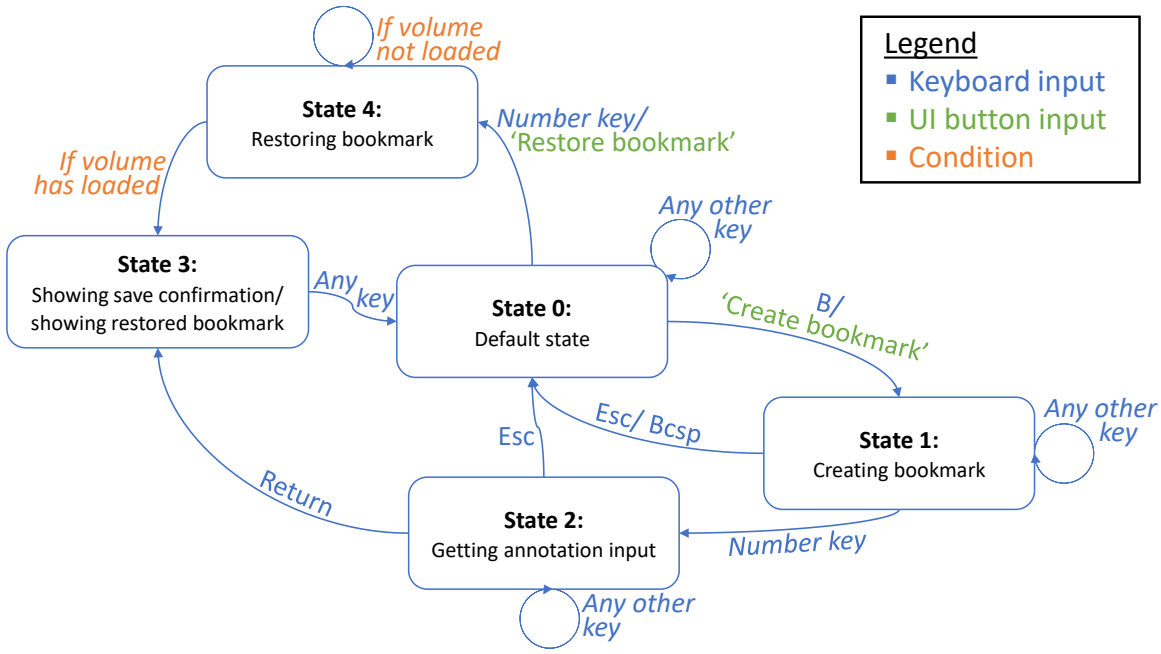



**Figure 3.3:** The state machine shown controls the creation and restoration of bookmarks.

**Snippet 3.5:** C#-JavaScript interface code for interacting with the browser's local storage to save and share bookmarks.

```
1 string evalMe = "localStorage.setItem('fpb-" + variables.fpbJSON.
    uniqueName + "-bookmark" + bookmarkNumber + "', '" +
    bookmarkString64ToSave + "');";
2 Application.ExternalEval (evalMe);
3 evalMe = "history.replaceState(null, null, '?b=" +
    bookmarkString64ToSave + "');";
4 Application.ExternalEval (evalMe);
```

Lines 3 and 4 of Snippet 3.5 shows how the base-64 bookmark string is appended to the base url in the web browser using the JavaScript function `replaceState`. The user can then simply copy the URL and send it to a friend or colleague to open on their own computer. When the volume is being loaded, the bookmarking C# script checks the URL of the webpage its running on, and if it finds a "?b=" in the URL string it will begin restoring the bookmark. This puts the bookmarking state machine into State 4, ready to show the bookmark and annotation text as soon as the volumetric model has loaded. In this way users can easily share specific views of the data across the globe.

### 3.2.5 Capturing screenshots

FPBioimage provides the capture of high resolution screenshots, shown in Snippet 3.6.

When the `takeScreenshot` function is called, a new `RenderTexture` is created to save the current view to. The size of this can be set either by the current pixel size of the main camera or by user input. In this way screenshots at a higher resolution than the current view can be created. This is advantageous, as the user can view the volumetric data in small window for fast performance, but then take a single screenshot at a high resolution for presentation and publication.

The new render texture is converted to a PNG-encoded byte stream, and then offered to the user as a download through the Javascript library `download.js`. `download.js` provides a single function `download(data, strFileName, strMimeType)`, which is called from Unity using the C#-Javascript interface function `ExternalCall`. This is shown in line 32 of Snippet 3.6.

**Snippet 3.6:** C# code for capturing a high-resolution screenshot and presenting it to the user as a download.

```
1  public void takeSreenshot(bool hiRes){
2    if (hiRes) {
3      bool wOK = int.TryParse (wRes.text, out ssWidth);
4      bool hOK = int.TryParse (hRes.text, out ssHeight);
5
6      if (!wOK || ssWidth < 1) {
7        ssWidth = 1920;
8      }
9      if (!hOK || ssWidth < 1) {
10       ssWidth = 1080;
11     }
12   } else {
13     ssWidth = Camera.main.pixelWidth;
14     ssWidth = Camera.main.pixelHeight;
15   }
16
17   // Create screenshot-sized render texture to render to
18   RenderTexture rt = new RenderTexture(ssWidth, ssWidth, 24);
19   mainCamera.targetTexture = rt;
20   Texture2D snapShot = new Texture2D(ssWidth, ssWidth, TextureFormat.
       RGB24, false);
21   mainCamera.Render();
22   RenderTexture.active = rt;
23   snapShot.ReadPixels(new Rect(0, 0, ssWidth, ssWidth), 0, 0);
```

```
24    mainCamera.targetTexture = null;
25    RenderTexture.active = null;
26    Destroy(rt);
27
28    // Encode snapshot to PNG byte stream
29    byte[] bytes = snapShot.EncodeToPNG ();
30
31    // Offer screenshot as download
32    Application.ExternalCall ("download", "" +
          System.Convert.ToBase64String (bytes), "Screenshot " + System.
          DateTime.Now.ToString ("yyyy-MM-dd") + " at " + System.DateTime.Now
          .ToString ("HH.mm.ss") + ".png", "image/png");
33  }
```

## 3.3 Results and discussion

The FPBioimage software designed in Unity, as detailed in Section 3.2, was compiled into JavaScript to run in a web browser. In this section, all screenshots and tests were performed in Google Chrome version 58 on the Windows 10 operating system. However FPBioimage also runs without issue on any modern web browser, including Mozilla Firefox, Microsoft Edge, and Apple Safari, including mobile versions of these browsers.

### 3.3.1 User interface

Much care has been taken to ensure FPBioimage is a user-friendly piece of software. Many offline software packages already exist for rendering volumetric data; FPBioimage's unique selling point is that it runs in a web browser, to make sharing data with colleagues across the world a simple process. For this reason it is important that a non-expert user can simply browse to a website and use the software without any training. At the same time, whilst being immediately accessible to new users the software contains a number of advanced rendering options for those wishing to extract more information from the data.

When opening up a web link with FPBioimage on the page, the user is presented with the view shown in Figure 3.4. Interacting with the volumetric model is responsive and intuitive. Using the mouse, the user can rotate the volume by clicking and dragging, or on a touch screen by dragging with their finger. Alternatively the arrow keys can be used on a keyboard. Pinching on a touch screen, or using the scroll wheel on a mouse, zooms in and out of the model. This creates a natural interaction with the data that requires no additional instruction.

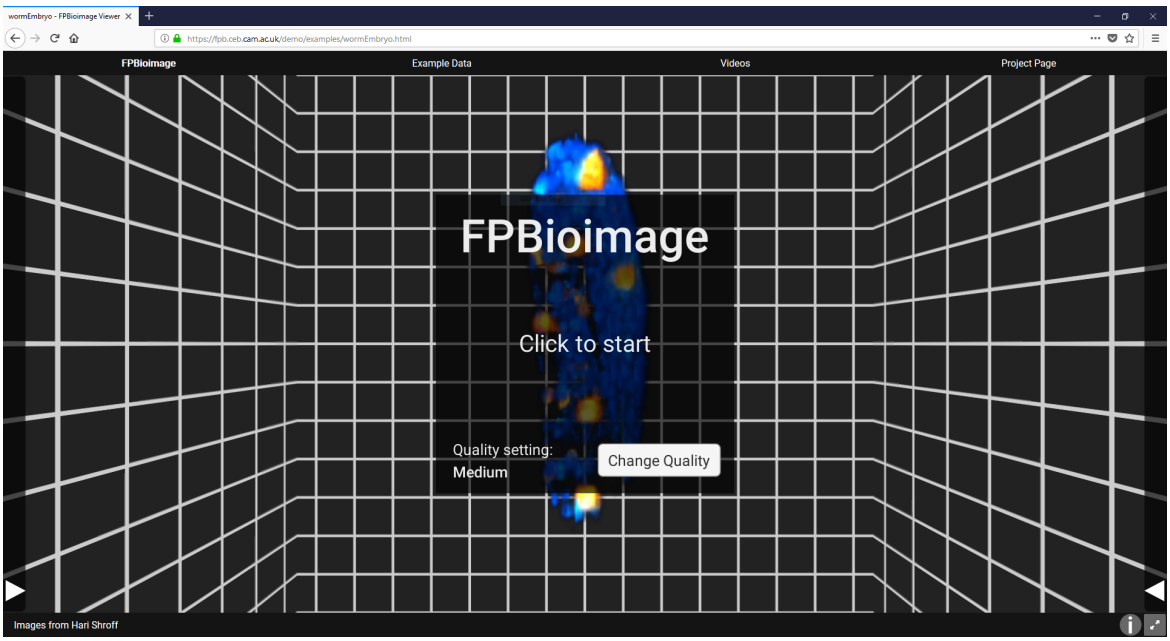



**Figure 3.4:** When a user opens an FPBioimage webpage, the data will be downloaded and the user will be presented with the simple interface shown. The model can be interacted with intuitively by clicking and dragging with the mouse, or finger on a touchscreen. The data shows a C. Elegans embryo captured by single plane illumination microscopy (SPIM), provided by Hari Shroff [8].

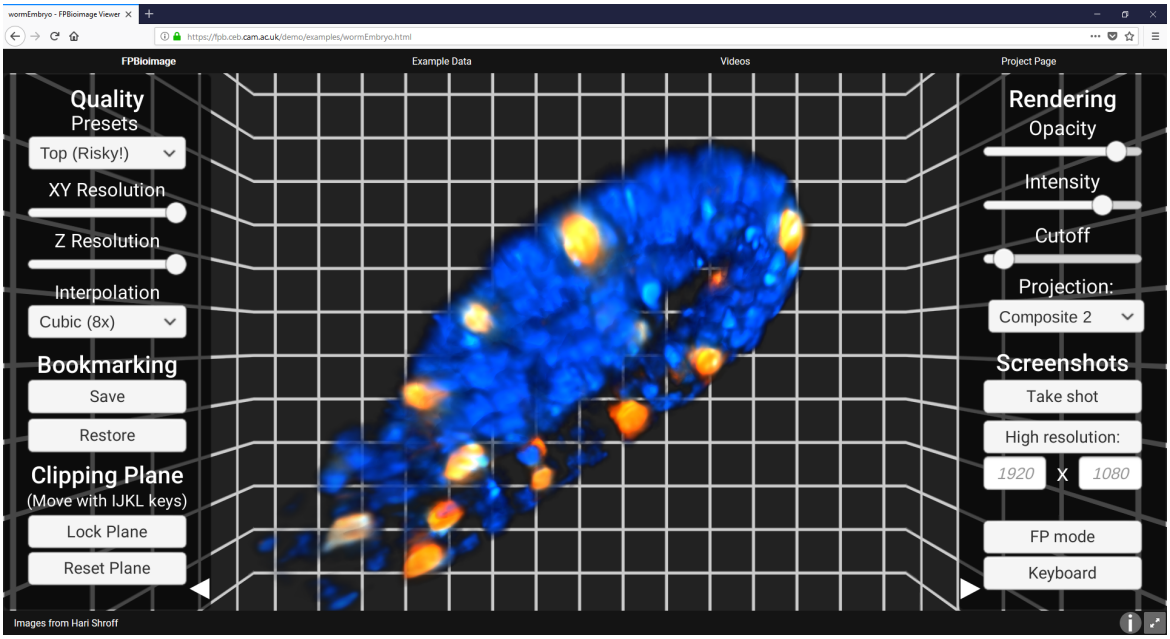



**Figure 3.5:** Clicking on the side panel arrows slides in advanced rendering options, clipping plane options, quality options, and bookmark and sharing options. Data from [8].

To access more advanced rendering features, Figure 3.5 a panel on the right-hand side of the screen which slides in. The user is presented with 3 rendering parameters: *Opacity*, *Intensity*, and *Threshold*. Matching these parameters up to the volume rendering code described in Section 3.2.2, it is easy to see how each parameter modifies the presentation of the data.

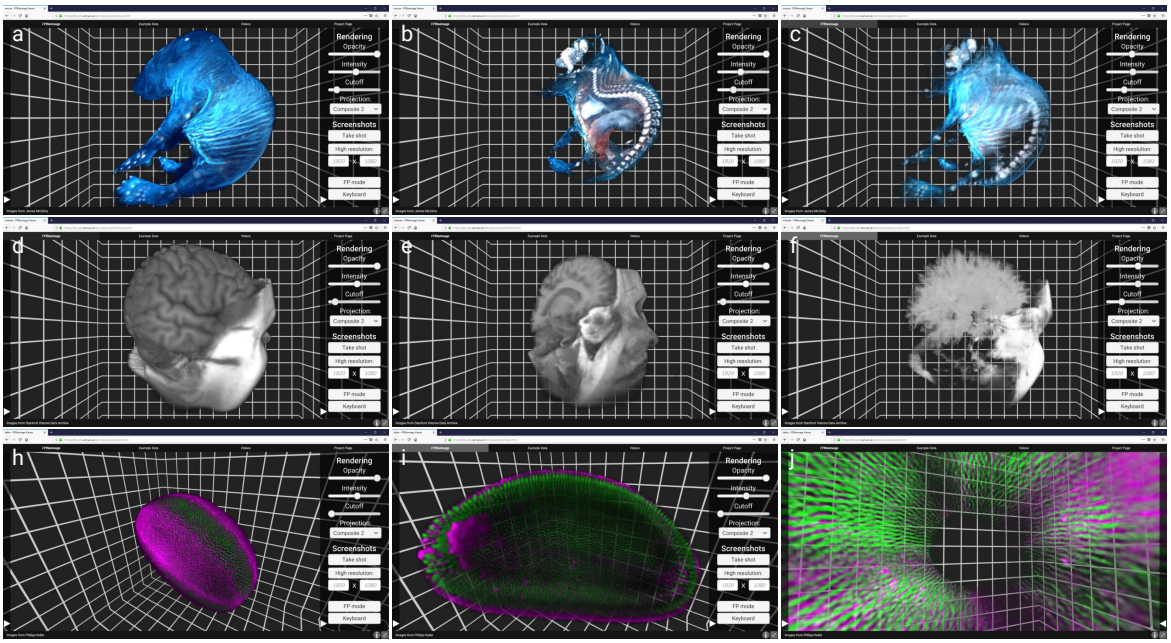



**Figure 3.6:** (a–c): Four-colour OPT data of a mouse embryo. (d–f): MRI data of a human head. (g–i): two-colour light-sheet microscopy data of a drosophila embryo. FPBioimage's built-in transparency feature has been used to render (c) and (f), revealing data within the volume; the cutting tool has been used for (c), (f), (h) and (i), removing outer voxels from the data; and h shows a bookmark in creation, which can be shared with other users. Reproduced from [9]. Raw data provided by: (a–c), J. McGinty [10]; (d–f), Stanford Volume Data Archive [11]; (g–i), P. Keller [12].

Lowering the *Opacity* value means that each voxel contributes less alpha to the ray marching through it; consequently, the volumetric data becomes more transparent. This makes features inside the model visible through the outer voxels, as shown in Figure 3.6. Combined with rotating the volume, this facilitates visualisation of how different structures are arranged in 3D space, providing answers that are not possible to obtain in any other way.

The *Intensity* value acts globally on the data at the end of the ray marching process. This simply increases or decreases the overall brightness of the model, to reveal contrast either at the bright end or the dark end of voxel values.

The *Threshold* slider sets the value below which voxels are not rendered. Most microscopy images, as well as images from other modalities such as MRI, use black voxels

around the imaging data to represent a lack of light or objects. Settings a small positive value for *Threshold* ensures that the volume does not appear as a black box when *Opacity* is set high. Larger values for *Threshold* can also be used to highlight bright objects which lie within the volume.

The next element in the right-hand UI tab is the *Projection* dropdown menu. As described in Section 3.2, FPBioimage provides several volumetric visualisation rendering pipelines. These include *Max. Intensity*, which renders a maximum intensity projection from any angle; *Composite*, which performs volumetric ray tracing with support for transparency; and *Iso-surface*, which reveals only voxels within a certain percentage of the *Threshold* cutoff. Screenshots of the various *Projection* methods rendering a [something!!] are shown in Figure 3.7.

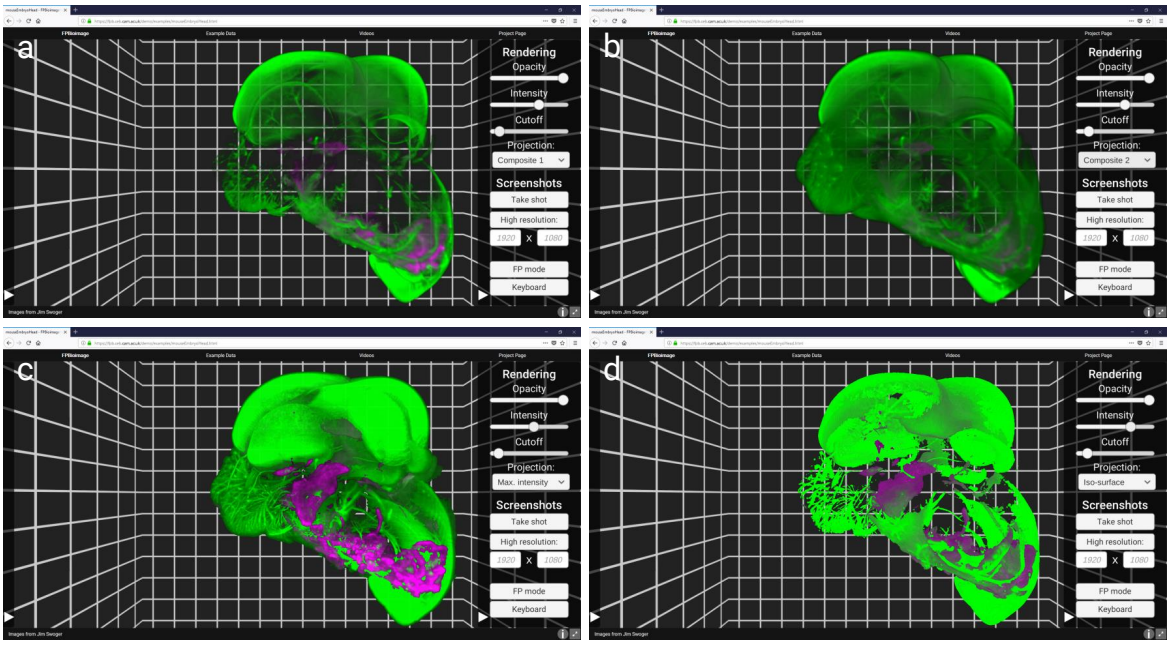
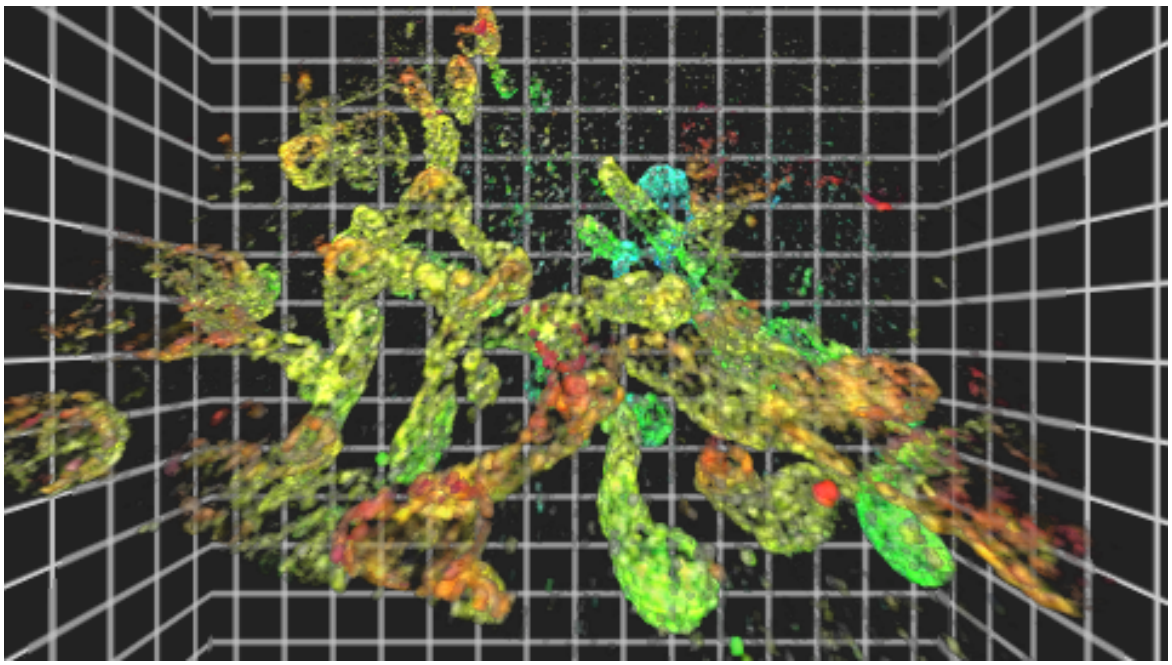
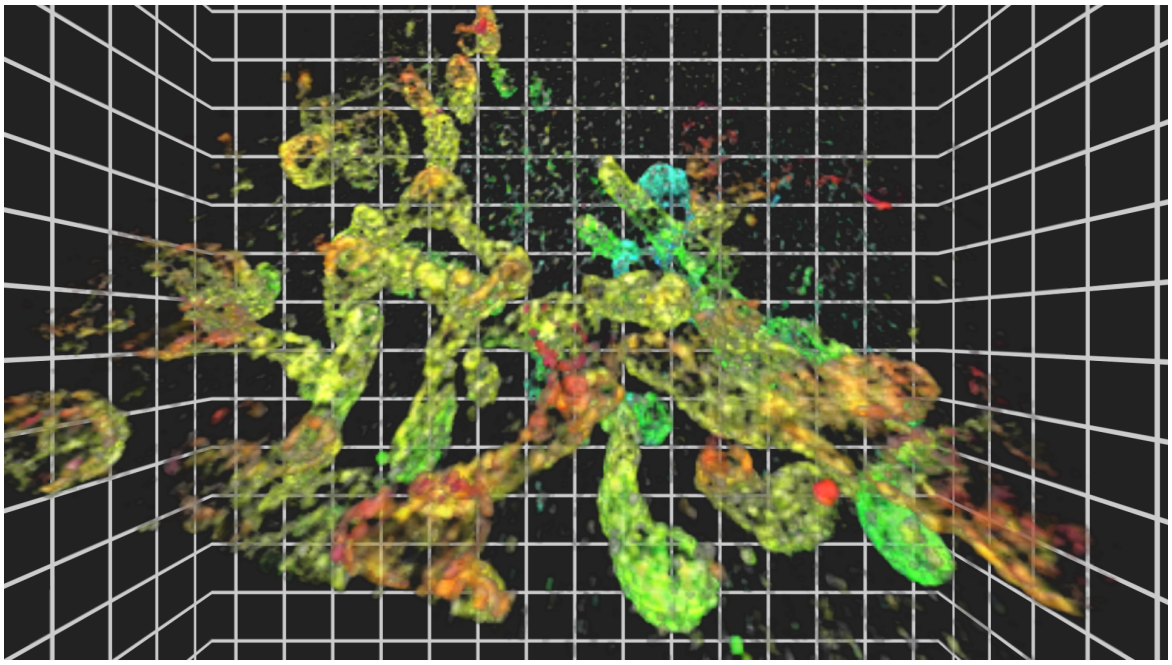



**Figure 3.7:** First Person Bioimage offers 4 different types of volumetric projection. (a) shows a maximum intensity projection, (b) a volumetric ray marching projection, (c) a variation on the volumetric ray marching projection, with alpha calculated from RGB voxel values, and (d) an iso-surface projection. Data is of a mouse embryo head captured with 'OPTiSPIM', provided by Jim Swoger [13].

Further down the right-hand UI tab are two UI buttons for capturing screenshots. The first, *Take shot*, captures a screenshot at the current resolution of the drawing canvas. Perhaps more usefully, the *High resolution* screenshot button captures a screenshot at the resolution defined in the two text boxes below. This allows the user to capture screenshots at a higher resolution than their display, in case high resolution images are required for presentation or publication. The advantage of this is exemplified in Figure 3.8.



**(a)**



**(b)**

**Figure 3.8:** High resolution screenshots can be created even on a small display. (a) shows a screenshot at the native 768x1024 display resolution; (b) shows a 1080p screenshot of the same view, created using the *High resolution* button. Images are of mitochondria captured with a whole-cell 4Pi single-molecule switching nanoscopy (W-4PiSMSN) microscope, provided by Joerg Bewersdorf [14].

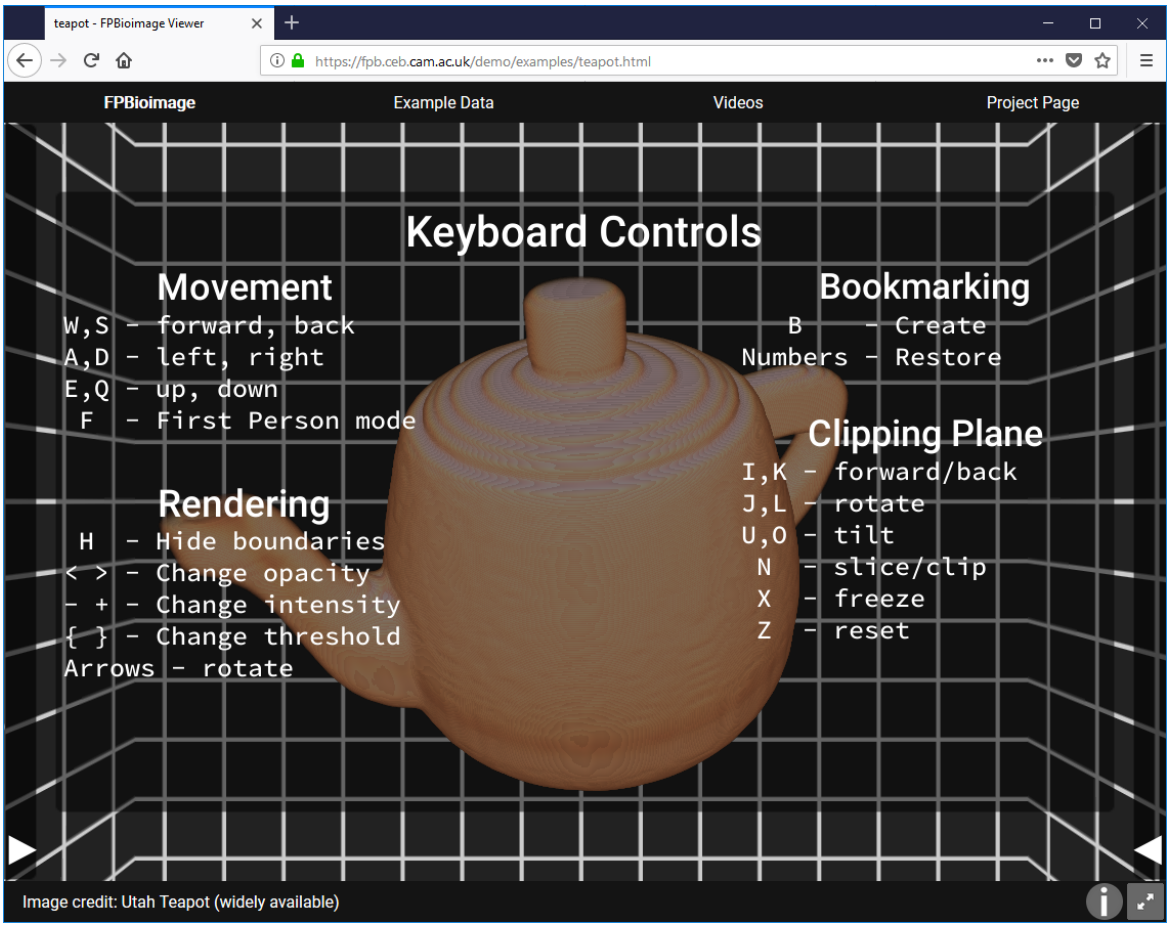



**Figure 3.9:** All rendering options in FPBioimage can be controlled by the keyboard, for mouse-free operation. The teapot in the background is the widely available Utah teapot created by Martin Newell [15].

The final two buttons in the right-hand UI tab are toggle buttons, labelled *FP mode* and *Keyboard*. *FP mode* toggles on the First Person mouse control that gives FPBioimage its full name. In this mode the user can use the mouse to look around, in a way that is familiar to anyone who has played a first-person perspective game. This can provide dramatic and immersive flights through the data controlled by the user, revealing much more information than can be provided on a pre-recorded video. First Person control is shown in Video [??] in the online version of this thesis, or in Supplementary Video [??] provided on the CD/USB drive.

At all times (except when creating a bookmark), the user can use the WASD keys to move around, as shown when the *Keyboard* button is active. Pressing the *Keyboard* button brings up a full list of controls, presented in Figure 3.9. Every rendering option is available through a keyboard shortcut, for complete mouse-free operation.

The left-hand UI tab reveals yet more parameters which can be controlled by a more advanced user.

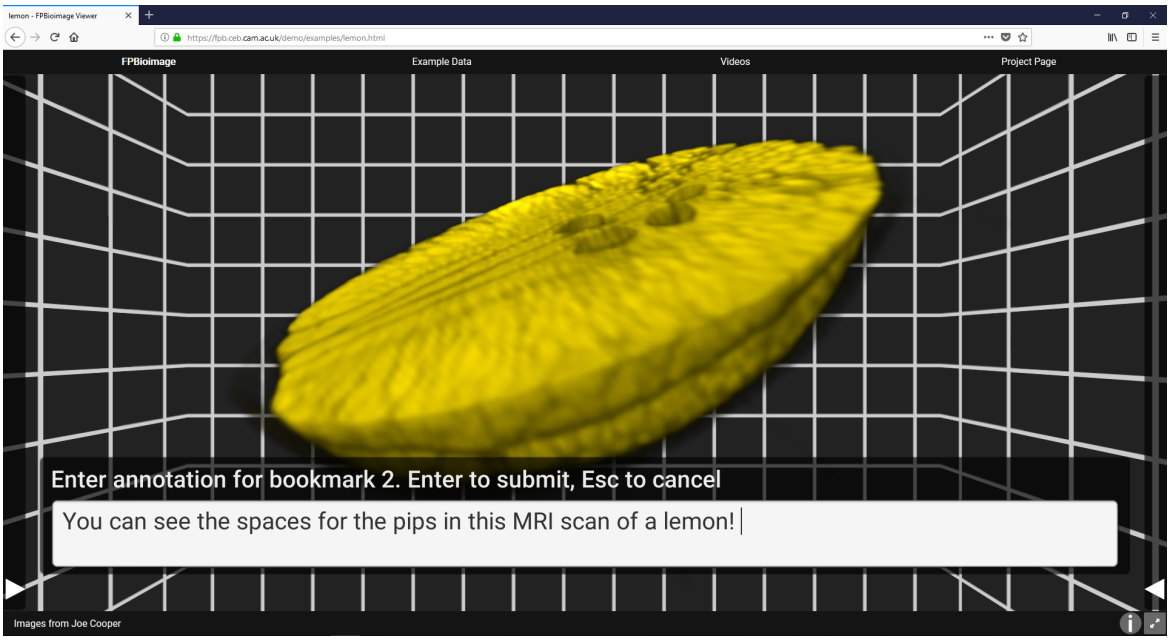
The first of these is *Quality*. Volume rendering in real time is a graphics intensive operation which has only recently become possible, and there is a clear trade off between quality and performance on most computers or mobile devices. A dropdown list with a number of preset options with suggestions of the type of device they are suitable for. For more advanced users, the individual quality settings *XY Resolution*, *Z Resolution*, and *Interpolation* can be adjusted independently. The *Quality* settings are saved to the web browser's Local Storage upon modification and are automatically restored on the user's next visit to an FPBioimage webpage. The performance-quality trade-off is discussed in more detail in Section 3.3.2.

The bookmarking functionality can be accessed from the UI using the *Save* and *Restore* buttons. This begins the process shown in Figure 3.3. Creation of a bookmark is shown in Figure 3.10a. When a bookmark is created or restored, the URL in the browser bar changes to encode the bookmark as a web link. This link can then be shared with other users, by email, text, or on social media, who will see exactly the same view when FPBioimage loads. This is demonstrated in Figure **??**, where the shared bookmark is opened on another computer, with a different browser and operating system.

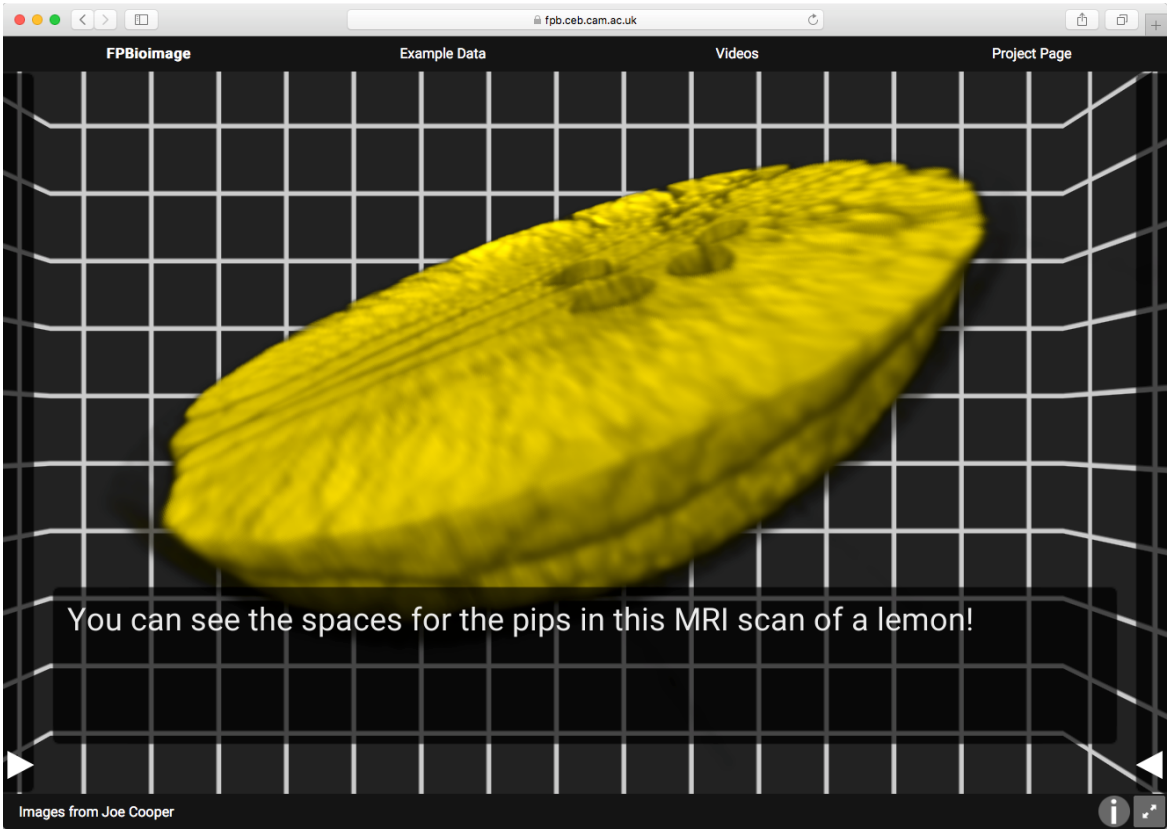
Finally, at the bottom of the left-hand UI tab are the UI buttons for the clipping plane. Mentioned briefly in Section 3.2.2, the clipping plane prevents voxels that are behind it from being rendered, revealing voxels inside the volumetric data that are otherwise obscured. The clipping plane's location is controlled with the IJKL keys, where I and K are used to move the clipping plane forward and back, and J and L rotate is around to cut the data at an arbitrary angle. Once an interesting cut has been made, the clipping plane can be locked in place with respect to the model by pressing the *Lock Plane* button. With the clipping plane locked, the user can move the camera or the model to view the cut volumetric data from a different angle. Use of the clipping plane can therefore provide insightful perspectives of the volumetric data that are not possible to obtain with any other method, as shown in Figure **??**.

### 3.3.2 Performance

Until recently, volume rendering was not possible in real time, and would be calculated given a specific camera perspective in advance to render either a single image or video. In [some year], [some company] produced the [some device name], a dedicated piece of hardware which could render [512×512×512] volumetric images in real time. This was achieved by realising the volume rendering algorithm described in Section 3.1.2 with physical devices, with [??] calculations running in parallel to compute all rays at once. [cite] Today, top-end




**(a)**




**(b)**

**Figure 3.10:** (a) shows creation of a bookmark, with an annotation, in the Firefox web browser running on Windows 10. The bookmark can be restored on the same computer, or shared as a URL to another user. (b) shows the same bookmarked view in Apple Safari running on macOS 10.13 High Sierra. The MRI scan of a lemon was provided by Joe Cooper.

graphics cards are able to perform the same calculations in real time on a Graphics Processing Unit (GPU), a highly parallelised device optimised for computing lighting models. [cite]

A primary aim of FPBioimage was that it should be user-friendly, with a particular focus on ease of use for non-expert users viewing data. It was therefore expected that a majority of users would not have access to expensive, high-performance graphics cards required for rendering volumetric data in real time at full resolution. Furthermore FPBioimage was designed so that it could be run on any device - according to [somewhere], [some percentage%] of users to the [Nature?] website accessed articles through a mobile browser. In order to support the full range of users, it was important to devise a scheme where the user could reduce the quality of the volume rendering in exchange for an increase in performance.

In the computer games industry, "performance" essentially means how many frames are rendered to the screen per second - assuming the graphics card is working at full capacity. If the graphics rendering does not require the full capacity of the graphics card, performance can be measured as the percentage of the graphics card's compute capabilities in use. In the tests presented in this chapter, performance is given in both frames per second (FPS) and graphics card usage percentage, measured using Windows Task Manager. Screenshots of the software under test are shown in Figure **??**.

"Quality" is harder to define. For a scene of simple 2D geometry quality could be measured by the XY-resolution of the screen, and we would expect performance to decrease as XY-resolution increases. For our volumetric data, however, we also have a Z-resolution, represented in Snippet 3.1 as `_Steps`. Furthermore visual quality is improved if, instead of just taking the nearest voxel to the tip of the marching ray, interpolation is performed between either the 2 or 8 closest voxels.

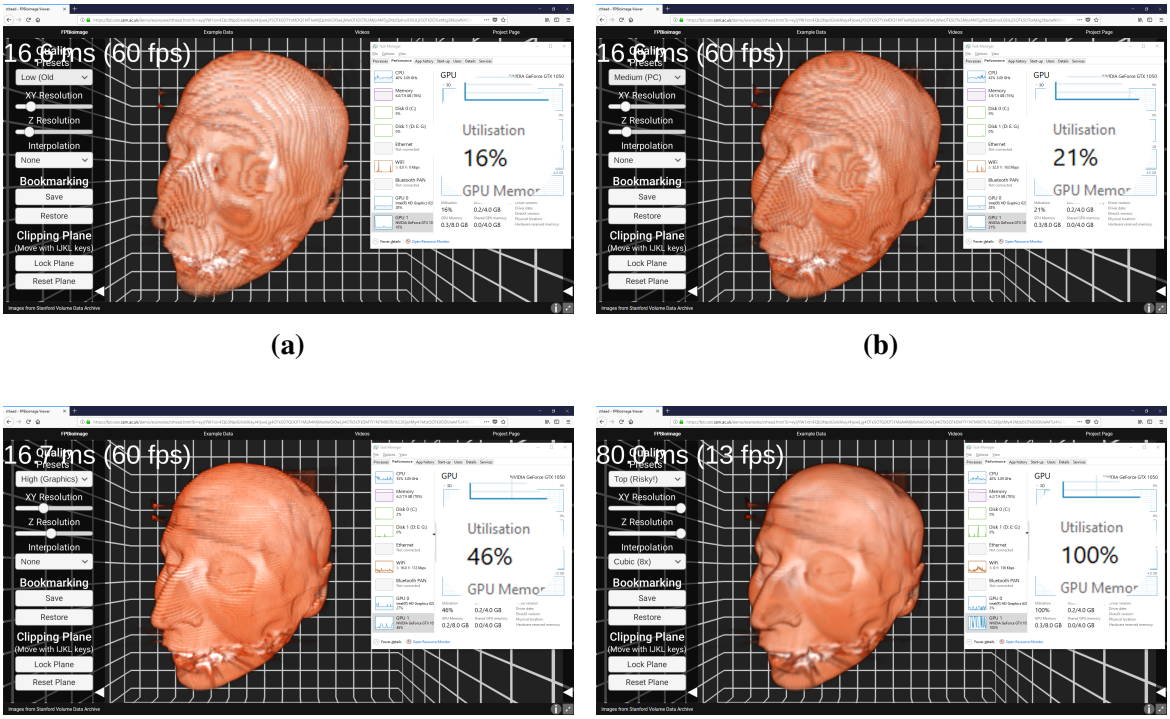
The three quality parameters can be adjusted as user inputs, or preset values can be selected. A table of quality against performance for different combinations of the three quality metrics, and for the preset values, is shown in Table 3.1.

As expected, the table shows that as the quality settings are increased in value, performance decreases. To choose the most useful preset values, the visual impact of the quality settings was compared to the performance trade-off.

As XY-resolution is increased, performance decreases dramatically. This is because the number of pixels being rendered increases with the square of the XY-resolution number. It is interesting that XY-resolution does not have such a dramatic affect on the perceived visual quality, as shown in Figure 3.11. Since there is little visual quality gain for a relatively large performance impact, XY-resolution is kept to moderate values even for the higher quality presets.

**Table 3.1:** The table shows how performance, measured by FPS and GPU percentage use, is affected by the FPBioimage quality settings. This test was performed on a midrange laptop, which cost £580 in 2017, with the following specification: 2-core i5-7200U @2.50 GHz, 8 GB RAM, NVIDIA GeForce GTX 1050 with 4 GB graphics RAM.

| Preset | | Quality | | | Performance | |
|---|---|---|---|---|---|---|
| Name | Device | XY-res. | Z-res. | Interp. | FPS | GPU use |
| **XY-resolution test** | | | | | | |
| - | | 256 | 256 | 1X | 60 | 14% |
| - | | 768 | 256 | 1X | 60 | 41% |
| - | | 1024 | 256 | 1X | 60 | 63% |
| - | | 2048 | 256 | 1X | 41 | 88% |
| **Z-resolution test** | | | | | | |
| - | | 450 | 64 | 1X | 60 | 15% |
| - | | 450 | 256 | 1X | 60 | 21% |
| - | | 450 | 512 | 1X | 60 | 30% |
| - | | 450 | 768 | 1X | 60 | 39% |
| **Interpolation test** | | | | | | |
| - | | 768 | 256 | 1X | 60 | 40% |
| - | | 768 | 256 | 2X | 60 | 49% |
| - | | 768 | 256 | 8X | 54 | 83% |
| **Preset values** | | | | | | |
| Very low | Old laptop | 256 | 64 | 1X | 60 | 11% |
| Low | Mobile | 400 | 100 | 1X | 60 | 15% |
| Medium | Laptop | 450 | 150 | 1X | 60 | 18% |
| High | Desktop | 768 | 350 | 1X | 60 | 49% |
| Very high | Graphics card | 1024 | 500 | 2X | 38 | 87% |
| Top | 3D workstation | 2048 | 768 | 8X | 9 | 97% |




**(a)** **(b)**

**(c)** **(d)**

**Figure 3.11:** The figure shows FPBioimage rendering with different quality presets: (a) Low; (b) Medium; (c) High; and (d) Top. The highest quality levels are designed for use on a high-end desktop computer; lower quality settings are provided to allow FPBioiamge to run on mobile devices. Performance was measured using frames per second (FPS) and graphics card percentage usage, shown on the screenshots. Results from these tests are presented in Table 3.1. The CT scan of a human head is from the Stanford Volumetric Data Archive [11].

Compared to the performance impact of XY-resolution, Z-resolution has a relatively small impact. From a Z-resolution of 256 steps to 768 steps the GPU use increases linearly. As shown in Figure 3.11, however, Z-resolution has a considerable impact on visual quality of 3D images. For an equivalent impact on performance, an increase in Z-resolution gives better visual quality than an increase in XY-resolution; therefore providing high Z-resolution is prioritised for the presets.

Interpolation of voxels has a big impact on visual quality, removing the staircase effect shown in Figure 3.11, but also has a big impact on performance. When 2X interpolation is used, two voxels from the two nearest Z-planes to the marching ray's current position are averaged (weighted by distance) to smooth the appearance of the volume. This requires an extra texture lookup at every z-step for every screen pixel, causing a significant performance impact particularly at high XY- or Z-resolutions. In 8X interpolation, the 8 nearest voxels to the marching ray's position are used, requiring 8 texture lookups at every z-step. The extra texture lookups explains the high impact on performance of interpolation.

The 'Top' preset setting sets all quality variables to maximum, and the user is warned that this setting is 'risky' because it can crash the web browser! This is an impractical setting on most personal machines - Table 3.1 shows that the GPU is working at full capacity, and only rendering 9 FPS. However, on dedicated graphics workstations where the GPU is able to render 60 FPS at top quality a fly-through video for publication can be recorded in real-time.

## 3.4 FPBioimage Suite

FPBioimage is a volumetric renderer designed to be used in a web browser. However, to assist users uploading their 3D data to share on the internet, plugins are available for the open-source image analysis platforms ImageJ, FIJI, and Icy. Furthermore, thanks to its open-source nature and an easy-to-use JSON interface, other groups have been able to integrate FPBioimage into their own software packages, for example OMERO-FPBioimage. Finally, for users who want to view volumetric data on their mobile, I have created dedicated Android and iOS apps, which support virtual reality for truly immersive interaction with the data. Together with FPBioimage, this collection of applications has been branded FPBioimage Suite.

### 3.4.1 Plugins

After the initial release of FPBioimage, uploading data to share with others was a convoluted process. Users had to download a copy of FPBioimage, install it on their own personal web

server, upload a stack of PNG files with a specific naming convention, edit some Javascript on a template webpage, and finally share the web link. This is not an issue for any web developer, but was beyond the ability of many scientists who gather volumetric data. Whilst many users enjoyed viewing data with FPBioimage, very few used it to share their own data.

In order to simplify the upload process, I created the FPBioimage Helper plugin for the popular open-source image analysis programs ImageJ, FIJI, and Icy. These programs all support volumetric image data in a wide variety of formats thanks to OME Bioformats. Users simply have to open their 3D data in the program, click one button, and the plugin will output a web link that the researcher can share with others.

By default, FPBioimage Helper will upload a simple webpage displaying the data to an Amazon Web Services (AWS) site. If the user would prefer to host their data on a private web server, for example to implement password protection or use their own website template, they can choose not to upload the data but simply save the formatted images and website to their local machine for their own purpose.

Using the plugin, rather than manually uploading PNG image slices, also provides faster download times for those viewing the data. Instead of saving every plane as a separate PNG file, the plugin arranges images into texture atlases as described in Section 3.2.2. A line of JSON (`'atlasMode': true,`) tell FPBioimage to expect atlases, rather than individual image slices, reducing the number of downloads to 8. This reduces loading times in four ways: the overhead time associated with each download is reduced from number-of-slices to just 8; overall download size is smaller, because the PNG compression algorithm is able to exploit more informational redundancy [cite]; the 8 texture atlases can be downloaded in parallel; and FPBioimage can skip the step of arranging slices into texture atlases, since it can send the downloaded atlases directly to the graphics card.

The plugin is distributed as an update site through the FIJI updater, and also through the Icy online plugin repository. This makes it easy to keep up-to-date, so that users get the access to the latest features with as little effort as possible.

### 3.4.2 OMERO-FPBioimage

OMERO is a web-based platform for managing scientific images. Created by OME, the same team behind Bioformats, it integrates with a number of other software packages to help users upload their imaging data to a remote server, so that it can be accessed from anywhere. Furthermore a web interface makes it easy for OMERO users to share their data with others, allowing them to perform further analysis or even meta-studies on independent image sets.

After reading about FPBioimage in Nature Photonics, an OMERO user noticed that the two programs have similar goals, and suggested integrating FPBioimage into OMERO

as the default volume renderer. Thanks to FPBioimage's clear documentation and simple JSON interface, the OMERO team were able to incorporate an FPBioimage viewer into their software before any contact with me.

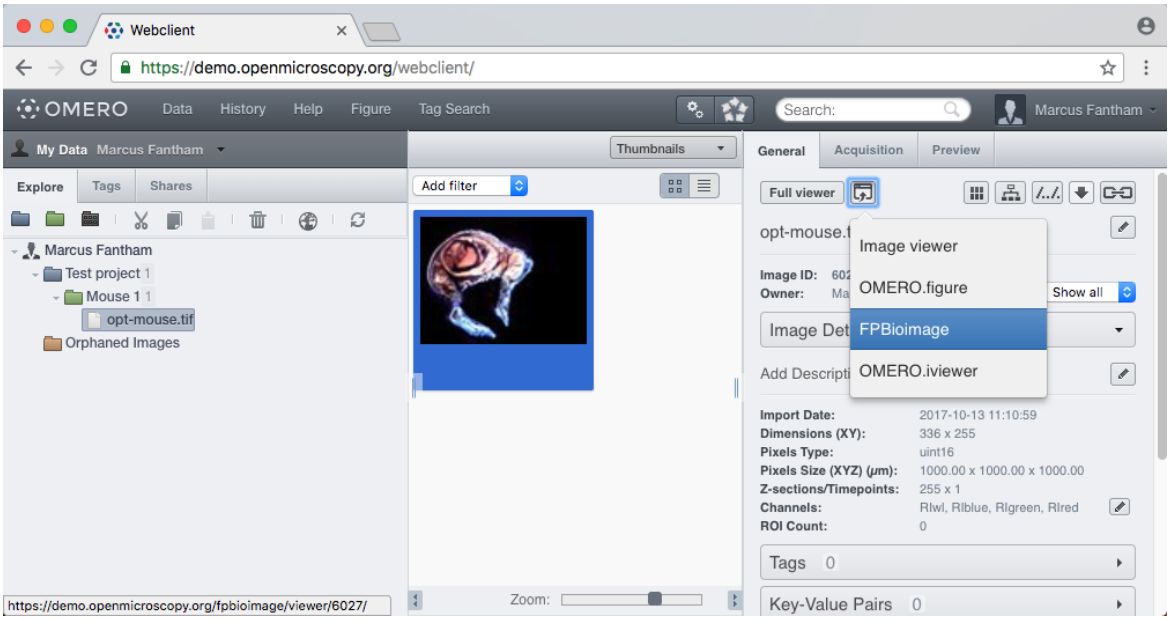



**Figure 3.12:** The screenshot shows OMERO web, which uses FPBioimage as its default volumetric image viewer.

If a user opens a volumetric image in OMERO, an option appears in the viewer options to open in FPBioimage (shown in Figure 3.12). When selecting this option, the OMERO server quickly creates a static webpage with an FPBioimage viewer and the relevant JSON to load the data into FPBioimage. Close collaboration with the OMERO team means that OMERO now sends texture atlases directly to FPBioimage, reducing loading time for users.

The viewer has proved popular with OMERO users, finding diverse uses from plant sciences to pathology. It has been adopted as OMERO's default volumetric viewer, and is featured prominently on their webpage. [https://www.openmicroscopy.org/omero/features/view/]

### 3.4.3 Mobile Apps

Since FPBioimage is a web-based application, it will run on any device with a web browser. This includes mobile devices, such as iPhones and Android smartphones, as well as tablet devices such as an iPad. However, mobile browsers have limited access to the device's computational resources, causing long loading times and poor performance, characterised by a low frame rate.

To allow FPBioimage to run on mobiles with full performance, native apps were made for the Android and iOS operating systems, which are available through their respective app stores. Since mobiles usually only run one app at a time, the app has full access to all the computational and graphics processing capability of the device.

For the mobile apps, a universal linking [cite] scheme has been devised. This means that if a user visits a website with an FPBioimage viewer on it, they will get the option to open the data in a native app on their phone instead of in the mobile browser. Since the mobile app uses the same rendering algorithm (presented in Section 3.2.2) as the online app, the visualisation will be exactly the same.

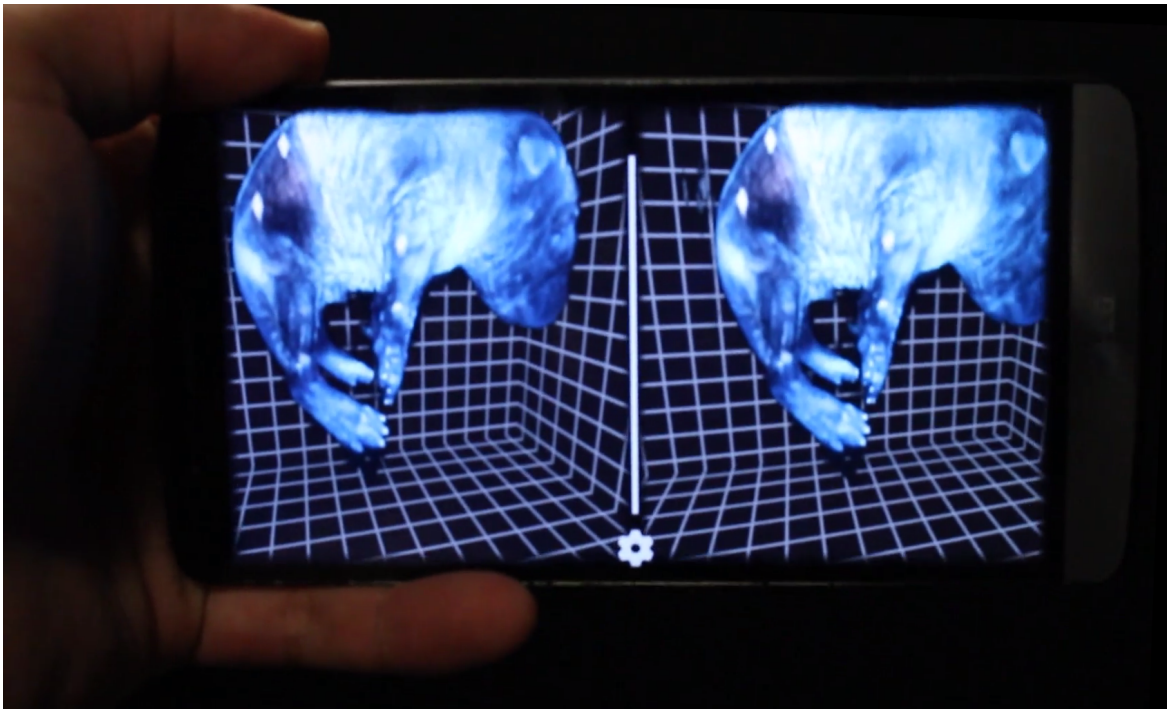


**Figure 3.13:** FPBioimage has native apps for Android and iOS. The photo shows a split-screen stereo view, which can be used in conjunction with Google Cardboard[16] to support virtual reality. Data from [10].

As part of the port to mobile, a new focus was put on FPBioimage touch controls. Users can intuitive rotate the model by touching and dragging. Users can also zoom in and out by a two-finger pinch, and move around the model using two fingers and dragging. As shown in Figure 3.13, the quality can be changed to accommodate a range of mobile chipsets, and the usual rendering options of opacity, intensity and threshold are available, along with the various projection methods.

Running on mobile brings one more advantage, thanks to the low-cost virtual reality (VR) device Google Cardboard. [cite cite] This is a VR viewer - originally made of cardboard but

now available in more robust plastic models - which utilises a smartphone's high-resolution screen and precise accelerometer to provide a low-cost VR experience. The user simply has to open their data in the mobile app, click the "Start VR" button, and put their phone into their Cardboard viewer. This option allows users to view their data in VR, providing a truly immersive experience and giving a perspective of the data not possible through any other means.

## 3.5 Conclusion

As an online volumetric rendering program, FPBioimage has a wide range of uses which have not been possible before.

As a volumetric renderer, FPBioimage offers a number of volumetric rendering modes found in most offline volume visualisation software, including composite ray marching and maximum intensity projections. However all other volume rendering programs have a complicated array of settings for changing transparency and colour maps, and no option for adjusting quality. With a focus on ease-of-use from its conception, FPBioimage provides an interface that anyone can use, even non-expert users.

Furthermore, because FPBioimage is designed to run on the web, or on devices with limited computational resources, the rendering algorithms have been optimised such that full-quality volume rendering can be performed in real time. This real-time performance has not been seen in any other software, even though they do not have such restrictions. Indeed, the ease-of-use coupled with high performance have generated many requests that FPBioimage could be used as the default volumetric viewer in other image analysis software.

In combination with the ImageJ, FIJI, or Icy plugins, scientists collecting volume data can share it with anyone in the world in one click. Data is uploaded to a public repository hosted on Amazon Web Services, at a permanent link, or can be uploaded to a private server for password protection. This allows researchers to get unprecedented instant feedback from international collaborators.

Researchers can embed FPBioimage in a presentation by utilising a slideshow plugin such as LiveSlides [cite], as shown in Figure 3.14. This allows an interactive flight through the data whilst talking about it, bringing a new dynamic to presentations and allowing detailed responses to questions which are not possible with a pre-recorded video.

Finally, for sharing volumetric data with the wider community FPBioimage can be used as a publication tool. A whole set of 3D images can be uploaded, such that readers can explore for themselves the findings of an experiment. [cite MOF paper?] It is hoped that

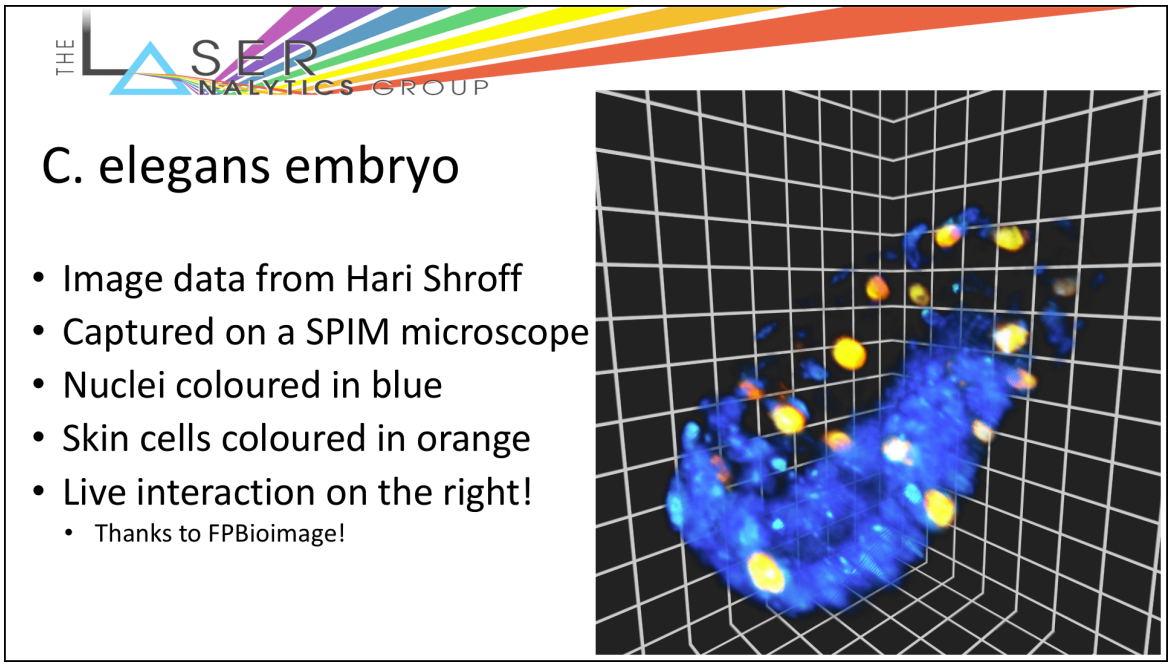



**Figure 3.14:** Using the Powerpoint plugin LiveSlides, FPBioimage can be embedded in a presentation, bringing an interactive dynamic to explaining volumetric data. Data from [8].

journals will adopt FPBioimage as a dynamic viewer for online publications, so that readers can interact with data alongside the text.

FPBioimage makes sharing of volumetric data with anyone in the world a one-click operation, and introduces a new paradigm for publishing 3D image data.

# Chapter 4

# MOFs: Metal organic frameworks for drug delivery

"One of the most exciting developments in recent porous-materials science"
*Teplensky et. al, JACS 2017* [17]

## 4.1 Introduction

### 4.1.1 siRNA as a cancer treatment

When the latest statistics say that one in two people born in the UK will develop some form of cancer in their lifetime [citation on CRUK], it hardly needs stating that cancer has affected almost everyone in the country in some way. Any successful efforts to discover new treatments for this set of diseases are therefore very welcome by the community. In this chapter, I describe a novel method of delivering therapeutic drugs to cells, utilising the LAG SIM for imaging and FPBioimage for 3D visualisation and analysis.

The availability of the human genome sequence [cite cite?] provides new opportunities for using DNA and other nucleic acids to target specific genes for therapeutic treatment. One such form of gene manipulation is RNA interference (RNAi) [cite cite cite]. RNAi prevents the expression of certain genes by degrading messenger RNA (mRNA) after transcription, thus preventing translation. RNAi can be performed with small interfering RNA (siRNA) [cite Science Nature 1999 Baulcombe Tuschl], which silences mRNA molecules by cleavage of the mRNA strand into two pieces.

If a specific gene is found to cause cancer in a given cell type, then appropriate siRNA can be synthesised to interfere with that gene. The siRNA will be 100% complementary to the gene sequence it is targeting, providing very high specificity. This means that, assuming the

siRNA is inside the cell, it can prevent the cancer without any other side effects. If the siRNA is inside a cell where, for some reason, the mRNA does not transcribe the cancer-causing gene, the siRNA will have nothing to bind to, and will simply degrade.

There are two difficulties with this potential treatment method. Firstly, discovering genes which are specific to cancer is not trivial. Complicated pathway from gene to disease. That said, now had 2 decades of research on siRNA, with billions of dollars poured in by pharmaceutical companies. A patent search shows ?? hits for siRNA, with ?? for siRNA + cancer.

The other issue with this treatment method is that siRNA is degraded by enzymes in extracellular space. [cite] Intuitively this makes sense - it would not be sensible for the body to allow arbitrary pieces of nucleic acid to enter the cell, since (as just described) it could affect the cell's ability to express genes. To allow siRNA to act as a treatment, we therefore require a system to protect the siRNA in extra-cellular space which can also enter the cells. This process is known as delivery. [cite?]

In the literature, we can find several methods under investigation for use as an siRNA delivery agent. Lipofectamine. Lipid nanoparticle. Virus/nucleocapsid thing. Naked siRNA supported with a strong sugary backbone? see [https://www.sciencedirect.com/science/article/pii/S1818087614000
for more details.

Collaborators in the Department of Chemical Engineering at Biotechnology in Cambridge University are experts in metal organic frameworks (MOFs), and we therefore worked together to develop a method of drug delivery utilising MOFs.

### 4.1.2 MOFs as a delivery vehicle

MOFs are a group of crystalline materials which self-assemble from a mixture of metal ions and organic linkers to form porous solids. They have a diverse use across many fields; because of their ability to contain other molecules within their pores, they have been applied to catalysis [cite cite], ion exchange [cite cite] and sensors [cite cite]. We recently published an application of MOFs for oxygen storage for use in emergency healthcare[cite]; the details of this work fall outside the scope of this thesis.

A MOF with the correct pore size could be loaded with a drug, protecting the contents from extracellular space. If the MOF is then able to enter cells and release its payload, then the system successfully delivers the drug to the cell. To be an effective drug delivery system, however, the MOF must also be biocompatible. At all stages of its journey to the cell the MOF must not produce a toxic or immunological response. This requirement extends of course to any of the products the MOF is broken down into. Choosing a biocompatible

MOF with an appropriate pore size remains a challenge, and will be discussed in the Results Section **??**.

Instead of targeting cancer-causing genes, we can target the cause of cancer itself. Cancer cells become immortal. One thing that happens is that the Krebs cycle stops working, i.e. mitochondria switch off. DCA (dichloroacetate) turns the Krebs cycle back on. But DCA has a half-life in the body of under an hour. Ie after an hour half the DCA that went in is already out. Again, a biocompatible MOF can be used to protect the DCA from being ejected so fast from the body. (Can it? Check the paper) More experiments on cells.

### 4.1.3 Structure of this chapter

The Results and methods Section 4.2 describes three experiments which were imaged on the LAG SIM.

The first experiment looks at whether drug release over time can be controlled by pre-treatment of the MOF-drug complex, and using calcein as a model drug.

The second experiment investigates MOF loaded with siRNA, for use as a cancer therapeutic.

The third describes MOFs loaded with DCA and a mitochondira-targetting cofactor, a drug combination designed to destroy cancerous cells.

In all experiments, synthesis of the MOFs and culturing of cell lines was performed by collaborators. Microscope setup, image capture and reconstruction, and the subsequent analysis was performed by me.

## 4.2 Results and methods

### 4.2.1 Temperature treatment of MOFs to modify release profile

A problem of drug delivery systems that has been noted in the literature is the so-called *burst release effect*. [cite?!] This is an undesirable characteristic of delivery systems where high concentrations of the drug are released from the carrier soon after loading, decaying quickly so that after a short time there is no payload left to release. This either leads to toxic levels of drug, or low concentrations which must be applied frequently to fulfil their therapeutic effect. An ideal drug delivery system would release its payload slowly and steadily over an extended period of time, maintaining non-toxic concentrations.

If a MOF is used as the delivery system, the payload drug is stored in the pores of the MOF, as shown in Figure 4.1. It was hypothesised that, after loading the MOF with drug,

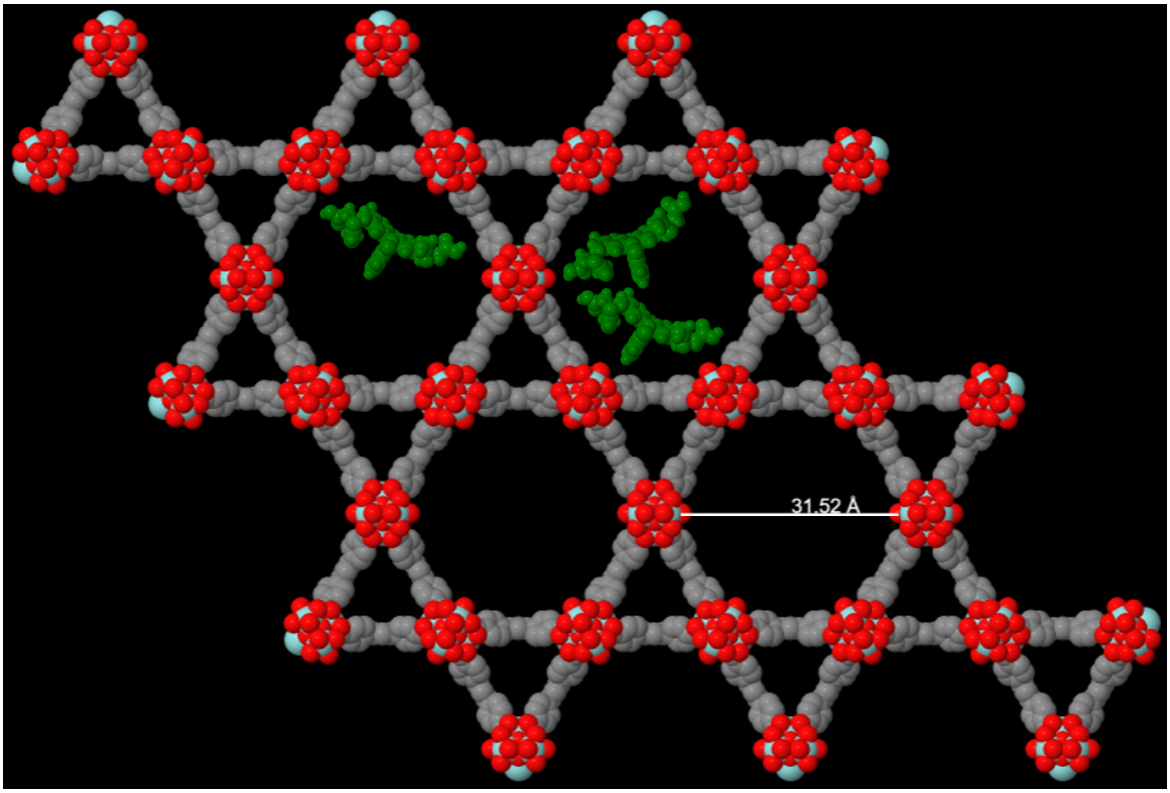



**Figure 4.1:** MOFs are crystalline structures of metal ions joined by organic linkers. The pore size of NU-1000 MOF, 31.5 Å, is large enough to store calcein molecules shown here in green.

release could be delayed by collapsing the pore structure. Pore collapse was achieved with a temperature treatment step after drug loading, before to the cells and imaging.

In this experiement a naturally fluorescent dye, calcein, was used as a model drug. MOFs developed by Norwestern University, known as NU-1000 and NU-901, were chosen and synthesised for their large pore size. Calcein was loaded into the MOFs by soaking 10 mg of either NU-1000 or NU-901 in a 10 mg/ml calcein solution for 3 days in a 37 °C shaking incubator. After this period, the supernatant was removed through centrifuging, and dried at 37 °C for 24 h.

Temperature treatment was then performed on non-control samples. Samples were placed in a high-temperature vacuum oven at 180 °C for 24 h to induce pore collapse.

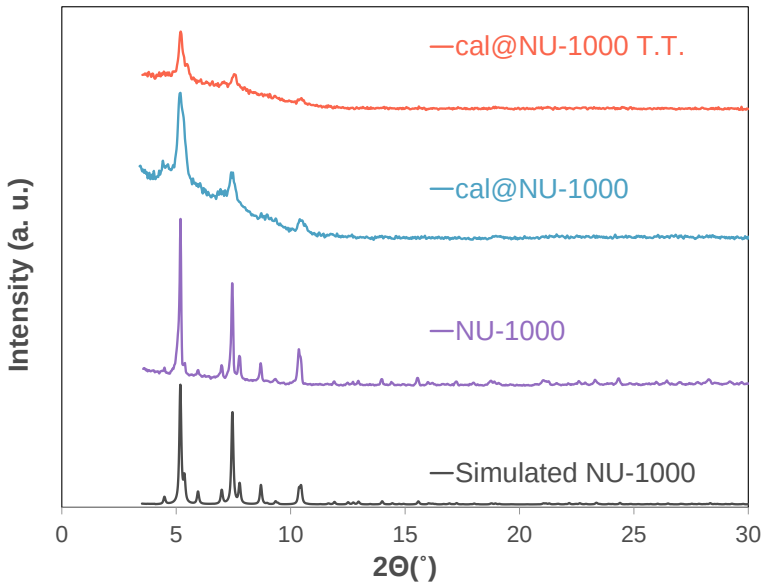



**Figure 4.2:** Experimental PXRF data (purple) matches well to simulated data (black). When calcein is loaded as a model drug (blue) the same Bragg peaks are still visible, showing that the crystalline structure is maintained; however the slight broadening of the peaks suggests calcein is successfully loaded into the MOF pores. After temperature treatment (red), pore collapse reduces the intensity of the peaks; however they are still visible, suggesting long-range crystalline order is preserved.

Pore collapse was verified by power X-ray diffraction patterns. Figure 4.2 shows that Bragg peaks broaden on addition of calcein, but are still visible at the same $\theta$, confirming the overall MOF structure is maintained. The slight broadening of loaded MOF is most likely a result of the loading process, which leads to some decrystallisation of the MOF. The decrease in peak intensity for temperature-treated MOF is due to a loss of crystalline structure from

the pore collapse; however peaks are still visible, suggesting that long-range order was not completely destroyed.

The release profile of calcein from NU-1000 MOF is shown in Figure 4.3, with the inset graph highlighting the burst release effect and its delay by temperature treatment. The effect of temperature treatment is particularly notable at the 24 h time point, where MOF has only released 30% of its payload, compared with 60% released by untreated MOF. This shows that the temperature treatment is successfully suppressing the burst release effect.

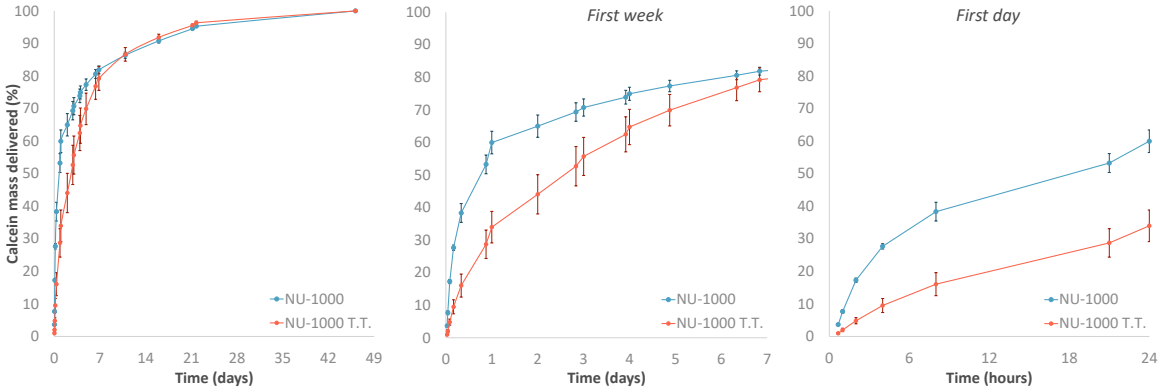
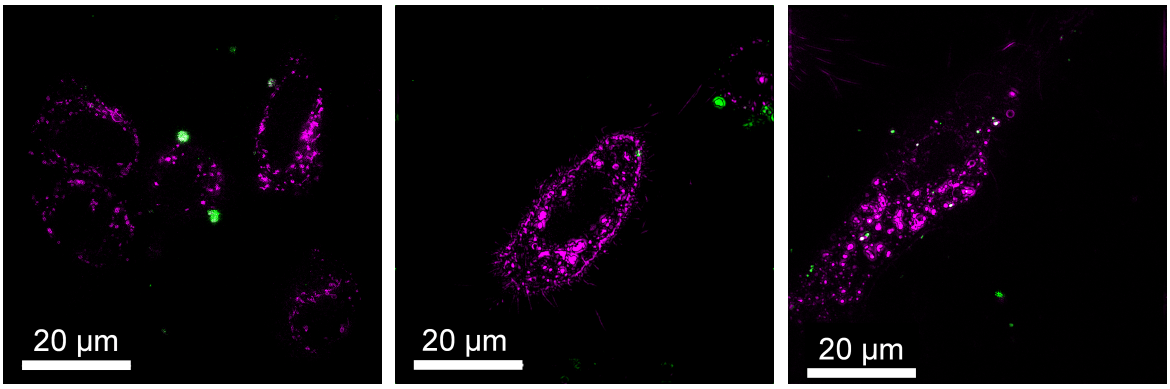



**Figure 4.3:** The figure shows three views of the same graph, with the time axis scaled differently to highlight the delay of payload release from MOF. The effect is particularly noticeable after 24 h, where just 30% of the payload is released from temperature-treated MOF compared to 60% released from untreated MOF. By delaying the release the MOF has longer to enter cells and will deliver a higher percentage of its payload, rather than releasing it in extracellular space.

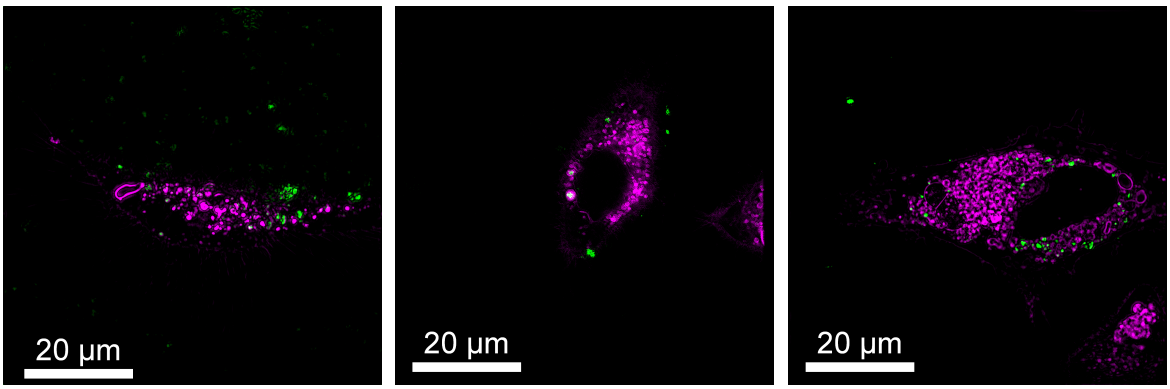
In order to visualise calcein release in a biological context, HELA cells were labelled with HCS NuclearMask™ Deep Red Stain to view the nucleus, and CellLight Lysosomes-RFP BacMam 2.0 or CellLight Early Endosomes-RFP BacMam 2.0 to view lysosomes or endosomes respectively. Calcein itself is fluorescent at 495/515 nm for excitation and emission wavelengths respectively, so had good spectral separation for imaging in 3-colours.

Images were captured on the LAG SIM, utilising optical sectioning to remove out of focus light. Representative image slices over a 24 h period are shown in Figure 4.4. Note that the nucleus was not captured for most images to minimise imaging time and thus prolong cell viability; a 3-colour image is presented in the SIM showcase, Figure 2.15. Also notable is that the MOF is highly autofluorescent at the same wavelength as calcein, such that the calcein released from the MOF cannot be seen on its own. In later experiments, detailed in Section 4.2.2, a drug with better spectral separation was used for clearer imaging.

An issue with examining 2D image slices is that it is unclear whether MOF has entered the cell, or is simply sitting on the cell membrane with out-of-focus light making it appear




**(a)** 30 minutes **(b)** 1 hour **(c)** 2 hours




**(d)** 6 hours **(e)** 8 hours **(f)** 24 hours

**Figure 4.4:** Images of Hela cells, with endosomes coloured in magenta, captured over a 24 h time period show the successful uptake of calcein-loaded NU-1000, coloured in green. The low quantity of uptake in the first 2 h highlights the need for temperature treatment to delay payload release, avoiding the burst release effect. These images can be viewed interactively in 3D at https://fpb.ceb.cam.ac.uk/MOF to confirm MOF is located within the cells, and not simply sitting on the cell membrane.

localised within the cell. Optical sectioning allowed images to be reconstructed in 3D, so that the cell could be viewed from any angle, verifying that MOF had indeed been taken inside the cell. To allow readers of the publication to verify this themselves, full 3D reconstructions of the slices in Figure 4.4 are hosted online with FPBioimage, and can be viewed at https://fpb.ceb.cam.ac.uk/MOF/.

The slices in Figure 4.4, together with their online 3D visualisations, show that MOF is successfully taken up by HeLa cells over a 24 h period. The MOF appears to be taken up by endocytosis, evident from the colocalisation of endosomes with MOFs, seen as white areas in the Figure. To verify this observation, an endocytosis study was performed. Inhibitors of various endocytosis pathways were added to the cells, and the fluorescence in the cytoplasm measured after a 24 h incubation period. Figure **??** firstly confirms that MOF uptake is an active process, as reducing the temperature to 4 °C causes a significant reduction in cytoplasmic fluorescence. Furthermore, the study shows that 0.3 M sucrose also causes a significant reduction in the amount of MOF uptake, implying MOF is endocytosed by the clathrin-mediated pathway.

Further evidence of endocytosis is provided by video-rate imaging of MOF uptake. The supplementary video [somewhere?] shows that MOF outside the cell moves with fast Brownian motion; once inside the cell, however, MOF is comparitvely static. This provides further evidence that MOF is taken up by the cells to deliver its payload.

The timescale over which MOF enters cells is important, and emphasises the need for the temperature treatment process. Figure 4.3 shows that, without temperature treatment, 30% of the payload is released within the first 4 h. The timelapse images in Figure 4.4 show that very little MOF has entered the cell by this time, indicating that the calcein payload has been dumped in extracellular space. For a real drug, this would significantly reduce theraputic effect.

### 4.2.2 Loading MOFs with siRNA for therapeutic effects

To quantify the therapeutic effect of delivering drugs to cells using MOFs, we loaded NU-1000 MOFs with small interfering ribonucleic acids (siRNA).

A HEK-293 cell line with the T-REx Flp-In™ system was used to express the fluorescent protein mCherry. Under normal conditions therefore, these cells are fluorescent at 587/610 nm for excitation and emission wavelengths respectively.

An siRNA sequence was designed to cleave the mRNA molecule and suppress expression of the mCherry protein. The 21-nucleotide-long sense strand which was most effective at suppressing fluorescent emission was 5'-AAGGAGTTCATGCGCTTCAAG-3'. Scrambled siRNA, used as a negative control, did not cause any suppression of fluorescence. A combi-

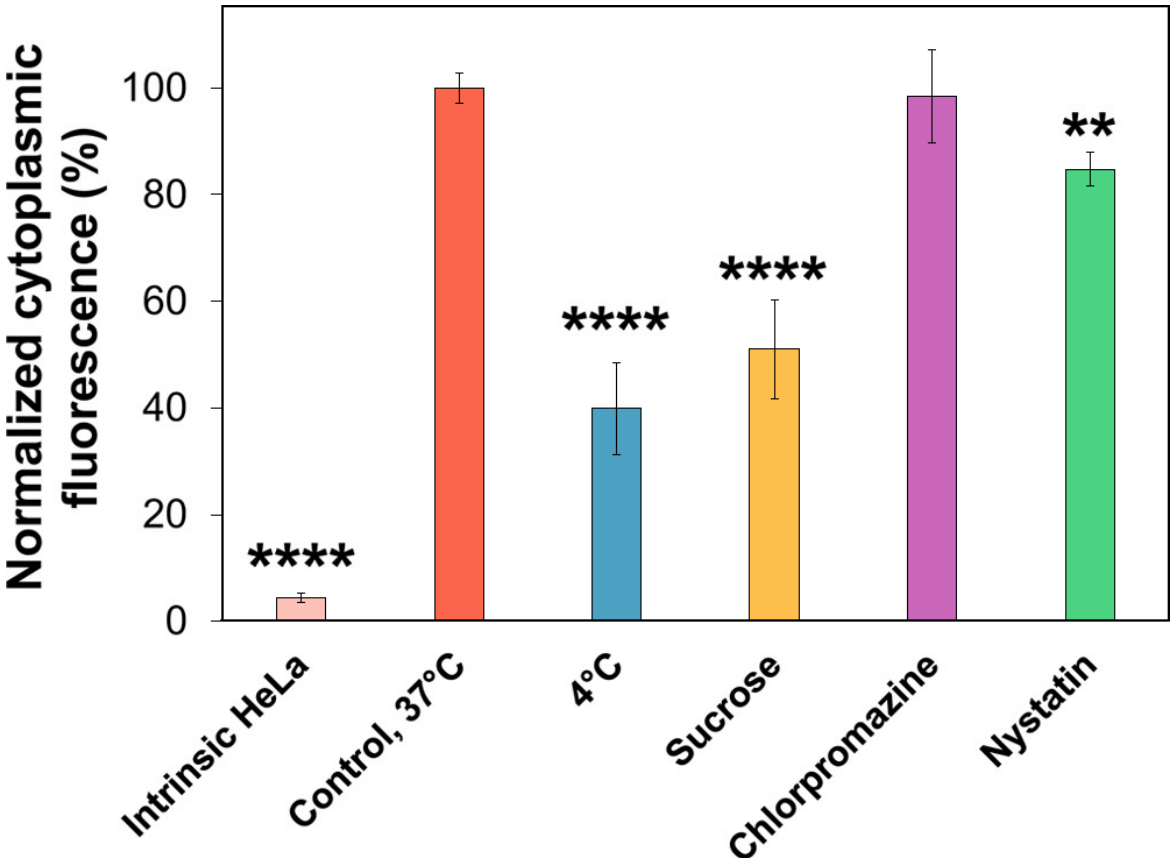



**Figure 4.5:** Cytoplasmic fluorescence was measured after incubating cells with NU-1000 MOF for 24 h under various conditions to measure how much MOF enters cells. The low bar after incubation at 4 °C shows that MOF uptake is an active endocytosis process. The low bar after incubation with sucrose suggests MOF is endocytosed by the clathrin-mediated pathway. $****: p < 0.0001$, $**: p < 0.01$.

nation of mCherry-fluorescent HEK-293 cells and mCherry-targeting siRNA creates a model drug delivery system which is quantifiable: the lower the fluorescence, the more effective the siRNA is at suppressing gene expression.

Without MOF, Figure 4.6a shows that siRNA is broken down by enzymes present in extra-celluar space - in this case, RNase. [cite] When the siRNA is loaded into MOF, it is protected from enzymes, demonstrated by the retention of a mark mark at 21 nt. The retention and broadening of low-$\theta$ peaks in Figure 4.6 confirms that the MOF is loaded inside the pores of the MOF, which protects it from enzymes.

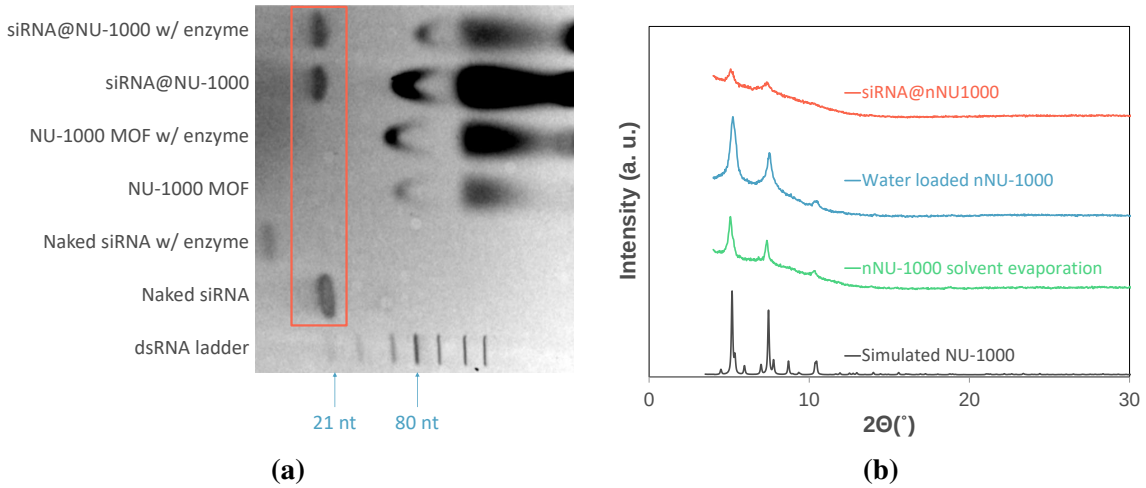


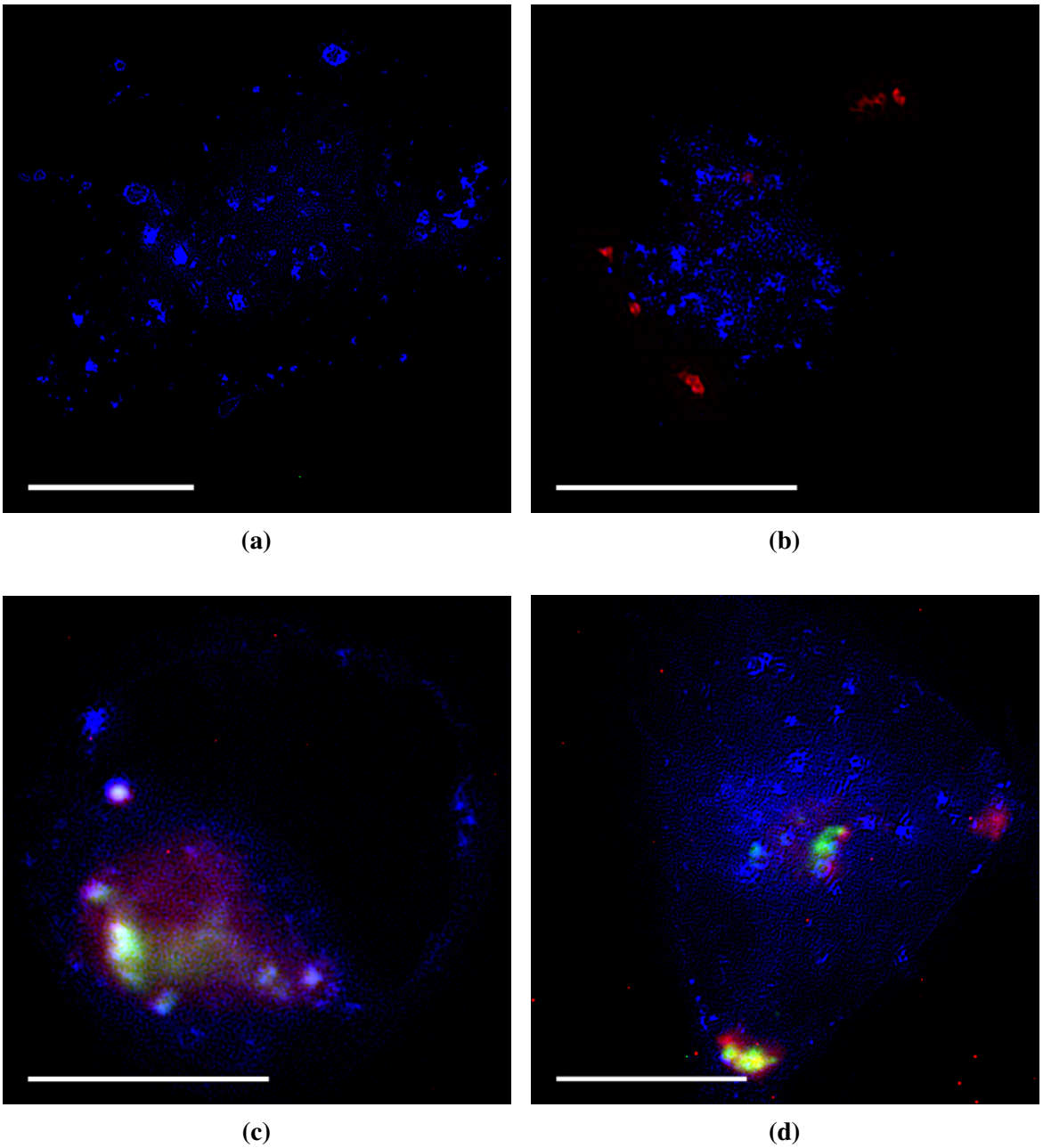

**Figure 4.6:** The gel in (a) has a mark at 21 nt when siRNA is present. When RNAase is added to the system, naked siRNA is broken down and not mark is present. Loading siRNA into NU-1000 protects the siRNA from degradation, and the mark at 21 nt is retained. PXRD plots shown in (b) confirm siRNA is contained within the pores of NU-1000, shown by broadening of the Bragg peaks.

To confirm that siRNA-loaded MOF could enter cells, we used LAG SIM to image cells, MOF, and siRNA. As seen in Section 4.2.1, this MOF is naturally fluorescent at 488 nm. The CellLight Early Endosomes-RFP BacMam 2.0 stain was used to visualise endosomes in cells. An Alexa Fluor 647 was tagged onto the 5' end of the siRNA sense strand. Multi-colour SIM imaging could then be used to assess cellular uptake of the MOF complex, and confirm that siRNA could enter the cell.

Figure 4.7 shows MOF uptake experiments under four conditions. MOF is coloured in green, endosomes in blue, and siRNA in red.

When naked siRNA was incubated with the cells, enzyme in the extracellular space broke down the siRNA. This can be seen in Figure 4.7a as an absence of red channel.

The current state-of-the-art for loading RNA into cells is using a transfection reagent such as Lipofectamine 2000. Figure 4.7b shows that this does indeed protect the siRNA



**(a)** **(b)**

**(c)** **(d)**

**Figure 4.7:** Images show endosomes coloured in blue, NU-1000 MOF in green, and siRNA in red, for HEK-293 cells incubated for 24 h under different experimental conditions. (a) shows cells incubated with naked siRNA; the absence of red signal suggests siRNA is degraded in extra-cellular space. (b) shows siRNA loaded into Lipofectamine 2000, which protects the siRNA from degradation, but does not appear to enter cells after 24 h. (c) shows MOF, siRNA, and endosomes colocalised, characterised by white spots; however despite being taken up by the cells, the siRNA has an insignificant therapeutic effect. Addition of an endosome release factor shown in (d) allows the siRNA-MOF complex to escape the endosome without being degraded, shown by yellow spots not colocalised with endosome. Scalebars are 10 µm.

from extracellular space, as red clumps of siRNA can be seen. However the siRNA does not appear to be located within the cell at the 1 h [CHECK] timepoint at which the cells were imaged.

Figure 4.7c shows MOF (green) and siRNA (red) colocalised and also within the cell boundary. Furthermore they are colocalised with the endosome stain (blue); the overlap of these three colours produces white. This corroborates with our previous findings from Section 4.2.1 that MOF is taken up by endocytosis.

Although it can be clearly seen in Figure 4.7c that MOF and siRNA are taken up by the cell, measuring the decrease in mCherry expression by flow cytometry revealed that simply entering the cell in this way is not enough to suppress gene expression. It was hypothesised that siRNA-MOF complexes were unable to escape the endosome when taken up by the cell, and so the siRNA was degraded by the acidic [?] environment of the endosome. [cite acidic?] To test this, endosome release factors were loaded into the MOF alongside the siRNA.

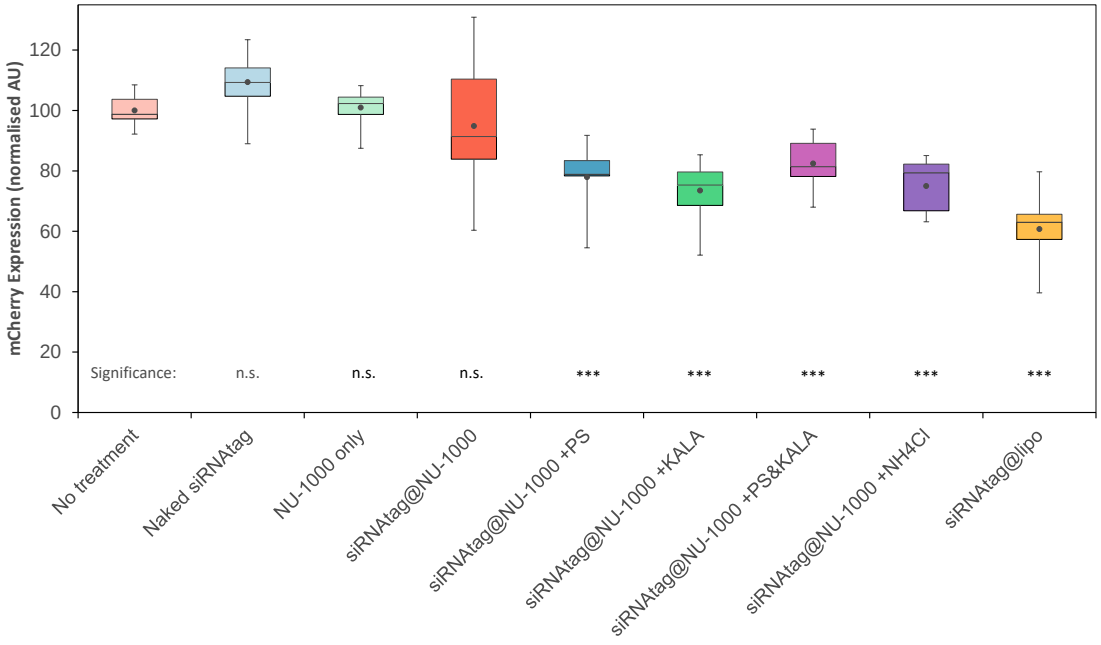



**Figure 4.8:** Using siRNA loaded in NU-1000 causes an insignificant decrease in the mCherry expression level. Under this condition siRNA is not able to escape the endosome to cleave the mCherry. Addition of endosome release factors, such as KALA, break down the endosome, allowing the siRNA to enter the cytoplasm and perform its intended function.

Three endosome release factors were used: Proton-Sponge®, an effective $H^+$ scavenger from Sigma-Aldrich [18]; KALA, a cationic amphipathic cell-penetrating peptide which binds oligonucleotides and disrupts cell membrane; and ammonium chloride, which has previously been shown to induce release of molecules from sub-cellular vesicles [19].

Figure 4.8 shows that, with the addition of any release cofactor, mCherry expression level was significantly reduced. The expected mechanism by which the siRNA is now able to suppress mCherry experssion is as follows: siRNA and the endosome release factor load into the pores of MOFs; these MOF complexes are taken up into the cell by endocytosis, and are initially located in endosomes; the MOF starts to break down due to the endosome environment, releasing the endosome release factor and siRNA; the endosome release factor breaks down the endosome membrane before the endosome can break down the siRNA, releasing the siRNA into the cytoplasm; and finally, the siRNA can cleave mRNA, silencing expression of mCherry.

The proposed mechanism for the successful gene silencing is confirmed by the SIM image in Figure 4.7d. In contrast to Figure 4.7b, yellow areas show that the siRNA-MOF complex is no longer colocalised with endosomes, showing that the cofactor successfully released it from the endosome into the cytoplasm. Furthermore, siRNA can be seen independently in parts of the cell, as red pixels not colocalised with any other colour.

### 4.2.3 MOFs for degrading mitochondria

As detailed in this chapter's introduction (Section 4.1), cells become cancerous when the Krebs' cycle stops working. Mitochondira 'switches off,' and so the cell continuously reproduces and does not undergo apoptosis, (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2567082/), leading to fatal tumours. This makes cancerous cells difficult to destroy. While chemotherapy treatments, such as radiation therapy, can kill the cancer cells, treatment is non-discriminatory and destroys all cells in the area, leading to unwelcome side-effects.

It has been shown that the surface chemistry of MOFs can be modified so that they are only taken up by cancer cells [20]. If the MOF is loaded with DCA, in this way it can kill off cancer cells by restarting the mitochondira Krebs cycle. The combination of DCA in a surface-modified MOF creates a targeted cancer therapy.

Inspired by our previous work loading a cofactor with siRNA, we investigated the effect of adding a mitochondria-targetting cofactor to the MOF alongside the DCA, with the aim of destroying more cancer cells with a lower concentration of MOF than previously shown. To get a better understanding of how the MOF was acting inside the cell, we used the LAG SIM to image HeLa cancer cells under different treatment conditions.

The UiO-66 MOF used for this study was not naturally fluorescent, so was also loaded with calcein to visualise it in the 488 nm excitation channel. Mitochondira were labelled with [something?] in the 561 nm channel, and the cell nucleus was visualised with [something?] in the 640 nm channel.

Sample SIM images are shown in Figure 4.9. We confirm, with 3D reconstructions obtained by optical sectioning, that the UiO-66 MOF is successfully taken up into HeLa cells. Furthermore, we can now visually observe the physical effect this treatment has on the mitochondria.

After 8 h incubation without treatment, the control condition in Figure **??** shows mitochondria in a healthy condition, with long stringy projections. After 30 min incubation with MOF and DCA, but no mitochondria-targeting cofactor, the mitochondria mostly retain this healthy shape. After the same period of time, but with [cofactor name?] loaded in the MOF as a cofactor, mitochondria have a noticably more circular shape. After 8 h of incubation without cofactor, the DCA does start to have an effect, with the mitochondria becoming circular; the effect after 8 h with cofactor was even more pronounced, with all labelled mitochondria now circular in shape.

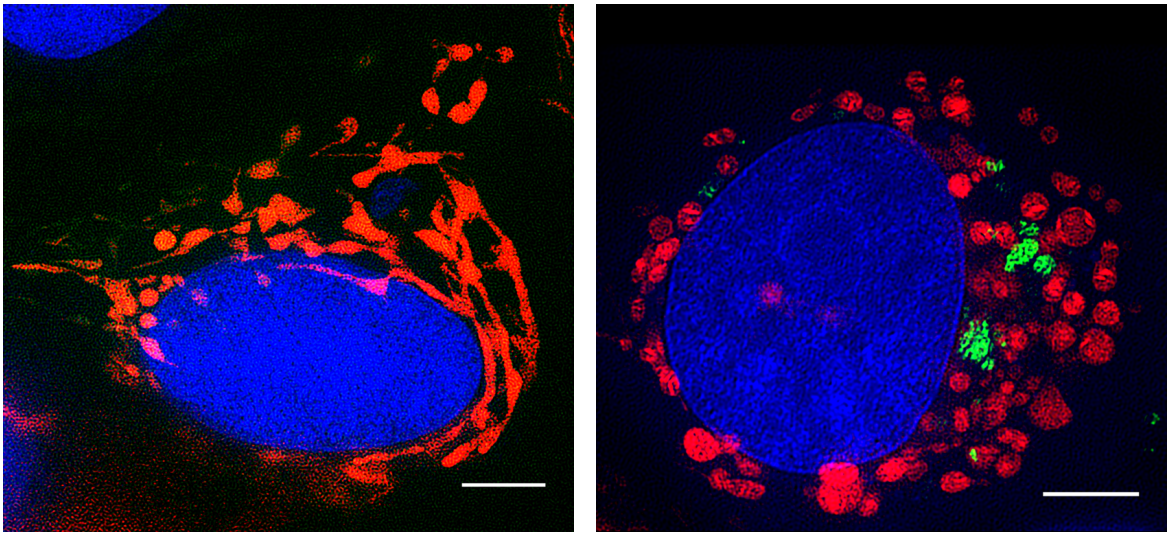
To confirm this visual inspection of mitochondrial shape quantitatively, analysis was performed in Cell Profiler [21]. Image stacks of 54 cells were captured and reconstructed with optical-sectioning LAG SIM, and from these images 6500 mitochondria were extracted using the [??] plugin, as shown in Figure **??**. The ellipticity of each classified mitochondria was then calculated with the [ROI statistics?] plugin, to produce a large list of mitochondria ellipticity under the different treatment conditions shown in Figure 4.9.

The mitochondria ellipticity statistics were analysed with a one-way ANOVA test in Graphpad Prism [22]. Figure 4.11 shows that the [chemical] cofactor causes a significant change in the ellipticity of mitochondria. This result correlates with findings from FACS measurements that MOF loaded with the [chemical] cofactor reduces cell viability more than MOF loaded with DCA alone.

## 4.3 Conclusion

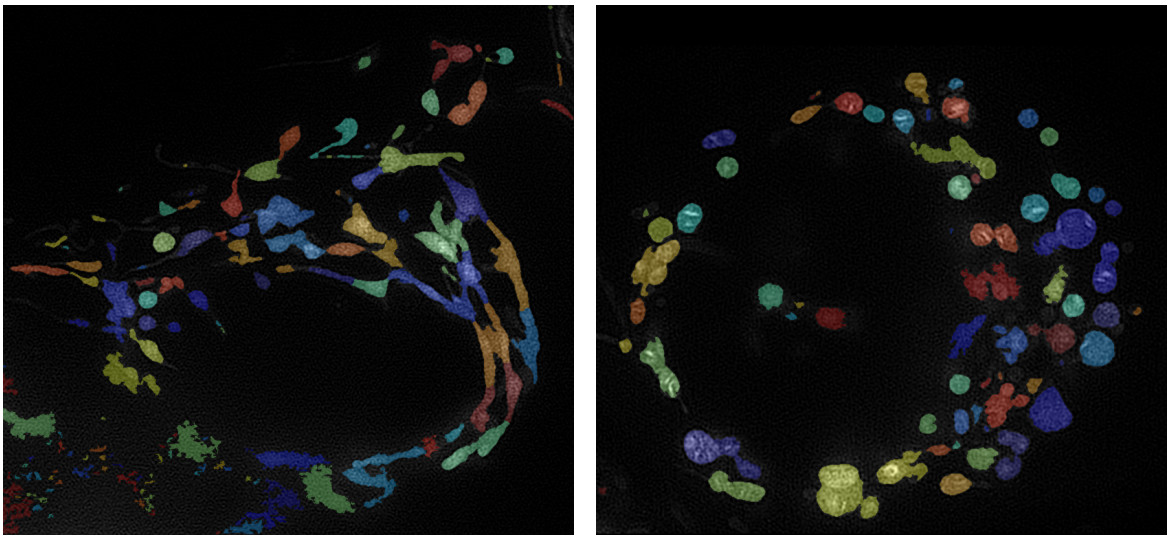
The three experiments presented in this chapter utilise MOFs as a drug delivery vehicle for cancer treatment. We have shown through a variety of techniques that MOFs can be successfully loaded with therapeutic molecules which would otherwise be degraded in extra-cellular space, and subsequently can delivery these drugs to the cell.

Furthermore, we have observed a significant decrease in cancer cell viability when MOFs are used as a delivery vehicle. Two techniques, based on delivering siRNA for gene silencing and DCA for mitochondrial degradation which restarts the Krebs' cycle, have emerged as successful treatment strategies on cancerous [cell type?] cells. Concerns regarding the burst release effect, where much of the MOF's therapeutic payload is released before the MOF has



**(a)** **(b)**

**Figure 4.9:** (a) shows healthy mitochondira in untreated cells. Treating the cells with DCA-loaded UiO-66 MOF, shown in (b), causes mitochondria to lose their long, elliptical shape.



**(a)** **(b)**

**Figure 4.10:** The Cell Profiler[**?** ] protocol identifies and segments mitochondria in the cell. The ellipticity of segments is calculated to provide statistics about the change in mitochondria shape caused by DCA-loaded UiO-66.

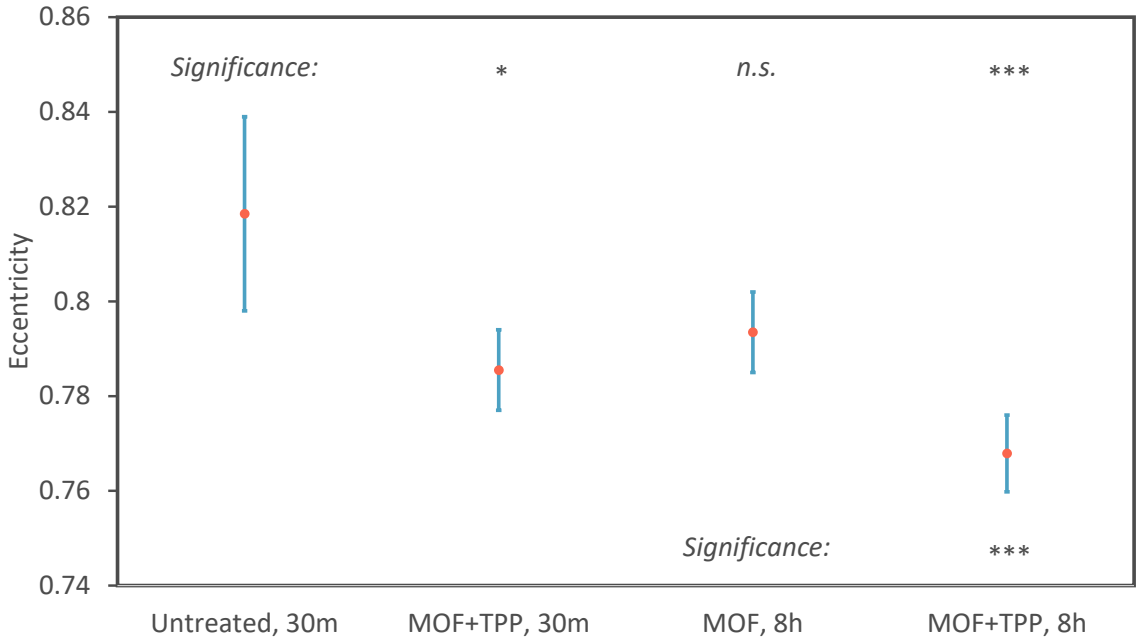



**Figure 4.11:** Mitochondria are affected by the DCA-UiO-66 treatment, making them more circular in shape. The reduction in eccentricity is not significant without addition of TPP as a mitochondira-targetting cofactor. $* : p < 0.05$, $*** : p < 0.001$

entered the cell, have been addressed with a temperature treatment strategy which induces pore collapse to delay release.

The utilisation of optical sectioning provided by the LAG SIM has allowed both visual confirmation of biological hypotheses and large datasets for statistical analysis.

Throughout the chapter, 3D reconstructions have shown that MOF is able to pass the cell membrane and enter cells; an endocytosis study showed that this occurs through the clathrin-mediated pathway. Inside the cells, MOFs are colocalised with endosomes; this can prevent the MOF's payload having a therapeutic effect without addition of an endosome release factor. When Proton-Sponge®, KALA or $NH_4Cl$ were loaded into the MOF as cofactors, a significant decrease in cell viability was measured, and furthermore SIM images confirmed that MOF was no longer colocalised with endosomes.

LAG SIM's fast imaging speed allows a large number of cells to be imaged in a single session. This firstly has the advantage that cells do not spend long on the microscope stage, where non-optimal conditions can cause apoptosis[check] of cells; but rather cells can be left in the incubator for most of a 24-hour time course study. Secondly, meaningful statistics can be gathered when a large number of cells are imaged, allowing visual observations to be rendered as statistically significant results.

Overall this chapter has demonstrated the LAG SIM's capability for facilitating hypothesis-driven research supporting and confirming measurements from other techniques and providing visualisation to develop pioneering treatments in the fight against cancer.

## 4.4 Future work

Using MOFs as a drug delivery vehicle has been demonstrated as an effective treatment for cultured [cell type] cells. The next step towards developing a drug that can be used to treat humans is to assess the treatment on a model organism. Work is now underway to apply the siRNA-MOF treatment to mice, to observe the effectiveness of the therapy at organism level and assess any toxicity issues.

This chapter has presented just 3 MOFs, NU-901, NU-1000, and UiO-66, which have desirable properties for use as a drug delivery vehicle. However our previous work [**?** ] has shown that due to the tens-of-thousands of MOF structures possible, selecting the perfect MOF is non-trivial. We have previously built an online tool for screening a database of drugs to select the one most-suited to a given application. Though the tool, accessible at https://aam.ceb.cam.ac.uk/mof-explorer/, is currently set up for optimising gas storage solutions, adding biologically relevant axes such as toxicity and payload release time could reveal an even better MOF than those already investigated.

# Chapter 5

# ER: Endoplasmic reticulum network dynamics

"I have also just seen some of the data from Meng on lysosomes and ER tubules! I have never seen anything like this. This is only possible through your amazing efforts."

*Gabi Kaminski, email 10-05-2018*

## 5.1 Introduction

### 5.1.1 Role of the Endoplasmic Reticulum

The Endoplasmic Reticulum (ER) is an interconnected network of tubular pipes and reservoirs present in most types of eukaryotic cells. Its function includes the biogenesis of membrane and secretory proteins, and delivery of these functional solutes throughout the cell. Since the ER network extends to the cell periphery, molecules can be transported to distant sites in the cell.

When demand is greater than capacity of ER to fold proteins, ER stress can occur. ER stress can also be caused by proteins that cannot properly fold inside the ER. A number of signal transduction proteins in the cell monitor proteins secreted by the ER, and if an imbalance is detected the ER stress response begins.

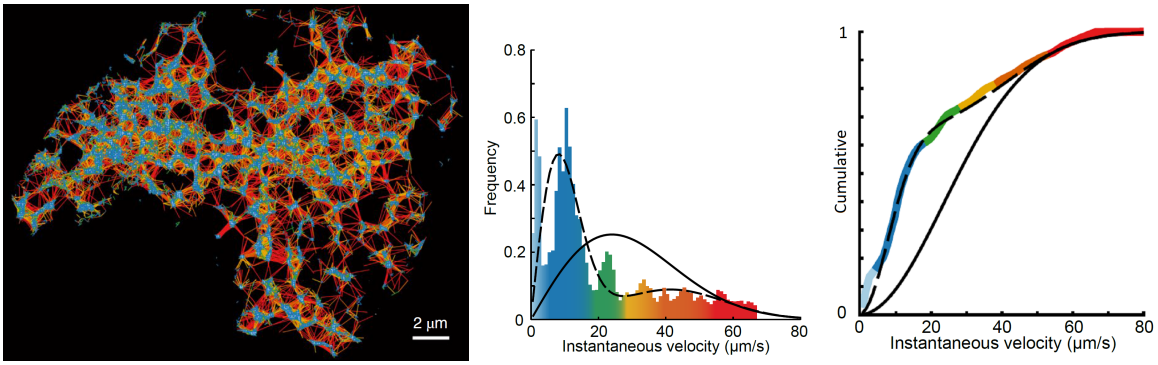
As the largest organelle in a cell, recent studies have shown ER forms membrane contacts with all other cellular organelles, with a higher number of contact sites than any other organelle-organelle interaction [23, 24]. Since ER is found in almost all human cells, it is therefore not surprising that a wide range diseases have been associated with ER stress, including neurodegeneration, atherosclerosis, type 2 diabetes, liver disease, metabolic diseases, and cancer [25].

The rate at which luminal ER content is distributed affects the efficiency of ER-mediated intracellular connectivity [26, 27]. Understanding the fundamental processes of ER protein mobility is therefore vital to curing diseases caused by mutations in ER proteins.

### 5.1.2 Flow in the ER

ER proteins are delivered around the cell by moving through a network of tubes which make up the ER. It has been shown with fluorescence recovery after photo-bleaching (FRAP) that protein mobility is an ATP-dependent process - that is, proteins move slower when ATP is depleted [28, 29].

More recently, we have shown with single particle tracking that molecules in the ER do not move with diffusive motion. Rather, they have a fast, directed velocity as they flow along ER tubes, but move with slow, diffusive motion at nodes where 3 or more tubes connect, as shown in Figure 5.1a.



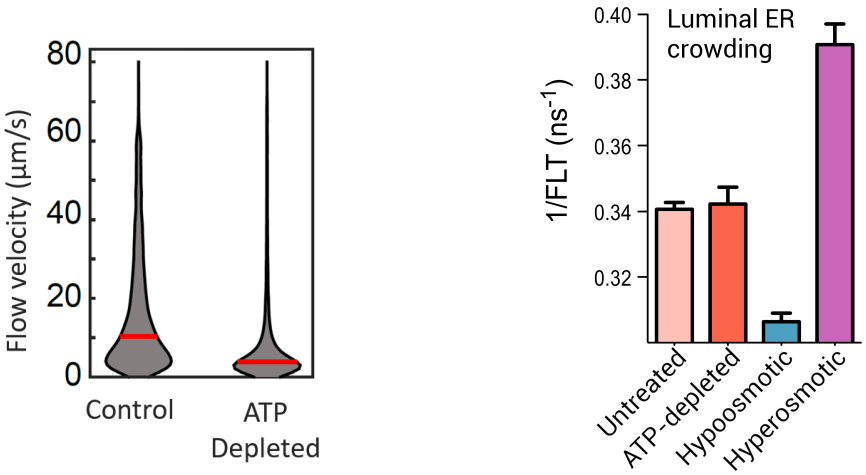

**(a)** **(b)** **(c)**

**Figure 5.1:** (a) shows tracks of single particles in the ER, colour-coded according to the scale in (b) and (c). Fast, directed movement occurs in tubules, and slow movement occurs in nodes. A diffusive model, shown by the black line in (b) and (c) does not fit the distribution of velocities; a much better fit is provided using a model which combines directed and diffusive flow, shown by the dashed line. Graphs were created from n = 80 563 tracks.

Figure 5.1b shows that applying data from 80 563 single particle tracks to a mathematical model with both diffusive and flow terms provided a better fit to experimental data than a model with diffusive motion only. This is particularly convincing in the cumulative frequency distribution shown in Figure 5.1c.

Consistent with previous studies, tracking directed flow revealed that depleting ATP causes a 4$\times$ decrease in mean flow velocity, shown in Figure 5.2a.

It was suggested that the decrease in protein mobility could be explained by an increase in the resistance to motion due to crowding of luminal ER proteins. To counter this argument,




**(a)** **(b)**

**Figure 5.2:** (a) shows that mean flow velocity decreased 4× when ATP was depleted from the cell. A FRET probe was used to assess crowding of luminal ER proteins, shown in (b). Although hypoosmotic conditions caused a decrease in crowding and hyperosmotic conditions caused an increase in crowding, due to expansion and contraction of the cell size respectively, depleting ATP did not change crowding of luminal proteins, countering the argument that a decrease in flow velocity is due to crowding of proteins.

a FRET-based probe was used to measure ER crowdedness. Treating with a hyperosmotic buffer induces cell shrinking, which increases crowding, decreasing the fluorescence lifetime. Conversely, a hypoosmotic buffer induces cell swelling, decreasing crowding, and the fluorescnce lifetime of our sensitive FRET probe increases. When depleting ATP, there is no change in lifetime; therefore we dismiss the hypothesis that the change in protein mobility is due to increased crowdedness, but relies on an active, ATP-dependent mechanism.

### 5.1.3 Structure of this chapter

As part of a collaboration with the authors who first observed directed ER flow, I investigated potential mechanisms for flow. Utilising the unique capabilities of structured illumination microscopy (SIM) and computational analysis, evidence is presented that ER luminal flow is caused by pinching of the ER membrane.

The Results and Methods section which follows details 3 key contributions which I personally made to the published study:

1. Measuring dynamic network arrangement with SIM imaging and computational analysis

2. Imaging tubule pinching with TIRF-SIM and extracting statistics regarding pinching frequency
3. Computational simulation of tubule pinching to investigate its biological advantage over diffusive transport

Conclusions and future work extending the findings presented here are given at the end of the Chapter. Further details of the full study can be found in *Nature Cell Biology*.

## 5.2 Results and methods

### 5.2.1 Network arrangement

Utilising the high-speed imaging capability of the LAG SIM, we observed that the ER is a dynamic network, with nodes and tubular endpoints constantly moving to rearrange the network. Imaging was performed at a raw frame rate of 99 Hz, corresponding to a reconstructed frame rate of 11 Hz.

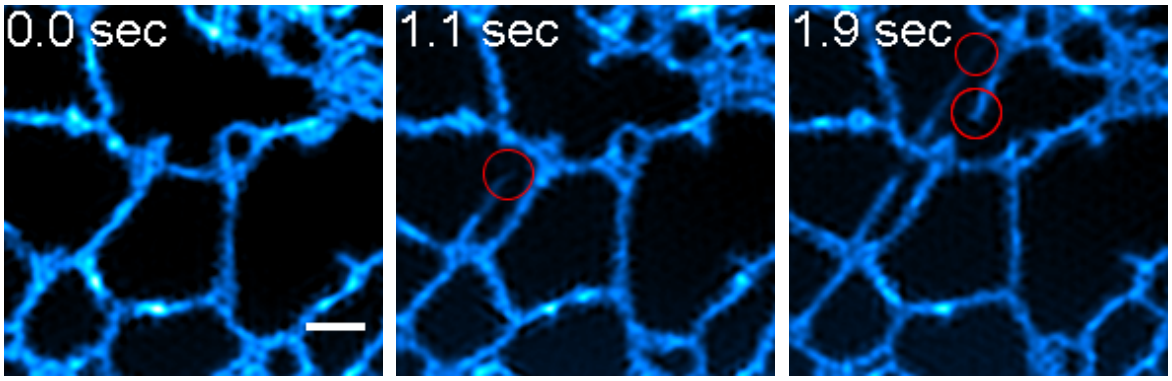
For optimal imaging conditions, COS-7 cells were selected for the investigation. These characteristically flat cells, with a maximum height of $\sim$10 µm, meant that the whole ER network could be imaged by LAG SIM in TIRF mode, which physically removes out-of-focus light.

The ER network was labelled with ER-Tracker$^{TM}$ Green, a fluorescent dye from ThermoFisher excited at 488 nm. This allowed SIM to achieve its highest resolution imaging of 85 nm.

To verify our observations, and ensure relevance to human pathology, we also used HEK-293 cells labelled with tetramethylrhodamine (TMR) attached to luminal ER proteins with HaloTag® technology. These thicker cells were imaged with optical sectioning LAG SIM, since TIRF only showed small fragments of ER close the coverglass. Images in this chapter are described as COS-7-ERT, COS-7-TMR, and HEK-293-ERT, and HEK-293-TMR to describe COS-7 and HEK-293 cells labelled with ER-Tracker$^{TM}$ Green or TMR.

Figure 5.3 shows a small area of the ER network in COS-7-ERT cells. The network's tubules and nodes are clearly visible, as well as large and small loops in the network and sheet structures. The time series shows that the ER network undergoes fast rearrangements, with a number of tubules growing and shrinking and nodes rearranging at any given time. The speed of this process highlights the need for the fast, high-resolution imaging technique provided by LAG SIM.

To draw statistics describing the network from the images, the following steps were performed:




**(a)** **(b)** **(c)**

**Figure 5.3:** The ER network is in a state of dynamic equilibrium, with a small number of nodes being created and destroyed at any given time to cause network rearrangement. Images are of COS-7-ERT; scalebar is 1 µm.

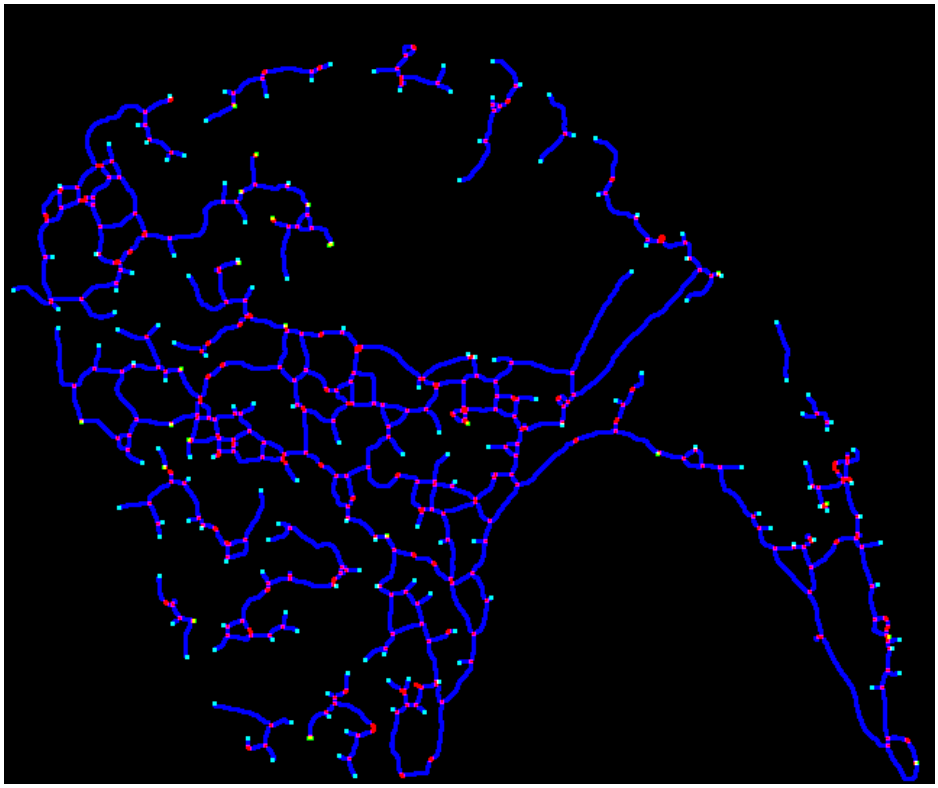
1. Generate a binary image of the network for each frame
2. Skeletonise the network, making it one pixel wide
3. Classify nodes, endpoints, and tubes with MATLAB `bwmorph` functions
4. Extract statistics from labelled binary image stacks

A common method for generating a binary image is thresholding, where all pixels with a value above a certain threshold are classified as `true` and all those with values below the threshold are assigned `false`. Whilst this scheme was effective at differentiating ER from background in the thin COS-7 cells, non-uniform background light in HEK-293 cells, visible in Figure 5.4a, meant that a simple binary threshold produced the poor network extraction shown in Figure 5.4b. Futhermore, the binary threshold required for each image set varies depending on exposure time, laser power, and expression of the fluorescent label that particular cell. Therefore a single value threshold was not appropriate to generate binary images.

Instead a machine learning technique was used to classify ER pixels from background pixels. Using the WEKA Segmentation plugin for ImageJ [30], a set of training data was generated by manually classifying ER pixels and background pixels. The Adaboost algorithm was used to quickly refine the classification, leading to a reliable and automated adaptive thresholding method. The output of WEKA Segmentation shown in Figure **??** successfully extracts the ER network even with the unideal imaging properties of HEK-293 cells.

After binary images were skeletonised, a classified network map, as shown in Figure 5.4d, was generated with MATLAB for every frame of every image set. From this, various statistics could be extracted, including tubule length distribution, tip growth rate, and node movement.

(a) (b) (c)



(d)

**Figure 5.4:** (a) shows the background of ER images is not uniform in some images. A simple threshold for separating ER from the background, shown in (b), does not work well on HEK-293 cells due to the variations in background intensity. A machine learning technique, implemented with the WEKA segmentation plugin for ImageJ, identifies the ER and background more reliably, as shown in (b). A MATLAB script was applied to the binary image to classify ER tubes, nodes, and end-points, labelled in (d) as blue, green, and red respectively. Tracking of nodes and endpoints, and analysis of tubule length distribution, could then be performed on the network. Images are of HEK-293-ERT; scalebar is 5 µm.

The distribution of tubule lengths, shown at at various time points in Figure 5.5, does not change over time. This shows that the ER is in a dynamic equilibrium. Although there are local changes in structure and connectivity, the overall network properties do not change.

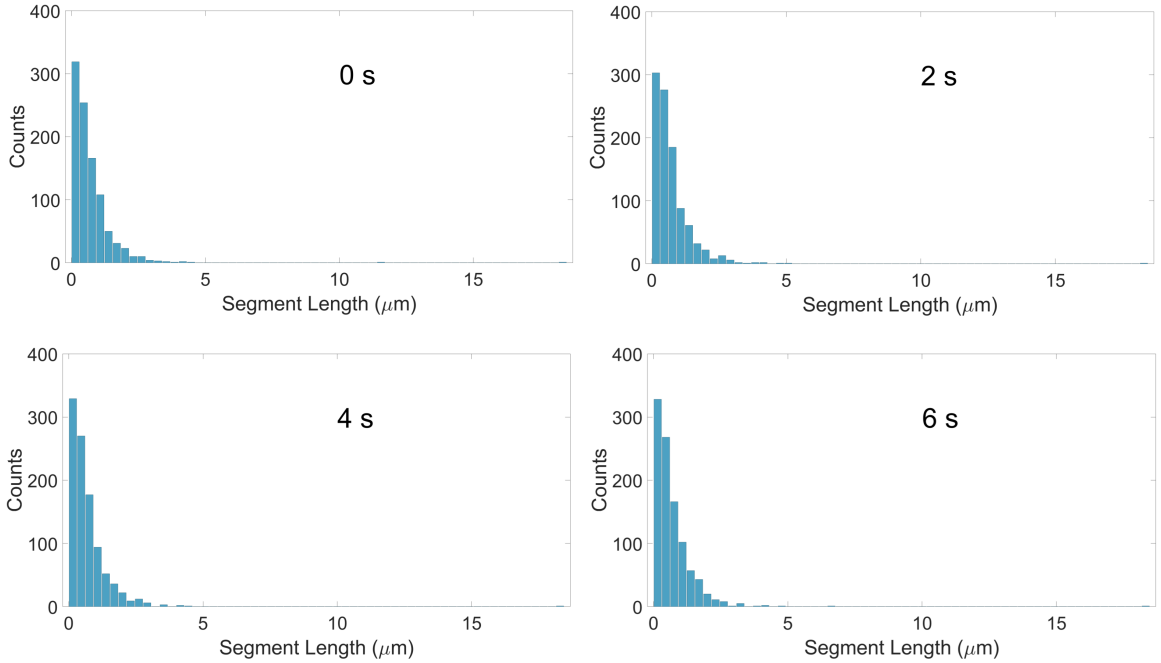



**Figure 5.5:** Although individual tubules are growing and shrinking at any given time, as shown in Figure 5.3, the distribution of tubule lengths does not vary over time, showing that the ER is in a dynamic equilibrium.

Movement of end points and nodes was initially suggested as a hypothesis for the distribution of particle movement inside the ER. However tracking of nodes revealed an average speed of just $(0.39 \pm 0.42)\,\mu\mathrm{m\,s^{-1}}$, around $60\times$ slower than the $22.9\,\mu\mathrm{m\,s^{-1}}$ mean speed of particles found in the single particle tracking analysis of Figure 5.1b. Furthermore, we calculated that just $(0.14 \pm 0.04)\,\%$ of tubules were growing at any given time. We conclude that tubule growth and node movement have a negligible contribution to particles' diffusion and flow velocities.

Experiments are now underway to understand the biochemistry behind network rearrangement, which is described in more detail in the Future Work Section 5.4.

### 5.2.2 Tubule pinching

The motion of luminal proteins suggested the possibility of nanoperistalsis-like propulsion [31], attainable by tubule contractions. Using the LAG SIM in TIRF mode, we obtained images at 11 Hz, with a resolution of 85 nm, and were able to observe pinching of the tubule.

ER tubules are 60 nm to 100 nm [cite!] in width, so observing pinching requires a microscope that can image beyond the diffraction limit. When ER tubules pinch, their width decreases to below the resolution limit of SIM; however this can still be observed as a decrease in intensity, confirmed by the simulation shown in Figures 5.6a-f.

To clearly visualise pinches, the 'Edges' colourmap was used in ImageJ, highlighting lines of constant intensity. Note that, whilst this allows pinching events to be accurately counted, there is not enough resolution in SIM to measure the width of the tubule when it is pinched. Nevertheless we believe that SIM is still the only technique capable of capturing these dynamic events, due to its combination of spatial resolution and imaging speed.

Counting the number of pinches in each tube over time gave the mean frequency of pinches as $(1.70 \pm 0.48)$ pinches/s in COS-7 cells. Figure 5.6i shows that when ATP was depleted, the frequency decreased 3.9×, to $(0.44 \pm 0.41)$ pinches/s. This decrease in pinching frequency is very similar to the velocity decrease observed with single particle tracking, suggesting that there is a strong link between tubule pinching and directed motion in the ER.

Further evidence for tubule pinching was gathered when the cell was artificially placed under excessive stress. Whilst acquiring time series videos of the ER on the LAG SIM, we occasionally observed a reversible blebbing behaviour of the ER. This phenomenom, shown in Figure 5.7, was particularly evident when imaging samples labelled with TMR fluorescent dye with the 561 nm laser power set to 95 mW or more, equivalent to [?? at the sample]. The same behaviour was observed when calcium manipulation [32] was performed on the cell. Notably, the ER recovers its original shape after the stress is removed, demonstrating the ability of ER tubules to undergo reversible pinching.

Furthermore, 3D images of a fixed ER network, obtained using optical sectioning SIM, reveal tubules pinched in 3D. This is available to view interactively with FPBioimage, at https://fpb.ceb.cam.ac.uk/ER. Despite tubules' size being at the resolution limit of SIM, the evidence from the different experiments presented here provide confidence that pinching is not an imaging artefact.

### 5.2.3 Graphical simulation of ER protein flow

In order to visualise the concept of luminal flow, and to understand its purpose, I wrote a 2D Unity program modelling an ER network with labelled proteins.

An ER network was automatically modelled with the following steps:

1. Create binary image of network, with method outlined in Section 5.2.1
2. Create a collision boundary from binary image with an edge-tracing algorithm
3. Create a simplified skeleton network from binary image

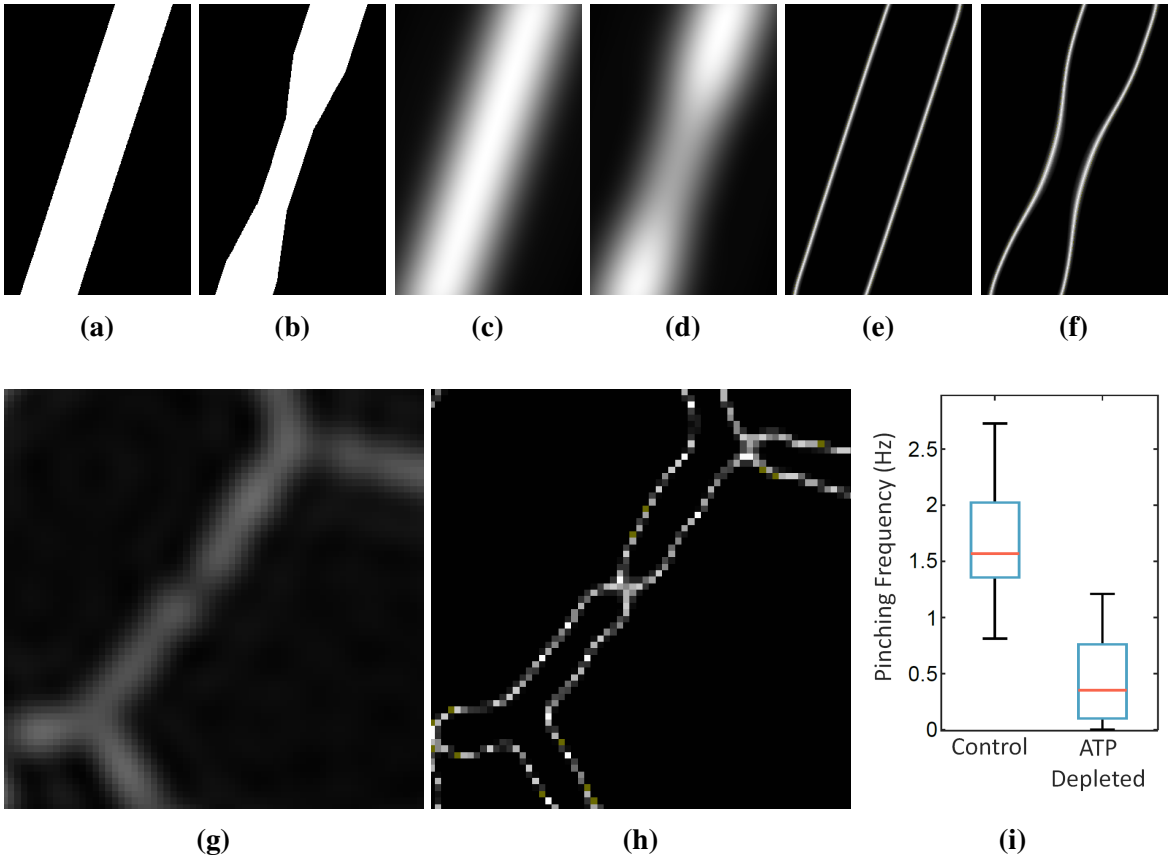
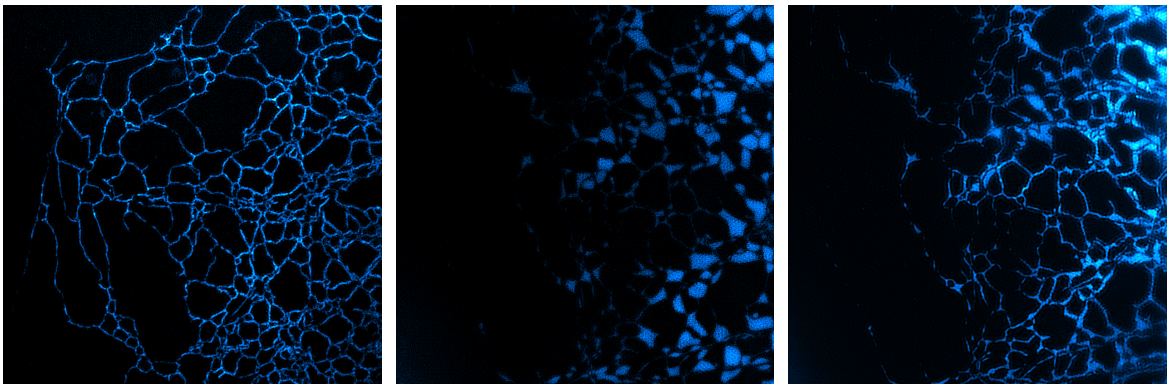



**Figure 5.6:** A simulation in (a-f) shows that pinching events can be observed even when the pinch width is below the resolution limit of SIM. (a) and (b) show an unpinched and pinched tubule at full simulation resolution, with a width of 80 nm. (c) and (d) show the tubule after convolution with a SIM-resolution point-spread function; (e) and (f) show that using the 'Edges' colourmap clearly shows the pinch. (g) shows experimental data of a tubule pinching in COS-7-ERT cells, highlighted in (h) by the 'Edges' colourmap. (i) shows that ATP depletion reduces pinching frequency 3.9×, showing that pinching is an ATP-dependent process.



(a) (b) (c)

**Figure 5.7:** When the cell shown in (a) is strongly perturbed from normal conditions, for example by phototoxicity or calcium manipulation, extreme pinching causes blebbing in the ER. (b) shows the cell after imaging for 30 s with 95 mW 561 nm laser light. When the perturbation is removed, the ER recovers its normal shape; (c) shows the same cell after 10 min of recovery time. This experiment confirms the ability of ER to undergo reversible pinching.

4. Spawn 'pinch points' on the skeleton, aligned with the skeleton direction and the width of the network at that point

The program will accept any binary network image as an input, and will perform these steps automatically using custom C# implementations of the Moore-Neighbourhood tracing algorithm for tracing tubule edges [cite], the Zhang-Suen algorithm for skeletonisation [cite], and the Ramer–Douglas–Peucker algorithm for simplifying the tubule skeleton and edges [cite].

Dynamics are modelled in the network at a particle level. Each particle which is spawned in the network has a script attached to it controlling its diffusive motion and its flow velocity when in the vicinity of a pinch.

Diffusive motion was modelled as Brownian motion. [**? ?** ] After travelling a random distance sampled from a distributed based on the mean free path, particles randomly change direction, as though collided into by an invisible particle [33]. Diffusion properties are set in the simulation with mean free path $\delta$, time between collisions $\tau$, mean velocity $v$, or the diffusion coefficient, related by $D = \frac{\delta^2}{2\tau} = \frac{v\delta}{2}$.

Pinching of the ER tubules simulated a force on particles in the vicinity of the pinch point. As the tubule contracts, this force acts away from the pinch, in a direction along the tubule, and decays exponentially over time. When the pinch point relaxes, the force acts in the opposite direction. The magnitude of the force is an adjustable parameter, to control overall energy in the system.

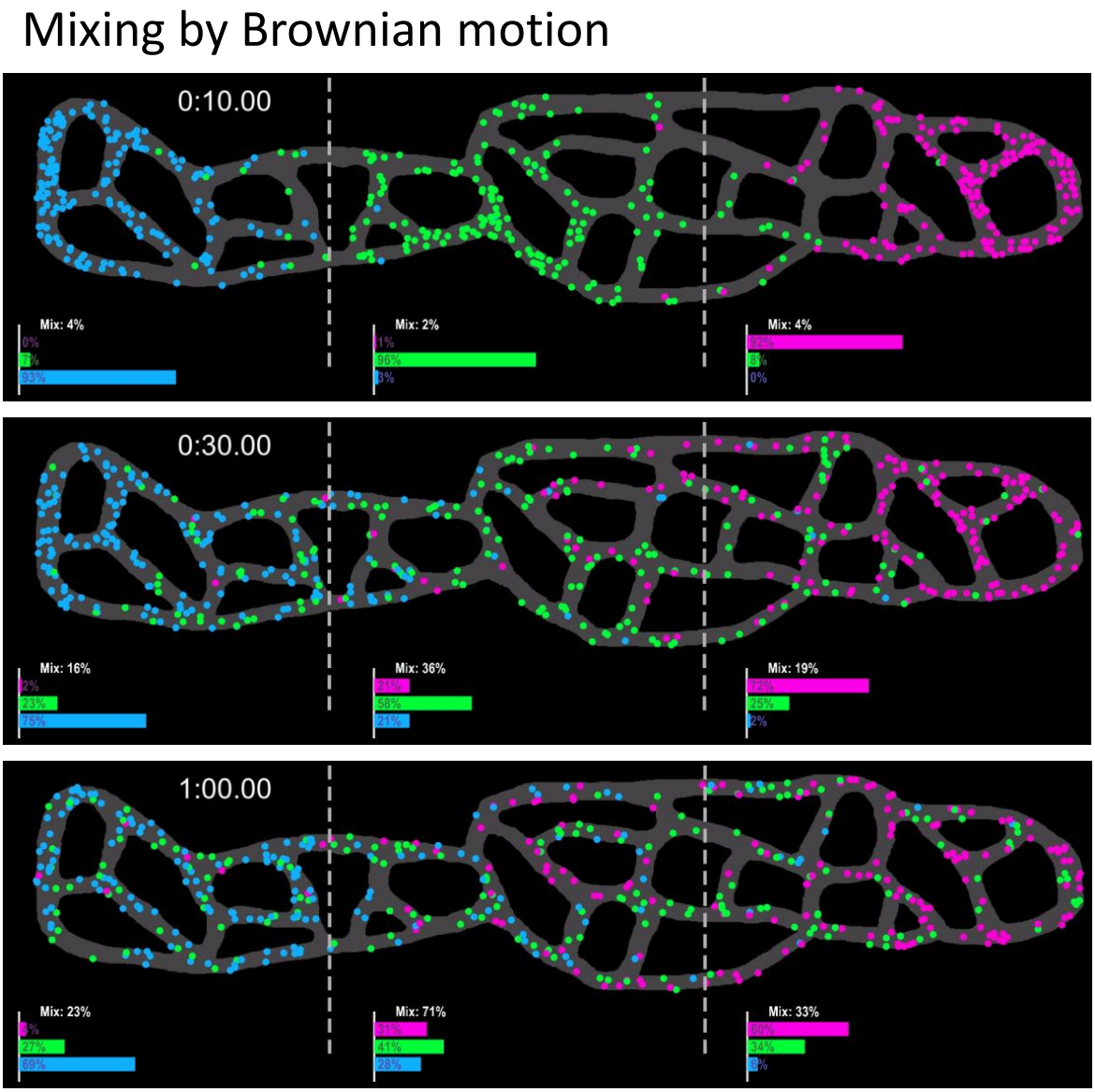



**Figure 5.8:** After 10 s of mixing by Brownian motion, most particles are still in the third of the network they started in. After 1 min of mixing, the middle section is 71% mixed, although the two edge sections are just 23% and 33% mixed, showing that particles do not transverse the network quickly under Brownian motion alone.

Inelastic collisions were modelled with the NVIDIA PhysX engine. Coefficient of restitution can be adjusted for each material, such that a collision with the ER membrane is different to a particle-particle collisions.

With this physics model, it was possible to simulate different scenarios to understand experimental observations. Firstly, the model was executed without any pinching, simulating purely diffusional motion. In the second scenario, pinching physics was turned on, such that

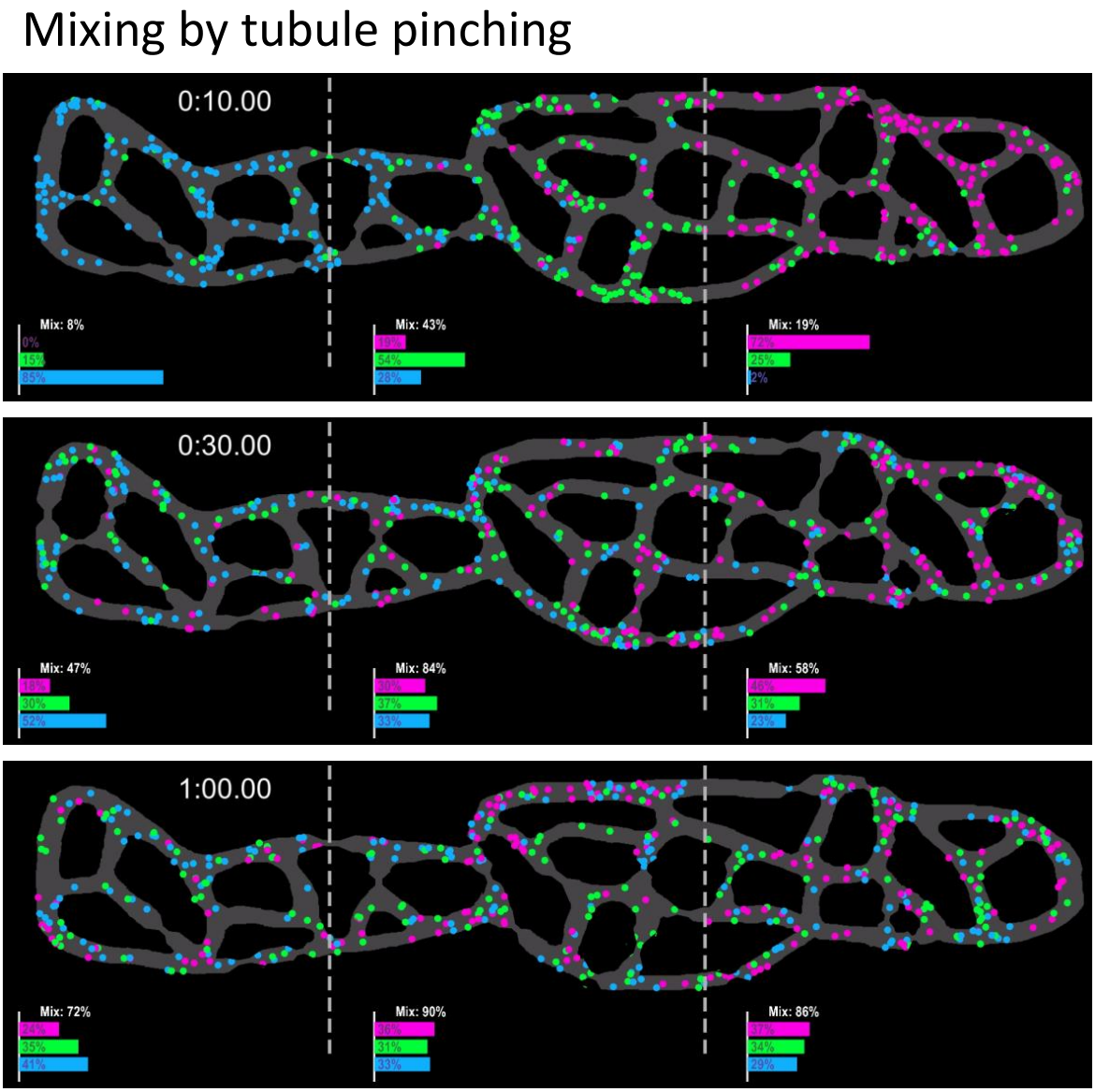



**Figure 5.9:** When the simulation is executed with tubule pinching, mixing occurs much faster than by Brownian motion despite the same average particle speed.

particles in the vicinity of a pinch point experienced a pushing force as the tubule pinched or a pulling force as it relaxed. The parameters were tuned to ensure an equal mean particle speed in both scenarios for a fair comparison.

The two scenarios were compared by measuring the distribution of particles in the network over time, as shown in Figures 5.8 and 5.9. Particles starting at the left hand side of the network were labelled blue; particles in the middle, green; and particles starting on

the right pink. By counting the number of particles in different areas of the network, a quantitative description of network mixing can be obtained.

Comparing Figures 5.8 and 5.9 shows that pinching provides the ER a faster way of moving particles stochastically around the network. Under Brownian motion alone, after one minute of simulation pink particles which started on the right-hand-side of the network make up just 6% of particles in the left-hand third of the network. Conversely with ER pinching dynamics, after one minute pink particles make up 24% of particles on the left-hand-side. Pinching dynamics allow particles are able to travel greater distances in the network in less time, even with the same average speed as the diffusional model.

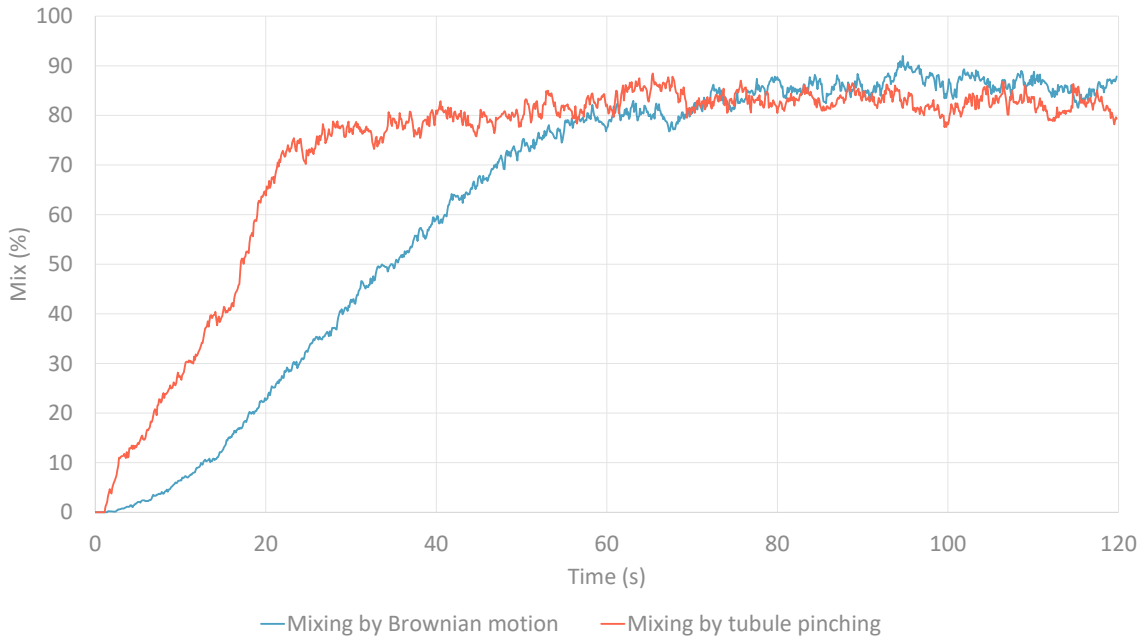



**Figure 5.10:** Mixing percentage for the middle third of the ER simulation was plotted against time for the two conditions. Mixing occurs about 3× faster with tubule pinching, despite the same average particle speed, allowing particles to reach remote parts of the network faster.

Furthermore, particle motion with ER pinching results in homogeneous mixing in a shorter space of time. After 30 s of simulation with pinching, the middle section of the graph is 84% mixed - where mixing percentage is measured as the sum of the second- and third-most abundant particle colour in an area divided by 2×the most abundant particle colour. Under pure Brownian motion, however, the middle section is only 36% mixed; and the edge sections even less so. Figure 5.10 shows how mixing in the middle section develops over time; pinching allows mixing to occur 3× faster than diffusive motion alone. Overall, particles are mixed faster with ER pinching.

Pinching events are uncoordinated, so that mixing with pinching is stochastic rather directed, which agrees with experimental observations. Particles gain enough speed from a pinch to move away from its influence when the pinch relaxes, so that particles do not stay stuck in one area but are able to freely mix in the network.

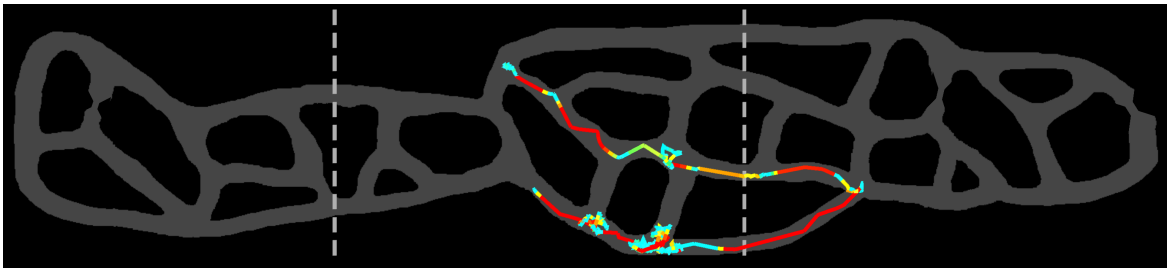


**Figure 5.11:** A single particle was tracked in the simulation, and its velocity recorded using the same colour scale as Figure 5.1. The simulated model is in good agreement with experimental data.

To prevent too much energy being added to the network from multiple pinching events, collisions were inelastic and drag was added at network junctions to slow the particles. Parameters were tuned to ensure the same average particle speed in the two mixing scenarios for a fair comparison. A plot of a single particle's trajectory through the network is shown in Figure 5.11, which is consistent with the experimental data in Figure **??**.

Understanding the reason for ER pinching allows us to associate diseases with ER malfunction. In cells with long projections, such as motor neurons, transporting luminal protein to remote parts of the cell quickly could be vital for correct operation.

## 5.3 Conclusion

In the *Nature Cell Biology* paper, we present evidence that the movement of proteins in the ER is not diffusive, but also has an ATP-dependent directed-flow component [34]. By utilising the fast high-resolution imaging capabilities of LAG SIM, as well as computational simulation and analysis, this chapter has suggested that contraction of ER tubes is responsible for luminal flow.

Imaging at 11 Hz with 85 nm resolution revealed ER networks in a dynamic equilibrium, constantly growing, shrinking, and rearranging. Network analysis and tracking showed that this is not responsible for flow, however.

The capacity of ER to undergo reversible contractions was revealed when high laser power was used to image ER labelled with TMR, which caused blebbing in the network. Rapid tubule pinching was revealed when re-examining individual segments of ER captured

in TIRF-SIM mode. The size and speed of the pinching phenomenon means that high-speed TIRF-SIM is currently the only technique able to visualise these events.

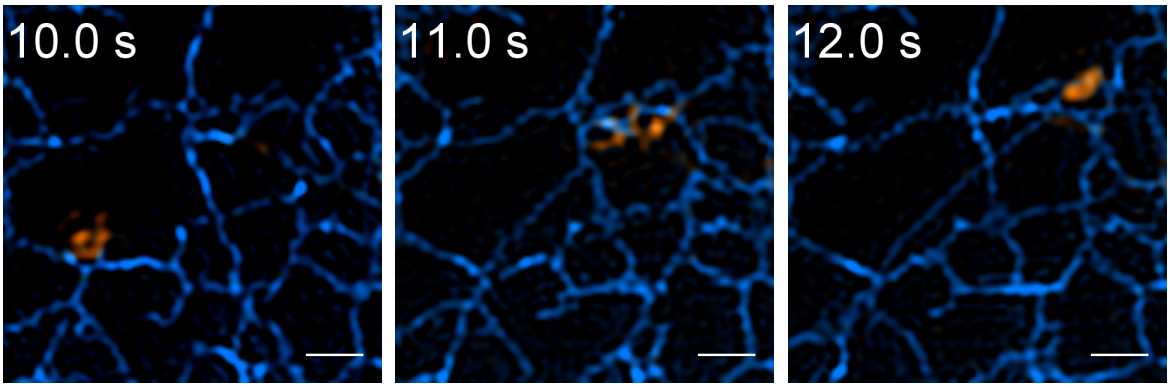
Using the NVIDIA PhysX engine to perform simulations of proteins moving in the ER provided insight into the advantages of active flow over diffusion. Tuning the simulation to ensure the same average particle speed in both the pinching and Brownian motion schemes to ensure a fair comparison showed that mixing occurs up to 3× faster in a network with pinching. Pinching therefore reduces the time taken for proteins to reach remote areas of the network, such as by the cell membrane.

Since ER is found in most eukaryotic cells, it is unsurprising that its malfunction is associated with a range of diseases. By contributing to the community's fundamental understanding of ER mechanisms, we advance our capability for finding cures.

## 5.4 Future research

The dynamic rearrangement of the ER network has been observed by several groups [cite cite cite]. Furthermore is it now known that ER exhibits strong contact with all other organelles. Nevertheless, the mechanisms for network rearrangement are not well understood.

By utilising high-speed multi-colour SIM we have observed lysosomes localised at the endpoints of growing ER tubules, which appear to pull the network into its new shape. Further investigation into this phenomenon is ongoing, but preliminary data is shown in Figure 5.12.




**(a)** **(b)** **(c)**

**Figure 5.12:** Preliminary multi-colour SIM data captured with the Optosplit shows lysosomes, coloured in orange, forming strong contacts with ER tubule endpoints, suggesting that lysosomes assist with rearrangement of the ER network. Scalebars are 1 µm.

# Chapter 6

# Conclusion

**Joerg Bewersdorf** So Marcus, are you an Engineer or a Scientist?
**Marcus** . . . Can I not be both?
*Excerpt from Skype conversation, 25-01-2018*

## 6.1 Wrapping up!

some words

# References


[1] Kelvinsong, "File:Animal Cell.svg - Wikimedia Commons." https://commons.wikimedia.org/wiki/File:Animal_Cell.svg, 2012. Accessed: 2018-10-10.

[2] L. J. Young, F. Ströhl, and C. F. Kaminski, "A guide to structured illumination TIRF microscopy at high speed with multiple colors," *Journal of visualized experiments: JoVE*, no. 111, 2016.

[3] K. O'Holleran and M. Shaw, "Polarization effects on contrast in structured illumination microscopy," *Optics letters*, vol. 37, no. 22, pp. 4603–4605, 2012.

[4] J. M. Bennett, *Handbook of Optics*, vol. 2. McGraw-Hill Inc., 2 ed., 1995.

[5] M. Müller, V. Mönkemöller, S. Hennig, W. Hübner, and T. Huser, "Open-source image reconstruction of super-resolution structured illumination microscopy data in ImageJ," *Nature communications*, vol. 7, p. 10980, 2016.

[6] K. O'Holleran and M. Shaw, "Optimized approaches for optical sectioning and resolution enhancement in 2d structured illumination microscopy," *Biomedical optics express*, vol. 5, no. 8, pp. 2580–2590, 2014.

[7] @Fyrd and @Lensco, "Can I Use... Support tables for HTML5, CSS3, etc." https://caniuse.com/#feat=webgl, 2013. Accessed: 2018-10-10.

[8] A. Kumar, Y. Wu, R. Christensen, P. Chandris, W. Gandler, E. McCreedy, A. Bokinsky, D. A. Colón-Ramos, Z. Bao, M. McAuliffe, G. Rondeau, and H. Schroff, "Dual-view plane illumination microscopy for rapid and spatially isotropic imaging," *nature protocols*, vol. 9, no. 11, p. 2555, 2014.

[9] M. Fantham and C. F. Kaminski, "A new online tool for visualization of volumetric data," *Nature Photonics*, vol. 11, no. 2, p. 69, 2017.

[10] J. Sharpe, U. Ahlgren, P. Perry, B. Hill, A. Ross, J. Hecksher-Sørensen, R. Baldock, and D. Davidson, "Optical projection tomography as a tool for 3d microscopy and gene expression studies," *Science*, vol. 296, no. 5567, pp. 541–545, 2002.

[11] M. Levoy, "Volume rendering—visible volume rendering," *Computer Graphics and Applications*, vol. 8, pp. 29–37, 1988.

[12] R. K. Chhetri, F. Amat, Y. Wan, B. Höckendorf, W. C. Lemon, and P. J. Keller, "Whole-animal functional and developmental imaging with isotropic spatial resolution," *Nature methods*, vol. 12, no. 12, p. 1171, 2015.

[13] J. Mayer, A. Robert-Moreno, R. Danuser, J. V. Stein, J. Sharpe, and J. Swoger, "Optispim: integrating optical projection tomography in light sheet microscopy extends specimen characterization to nonfluorescent contrasts," *Optics letters*, vol. 39, no. 4, pp. 1053–1056, 2014.

[14] F. Huang, G. Sirinakis, E. S. Allgeyer, L. K. Schroeder, W. C. Duim, E. B. Kromann, T. Phan, F. E. Rivera-Molina, J. R. Myers, I. Irnov, *et al.*, "Ultra-high resolution 3d imaging of whole cells," *Cell*, vol. 166, no. 4, pp. 1028–1040, 2016.

[15] A. Torrence, "Martin newell's original teapot," in *ACM SIGGRAPH 2006 Teapot Copyright restrictions prevent ACM from providing the full text for the Teapot exhibits*, p. 29, ACM, 2006.

[16] Google, "Google Cardboard - Google VR." https://vr.google.com/cardboard/, 2018. Accessed: 2018-10-10.

[17] M. H. Teplensky, M. Fantham, P. Li, T. C. Wang, J. P. Mehta, L. J. Young, P. Z. Moghadam, J. T. Hupp, O. K. Farha, C. F. Kaminski, and D. Fairen-Jimenez, "Temperature treatment of highly porous zirconium-containing metal–organic frameworks extends drug delivery release," *Journal of the American Chemical Society*, vol. 139, no. 22, pp. 7522–7532, 2017.

[18] S. Özkar and R. G. Finke, "Nanocluster formation and stabilization fundamental studies. 2. proton sponge as an effective h+ scavenger and expansion of the anion stabilization ability series," *Langmuir*, vol. 18, no. 20, pp. 7653–7662, 2002.

[19] L. D. Cervia, C.-C. Chang, L. Wang, and F. Yuan, "Distinct effects of endosomal escape and inhibition of endosomal trafficking on gene delivery via electrotransfection," *PloS one*, vol. 12, no. 2, p. e0171699, 2017.

[20] I. Abánades Lázaro, S. Haddad, J. M. Rodrigo-Muñoz, C. Orellana-Tavra, V. del Pozo, D. Fairen-Jimenez, and R. S. Forgan, "Mechanistic investigation into the selective anticancer cytotoxicity and immune system response of surface-functionalized, dichloroacetate-loaded, uio-66 nanoparticles," *ACS applied materials & interfaces*, vol. 10, no. 6, pp. 5255–5268, 2018.

[21] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, *et al.*, "Cellprofiler: image analysis software for identifying and quantifying cell phenotypes," *Genome biology*, vol. 7, no. 10, p. R100, 2006.

[22] GraphPad Software, "Graphpad Prism version 7 for Windows." https://www.graphpad.com/, 1995. Accessed: 2018-10-10.

[23] M. J. Phillips and G. K. Voeltz, "Structure and function of er membrane contact sites with other organelles," *Nature reviews Molecular cell biology*, vol. 17, no. 2, p. 69, 2016.

[24] A. M. Valm, S. Cohen, W. R. Legant, J. Melunis, U. Hershberg, E. Wait, A. R. Cohen, M. W. Davidson, E. Betzig, and J. Lippincott-Schwartz, "Applying systems-level spectral imaging and analysis to reveal the organelle interactome," *Nature*, vol. 546, no. 7656, p. 162, 2017.

[25] L. Ozcan and I. Tabas, "Role of endoplasmic reticulum stress in metabolic disease and other disorders," *Annual review of medicine*, vol. 63, pp. 317–328, 2012.

[26] C. A. Hübner and I. Kurth, "Membrane-shaping disorders: a common pathway in axon degeneration," *Brain*, vol. 137, no. 12, pp. 3109–3121, 2014.

[27] C. Blackstone, C. J. O'kane, and E. Reid, "Hereditary spastic paraplegias: membrane traffic and the motor pathway," *Nature Reviews Neuroscience*, vol. 12, no. 1, p. 31, 2011.

[28] M. J. Dayel, E. F. Hom, and A. Verkman, "Diffusion of green fluorescent protein in the aqueous-phase lumen of endoplasmic reticulum," *Biophysical journal*, vol. 76, no. 5, pp. 2843–2851, 1999.

[29] S. Nehls, E. L. Snapp, N. B. Cole, K. J. Zaal, A. K. Kenworthy, T. H. Roberts, J. Ellenberg, J. F. Presley, E. Siggia, and J. Lippincott-Schwartz, "Dynamics and retention of misfolded proteins in native er membranes," *Nature cell biology*, vol. 2, no. 5, p. 288, 2000.

[30] I. Arganda-Carreras, V. Kaynig, C. Rueden, K. W. Eliceiri, J. Schindelin, A. Cardona, and H. Sebastian Seung, "Trainable weka segmentation: a machine learning tool for microscopy pixel classification," *Bioinformatics*, vol. 33, no. 15, pp. 2424–2426, 2017.

[31] S. Nadeem and E. Maraj, "The mathematical analysis for peristaltic flow of nano fluid in a curved channel with compliant walls," *Applied Nanoscience*, vol. 4, no. 1, pp. 85–92, 2014.

[32] K. Subramanian and T. Meyer, "Calcium-induced restructuring of nuclear envelope and endoplasmic reticulum calcium stores," *Cell*, vol. 89, no. 6, pp. 963–971, 1997.

[33] T. Lucretius Carus, *Titvs Lvcretivs Carvs: De rerum natura, 66 BC*. Amstelodami: Apud Ioann: Ianssonium, 1631.

[34] D. Holcman, P. Parutto, J. E. Chambers, M. Fantham, L. J. Young, S. J. Marciniak, C. F. Kaminski, D. Ron, and E. Avezov, "Single particle trajectories reveal active endoplasmic reticulum luminal flow," *Nature cell biology*, p. 1, 2018.